AIX версии 7.2

Управление сетями и средствами связи

AIX версии 7.2

Управление сетями и средствами связи

Примечание

Перед началом работы с этим изданием и описанным в нем продуктом ознакомьтесь с информацией, приведенной в разделе “Примечания” на стр. 695.

Данное издание относится к AIX версии 7.2, а также ко всем последующим выпускам и модификациям, если в соответствующих изданиях не будет оговорено обратное.

Содержание

Об этом документе

В этом документе разработчики приложений найдут полную информацию о включении приложений для поддержки глобализации в операционной системе AIX. Кроме того, она содержит информацию для системных администраторов о включении сетевых сред для поддержки глобализации в операционной системе AIX. Документ содержит основную информацию о глобализации и рекомендации по ее применению. В разделах рассмотрены локали, кодовые наборы, методы ввода, процедуры, конвертеры, преобразования символов, национальная информация и средства обмена сообщениями.

Выделение текста

В данном документе применяются следующие специальные обозначения:

Элемент	Описание
Полужирный шрифт	Полужирным шрифтом выделены команды, процедуры, ключевые слова, имена файлов, структуры, каталоги и другие системные объекты с предопределенными именами. Кроме того, полужирным шрифтом выделены названия графических объектов, выбираемых пользователем - кнопок, меток, значков и т.д.
Курсив	Курсивом выделены те параметры, имена или значения которых задаются пользователем.
`Непропорциональный шрифт`	Непропорциональным шрифтом выделены конкретные значения, текст, который вы можете увидеть на экране, фрагменты программных кодов, системные сообщения и данные, которые вам будет предложено ввести.

Учет регистра символов в AIX

В операционной системе AIX учитывается регистр символов, т.е. различаются прописные и строчные буквы. Например, с помощью команды **ls** можно просмотреть список файлов. Если ввести `LS`, то будет выдано сообщение о том, что команда `не найдена`. Точно так же, **FILEA**, **FiLea**, и **filea** - это имена трех различных файлов, даже если эти файлы находятся в одном каталоге. Во избежание нежелательных последствий всегда проверяйте правильность регистра букв.

ISO 9000

При разработке и производстве данного продукта использовались зарегистрированные системы ISO 9000.

Управление сетями и средствами связи

Системные администраторы и пользователи выполняют множество различных задач, связанных с обменом данных по сети. Системные администраторы надут в этом разделе сведения о выполнении таких задач, как настройка параметров TCP/IP, повышение защищенности сети и наблюдение за работой системы. Пользователи найдут полную информацию о применении программ и служб связи в операционной системе. Также приведена информация о настройке и устранении неполадок почтовых служб, MH (Обработчик сообщений), NFS (Сетевая файловая система), HA-NFS (NFS высокой готовности), TCP/IP, BNU (Основные сетевые утилиты), последовательных устройств связи и терминалов, ATE (Эмуляция асинхронного терминала) и SNMP (Простой протокол управления сетью). Кроме того приведена информация о процедурах получения и отправки почты и сообщений, передачи файлов (команда **ftp**), печати файлов в удаленной системе или из удаленной системы, выполнения команд в других системах, обмена данными между локальной и удаленной системой, а также настройки соединений. Данный раздел можно найти и на компакт-диске документации, который поставляется вместе с операционной системой.

Управление сетью и связью - новое

В этом разделе описаны новые или измененные возможности по теме Управление сетью и связью.

Как узнать об изменениях и добавлениях

В данном файле PDF новая и измененная информация может выделяться значками (|) в левом поле.

Октябрь 2017 года

Ниже приведено краткое описание изменений, внесенных в разделы из этой книги:

- Удалена устаревшая информация об адаптерах с поддержкой режима асинхронной передачи (ATM).

Апрель 2017 года

- Обновлена информация об удалении адаптера из EtherChannel в разделе “Внесение изменений в канал EtherChannel с помощью динамической настройки адаптеров” на стр. 389.

Ноябрь 2016 года

- Обновлена информация о сетевой установке с помощью EtherChannel в разделе “Замечания по настройке канал EtherChannel” на стр. 379.

Средства связи и сети

Понимание основных принципов функционирования компьютерных сетей чрезвычайно важно. Эта информация предназначена для системных администраторов, которые не знакомы с принципами работы сетей. Пользователи, ранее работавшие с сетями UNIX, могут пропустить этот раздел.

Сетью называется совокупность нескольких компьютеров и соединяющих их кабелей. *Физическая* сеть - это аппаратное обеспечение (карты адаптеров, кабели и телефонные линии). Программное обеспечение и структура сети образуют *логическую* сеть. Существует множество различных типов сетей и эмуляторов, предназначенных для выполнения различных функций.

Связь

В сетях могут быть реализованы различные пользовательские и прикладные функции связи.

В частности, они применяются для выполнения следующих задач:

- Отправка сообщений электронной почты (e-mail)
- Эмуляция терминала и работа с удаленным компьютером
- Передача данных
- Запуск программ на удаленном компьютере.

Одно из наиболее популярных применений компьютерных сетей - электронная почта, позволяющая пользователям обмениваться сообщениями. Пользователи могут работать с одной системой (в этом случае сеть не нужна) или с разными системами, находящимися в разных зданиях или даже в разных странах. Нижние уровни программного и аппаратного обеспечения, а также физическая сеть, позволяют пользователю создавать, отправлять, получать и обрабатывать сообщения, письма, заметки, приглашения и файлы данных. При этом взаимодействие возможно между любыми пользователями, подключенными к физической сети. В программе электронной почты предусмотрены такие функции, как составление комментариев к сообщениям, задание последовательности сообщений, упаковка сообщений, сортировка данных, а также управление папками почты.

Кроме того, с помощью сети один компьютер может *имитировать* другой компьютер или терминал, обеспечивая доступ к его информации и функциям. Функции удаленного входа в систему позволяют пользователям с помощью интерактивного интерфейса командной строки входить в удаленные системы и работать в них с программами и файлами точно так же, как на собственном компьютере.

Сети позволяют передавать данные из одной системы в другую. Файлы, каталоги и целые файловые системы можно переносить с одного компьютера на другой для создания резервной копии или дубликата данных на случай сбоя системы. Обычно в протоколе бывает предусмотрена защита с помощью пароля. Протокол передачи файлов обеспечивает взаимосвязь типа клиент/сервер между пользователем, отправившим запрос, и удаленной системой. Часто в протоколе передачи файлов бывают предусмотрены функции, позволяющие пользователям с правами доступа на чтение/запись просматривать, создавать и удалять файлы и каталоги.

Существует еще несколько протоколов, позволяющих пользователям и приложениям одной системы вызывать процедуры и выполнять программы в другой системе. Такие протоколы могут применяться в различных средах, в том числе для вызова сложных вычислительных процедур в инженерных и научных приложениях.

Сети

Сложность современных компьютерных сетей привела к возникновению нескольких концептуальных моделей, объясняющих принципы работы сети.

Одна из наиболее распространенных - это Справочная модель взаимодействия открытых систем (OSI), разработанная Международной организацией по стандартизации. Другое название этой модели - "семиуровневая модель OSI".

Уровни модели OSI нумеруются следующим образом:

Элемент	Описание
7	**Прикладной уровень**
6	**Уровень представления**
5	**Уровень сеанса**
4	**Транспортный уровень**
3	**Сетевой уровень**
2	**Канальный уровень**
1	**Физический уровень**

Первые три уровня зависят от того, какую физическую сеть вы используете. Уровни 4-7 независимы от типа сети и выполняют функции более высокого уровня. Уровни сети в данной модели выделяются по принципу выполнения конкретной функции передачи данных (а не использования какого-либо конкретного

протокола). Ниже описаны функции каждого из уровней, начиная с нижнего (аппаратного уровня) и заканчивая высшим (уровень, на котором происходит взаимодействие с пользователем):

Элемент	Описание
Прикладной уровень	Включает в себя все приложения, работающие с сетью.
Уровень представления	Отвечает за представление данных приложениям в необходимом формате.
Уровень сеанса	Устанавливает соединение между приложениями.
Транспортный уровень	Обеспечивает безошибочную передачу данных.
Сетевой уровень	Устанавливает соединение с другими компьютерами сети.
Канальный уровень	Обеспечивает надежную доставку данных по физической линии связи (которая, как правило, ненадежна).
Физический уровень	Определяет тип физической линии связи. Например, оптоволоконный кабель, необходимый для применения протокола Оптоволоконного интерфейса распределенных данных (FDDI), представляет собой одну из частей физического уровня.

Примечание: Справочная модель OSI весьма полезна для обсуждения принципов функционирования сетей, однако многие сетевые протоколы не полностью соответствуют этой модели. Например, в наборе протоколов TCP/IP прикладной уровень и уровень представления, уровень сеанса и транспортный уровень, а также канальный и физический уровни попарно объединены.

Каждый уровень модели OSI взаимодействует с соответствующим уровнем в удаленной системе, как показано на рисунке Модель OSI.

На этом рисунке показаны различные уровни модели OSI.

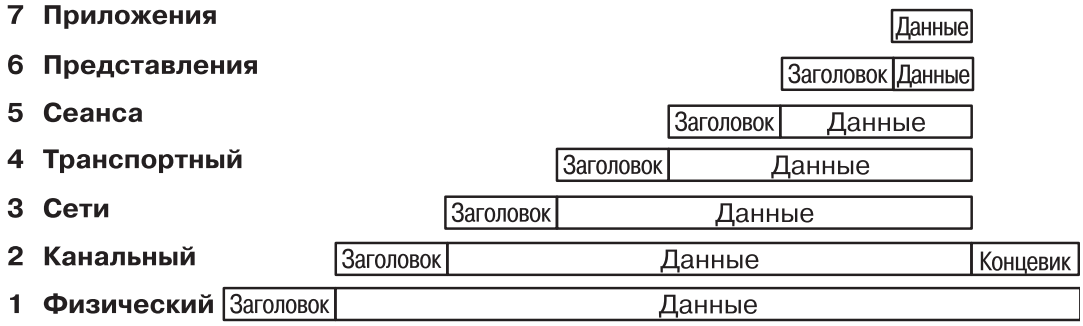


Рисунок 1. Модель OSI

Передача данных возможна только между смежными уровнями. На каждом уровне к пакету данных, полученному с более высокого уровня, добавляется заголовок (а на канальном уровне еще и концевик).

Сети применяются отдельными пользователями и организациями для решения следующих задач:

- Ввод данных
- Формирование запросов к данным
- Удаленная пакетная загрузка данных
- Совместное использование ресурсов
- Совместное использование данных
- Электронная почта.

Под вводом данных понимается запись данных напрямую в локальный или удаленный файл. При подобном одношаговом способе увеличивается точность и скорость передачи данных. Запрос к данным - это возможность поиска конкретной информации в файлах. Под обновлением данных понимается возможность изменения, добавления и удаления данных, располагающихся в локальных и удаленных файлах. Возможность удаленного выполнения пакетных заданий позволяет обрабатывать большие объемы данных ночью и в периоды малой загрузки системы. Все перечисленные возможности делают соединение компьютеров в сети уже не только полезным, но и необходимым.

Еще одна функция сети - совместное использование ресурсов: данных, программ, дискового пространства, периферийных устройств (например, принтеров, модемов и жестких дисков). Эта функция позволяет сэкономить ресурсы (например, в случае периферийных устройств), избавляет от необходимости хранения нескольких экземпляров программ и обеспечивает согласованность данных (в случае программ и файлов).

Физические сети

На физическом уровне сеть состоит из кабелей (коаксиальных, витых пар, оптоволоконных или телефонных), соединяющих различные устройства; карт адаптеров, используемых на подключенных хостах; а также концентраторов, повторителей, маршрутизаторов и мостов, применяемых в сети.

На физическом уровне сети различаются по размеру и типу используемого аппаратного обеспечения. Есть два типа сетей - *локальные* (LAN) и *глобальные* (WAN). Физический размер локальных сетей ограничен и составляет от 1 до 10 км. Например, это может быть сеть, расположенная в одном здании или принадлежащая небольшой организации. Глобальная сеть обеспечивает связь в больших географических масштабах по сравнению с локальной сетью, например, между различными регионами страны или между континентами. Существует и промежуточный класс сетей - *сети городского масштаба* (MAN). В данном руководстве сети городского масштаба не выделяются в отдельный класс: они рассматриваются как глобальные сети.

Обычно локальные сети на физическом уровне используют обычный Ethernet, Ethernet IEEE 802.3, или Token-Ring, а глобальные и асинхронные сети - линии, предоставляемые операторами связи. Реализация сетей обычно соответствует определенным сетевым стандартам, разрабатываемым такими организациями, как Ассоциация электронной промышленности (EIA) и Международное телекоммуникационное объединение (ITU).

Сетевые системы

Для организации сетевого взаимодействия необходимо аппаратное и программное обеспечение. Аппаратные и программные средства связи представляют собой оборудование, а также программное обеспечение, которые управляет работой этого оборудования и обеспечивает возможность взаимодействия с сетью.

Аппаратное обеспечение - это оборудование, подключенное к физической сети. *Программное обеспечение* - это программы и драйверы устройств, предназначенные для работы в конкретной системе. Аппаратное обеспечение системы включает карты адаптеров или другие устройства, обеспечивающие связь между программным обеспечением системы и физической сетью. Для карты адаптера в системе должно быть предусмотрено специальное гнездо ввода/вывода (I/O). Карта адаптера соединяет *терминальное оборудование* (DTE) с *конечным оборудованием для передачи данных* (DCE); то есть, она обеспечивает локальную адресацию портов DTE на физическом уровне. Другие устройства, например, модемы, могут быть подключены к одному из стандартных портов компьютера.

Карта адаптера обеспечивает подготовку поступающих и отправляемых данных, выполняет поиск адресов, обеспечивает защиту драйверов, получателей, а также защиту системы от перегрузки. Таким образом, карта адаптера освобождает процессор системы от выполнения многих задач связи. Карты адаптеров обеспечивают поддержку стандартов физических сетей (например, EIA 232D, Smartmodem, V.25 bis, EIA 422A, X.21 или V.35), а также поддерживают некоторые программные *протоколы* (например, SDLC, HDLC и бисинхронные протоколы). Если адаптер не поддерживает программный протокол, то такая поддержка должна предоставляться драйвером адаптера.

Протоколы

Все программное обеспечение средств связи основано на *протоколах*, представляющих собой наборы семантических и синтаксических правил, которые определяют, каким образом должны действовать различные устройства для установления соединения.

Протоколы указывают, каким образом должна осуществляться доставка информации, в каком формате ее нужно отправлять в пункт назначения и по какому маршруту она должна передаваться. Кроме того, в протоколах определены правила обмена сообщениями и подтверждениями.

Протоколы существуют на разных уровнях внутри ядра и с ими нельзя работать непосредственно. Однако с протоколами можно работать косвенным путем, задавая необходимые действия на уровне интерфейса прикладных программ (API). Выбирая какие-либо опции и команды при запуске программ передачи файлов, удаленного входа в систему или эмуляции терминала, пользователь тем самым определяет набор применяемых протоколов.

Адреса

Адреса присваиваются как программам, так и устройствам. Адрес представляет собой средство, с помощью которого передающая или управляющая система выбирает систему, на которую необходимо отправить данные.

Другими словами, адрес служит идентификатором расположения отправителя и получателя. Физический адрес - это уникальный код, присваиваемый каждому устройству или рабочей станции, подключенной к сети.

Например, в сети Token-Ring просмотреть физический адрес адаптера можно командой **netstat -iv**. Это адрес физической сети. Кроме него команда **netstat -iv** позволяет просмотреть информацию об адресах уровня класса и пользователя. Адреса часто определяются программным обеспечением, но могут также задаваться и пользователем.

Домены

Во многих сетях применяется принцип адресации, основанный на концепции *доменов*. Домены помещают ресурсы обработки данных в сети под общий контроль.

Например, структура сети Internet наглядно показывает, каким образом структура доменов определяет адрес протокола Internet (IP-адрес). Internet представляет собой обширную сеть, объединяющую множество различных сетей меньшего размера. Для выполнения маршрутизации и адресации в сети Internet адреса классифицируются в соответствии с иерархией доменов; на верхнем уровне этой иерархии находятся такие домены, как `com` - для коммерческих организаций, `edu` - для образовательных учреждений и `gov` - для правительственных организаций.

Внутри домена `com` имеется множество доменов меньшего размера, соответствующих отдельным компаниям, например домен `ibm`. Внутри домена `ibm.com` есть еще более мелкие домены, соответствующие IP-адресам разных филиалов, например, `austin.ibm.com` или `raleigh.ibm.com`. На этом уровне мы начинаем различать имена *хостов*. В данном контексте хост - это любой компьютер, подключенный к сети. Внутри домена `austin.ibm.com` могут быть хосты с именами `hamlet` и `lear`, их адреса имеют вид `hamlet.austin.ibm.com` и `lear.austin.ibm.com`, соответственно.

Шлюзы и мосты

Сеть Internet состоит из множества сетей разных типов, в которых часто используется различное аппаратное и программное обеспечение. *Шлюзы* и *мосты* позволяют соединять такие сети друг с другом.

Мост - это устройство, соединяющее две локальные сети, которые, возможно, используют одну и ту же процедуру управления логическим каналом связи (LLC), например, Ethernet, но разные процедуры управления доступом к среде передачи данных (MAC). Шлюз - это более широкое понятие, чем мост. Он действует на более высоком уровне, чем уровень передачи данных, и применяется для преобразования интерфейсов и протоколов. Шлюзы позволяют передавать данные через сети различного типа, входящие в состав Internet.

Маршрутизация данных

Применение доменных имен для адресации и шлюзов для преобразования сильно упрощает *маршрутизацию* передаваемых данных. Маршрутизация - это определение маршрута, по которому сообщение доставляется в пункт назначения.

Имя домена эффективно определяет назначение сообщения. В большой сети, такой, как Internet, информация передается из одной сети в другую сеть до тех пор, пока не достигнет пункта назначения. Каждая сеть

сравнивает имя получателя с именами известных ей доменов и направляет данные на следующий логический узел. Таким образом, каждая сеть, которая принимает данные, вносит свой вклад в процесс маршрутизации.

Локальные и удаленные узлы

Хосты, находящиеся в данной сети, используют физическую сеть. Хост называется *узлом* сети. Узел представляет собой подключенное к сети устройство, к которому можно обращаться по его адресу, и на котором могут запускаться службы хостов. С точки зрения сетевого взаимодействия узлы делятся на *локальные* и *удаленные*.

Локальным может быть устройство, файл или система, к которым можно обращаться непосредственно из вашей системы, без использования линии связи. *Удаленным* может быть устройство, файл или система, к которым ваша система должна обращаться по линии связи. Локальные файлы находятся в вашей системе, а удаленные - на файловом сервере или на другом узле, с которым вы соединяетесь с помощью физической сети, например, сети Ethernet, Token-Ring или телефонной линии.

Клиент и сервер

Сервер - это компьютер, на котором хранятся данные, или который выполняет определенные служебные функции для других компьютеров сети. *Клиент* - это компьютер, запрашивающий некоторую функцию или данные у сервера.

К широко распространенным типам серверов относятся файловые серверы, на которых хранятся файлы; серверы имен, на которых хранятся адреса и имена; серверы приложений, на которых хранятся программы и приложения; а также серверы печати, хранящие и передающие задания клиентов на печать.

Клиент может запрашивать с сервера обновленные версии программ или загружать приложения с базового сервера. Для получения имени или адреса клиент может обращаться к серверу имен. Для ввода данных, формирования запросов или обновлении записей клиент может обращаться к файловому серверу.

Связь с другими операционными системами

К одной сети могут быть подключены компьютеры разных типов. Например, это могут быть компьютеры различных производителей или различные модели компьютеров одного производителя. Современные программы связи позволяют преодолеть различия в архитектуре операционных систем.

В некоторых случаях для работы таких программ в сети необходимо предварительно установить специальное программное обеспечение. Иногда для работы программ может потребоваться установить в сети протоколы связи, например, TCP/IP или Системной сетевой архитектуры (SNA).

Приложения эмуляции хоста

Эмулятор - это приложение, позволяющее вашей системе работать так, как будто она использует другой терминал или принтер.

Эмулятор терминала подключается к удаленному хосту для получения доступа к данным или приложениям. Некоторые эмуляторы терминалов обеспечивают обмен файлами с удаленным хостом. Другие эмуляторы предоставляют интерфейс прикладных программ (API), позволяющий организовать взаимодействие между программами и автоматизировать выполнение задач на удаленном хосте.

Эмулятор принтера позволяет напечатать файлы на локальном принтере или сохранять их для печати или редактирования и дальнейшей печати.

Существует несколько приложений, эмулирующих различные типы терминалов. В данном разделе приведена информация об эмуляторах терминалов и принтеров.

Примечание: Команда **bterm** эмулирует терминалы, работающие в двунаправленном режиме (BIDI).

Команды TCP/IP для эмуляции

В программное обеспечение Протокола управления передачей/Протокола Internet (TCP/IP) включены команды **telnet** и **rlogin**, позволяющие подключаться к удаленным системам TCP/IP.

Элемент	Описание
telnet	Эта команда позволяет подключиться к удаленному хосту с помощью протокола **TELNET**. Главное ее отличие от команды **rlogin** состоит в том, что telnet - это защищенная команда. *Защищенная* команда должна удовлетворять требованиям, предъявляемым на всех уровнях защиты вашего компьютера. В системах с повышенными требованиями к защите должны выполняться только защищенные команды. Требования к защищенным командам, процессам и программам устанавливаются и поддерживаются стандартами Министерства обороны США.
tn	Эта команда выполняет те же функции, что и команда **telnet**.
rlogin	С помощью этой команды пользователь может войти в удаленную систему. Отличие этой команды от команды **telnet** в том, что она является *ненадежной* командой и может быть запрещена системным администратором, если к защите сети предъявляются повышенные требования.

Дополнительная информация о **TCP/IP** приведена в разделе “Протокол TCP/IP” на стр. 102.

Команды BNU для эмуляции

Программное обеспечение Основные сетевые утилиты (BNU) предоставляет команды **ct**, **cu** и **tip**, позволяющие подключиться к удаленной системе, работающей под управлением AIX.

Элемент	Описание
ct	Эта команда позволяет пользователю удаленного терминала, например 3161, установить соединение с другим терминалом по телефонной линии. После этого пользователь удаленного терминала может войти в систему и работать с ней. Команда **ct** аналогична команде **cu**, но диапазон ее возможностей не так широк. Например, если соединение с удаленной системой установлено с помощью команды **ct**, то пользователь не может вводить команды локальной системы. Однако в команде **ct** предусмотрен режим автоматического набора одного или нескольких номеров до установления соединения.
cu	Эта команда позволяет подключить терминал к другому терминалу, на котором может быть установлена операционная система UNIX или другая система. После установления соединения пользователь может одновременно работать в обеих системах и выполнять команды в любой из них без прерывания соединения BNU. Если удаленный терминал работает под управлением операционной системы UNIX, то между двумя системами можно передавать файлы ASCII. Команду **cu** можно также использовать для установления соединения с несколькими системами, после чего можно выполнять команды в любой из этих систем.
tip	Эта команда подключает терминал к удаленному терминалу и позволяет работать с ним как при прямом подключении. Команда **tip** может применяться для обмена файлами с удаленной системой. Вы можете записать протокол работы с командой **tip**. **Примечание:** Для использования команды **tip** должно быть установлено соединение с удаленной системой.

Дополнительная информация о BNU приведена в разделе “Основные сетевые утилиты” на стр. 438.

Эмуляция асинхронного терминала

Программа эмуляции асинхронного терминала (ATE) позволяет установить соединение с системами, которые поддерживают асинхронные терминалы, включая системы с поддержкой RS-232C и RS-422A.

Работая с ATE, удаленная система обращается к вашему терминалу как к асинхронному дисплею или терминалу DEC VT100.

ATE позволяет выполнять команды в удаленной системе, отправлять и получать файлы, а также проверять целостность данных в передаваемых файлах. Кроме того, вы можете записывать данные, поступающие из удаленной системы, в файл *приема*. Управлять ATE можно с помощью меню и подкоманд.

После установки программы ATE с ней смогут работать только пользователи с правами доступа root и члены группы UUCP.

Дополнительная информация об ATE приведена в разделе “Эмуляция асинхронного терминала” на стр. 635.

Команды средств связи системы

Здесь описаны различные команды, позволяющие просмотреть информацию о пользователях системы, определить имя системы, с которой вы работаете, а также просмотреть список пользователей, работающих в других системах.

Описание различных команд, используемых для просмотра информации о системе и пользователе приведены в следующих разделах.

Просмотр своего имени пользователя

С помощью команды **whoami** можно определить свое имя пользователя.

Для отображения имени текущего пользователя введите:

```
whoami
```

Будет показана примерно следующая инфомрация:

```
denise
```

В данном примере именем пользователя является `denise`.

Просмотр имени системы

С помощью команды **uname** можно определить имя системы.

1. Для просмотра сетевого имени вашей системы введите:

   ```
   uname -n
   ```

   Будет показана примерно следующая информация:

   ```
   barnard
   ```

   В данном примере именем системы является `barnard`.
2. Для того чтобы узнать имя узла другой системы, попросите пользователя этой системы ввести команду **uname-n**.

Определение прав доступа вашей системы

С помощью команды, **host** можно определить, имеет ли ваша система доступ к информации, определяющей другую систему.

Для работы с другой системой сети у вашей локальной сети должен быть доступ к информации, определяющей другую систему. Для того чтобы выяснить, есть ли у вашей локальной системы доступ к этой информации, введите команду **host** и имя удаленной системы. как показано в следующем примере:

Для определения, есть ли у вашей системы информация о маршруте доступа к системе `zeus`, введите:

```
host zeus
```

При наличии доступа будет показана примерно следующая информация:

```
zeus - 192.9.200.4  (300,11,310,4)
```

Вы можете отправить сообщение в систему `zeus`. Адрес `192.9.200.4` применяется системой для доставки почты. При отсутствии доступа будет показана примерно следующая информация:

```
zeus: неизвестный хост
```

Если получено сообщение `неизвестный хост`, это может свидетельствовать о следующих ошибках:

- Неверное имя хоста (проверьте правильность адреса)
- Хост находится в вашей сети, но не определен в конфигурации вашей системы (обратитесь к администратору сети)
- Хост находится в другой сети (см. раздел “Отправка почты пользователям другой сети” на стр. 23) и требует более подробной адресации
- Хост не подключен к вашей сети

Кроме того, сообщение о `неизвестном хосте` может быть получено, если сеть не работает, а ваша локальная система получает сетевые адреса из другой системы сети.

Просмотр информации о работающих в системе пользователях

С помощью команды **finger** или **f** можно просмотреть информацию о пользователях, которые в данный момент работают в системе.

Эта информация включает: идентификатор пользователя, его полное имя, имя терминала, дату и время входа в систему.

1. Для просмотра списка текущих пользователей хоста `@alcatraz` введите следующую команду:

   ```
   finger @alcatraz
   ```

   Будет показана примерно следующая информация:

   ```
   brown  Console Март 15 13:19
   smith   pts0    Март 15 13:01
   jones   tty0    Март 15 13:01
   ```

   Пользователь `brown` вошел в систему с консоли, пользователь `smith` - с устройства pts0, а пользователь `jones` - с tty0.
2. Для получения информации о пользователе `brown` из предыдущего примера введите:

   ```
   finger brown@alcatraz
   ```

   or

   ```
   finger brown
   ```

   Будет показана примерно следующая информация:

   ```
   Идентификатор пользователя: brown
   Полное имя: Marta Brown
   Каталог:/home/brown    Оболочка: /bin/ksh
   Дата входа: 8 мая 07:13:49 консоль
   No Plan.
   ```

Управление электронной почтой

С помощью электронной почты пользователи одной или нескольких систем в сети могут обмениваться электронными сообщениями (e-mail). В этом разделе описана система доставки почты, стандартный пользовательский интерфейс, **Протокол доступа к сообщениям Internet (IMAP)** и **Почтовый протокол (POP)**.

Почтовая система состоит из пользовательского интерфейса, программы маршрутизации сообщений и программы доставки сообщений. Она передает сообщения от одного пользователя другому на одном хосте, между хостами в одной сети и между сетями. Кроме того, она изменяет заголовки сообщений с целью преобразовать сообщение в формат, применяемый на целевом хосте.

Пользовательский интерфейс электронной почты позволяет создавать, отправлять и принимать сообщения. В почтовой системе предусмотрено два пользовательских интерфейса: **mail** и **mhmail**. Команда **mail** - это стандартный пользовательский интерфейс, существующий во всех системах UNIX. Команда **mhmail**

запускает пользовательский интерфейс Обработчик сообщений (Message Handler - MH), содержащий расширенный набор функций, и предназначенный для опытных пользователей.

Программа маршрутизации сообщений рассылает сообщения адресатам. В почтовой системе роль такой программы играет программа **sendmail**, которая входит в состав Базовой операционной системы (BOS) и устанавливается вместе с ней. Программа **sendmail** - это демон, выполняющий маршрутизацию на основе информации из файлов `/etc/mail/sendmail.cf` и `/etc/mail/aliases`.

Команда **sendmail** применяет различные *программы доставки сообщений*, в зависимости от типа маршрута к целевой системе.

На рисунке приведена схема, в вершине которой расположены записи Почта и MH. Ниже расположены

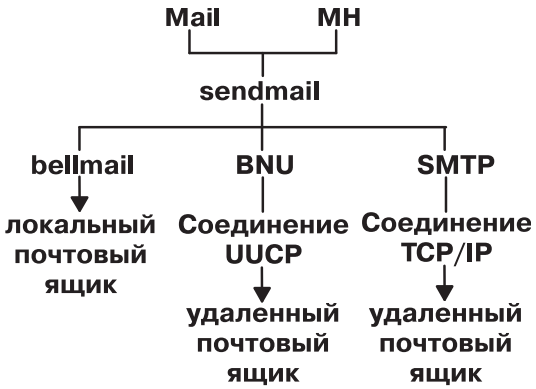


Рисунок 2. Почтовые программы, применяемые командой sendmail

записи bellmail, BNU и SMTP. Под предыдущим уровнем расположены локальный почтовый ящик, соединение UUCP и соединение **TCP/IP**, соответственно. Под соединениями UUCP и **TCP/IP** расположены удаленные почтовые ящики.

На рисунке представлены следующие этапы работы с почтой:

- Для доставки локальной почты программа **sendmail** передает сообщения программе **bellmail**. Программа **bellmail** в свою очередь, отправляет всю локальную почту в системный почтовый ящик пользователя, расположенный в каталоге /var/spool/mail.
- Для доставки почты по каналу связи Программы копирования UNIX-UNIX (UUCP) программа **sendmail** передает сообщения с помощью Основных сетевых утилит (BNU).
- Для доставки почты с помощью протокола **TCP/IP** программа **sendmail** устанавливает соединение **TCP/IP** с удаленной системой, а затем передает ей сообщение с помощью протокола **SMTP**.

Пользовательские программы работы с почтой

Для работы с почтой необходимо выбрать одну из следующих программ. Можно выбрать почтовую программу (**mail**), обработчик сообщений (**mh**) или команду **bellmail**.

Эти программы позволяют создавать, получать, отправлять и сохранять почту. Кроме того, вам понадобится транспортная программа-посредник **sendmail**, которая распределяет почту, поступающую из других систем или пакетов, и передает все отправляемые сообщения аналогичной программе в одной или нескольких удаленных системах.

> **Примечание:** Программы **mail** и **mh** применяют несовместимые способы хранения почты, поэтому вы должны выбрать только одну из них.

Интерфейс почтовой программы

Программа **mail** обеспечивает пользовательский интерфейс, позволяющий работать с почтой для пользователей как локальной, так и удаленных систем.

Почтовым сообщением может быть текст, введенный с помощью текстового редактора, или файл ASCII. Кроме текстового сообщения или файла можно отправить следующие виды почты:

Элемент	Описание
системное сообщение	Сообщает пользователям об обновлении системы. Системное сообщение - это оповещающее сообщение, отправляемое только по локальной сети.
секретная почта	Применяется для пересылки секретной информации. Сообщения секретной почты шифруются. Для просмотра такого сообщения получатель должен ввести пароль.
сообщение об отсутствии	Сообщает пользователям о вашем отсутствии. Если во время вашего отсутствия система получит адресованное вам сообщение, то пошлет его отправителю уведомление. Кроме того, система может переслать по указанному адресу любое сообщение, полученное во время вашего отсутствия.

При получении почты с помощью подкоманд программы **mail** можно выполнить следующие операции:

- Оставить почту в системном почтовом ящике.
- Прочитать и удалить почту.
- Отправить почту.
- Добавить комментарии к сообщениям.
- Сохранить почту в личном почтовом ящике (mbox).
- Сохранить почту в созданной папке.
- Создать или изменить файл псевдонимов или список рассылки.

Установка программы **sendmail** выполняется автоматически.

Дополнительная информация о программе **mail** приведена в разделе “Функции почты” на стр. 12.

Обработчик сообщений (mh)

Программа **mh** - это набор команд, позволяющих выполнять все необходимые функции обработки почты прямо из командной строки.

Эти команды обеспечивают более широкие возможности по сравнению с подкомандами программы **mail**. Поскольку их можно вводить в любой командной строке, они существенно упрощают и ускоряют процесс создания сообщений и обработки полученной почты. Например, вы можете просмотреть почтовое сообщение, найти файл или запустить программу для поиска той или иной информации, затем ответить на полученное сообщение, - и все это в одной и той же оболочке.

Программа **mh** позволяет создавать, рассылать, получать, просматривать, обрабатывать и сохранять сообщения с помощью следующих команд:

Элемент	Описание
ali	Создает список почтовых псевдонимов и адресов.
anno	Аннотирует сообщения.
ap	Анализирует и форматирует адреса.
burst	Разворачивает сообщения.
comp	Запускает редактор для создания или изменения сообщения.
dist	Перераспределяет сообщение на дополнительные адреса.
dp	Анализирует и форматирует даты.
folder	Выбирает и показывает папки и сообщения.
folders	Перечисляет все папки и сообщения, находящиеся в почтовом каталоге.
forw	Пересылает сообщения.
inc	Помещает новую почту в папку.
mark	Создает, изменяет или показывает последовательность сообщений.
mhl	Показывает отформатированный список сообщений.
mhmail	Отправляет или получает почту.
mhpath	Показывает полные имена сообщений и папок.
msgchk	Проверяет наличие новых сообщений.

Элемент	Описание
msh	Создает оболочку для работы с почтой (mh).
next	Показывает следующее сообщение.
packf	Сжимает содержимое папки.
pick	Выбирает сообщения по содержанию, а также создает или изменяет последовательность.
prev	Показывает предыдущее сообщение.
refile	Перемещает файлы между папками.
repl	Отвечает на сообщение.
rmf	Удаляет папки вместе с сообщениями.
rmm	Удаляет активные сообщения.
scan	Показывает список сообщений (по одному на строке).
send	Отправляет сообщение.
show	Показывает сообщения.
sortm	Сортирует сообщения.
vmh	Показывает интерфейс для работы с командами **mh**.
whatnow	Запускает интерфейс для работы с черновиком.
whom	Позволяет работать с адресами **mh**.

Дополнительная информация о командах **mh** приведена в книге *Справочник по командам, том 3*.

Команда bellmail

bellmail - это команда электронной почты UNIX, разработанная фирмой AT&T. Она обрабатывает почту пользователей локальной системы, а также удаленных систем, доступных с помощью Основных сетевых утилит(BNU) (другое название - программа копирования UNIX-UNIX (UUCP)).

Эти программы поддерживают только сети, в которых системы соединены с помощью модема или выделенных двухточечных линий. Команда открывает оболочку, позволяющую выполнять следующие операции:

- Взять данные из стандартного потока ввода (т.е. введенные с клавиатуры или из существующего файла), добавить один или несколько адресов (заданных в качестве параметров команды), системное время, и поместить полученную информацию в почтовые ящики всех перечисленных адресатов (`/var/spool/mail/`*`ИД-пользователя`*).
- Просмотреть сообщения, находящиеся в вашем системном почтовом ящике.
- Добавить почтовые сообщения в личный почтовый ящик (`$HOME/mbox`) или в другой файл.
- Отправить сообщение пользователю другой системы с помощью BNU.
- Автоматически перенаправить всю почту из системного почтового ящика в другую систему, указав опцию *.forward* в начале файла системного почтового ящика.

Для эффективного применения этого обработчика электронной почты требуются опыт работы в качестве пользователя UNIX. Дополнительная информация приведена в описании команды **bellmail** в книге *Справочник по командам, том 1*.

Функции почты

Здесь описаны функции программы **mail**.

С помощью программы работы с почтой (**mail**) можно получать, создавать и оправлять сообщения пользователям локальной или удаленной системы.

Хранение почты

В зависимости от ситуации, возможны различные способы сохранения почты.

Направляемая вам почта сохраняется в специальном системном каталоге. Этот системный каталог содержит файлы для всех пользователей локальной системы. Ваша почта хранится в этом каталоге до тех пор, пока вы ее не обработаете.

Системный почтовый ящик:

Системный почтовый ящик аналогичен обычному почтовому ящику, в который почтальон кладет письма, адресованные владельцу этого ящика.

Системный почтовый ящик, по аналогии с обычным, - это файл, куда помещаются сообщения, адресованные определенному пользователю. Если такого файла не существует, то при получении почты система сама создает его. После удаления сообщений удаляется и сам файл.

Системные почтовые ящики хранятся в каталоге `/var/spool/mail`. Каждому системному почтовому ящику присвоено имя, совпадающее с ИД пользователя, которому принадлежит этот ящик. Например, если ваше ИД пользователя - `karen`, то ваш системный почтовый ящик будет располагаться в файле `/var/spool/mail/karen`

Личный почтовый ящик по умолчанию:

Личный почтовый ящик аналогичен обычной папке для входящей корреспонденции в офисе. Сначала вы помещаете полученную почту в папку для входящих сообщений, а затем передаете ее в архив или уничтожаете.

У каждого пользователя есть личный почтовый ящик. Если при просмотре почты, находящейся в системном почтовом ящике, сообщения не были помечены для удаления или сохранения в файл, то они будут сохранены в личном почтовом ящике, `$HOME/mbox` (`$HOME` - домашний каталог). Если таких сообщений нет, то файл `mbox` не создается.

Файл dead.letter для неполных сообщений:

Если вам пришлось прервать создание сообщения для выполнения других задач, то система сохранит незаконченное сообщение в файле `dead.letter` в каталоге `$HOME`.

Если файл `dead.letter` не существует, то система создает его. В дальнейшем вы сможете отредактировать этот файл и закончить составление сообщения.

> **Внимание:** Не используйте файл `dead.letter` для хранения сообщений. Содержимое файла `dead.letter` постоянно обновляется другими незаконченными сообщениями.

Папки почты:

Папки предназначены для хранения сообщений. С помощью программы работы с электронной почтой можно поместить в указанную папку сообщение из системного почтового ящика, личного почтового ящика или из другой папки.

Папка - это текстовый файл. Каждая папка помещается в каталог, указанный в файле `.mailrc` с помощью опции **set folder**. Перед сохранением сообщения в папке необходимо создать этот каталог. После этого программа работы с электронной почтой будет автоматически создавать в нем папки. Если имя каталога в файле `.mailrc` не указано, то папки будут создаваться в текущем каталоге. См. См. раздел “Упорядочение почты” на стр. 19.

> **Примечание:** Отправлять и получать почту можно с помощью любой из нескольких программ, включая Message Handler (MH) и **bellmail**. Выбор применяемой программы зависит от конфигурации системы. За дополнительной информацией обратитесь к системному администратору.

Обработка и получение почты

Программа **mail** позволяет просмотреть каждое сообщение, находящееся в почтовом ящике, а затем удалить его или сохранить в отдельном почтовом каталоге.

При получении почты командная оболочка направляет вам уведомление. Уведомление отображается перед следующим приглашением в том случае, если задана переменная среды **MAIL** и с момента предыдущей проверки наличия почты прошел интервал времени, указанный в переменной **MAILCHECK**. Текст уведомления о получении новой почты задается с помощью переменной среды **MAILMSG**. В зависимости от применяемой оболочки (bourne, korn или C), будет показано примерно следующее уведомление:

```
У ВАС НОВАЯ ПОЧТА
```

Запуск почтового ящика:

Команда **mail** позволяет прочитать, а затем удалить сообщение из почтового ящика.

Системный почтовый не предназначен для хранения сообщений. Сохраняйте сообщения в личном почтовом ящике и в почтовых каталогах.

Проверка личного почтового ящика:

С помощью команды **mail** можно проверить наличие новых сообщений в личном почтовом ящике.

В командной строке введите команду **mail**:

```
mail
```

Если почтовый ящик пуст, то появится следующее сообщение:

```
Почта для ваш-ИД отсутствует
```

Если в почтовом ящике есть какие-либо сообщения, то появится список этих сообщений:

```
Mail  Для просмотра справки введите ?.
"/usr/mail/lance": 3 сообщений 3 новых
  >N    1 karen вторник 27 апреля 16:10 12/321 "Собрание отдела"
   N       2 lois  вторник 27 апреля 16:50 10/350 "Новости"
   N       3 tom   вторник 27 апреля 17:00 11/356 "Новые программы"
```

Текущее сообщение всегда обозначается символом больше (>). Каждая строка списка содержит следующие поля:

Элемент	Описание
состояние	Указывает класс сообщения.
номер	Указывает номер сообщения, присвоенный ему программой работы с электронной почтой.
отправитель	Указывает адрес отправителя сообщения.
дата	Указывает дату получения сообщения.
размер	Указывает число строк и символов в сообщении (включая заголовок).
тема	Указывает тему сообщения, если она задана.

В качестве состояния может быть указано одно из следующих значений:

Элемент	Описание
N	Новое сообщение.
P	Сообщение, которое будет сохранено в почтовом ящике.
U	Непрочитанное сообщение. Сообщение, которое не было прочитано при последнем обращении к программе работы с электронной почтой.
*	Сообщение, которое было сохранено в файле или папке.

Пустое поле состояния означает, что сообщение было прочитано, но не было сохранено или удалено.

Проверка личного почтового ящика или почтовой папки:

С помощью команды **mail** можно проверить личный почтовый ящик или почтовую папку.

Введите в командной строке команду **mail** одним из следующих способов:

1. Для просомтра списка сообщений в личном почтовом ящике `$HOME/mbox` введите:
   ```
   mail -f
   ```
   Если личный почтовый ящик пуст, то появится примерно следующее сообщение:
   ```
   "/u/george/mbox": 0 сообщений
   ```
   or
   ```
   Файл или каталог не существует
   ```
2. Для просомтра списка сообщений в папке `dept` введите:
   ```
   mail -f +dept
   ```
   Если личная почтовая папка пуста, то появится примерно следующее сообщение:
   ```
   Файл или каталог не существует
   ```

опции отображения содержимого почтового ящика:

Для обработки сообщений, находящихся в почтовом ящике, воспользуйтесь подкомандами программы работы с почтой.

`Предварительные требования`

1. В системе должна быть установлена программа работы с электронной почтой.
2. Программа работы с почтой должна быть запущена.
3. В вашем почтовом ящике должно быть по крайней мере одно сообщение.

Просмотр группы сообщений:

С помощью подкоманды **h** можно просмотреть сообщение, содержащееся в заданном списке сообщений, не просматривая все сообщения.

В приглашении программы работы с почтой введите подкоманду **h**, как в следующих примерах:

Элемент	Описание
h	За один раз отображается примерно 20 сообщений. Точное число сообщений зависит от типа применяемого терминала и значения опции **set screen**, указанной в файле `.mailrc`. Если повторно ввести подкоманду **h**, будет показана та же группа сообщений.
h 21	Будут показаны сообщения с номерами от 21 до 40 включительно (если в почтовом ящике есть сообщения с такими номерами). Вводите подкоманду **h**, указывая в ней следующий номер сообщения, до тех пор, пока не просмотрите все сообщения.
h 1	Для перехода к группе первых 20 сообщений укажите любое число от 1 до 20.

Прокрутка содержимого почтового ящика:

Подкоманда **z** позволяет выполнить прокрутку содержимого почтового ящика.

В приглашении программы работы с почтой введите подкоманду **z**, как в следующих примерах:

Элемент	Описание
z	За один раз отображается примерно 20 сообщений. Точное число сообщений зависит от типа применяемого терминала и значения опции **set screen**, указанной в файле `.mailrc`. Для просмотра следующих 20 сообщений повторно введите подкоманду **z**.
z +	Если вы укажете аргумент знак плюс (**+**), то будут показаны следующие 20 сообщений. Будут показаны сообщения с номерами от 21 до 40 включительно (если в почтовом ящике есть сообщения с такими номерами). Вводите подкоманду **z+** до тех пор, пока не просмотрите все сообщения. Появится следующее сообщение: `На последнем экране сообщений.`
z -	Если вы укажете аргумент знак минус (**-**), то будут показаны предыдущие 20 сообщений. Когда вы достигните первой группы сообщений, появится следующее сообщение: `На первом экране сообщений.`

Фильтрация сообщения на основе определенной информации:

Для того, чтобы фильтровать сообщения по определенной информации, в командной строке почтового ящика можно воспользоваться подкомандой **f**, как показано в следующих примерах:

Элемент	Описание
f	Показывает заголовок текущего сообщения.
f `1 4 7`	Показывает заголовки сообщений 1, 4 и 7.
f `1-10`	Показывает заголовки сообщений с 1 по 10.
f `*`	Показывает все сообщения.

f `ron`	Показывает сообщения, полученные от пользователя `ron`, если они есть. Будут найдены все сообщения, в адресе отправителя которых содержится указанная подстрока (без учета регистра); т.е. адресу `ron`, указанному в верхнем или нижнем регистре, будут соответствовать следующие адреса отправителей:
	`RoN`
	`ron@topdog`
	`hron`
	`rOn`

f`meet`	Показывает все сообщения, у которых в поле **Subject:** содержится подстрока `meet`. Указанная в команде символьная строка рассматривается в качестве шаблона для поиска в поле **Subject:**. Поиск подстроки в поле **Subject:** выполняется без учета регистра; т.е. шаблону `meet` будут соответствовать, например, следующие поля Subject:
	`Meeting on Thursday`
	`Come to meeting tomorrow`
	`MEET ME IN ST. LOUIS`

Номер текущего сообщения:

Подкоманда **=** отображает номер текущего сообщения.

В приглашении почтового ящика введите подкоманду **=**, как в следующем примере:

Элемент	Описание
=	Будет показан номер текущего сообщения.

Общее число сообщений в почтовом ящике:

Для проверки общего числа сообщений в почтовом ящике используйте подкоманду **folder**.

В приглашении почтового ящика введите подкоманду **folder**, как показано в следующем примере:

Элемент	Описание
folder	Показывает информацию о папке или почтовом ящике. Будет показана примерно следующая информация: `"/u/lance/mbox": 29 сообщений`

Опции чтения почты:

Почту можно прочитать несколькими способами. Здесь приведены примеры для каждого метода.

Выберите наиболее удобный для себя способ чтения почты. Прежде чем читать сообщение, убедитесь в соблюдении следующих условий:

1. В системе должна быть установлена программа работы с электронной почтой.
2. Программа работы с почтой должна быть запущена.
3. В почтовом ящике системы должно быть по крайней мере одно сообщение.

Чтение сообщений в почтовом ящике:

Для чтения сообщений, поступивших в почтовый ящик, введите подкоманду **t** или **p**.

В командной строке почтового ящика можно воспользоваться подкомандой **t** или **p**, как показано в следующих примерах:

Элемент	Описание
`3`	Если вы укажете номер сообщения, то по умолчанию будет показан текст этого сообщения.
t	Если вы укажете подкоманду **t**, то по умолчанию будет показан текст текущего сообщения.
t `3`	Будет показан текст сообщения 3.
t `2 4 9`	Будет показан текст сообщений 2, 4 и 9.
t `2-4`	Будет показан текст сообщений с номерами от 2 до 4.
t	Если вы укажете подкоманду **p**, то по умолчанию будет показан текст текущего сообщения.
p `3`	Будет показан текст сообщения 3.
p `2 4 9`	Будет показан текст сообщений 2, 4 и 9.
p `2-4`	Будет показан текст сообщений с номерами от 2 до 4.

Чтение следующего сообщения в почтовом ящике:

Для чтения следующего сообщения в почтовом ящике используется подкоманда **n**.

В приглашении программы работы с почтой введите подкоманду (**n**)ext или знак плюс (+), как показано в следующем примере:

Элемент	Описание
n или +	Будет показан текст следующего сообщения, которое при этом станет текущим.

Кроме того, для просмотра следующего сообщения можно нажать клавишу Enter.

Чтение предыдущего сообщения в почтовом ящике:

Для чтения предыдущего сообщения используется подкоманда **-**.

В приглашении программы работы с почтой введите подкоманду **-**, как в следующем примере:

Элемент	Описание
-	Будет показано предыдущее сообщение.

Удаление почты:

Вы можете удалить текущее сообщение, сообщение с заданным номером, а также группу сообщений.

Указав комбинацию подкоманд, можно удалить текущее сообщение и перейти к просмотру следующего сообщения. Убедитесь, что следующие условия выполнены:

1. В системе должна быть установлена программа работы с электронной почтой.
2. В почтовом ящике системы должно быть по крайней мере одно сообщение.
3. Программа работы с почтой должна быть запущена.

Удаление сообщений:

Для удаления сообщений используются различные виды подкоманды **d**.

В приглашении программы работы с почтой введите подкоманду (**d**)elete, как в следующих примерах:

Элемент	Описание
d	Удаляет текущее сообщение.
dp или **dt**	Удаляет текущее сообщение и показывает следующее. Также для этого можно добавить опцию **set autoprint** в файл `.mailrc`, после чего подкоманда **d** будет работать как сочетание команд **dp** или **dt**.
d `4`	Удаляет сообщение 4.
d `4-6`	Удаляет группу сообщений с 4 по 6.
d `2 6 8`	Удаляет сообщения 2, 6 и 8.

Восстановление сообщений:

Эти атрибуты можно изменить с помощью команды **u**.

В командной строке почтового ящика можно воспользоваться подкомандой **u**, как показано в следующих примерах:

Элемент	Описание
u	Восстанавливает текущее сообщение.
u `4`	Восстанавливает сообщение 4.
u `4-6`	Восстанавливает сообщения с номерами 4, 5 и 6.
u `2 6 8`	Восстанавливает сообщения 2, 6 и 8.

Выход из программы просмотра почты:

Перед выходом из программы просмотра почты убедитесь в выполнении следующих условий.

1. В системе должна быть установлена программа работы с электронной почтой.
2. В почтовом ящике системы должно быть по крайней мере одно сообщение.
3. Программа работы с почтой должна быть запущена.

Выход с сохранением изменений:

Для выхода из программы почты и сохранения изменений используйте подкоманду **q**.

При выходе из системного почтового ящика:

Элемент	Описание
q	Подкоманда **q** завершает работу с системным почтовым ящиком и показывает приглашение операционной системы. После этого все сообщения, выбранные для удаления, действительно удаляются из почтового ящика без возможности восстановления. Прочитанные сообщения будут сохранены в личном почтовом ящике (**mbox**). Если ни одно сообщение не было прочитано, то сообщения остаются в почтовом ящике системы до тех пор, пока они не будут обработаны.

При выходе из личного почтового ящика или почтовой папки:

Элемент	Описание
q	Подкоманда **q** оставляет прочитанные и непрочитанные сообщения в личном почтовом ящике или почтовой папке до тех пор, пока они не будут обработаны.

Выход без сохранения изменений:

Для выхода из программы почты без сохранения изменений используйте подкоманду **x** или **ex**.

Элемент	Описание
x или **ex**	Подкоманды **x** и **ex** позволяют завершить работу с почтовым ящиком и вернуться в операционную систему без изменения списка сообщений. Программа игнорирует все изменения, внесенные перед вводом подкоманды **x**, однако если вы сохранили сообщение в другой папке, то сохранение будет выполнено.

Упорядочение почты:

Для структуризации почты рекомендуется сохранять сообщения в отдельных папках.

Можно создать неограниченное число папок. Присвойте каждой папке имя, отражающее тему сообщений, которые будут в ней храниться. В этом случае имена задаются по тем же правилам, что и имена папок в обычном офисном архиве. Каждая папка является текстовым файлом и помещается в каталог, указанный в файле `.mailrc` с помощью опции **set folder**. Перед сохранением сообщения в папке необходимо создать этот каталог. После этого программа работы с электронной почтой будет автоматически создавать в нем папки. Если в опции **set folder** файла `.mailrc` каталог не задан, то папки будут создаваться в текущем каталоге. С помощью программы работы с электронной почтой можно поместить в указанную папку сообщение из системного почтового ящика, личного почтового ящика или из другой папки.

Подкоманды **s** и **w** предназначены для добавления сообщения к файлу или папке. Обе команды добавляют содержимое сообщения к уже существующему файлу или создают новый файл, если указанный файл не существует. При этом прежняя информация файла сохраняется. При сохранения сообщения из системного почтового ящика в файле или папке, оно заносится в указанную папку или файл, а затем удаляется из почтового ящика. При сохранении сообщения из личного почтового ящика или папки в другом файле или папке, оно остается в личном почтовом ящике и копируется в указанный файл или папку. Подкоманда **s** помещает сообщение в конец папки вместе с заголовком, поэтому вы можете работать с папкой точно также, как с почтовым ящиком. Подкоманда **w** добавляет к файлу сообщение без заголовка, поэтому вы можете работать с папкой так же, как с обычным файлом.

Перед упорядочиванием почты убедитесь, что следующие предварительные требования выполнены:

1. В системе должна быть установлена программа работы с электронной почтой.
2. В системном почтовом ящике, в личном почтовом ящике или в папке должно быть по крайней мере одно сообщение.
3. Программа работы с почтой должна быть запущена.

Создание каталога для хранения почты:

Сообщения можно сохранить в папке каталога почтового ящика с помощью подкоманды **set folder**.

Для сохранения сообщений в папках воспользуйтесь следующей процедурой:

1. Для того чтобы определить, указана ли опция **set folder** в файле `.mailrc`, введите следующую подкоманду в приглашении программы работы с почтой:

   ```
   set
   ```

   Подкоманда **set** предназначена для просмотра списка опций, указанных в файле `.mailrc`.

   Если опция **set folder** указана, то появится примерно следующее сообщение:

   ```
   folder /home/george/letters
   ```

   В данном примере `letters` - это каталог, в котором будут храниться почтовые папки.
2. Если опция **set folder** не указана, то добавьте в файл `.mailrc` строчку, аналогичную следующей:

   ```
   set folder=/home/george/letters
   ```

   В этом примере значение `/home/george` задает домашний каталог пользователя george, а `letters` - каталог, в котором будут храниться почтовые папки. Опция **set folder** позволяет сохранять сообщения в каталоге `letters` с помощью знака плюса.
3. Создайте каталог `letters` в своем домашнем каталоге. Перейдите в домашний каталог и введите в командной строке:

   ```
   mkdir letters
   ```

а охранение сообщений с заголовками:

Для сохранения сообщений с заголовками применяется подкоманда **s**.

Существуют следующие способы применения подкоманды **s**:

Элемент	Описание
s `1-4 notes`	Сохраняет сообщения 1, 2, 3 и 4 вместе с их заголовками в папке `notes` из текущего каталога. Программа работы с электронной почтой отправит следующее сообщение: `"notes" [добавлено] 62/1610`
s +`admin`	Сохраняет текущее сообщение в папке `admin` из почтового каталога. Если в файле `.mailrc` в качестве почтового каталога был задан каталог `/home/george/letters`, появится следующее сообщение: `"/home/george/letters/admin" [добавлено] 14/321`
s `6 +admin`	Сохраняет сообщение `6` в папке `admin` почтового каталога. Если в файле `.mailrc` в качестве почтового каталога был задан каталог `/home/george/letters`, появится следующее сообщение: `"/home/george/letters/admin" [добавлено] 14/321`

Сохранение сообщений без заголовков:

Подкоманда **w** позволяет сохранить сообщение как файл, а не как папку.

Для просмотра или редактирования файла, сохраненного с помощью подкоманды **w**, можно воспользоваться редактором **vi** или любым другим текстовым редактором. В приглашении почтового ящика введите подкоманду **w**, как показано в следующих примерах:

Элемент	Описание
w 6 pass	Сохраняет только текст сообщения 6 в файле pass текущего каталога. Если файл pass не существует, то появится следующее сообщение: "pass" [Новый файл] 12/30 Если файл pass существует, то появится следующее сообщение: "pass" [Добавлено] 12/30
w 1-3 safety	Сохраняет только текст сообщений 1, 2 и 3 в файле safety текущего каталога. Указанные в примере сообщения будут последовательно добавлены к заданному файлу. Если файл safety не существует, то появится следующее сообщение: "safety" [новый файл] 12/30

Определение текущего почтового ящика или папки:

С помощью команды **folder** можно определить текущий почтовый ящик или папку.

При запуске команды **mail** появляется имя текущего почтового ящика. Однако если в дальнейшем вы будете переходить от одного почтового ящика к другому, то вы можете забыть имя текущего почтового ящика. В приглашении почтового ящика введите подкоманду **folder**, как показано в следующем примере:

Элемент	Описание
folder	Эта подкоманда предназначена для определения имени текущего почтового ящика или папки. Если имя текущего почтового ящика - /home/lance/mbox, то появится следующее сообщение: /home/lance/mbox: 2 сообщения 1 удалено Оно говорит о том, что имя текущего почтового ящика - /home/lance/mbox, и в нем содержится два сообщения, одно из которых выбрано для удаления.

Переход к другому почтовому каталогу:

Переход к другому почтовому ящику аналогичен завершению работы с почтовым ящиком или папкой.

При завершении работы с данным почтовым ящиком все сообщения, помеченные для удаления, будут удалены. После этого удаленные сообщения нельзя будет восстановить. В приглашении программы работы с почтой введите подкоманду **file** или **folder**, как в следующем примере:

Элемент	Описание
folder +project	Подкоманды **file** и **folder** предназначены для перехода от почтового ящика, из которого была запущена программа работы с электронной почтой, к другому почтовому ящику. Если при переходе от файла mbox к папке mbox из файла mbox будут удалены все сообщения, то программа работы с электронной почтой отправит следующее сообщение: /home/dee/mbox удален +project: 2 сообщения 2 новых После этой строки будет показан список сообщений из папки project.

Создание и отправка почты

Программа для работы с электронной почтой (**mail**) позволяет создавать и отправлять сообщения, отвечать на полученные сообщения, пересылать их другим пользователям, а также отправлять другим пользователям файлы ASCII.

Файл ASCII может содержать документ, написанный с помощью текстового редактора, или исходный текст программы.

Электронные сообщения и файлы можно отправить пользователям локальной системы, пользователям системы, находящейся в вашей сети, или пользователям удаленной сети. Для получения почты адресату не обязательно работать в системе в момент отправки информации. Почта отправляется по адресу пользователя.

Отправка почты:

Почта отправляется по адресу пользователя. Адрес включает имена пользователя и системы, необходимые для доставки сообщения получателю.

Для отправки сообщения другому пользователю обычно нужно ввести команду **mail** и указать адрес получателя следующим образом:

```
mail Пользователь@Адрес
```

Формат параметра *Адрес* зависит от расположения получателя. Общие принципы указания адреса такие же, как и при передаче обычной записки коллеге в офисе. Для того чтобы отправить записку сотруднику Ryan из небольшого отдела, в котором работает лишь несколько человек, вы можете просто написать фамилию на конверте и положить его в ящик для корреспонденции. Однако если этот сотрудник работает в другом отделе, то вы должны указать на конверте более подробную информацию:

```
Ryan
Payroll
```

Если же Ryan работает в другом филиале, то для гарантированной доставки сообщения вам придется указать еще больше сведений:

```
Ryan
Payroll
Gaithersburg
```

При отправке электронной почты применяются те же правила адресации:

Элемент	Описание
mail `ryan`	Для отправки электронного сообщения пользователю локальной системы нужно указать только имя этого пользователя.
mail `ryan@tybalt`	Для отправки электронного сообщения пользователю локальной сети укажите полный адрес системы (узла).
mail `ryan@mars.aus.dbm.com`	Для отправки электронного сообщения пользователю другой сети укажите полный адрес системы и адрес сети.
mail `dept71`	С помощью списка псевдонимов или списка рассылки можно отправить сообщение определенной группе пользователей. Для этого нужно создать в файле `.mailrc` список псевдонимов или список рассылки. Информация о создании псевдонимов приведена в разделе “Список псевдонимов и рассылки” на стр. 38.

Отправка почты нескольким пользователям:

Для того чтобы отправить почту нескольким пользователям одновременно, перечислите имена этих пользователей через пробелы.

Например:

```
ryan@tybalt suemc@julius dmorgan@ophelia
```

Отправка почты пользователям локальной системы:

Для отправки электронного сообщения пользователю локальной системы (то есть пользователю, чье имя указано в файле `/etc/passwd`) укажите в качестве адреса имя этого пользователя.

Введите команду **mail** в следующем формате:

```
mail имя-пользователя
```

Элемент	Описание
mail `ryan`	Если в системе есть пользователь Ryan с ИД пользователя `ryan`, то эта команда запустит программу работы с электронной почтой, предложить создать сообщение и попытается отправить его локальному пользователю с именем `ryan`. При успешной доставке сообщения никакого уведомления показано не будет. Если пользователь Ryan отсутствует в локальной системе, то программа работы с электронной почтой немедленно поместит в ваш почтовый ящик сообщение об ошибке.

Отправка почты пользователям сети:

Для отправки сообщений пользователям сети используйте команду **mail**. В качестве адреса укажите имя пользователя и имя системы.

Для отправки сообщения другому пользователю локальной сети введите следующую команду:

Элемент	Описание
mail *имя-пользователя@имя-системы*	Например, если Ryan - это пользователь системы `zeus`, то для создания и отправки ему сообщения введите команду: `mail ryan@zeus` Эта команда запустит программу работы с электронной почтой, предложит создать сообщение и попытается отправить его пользователю `ryan` в системе `zeus`. При успешной доставке сообщения никакого уведомления показано не будет и на экране появится командная строка терминала. Если был задан неверный электронный адрес, то будет показано сообщение об ошибке.

Примечание: Для отправки сообщения пользователю другой системы локальной сети нужно знать имя этой системы и имя пользователя. Дополнительные сведения о просмотре сведений, идентифицирующих пользователей, приведены в “Команды средств связи системы” на стр. 8.

Отправка почты пользователям другой сети:

Если ваша сеть соединена с другими сетями, то вы можете отправлять почтовые сообщения пользователям этих сетей.

В этом случае адрес задается в зависимости от способа соединения сетей и принятых в этих сетях способов адресации. В зависимости от сетевой конфигурации, выполните одно из следующих действий:

- Если используется центральная база данных имен и адресов, воспользуйтесь командой **mail**, как пока зано в следующем примере:

  mail *имя-пользователя@имя-системы*

  Если в сети применяется централизованная база данных имен, то для отправки электронных сообщений пользователям других сетей дополнительную информацию указывать не нужно. В этом случае применяется тот же формат адреса, что и для адресов пользователей локальной сети.

  Такой тип адресации применяется в сетях, структура которых позволяет поддерживать централизованную базу данных имен.

- Если в вашей сети применяется адресация имен доменов, используйте команду **mail** как показанов следующем примере:

  mail *имя-пользователя@имя-системы.имя-домена*

  Для больших сетей, а также для физически распределенных сетей различных типов невозможно создать централизованную базу данных имен. Параметр *имя-домена* определяет положение удаленной сети относительно локальной сети в общей структуре сетей, связанных друг с другом.

  Например, при вводе команды

  mail `kelly@merlin.odin.valryanl`

  ваша почта будет отправлена пользователю `kelly` в системе `merlin`, находящейся в локальной сети `odin`, которая подключена к другой сети домена `valryanl`.

Адресация почты по соединению BNU или UUCP:

Можно отправлять сообщения пользователям других систем с помощью Основных сетевых утилит (BNU) или программы копирования UNIX-UNIX (UUCP).

Для отправки сообщения пользователю другой системы, подключенной к вашему компьютеру с (BNU) или другой версии (UUCP), необходимо знать следующую информацию:

- Имя получателя
- Имя системы получателя
- Физический маршрут к системе получателя

Информацию о маршруте к системе получателя можно получить у администратора, отвечающего за подключение вашей системы к другим системам сети.

Если ваш компьютер подключен к BNU или UUCP: В командной строке системы вызовите команду **mail**, как показано в следующих примерах:

Элемент	Описание
mail *маршрут-UUCP*!*имя-пользователя*	Такой формат можно применять в том случае, если в локальной системе установлено соединение BNU или UUCP с удаленными системами. Параметр *имя-пользователя* - это имя получателя в удаленной системе. Параметр *маршрут-UUCP* описывает физический маршрут, по которому сообщение должно передаваться в сети UUCP. Если ваша система непосредственно подключена к удаленной системе (без промежуточных систем UUCP), то в этом параметре нужно указать имя удаленной системы.
mailarthur!lancelot!merlin!ken	Если сообщение должно быть передано через несколько промежуточных систем UUCP, то в этом параметре нужно указать список промежуточных систем. Список начинается с ближайшей системы и заканчивается наиболее удаленной системой, при этом имена систем отделяются друг от друга восклицательными знаками (!). В приведенном выше примере сообщение будет передано в систему `merlin` через системы `arthur` и `lancelot` (в указанном порядке).
mail merlin!ken	Если в локальной системе установлено соединение UUCP с системой `merlin` (без промежуточных узлов), то можно отправить сообщение пользователю `ken` в этой системе.

Если соединение BNU или UUCP установлено в другой системе: если ваша система подключена к локальной или глобальной сети, то в некоторых системах этой сети может быть установлено соединение BNU или другое соединение UUCP с удаленной системой. В этом случае по такому соединению можно отправлять сообщения пользователям удаленной системы UUCP. Из командной строки системы вызовите команду **mail**, как показано в следующих примерах:

mail @arthur:merlin!ken
: Эта команда отправляет почту пользователю `ken` в системе UUCP `merlin` с хоста Internet `arthur`. Ограничитель @ применяется в адресах Internet, ограничитель ! - в адресах UUCP, а ограничитель : объединяет два адреса. Обратите внимание, что в этом случае сообщение не предназначено каким-либо пользователям промежуточных систем, поэтому в адресе домена перед ограничителем @ имя пользователя не указано.

mail@arthur:odin!acct.dept!kelly
: Эта команда отправляет почту пользователю `kelly` в системе UUCP `acct.dept` через промежуточную систему `odin` с хоста Internet `arthur`.

mail@odin.uucp:@dept1.UUCP:@dept2:bill@dept3
: Эта команда отправляет почту пользователю `bill@dept3` через промежуточные системы UUCP `odin` и `dept1`, а затем - через промежуточные системы локальной сети `dept2` и `dept3`. Для применения такой формы записи адресов UUCP необходимо соответствующим образом настроить файл `/etc/sendmail.cf`. За дополнительной информацией обратитесь к системному администратору.

Если вы часто отправляете почту пользователям других сетей, то можно создать псевдонимы, содержащие адреса этих пользователей. См. См. раздел “Список псевдонимов и рассылки” на стр. 38.

Запуск редактора электронной почты:

Для создания сообщений в программе работы с электронной почтой **mail** предусмотрен строковый редактор.

1. В системе должна быть установлена программа работы с электронной почтой.
2. Программа работы с почтой должна быть запущена.

В этом редакторе ввод каждой строки сообщения следует завершать нажатием клавиши Enter, после чего будет показана новая строка для ввода текста. После нажатия клавиши Enter в строку нельзя вносить изменения. Однако до нажатия клавиши Enter введенную в строке информацию можно изменять с помощью клавиш Backspace и Delete. Кроме того, с помощью подкоманд редактора можно перейти к редактированию сообщения в полноэкранном редакторе.

Если сообщение создается с помощью редактора электронной почты, то поля **date:** и **from:** заполняются автоматически. Вы можете указать значения в полях **subject:** и **cc:**. Эти поля аналогичны тексту обычного делового письма.

В редакторе электронной почты предусмотрен ряд подкоманд, выполняющих различные операции над сообщением. Каждая команда вводится с новой строки и начинается со специального *escape-символа*. По умолчанию роль escape-символа играет тильда (~). Вы можете заменить его на любой другой символ, добавив опцию **set escape** в файл `.mailrc`.

Из командной строки системы или из командной строки почтового ящика можно вызвать подкоманду **mail**, как показано в следующих примерах:

Элемент	Описание
mail *Пользователь@Адрес*	Эту команду следует вводить в командной строке. Сообщение будет отправлено по адресу *Пользователь@Адрес*. Формат параметра *Адрес* зависит от расположения получателя.
m*Пользователь@Адрес*	Эту команду следует указывать в приглашении программы работы с почтой. Сообщение будет отправлено по адресу *Пользователь@Адрес*. Формат параметра *Адрес* зависит от расположения получателя.

Текстовый редактор электронной почты также запускается при вводе подкоманд **R** или **r** для составления ответа на сообщение. Дополнительная информация о создании о отправке ответов на сообщения приведена в разделах “Отправка почты” на стр. 30 и “Ответ на почту” на стр. 31.

Редактирование сообщений:

Вы можете добавить информацию к существующему сообщению, введя в приглашении программы работы с почтой подкоманду (**e**)dit или (**v**)isual.

В редакторе электронной почты информацию, указанную в какой-либо строке, нельзя изменять после нажатия клавиши Enter и перехода к следующей строке. Если необходимо изменить сообщение перед его отправкой, сделайте это с помощью другого редактора.

Прежде чем изменять сообщение в другом редакторе, убедитесь в соблюдении следующих условий:

1. В системе должна быть установлена программа работы с электронной почтой.
2. В файле `.mailrc` должен быть определен альтернативный редактор. Это можно сделать с помощью директивы

   `set EDITOR=`*`путь`*

Она задает редактор, активируемый подкомандой **~e**. Параметр *полное-имя* задает полное имя программы альтернативного редактора. Например, если указать параметр `set EDITOR=/usr/bin/vi`, то при выполнении подкоманды **~e** будет запускаться редактор **vi**.

3. Для того чтобы добавить информацию к сообщению из своего почтового ящика, необходимо запустить команду **mail** для просмотра сообщения из системного почтового ящика, почтового ящика другого пользователя или из папки.
4. Для создания сообщения с помощью альтернативного редактора необходимо перейти к приглашению редактора электронной почты.

Добавление информации к определенному сообщению почтового ящика:

Для того, чтобы добавить информацию в сообщение почтового ящика, введите подкоманду **e** или **v**, а затем укажите номер сообщения.

В командной строке почтового ящика можно воспользоваться подкомандой **e** или **v**, как показано в следующих примерах:

Элемент	Описание
e 13	Эта команда позволит добавить информацию к сообщению номер 13 с помощью редактора **e** или другого редактора, определенного в файле `.mailrc`.
v 15	Эта команда позволит добавить информацию к сообщению номер 15 с помощью редактора **vi** или другого редактора, определенного в файле `.mailrc`.

Если вы не укажете номер сообщения, то команда **mail** запустит редактор для работы с текстом текущего сообщения. После завершения работы редактора вновь будет показано приглашение программы работы с почтой и вы сможете продолжить обработку сообщений почтового ящика.

Изменение текущего сообщения в редакторе электронной почты:

Работая с редактором электронной почты, в начале строки можно ввести подкоманду **~e** или **~v**, как показано в следующих примерах.

Элемент	Описание
~e	Запускает редактор **e** или другой редактор, определенный в файле `.mailrc`.
~v	Запускает редактор **vi** или другой редактор, определенный в файле `.mailrc`.

Эти команды позволяют изменять текст текущего сообщения. После завершения работы указанного редактора вновь будет показано приглашение программы работы с почтой.

Просмотр сообщения в редакторе электронной почты:

С помощью подкоманды **~p** можно просмотреть сообщение в редакторе электронной почты.

1. В системе должна быть установлена программа работы с электронной почтой.
2. Для просмотра сообщения с помощью редактора электронной почты этот редактор должен быть запущен. Дополнительная информация приведена в разделе “Запуск редактора электронной почты” на стр. 25.

Работая с редактором электронной почты, в начале строки введите подкоманду**~p**, как показано в следующем примере:

Элемент	Описание
~p	Будет показан заголовок и текст сообщения. Отображаемый текст прокручивается по мере вывода на экран. После текста сообщения будет показано приглашение редактора (продолжение).

Если сообщение не помещается в окне целиком и для терминала не задан размер страницы (это можно сделать с помощью команды **stty**), то по мере прокрутки текста первые строки сообщения будут исчезать за верхней границей экрана. Для просмотра больших сообщений запустите другой редактор с помощью соответствующих подкоманд. Дополнительная информация приведена в разделе “Редактирование сообщений” на стр. 25.

Выход из редактора электронной почты:

Для выхода из редактора электронной почты без отправки сообщения введите команду **~q** или нажмите комбинацию клавиш прерывания (обычно это Alt-Pause или Ctrl-C).

1. В системе должна быть установлена программа работы с электронной почтой.
2. Для просмотра сообщения с помощью редактора электронной почты этот редактор должен быть запущен. Дополнительная информация приведена в разделе “Запуск редактора электронной почты” на стр. 25.

Если был введен какой-либо текст, то команда **mail** сохранит его в файле dead.letter.

Работая с редактором электронной почты, в начале строки можно ввести подкоманду**~q**, как показано в следующем примере:

Элемент	Описание
~q	Завершает работу редактора электронной почты без отправки сообщения. Если вы ввели какой-либо текст, то он будет сохранен в файле dead.letter в вашем домашнем каталоге. В окне терминала будет показано системное приглашение.
Ctrl-C	Вы можете завершить работу редактора, нажав клавишу прерывания (Ctrl-C или Alt-Pause). Появится следующее сообщение: (Прерывание -- для уничтожения письма нажмите еще раз) Нажмите клавишу прерывания еще раз. (Последнее прерывание -- письмо сохранено в dead.letter) Сообщение не отправляется. Если вы ввели какой-либо текст, то он будет сохранен в файле dead.letter в вашем домашнем каталоге. В окне терминала будет показано системное приглашение.

> **Примечание:** При выходе из редактора электронной почты без отправки сообщения содержимое файла dead.letter заменяется на текущее неотправленное сообщение. Инструкции по просмотру этого файла приведены в разделе “Опции добавления к сообщению файла и или другого сообщения”.

Опции добавления к сообщению файла и или другого сообщения:

Перед добавление файла или определенного сообщения к почтовому сообщению следует выполнить некоторые требования.

Предварительные требования

1. В системе должна быть установлена программа работы с электронной почтой.
2. Вам должны быть известны имя и адрес получателя сообщения.
3. Должен быть запущен редактор электронной почты.

Добавление файлов к сообщению:

С помощью подкоманды **~r** можно добавить файлы в сообщение.

Работая с редактором электронной почты, в начале строки можно ввести подкоманду **~r**, как показано в следующем примере:

Элемент	Описание
~r `schedule`	Где `schedule` - это имя добавляемого файла. В данном примере в конец текущего сообщения будет вставлено содержимое файла `schedule`.

Добавление в сообщение определенного собщения:

С помощью подкоманды **~f** или **~m** можно добавить в электронное сообщение определенное сообщение.

Работая с редактором электронной почты, в начале новой строки можно ввести подкоманду **~f** или **~m**, как показано в следующих примерах:

Элемент	Описание
~f *`список-сообщений`*	Эта команда добавит одно или несколько указанных сообщений в конец текущего сообщения, но *не* выделит добавленные сообщения отступом. Кроме того, эта команда применяется для добавления ссылок на сообщения, которые из-за слишком широких полей не могут быть вставлены с помощью команды **~m**. **Примечание:** Значением параметра *список-сообщений* должен быть список целых чисел, соответствующих номерам существующих сообщений, находящихся в текущем почтовом ящике или папке. Можно также указывать диапазоны номеров. Например: **~f 1-4** В конец текущего сообщения будут добавлены сообщения 1, 2, 3 и 4. Эти сообщения будут выровнены по левому краю (без отступа).
~m `2`	В конец текущего сообщения будет вставлено сообщение с указанным номером. Сообщение будет вставлено с отступом, равным одному шагу табуляции. В данном примере к текущему сообщению добавляется сообщение 2.
~m `1 3`	В конец текущего сообщения добавляются сообщения 1 и 3 с отступом, равным одному шагу табуляции.

Добавление к текущему сообщению содержимого файла dead.letter:

С помощью подкоманды **~d** можно добавить содержимое `dead.letter` в сообщение.

Работая с редактором электронной почты, в начале новой строки можно ввести подкоманду **~d**, как показано в следующем примере:

Элемент	Описание
~d	Эта команда восстанавливает содержимое файла `dead.letter` и добавляет его в конец текущего сообщения. Затем введите в приглашении `(продолжение)` команду добавления следующего сообщения или отправки составленного сообщения.

Изменение заголовка:

Заголовок сообщения содержит информацию о маршрутизации и тему сообщения. Необходимо указать по крайней мере одного получателя сообщения.

1. В системе должна быть установлена программа работы с электронной почтой.
2. Запустите текстовый редактор электронной почты и приступайте к редактированию сообщения. Дополнительная информация приведена в разделе Начало работы с редактором электронной почты.

Другие поля заголовка являются необязательными. В заголовке может быть указана следующая информация:

Элемент	Описание
To:	Адрес или адреса получателей сообщения.
Subject:	Тема сообщения.
Cc:	Один или несколько адресов, по которым нужно отправить копию сообщения. Содержимое этих полей отправляется всем получателям вместе с текстом сообщения.
Bcc:	Один или несколько адресов, по которым нужно отправить *секретную* копию сообщения. Содержимое этого поля *не* передается другим получателям сообщения.

Программу работы с электронной почтой можно настроить таким образом, чтобы она запрашивала или не запрашивала значения этих полей. Для этого необходимо разместить соответствующие записи в файле `.mailrc`. Дополнительная информация приведена в разделе “Опции настройки почтовой программы” на стр. 35.

Установка и сброс поля Subject::

С помощью подкоманды **~s** можно указать в поле **Subject:** конкретную фразу или предложение.

Эта подкоманда позволяет указать в поле **Subject:** отдельное словосочетание или предложение. При этом содержимое поля **Subject:** (если поле было заполнено) заменяется на новое. Работая с редактором электронной почты, в начале новой строки можно ввести подкоманду **~s**, как показано в следующем примере:

Элемент	Описание

~s `Рыбалка на озере`	В результате текущее содержимое поля **Subject:**:
	`Subject: Отпуск`
	
	Изменится на:
	`Subject: Рыбалка на озере`
	Примечание: Данная подкоманда не позволяет добавить данные в поле **Subject:** . Изменить эту информацию можно с помощью подкоманды **~h**, как это описано в разделе “Изменение заголовка” на стр. 28.

Добавление адресов в поля To:, Cc: и Bcc::

Для добавления пользователей к полям заголовков используйте команды **~t**, **~c** или **~b**.

Работая с редактором электронной почты, в начале новой строки можно ввести подкоманду **~t**, **~c** или **~b**, как показано в следующих примерах:

Элемент	Описание

~t `geo@austin mel@gtwn`	Текущий список адресов, указанных в поле **To:**
	`To: mark@austin`
	
	будет изменен на следующий:
	`To: mark@austin geo@austin mel@gtwn`

~c `geo@austin mel@gtwn`	Текущий список адресов, указанных в поле **Cc:**
	`Cc: mark@austin amy`
	
	будет изменен на следующий:
	`Cc: mark@austin amy geo@austin mel@gtwn`

~b `geo@austin mel@gtwn`	Текущий список адресов, указанных в поле **Bcc:**
	`Bcc: mark@austin`
	
	будет изменен на следующий:
	`Bcc: mark@austin geo@austin mel@gtwn`

Примечание: Подкоманды **~t**, **~c** и **~b** не предназначены для изменения или удаления содержимого полей **To:**, **Cc:** и **Bcc:**. Изменить эту информацию можно с помощью подкоманды **~h**, как это описано в разделе “Изменение заголовка” на стр. 28.

Изменение формата сообщения в редакторе электронной почты:

Перед отправкой созданного сообщения можно улучшить его внешний вид, отформатировав сообщение с помощью команды **fmt**.

Прежде чем переформатировать сообщение, убедитесь в соблюдении следующих условий:

1. В системе должна быть установлена программа работы с электронной почтой.
2. В системе должна быть установлена команда **fmt**.

Работая с редактором электронной почты, в начале новой строки можно ввести команду **fmt**, как показано в следующем примере:

Элемент	Описание
~| fmt	В данном примере команда форматирует каждый абзац сообщения в соответствии с заданными размерами полей (абзацы должны отделяться друг от друга пустой строкой). Подкоманда конвейера **|** помещает сообщение в стандартный поток ввода команды, а затем заменяет сообщение выводом команды.

Внимание: Не применяйте команду **fmt** для форматирования сообщений, содержащих вложенные сообщения или отформатированную информацию из внешних файлов. Это ограничение связано с тем, что команда **fmt** изменяет формат заголовка вложенного сообщения и формат предварительно отформатированной информации. Для форматирования таких сообщений воспользуйтесь полноэкранным редактором, который можно вызвать подкомандой **~e** или **~v**.

Проверка орфографии в редакторе электронной почты:

Команда **spell** позволяет поверить правописание в вашем сообщение.

Прежде чем проверять правописание в сообщении, убедитесь в соблюдении следующих условий:

1. В системе должна быть установлена программа работы с электронной почтой.
2. В системе должны быть установлены программы форматирования текста.

Для проверки правописания в сообщении введите в редакторе электронной почты команду **spell**:

1. Сохраните сообщение во временном файле. Например, для записи сообщения в файл `checkit` нужно ввести такую команду:

   ```
   ~w checkit
   ```

2. Запустите команду **spell**, указав в ней в качестве параметра имя временного файла. Введите:

   ```
   ~! spell checkit
   ```

   В данном примере подкоманда восклицательный знак (!) запускает оболочку, выполняет указанную команду и вновь показывает приглашение команды работы с почтой. Результат работы команды **spell** - список неизвестных слов, за которыми следует восклицательный знак (!) , означающий возврат к программе работы с электронной почтой.
3. Просмотрите список слов. Исправьте ошибки с помощью редактора.
4. Удалите временный файл с помощью команды:

   ```
   ~! rm checkit
   ```

Отправка почты:

С помощью этой процедуры можно отправлять сообщения.

- В системе должна быть установлена программа работы с электронной почтой.

- Вам должны быть известны имя и адрес получателя сообщения.

1. В командной строке введите команду **mail**, а затем укажите имена и адреса получателей сообщения. Например:

   ```
   >mail jan@brown
   ```

   Появится строка:

   ```
   Subject:
   ```

2. Укажите тему сообщения. Например:

   ```
   Subject: Собрание отдела
   ```

   и нажмите Enter. Теперь вы можете задать текст сообщения.

3. Введите сообщение. Например:

   ```
   Сегодня в семь часов вечера состоится собрание отдела.
   Присутствие всех сотрудников обязательно.
   ```

4. Для отправки сообщения, созданного в редакторе электронной почты, введите символ конца текста.Обычно это сочетание клавиш Ctrl-D или точка.

   При этом появится содержимое поля **Cc:**:

   ```
   Cc:
   ```

5. Введите имена и адреса пользователей, которым нужно отправить копии сообщения. Например:

   ```
   Cc:  karen@hobo cliff@cross
   ```

   Примечание: Если копии сообщения отправлять не нужно, то оставьте поле пустым и нажмите Enter.

После нажатия клавиши Enter сообщение будет доставлено по указанному адресу.

Примечание: Если ввести адрес, который неизвестен системе или не определен в списке псевдонимов или списке рассылки, то появится сообщение об ошибке с указанием имени пользователя: `[ИД пользователя]... Неизвестный пользователь`.

Ответ на почту:

Для ответа на сообщения электронной почты из командной строки почтового ящика можно вызвать подкоманду **r** или **R**, как показано в следующих примерах.

1. В системе должна быть установлена программа работы с электронной почтой.
2. В почтовом ящике системы должно быть по крайней мере одно сообщение.

Элемент	Описание
r	Эта подкоманда создает новое сообщение, адресованное отправителю выбранного сообщения. Копии этого сообщения будут отправлены всем пользователям, указанным в поле **Cc:** (если оно заполнено). В поле **Subject:** указывается ссылка на выбранное сообщение. По умолчанию подкоманда **r** создает ответ на текущее сообщение. Для ответа на другое сообщение укажите его номер после команды **r**.
R	Создает ответ, адресованный только отправителю данного сообщения. По умолчанию подкоманда **R** создает ответ на текущее сообщение.

R 4	Создает ответ, адресованный только отправителю данного сообщения. Для ответа на другое сообщение укажите его номер после команды **R**. В данном примере создается ответ на сообщение номер 4. В заголовке ответа будет указана примерно следующая информация:
	`To: karen@thor`
	`Subject: Re: Собрание отдела`
	
	Введите текст ответа:
	`Я приду.`
	
	После ввода текста введите точку (**.**) или нажмите клавиши Ctrl-D, чтобы отправить сообщение. После отправки ответа будет показано приглашение программы работы с почтой.

Создание нового сообщения в программе работы с почтой:

Для создания новых сообщений из командной строки почтового ящика можно вызвать подкоманду **m**, как показано в следующем примере:

Элемент	Описание
m *Адрес*	Параметр *Адрес* имеет в качестве значения существующий адрес пользователя. Эта подкоманда запускает редактор электронной почты для создания нового сообщения. После отправки сообщения вновь будет показано приглашение программы работы с почтой.

Пересылка почты:

После прочтения почты можно переслать некоторые электронные сообщения другому пользователю.

1. В системе должна быть установлена программа работы с электронной почтой.
2. Для пересылки сообщения необходимо запустить команду **mail**. Запомните номер электронного сообщения, которое нужно переслать.

Для выполнения этой задачи предназначены подкоманды **~f** и **~m**.

Если вы не сможете некоторое время получать почту по обычному электронному адресу, то можете указать, что в течение этого времени почта должна пересылаться на другой сетевой адрес. Для этого необходимо создать файл `.forward`. См. раздел “Файлы .forward” на стр. 33. В качестве нового адреса может быть указан любой допустимый адрес локальной или удаленной сети. Это может быть адрес коллеги, который будет получать вашу почту во время вашего отсутствия. При пересылке почты копии поступающих сообщений не будут отправляться в ваш почтовый ящик. Вся почта будет пересылаться по указанному адресу (или списку адресов).

Пересылка выбранных сообщений из почтового ящика:

С помощью этой процедуры можно переслать определенные сообщения электронной почты из почтового ящика.

Для пересылки выбранных сообщений выполните следующие действия:

1. Создайте новое сообщение с помощью подкоманды **m** и укажите в нем адрес получателя. Для этого в приглашении программы работы с почтой введите:

   `m` *`пользователь@хост`*

   , где *пользователь* - это имя пользователя, которому будет пересылаться сообщение, а *хост* - имя системы этого пользователя. Если вы укажете имя пользователя локальной системы, то часть адреса *`@Хост`* можно опустить.
2. Укажите тему сообщения в поле **Subject:**.
3. Задайте номер пересылаемого сообщения:

   `~f` *`номер-сообщения`*

   ИЛИ

   `~m` *`номер-сообщения`*

   номер-сообщения задает номер пересылаемого сообщения.

   В результате выполнения команды **mail** будет показано примерно следующее сообщение:

   ```
   Интерполяция: 1
   (продолжение)
   ```
4. Для выхода из программы работы с почтой введите точку (**.**) или пустую строку. В поле **Cc:** введите дополнительные адреса, по которым нужно переслать электронное сообщение.

Пересылка всей почты:

Для пересылки всей почты по другому адресу выполните данную процедуру.

Для пересылки всей почты по другому адресу выполните следующие действия:

1. Для перехода в домашний каталог введите команду **cd** без параметров. Например, для пользователя `mary` введите:

   ```
   cd
   pwd
   ```

   Появится строка:

   ```
   /home/mary
   ```

2. Создайте в домашнем каталоге файл `.forward`. См. См. раздел “Файлы .forward”.

   Примечание: Вы не будете получать почту до тех пор, пока не удалите файл `.forward`.

Файлы .forward:

В файле `.forward` содержатся сетевые адреса пользователей, которым должна пересылаться почта.

Адреса нужно указывать в формате *пользователь@хост*. *Пользователь* - это имя пользователя, которому будет пересылаться почта, а *хост* - имя системы пользователя. Если вы укажете имя пользователя локальной системы, то часть адреса `@`*`Хост`* можно опустить. Файл `.forward` можно создать с помощью команды **cat**.

```
cat > .forward
mark
joe@saturn
[END OF FILE]
```

`[END OF FILE]` - это символ конца файла, которому на большинстве терминалов соответствует комбинация клавиш Ctrl-D. Он должен быть указан в отдельной строке.

Файл `.forward` содержит адреса пользователей, которым будет пересылаться ваша почта. В данном примере почта будет пересылаться пользователю `mark` локальной системы и пользователю `joe` системы `saturn`.

В файле должны быть указаны допустимые адреса. Если файл пуст (имеет нулевой размер), то почта не пересылается, а сохраняется в вашем почтовом ящике.

Примечание: Вы не будете получать почту до тех пор, пока не удалите файл `.forward`.

Отмена пересылки почты:

Для отмены пересылки почты удалите файл `.forward` следующим образом.

Выполните команду **rm**, чтобы удалить файл `.forward` из домашнего каталога:

```
rm .forward
```

Отправка сообщения об отсутствии:

С помощью данной процедуры можно подготовить и отправить сообщение об отсутствии.

В системе должна быть установлена программа работы с электронной почтой.

1. Для того чтобы задать сообщение об отсутствии, перейдите в каталог `$HOME` (начальный каталог) и введите команду:

   ```
   vacation -I
   ```

В результате выполнения этой команды будут созданы файлы .vacation.dir и .vacation.pag, где будут храниться имена отправителей полученных сообщений.

2. Измените файл .forward. Допустим, пользователь carl добавил следующий оператор в файл .forward:

   ```
   carl, |"/usr/bin/vacation carl"
   ```

   Первая запись пользователя carl - имя пользователя, которому пересылается сообщение. Вторая запись пользователя carl - это имя отправителя сообщения об отсутствии. Отправитель почтового сообщения будет получать от пользователя carl одно сообщение об отсутствии каждую неделю, вне зависимости от того, сколько сообщений он отправил пользователю carl. Если во время вашего отсутствия почта пересылается другому пользователю, то сообщение отправителя также будет пересылаться по адресу, указанному в файле .forward.

   Частоту отправки сообщения об отсутствии можно изменить с помощью флага **-f**. Допустим, пользователь carl добавил следующий оператор в файл .forward:

   ```
   carl, |"/usr/bin/vacation -f10d carl"
   ```

   Отправитель сообщения будет получать от пользователя carl по одному сообщению о его отсутствии каждые десять дней, независимо от числа сообщений, которые были отправлены пользователю carl.

3. Для того чтобы сообщение об отсутствии отправлялось всем пользователям, которые присылают вам сообщения, создайте файл $HOME/.vacation.msg и добавить в этот файл нужное сообщение. Ниже приведен пример сообщения об отсутствии:

   ```
   From: carl@odin.austin (Carl Jones)
   Subject: Я в отпуске.
   Я буду в отпуске до 1 октября. Если у вас срочная информация, отправьте,
   пожалуйста, сообщение пользователю Jim Terry <terry@zeus.valhalla>.
     --carl
   ```

   Отправитель будет получать сообщение, указанное в файле $HOME/.vacation.msg, или, если этот файл не существует, сообщение по умолчанию из файла /usr/share/lib/vacation.def. Если ни один из этих файлов не существует, отправитель сообщения не будет получать автоматический ответ, при этом никаких сообщений об ошибке выдаваться не будет.

Для того, чтобы прекратить отправку сообщений об отсутствии, удалите файлы .forward, .vacation.dir, .vacation.pag и .vacation.msg из каталога $HOME (домашнего каталога):

```
rm .forward .vacation.dir .vacation.pag .vacation.msg
```

Отправка и получение секретной почты:

Для отправки секретной почты введите в командной строке команду **xsend**, как показано в следующих примерах.

1. В системе должна быть установлена программа работы с электронной почтой.
2. Должен быть задан пароль с помощью команды **enroll**.

Элемент	Описание
xsend barbara	В данном примере секретная почта отправляется пользователю barbara. При нажатии клавиши Enter появляется строка для ввода текста сообщения. После ввода сообщения нажмите комбинацию клавиш Ctrl-D или введите точку (.), чтобы закрыть редактор почты и отправить сообщение. Команда **xsend** шифрует сообщение перед отправкой.

1. Для получения секретной почты введите в командной строке:

   ```
   mail
   ```

   Появится список сообщений, находящихся в вашем почтовом ящике. Программа работы с секретной электронной почтой уведомит вас о получении секретного сообщения. В строке сообщения будет содержаться примерно следующая информация:

   ```
   Почта [5.2 UCB] Для просмотра справки введите ?.
   "/usr/spool/mail/linda": 4 сообщения 4 новых
   >N 1 robert среда 14 апреля 15:23 4/182 "секретная почта от robert@Zeus"
   ```

Это сообщение говорит о том, что вы можете прочитать полученное секретное сообщение с помощью команды **xget**.

2. Введите в командной строке:

   ```
   xget
   ```

   Появится приглашение для ввода пароля, который был предварительно задан с помощью команды **enroll**. После ввода пароля появится приглашение команды **xget** и список всех секретных сообщений. Вы можете просмотреть любое секретное сообщение с помощью программы работы с электронной почтой. Если вы хотите оставить прочитанные и непрочитанные сообщения в секретном почтовом ящике и запретить удаление сообщений с помощью команды **xget**, введите подкоманду **q**.

Справочная информация почтовой программы

Для просмотра справки по программе работы с почтой введите команду **?**, **man** или **info**.

Элемент	Описание
Для просмотра справки в программе работы с почтой	Введите **?** или `help` в приглашении программы работы с почтой. Символ **?** и **help** показывают список подкоманд почтового ящика. Для просмотра всех подкоманд (без описаний) можно воспользоваться подкомандой **(l)**ist.
Для просмотра справки в редакторе сообщений	Введите **~?** в приглашении редактора сообщений. Подкоманда **~?** показывает подкоманды редактора сообщений.
Для просмотра справки по секретным сообщениям	Введите **?** в приглашении редактора сообщений. Символ **?** показывает подкоманды работы с секретной почтой.
Для просмотра справки по руководству работы с почтой	Введите `man mail` в командной строке. В данном примере, `mail` - это имя команды, информацию о которой нужно найти. Система предоставит необходимую информацию о команде **mail** в формате ASCII. Для просмотра всей информации после появления маркера продолжения (:) нажмите клавишу Enter. Команда **man** предоставляет информацию о командах, подкомандах и файлах. Информация предоставляется в формате ASCII.

Опции настройки почтовой программы

Пользователи могут настраивать программу работы с электронной почтой, изменяя команды и опции, указанные в файлах `.mailrc` и `/usr/share/lib/Mail.rc`.

Дополнительная информация об опциях программы работы с почтой приведена в разделе “Опции включения и отключения почты” на стр. 36.

Ниже перечислены настраиваемые параметры программы работы с почтой:

- **Запрос темы сообщения.** После ввода команды **mail** программа предлагает указать тему сообщения в поле **Subject:**. В этом поле можно указать краткое описание сообщения. Тема сообщения указывается при его получении и позволяет быстро выбрать нужное сообщение в списке. См. раздел “Поля Subject: и Carbon Copy (Cc:)” на стр. 37.
- **Запрос списка пользователей для получения копии сообщения.** В файле `.mailrc` можно указать, чтобы при отправке сообщения программа работы с почтой должна запрашивать имена других пользователей, которым следует отправить копии сообщения. См. раздел “Поля Subject: и Carbon Copy (Cc:)” на стр. 37.
- **Псевдонимы или списки рассылки.** Если вам приходится рассылать почту в большей сети или часто отправлять одно и тоже сообщение сразу нескольким пользователям, то ввод длинных списков становится утомительным. Для упрощения этого процесса создайте в файле `.mailrc` псевдоним или список рассылки.

Псевдоним - это имя, которое можно указывать вместо полного адреса пользователя. *Список рассылки* - это имя, которое можно применять вместо группы пользовательских адресов. См. См. раздел “Список псевдонимов и рассылки” на стр. 38.

- **Число строк, отображаемых при просмотре сообщения.** Вы можете изменить число строк, отображаемых на экране при просмотре заголовков или текста сообщений. См. См. раздел “Изменение числа строк в заголовке или тексте сообщения” на стр. 39.
- **Информация, которая показывается о сообщениях.** Вы можете отключить отображение отдельных полей заголовков, например поля `message-id`. См. См. раздел “Отображение информации в сообщении” на стр. 40.
- **Каталог для хранения сообщений.** Вы можете создать специальный каталог для сообщений. Для обозначения этого каталога при сохранении сообщений или просмотре содержимого папок можно применять подкоманду знак плюс (+). См. См. раздел “Создание папок по умолчанию для хранения сообщений” на стр. 42.
- **Файл протокола для записи отправляемых сообщений.** Вы можете указать, что программа **mail** должна сохранять все отправляемые сообщения в файле или подкаталоге вашего домашнего каталога. См. См. раздел “Создание папок по умолчанию для хранения сообщений” на стр. 42.
- **Редакторы для ввода сообщений**. Для редактирования сообщений, помимо внутреннего редактора программы работы с почтой можно применять два других редактора. См. См. раздел “Текстовые редакторы для ввода текста сообщений” на стр. 43.

Дополнительная информация о надстройке программы работы с почтой приведена в следующих разделах:

Опции включения и отключения почты:

В программе работы с почтой предусмотрено два типа опций: опции-переключатели и опции, для которых можно задавать значения.

Опции-переключатели можно установить или сбросить (с помощью команд **set** и **unset** соответственно), а опциям со значением можно с помощью команды **set** присвоить определенное значение.

> **Примечание:** Команда **unset** *опция* эквивалентна команде **set no** *опция*.

Команда **pg** предназначена для просмотра файла `/usr/share/lib/Mail.rc`. Содержимое файла `/usr/share/lib/Mail.rc` определяет конфигурацию программы работы с почтой. Вы можете изменить системную конфигурацию программы работы с почтой, создав файл `$HOME/.mailrc`. При работе команды **mail** опции, перечисленные в файле **.mailrc**, переопределяют аналогичные опции файла `/usr/share/lib/Mail.rc`. Опции `.mailrc` можно изменять и применять их в программе работы с почтой.

Для выполнения записанных в файле команд работы с почтой введите подкоманду **source**.

`Предварительные требования`

В системе должна быть установлена программа работы с электронной почтой.

Включение опций почты:

Ниже приведены некоторые распространенные подкоманды, применяемые для настройки программы работы с электронной почтой:

Элемент	Описание
set	Включает опции программы работы с электронной почтой.
source	Включает опции программы работы с почтой, записанные в файле. При просмотре почты, вы можете ввести эту подкоманду в приглашении программы работы с почтой. `source` *полное-имя* где *полное-имя* задает путь и имя файла, в котором перечислены опции программы работы с почтой. Указанные в этом файле опции переопределяют на время текущего сеанса аналогичные опции, заданные ранее. Кроме того, вы можете изменять параметры текущего сеанса с помощью подкоманд программы работы с почтой.

Вы можете задать эти опции в приглашении программы работы с почтой или указать их в файле `.mailrc`.

Просмотр активных опций программы работы с почтой:

Для просмотра всех активных опций, перечисленных в файле `.mailrc`, укажите при чтении почты подкоманду **set** без аргументов.

С помощью этой информации можно также определить, выбран ли каталог для сообщений и заносятся ли в протокол все отправляемые сообщения.

В приглашении программы работы с почтой введите:

```
set
```

Будет показано примерно следующее сообщение:

```
ask
metoo
toplines 10
```

В данном примере активны две опции-переключателя: **ask** и **metoo**. В списке нет записи **askcc**. Это значит, что опция **askcc** не активна. Опции **toplines** присвоено значение `10`. Описание опций **ask**, **metoo**, **askcc** и **toplines** приведено в разделе .mailrc File Format книги *Справочник по файлам* .

Отключение опций почты:

Ниже приведены некоторые распространенные подкоманды, применяемые для настройки программы работы с электронной почтой:

Элемент	Описание
unset	Отключает опции программы работы с почтой.
unalias	Удаляет указанные псевдонимы.
ignore	Не отображает поля заголовков сообщений.

Вы можете задать эти опции в приглашении программы работы с почтой или указать их в файле `.mailrc`.

Примечание: Команда **unset** *опция* эквивалентна команде **set no** *опция*.

Поля Subject: и Carbon Copy (Cc:):

При изменении полей **Subject:** и **Cc:** следует выполнить следующие предварительные условия.

`Предварительные требования`

В системе должна быть установлена программа работы с электронной почтой.

Включение и отключение поля Subject::

С помощью команд **set** и **unset** можно включить и выключить поле **Subject:**.

В следующих примерах показано, как можно включить или отключить запрос поля **Subject:**

Элемент	Описание
set ask	Эта команда изменяет в файле `.mailrc` опцию `ask` и включает запрос поля **Subject:**.
unset ask	Эта команда изменяет в файле `.mailrc` опцию `ask` и выключает запрос поля **Subject:**.

Включение и отключение запроса поля копии (Cc:):

С помощью команд **set** и **unset** можно включить и выключить поле **Cc:**.

В следующих примерах показано, как можно включить или отключить запрос поля **Cc:**

Элемент	Описание
set askcc	Эта команда изменяет в файле `.mailrc` опцию **askcc** и включает запрос поля копий (**Cc:**).
unset askcc	Эта команда изменяет в файле `.mailrc` опцию **askcc** и включает запрос поля копий (**Cc:**).

Список псевдонимов и рассылки:

Создав списки псевдонимов и рассылок можно упростить управление адресами.

Прежде чем создавать список псевдонимов или список рассылки, убедитесь в соблюдении следующих условий:

1. В системе должна быть установлена программа работы с электронной почтой.
2. Вы должны знать имена и адреса пользователей, которых собираетесь включить в список псевдонимов и список рассылки.

Ниже описаны способы создания псевдонимов или списков рассылки:

Элемент	Описание

alias	`kath kathleen@gtwn`
	
	В данном примере псевдоним `kath` будет соответствовать пользователю `kathleen` с адресом `gtwn`. После того, как эта строка будет добавлена в файл `$HOME/.mailrc`, для отправки сообщения пользователю kathleen можно будет ввести следующую команду:
	`mail kath`
	
	Теперь для отправки почты пользователю Kathleen можно использовать псевдоним `kath`.

alias	`dept dee@merlin anne@anchor jerry@zeus bill carl`
	
	После того, как эта строка будет добавлена в файл `$HOME/.mailrc`, для отправки сообщения всем пользователям своего отдела достаточно будет ввести следующую команду:
	`mail dept`
	
	Созданное сообщение будет отправлено следующим пользователям: `dee` в системе `merlin`, `anne` в системе `anchor`, `jerry` в системе `zeus`, а также пользователям `bill` и `carl` в локальной системе.

Для просмотра списков псевдонимов и рассылок введите в приглашении команды работы с почтой:

`alias`

ИЛИ

`a`

Будут перечислены все определенные псевдонимы и списки рассылки.

Изменение числа строк в заголовке или тексте сообщения:

Изменяя файл `.mailrc`, вы можете настроить число строк, отображаемых при просмотре содержимого почтового ящика или текста сообщений.

Для внесения таких изменений, в системе должна быть установлена почтовая программа.

Изменение числа строк, отображаемых в списке сообщений:

Каждому сообщению, находящемуся в почтовом ящике, соответствует одна строка заголовка, показанная в списке сообщений. Если сообщений больше 24, то первые заголовки не будут видны на экране. Опция **set screen** позволяет изменять число одновременно отображаемых строк списка.

Для изменения числа одновременно отображаемых строк в списке сообщений укажите в файле **$HOME/.mailrc** следующую опцию:

```
set screen=20
```

В данном примере система будет одновременно показывать 20 заголовков сообщений. Для просмотра дополнительных групп заголовков введите подкоманду **h** или **z**. Эту подкоманду можно также указать в приглашении программы работы с почтой.

Изменение числа строк в длинном сообщении:

Если вы просматриваете сообщение длиной более 24 строк, то первые строки сообщения не будут видны на экране. Для просмотра длинных сообщений можно использовать команду **pg**, если включена опция **set crt** в файле `.mailrc`.

Опция **set crt** задает число строк в сообщении, при превышении которого запускается команда **pg**.

Например, если вы введете подкоманду **t** для просмотра большого сообщения, то на экране будет отображаться только одна страница. Внизу страницы будет показано приглашение с двоеточием, означающее, что есть еще непросмотренные страницы. Для просмотра следующей страницы сообщения нажмите клавишу Enter. После вывода на экран последней страницы появится примерно следующее приглашение:

```
EOF:
```

В этом приглашении можно ввести любую допустимую подкоманду **pg**. Вы можете просмотреть предыдущие страницы, начать поиск в сообщении определенной строки символов, или прекратить просмотр сообщения и вернуться в командную строку программы работы с почтой.

В файле `.mailrc` опцию **set crt** нужно указывать в следующем формате:

set crt=*число-строк*

Например:

```
set crt=20
```

указывает, что сообщение, передаваемое программе **pg**, должно состоять не менее чем из 20 строк. Программа **pg** будет запускаться при просмотре сообщений длиной более 20 строк.

Изменение числа строк, отображаемых в верхней части сообщения:

Подкоманда **top** позволяет просмотреть сообщение без чтения всего документа.

Вы можете указать в опции **toplines** число отображаемых строк:

set toplines=*число-строк*

В этой подкоманде переменная *число-строк* определяет число верхних строк, включая все поля заголовков, которые должны отображаться после ввода подкоманды **top**.

Например, если в файле `.mailrc` пол ьзователя Amy указана следующая строка:

```
set toplines=10
```

Когда Amy введет команду **mail** для просмотра новых сообщений, на экране будет показана примерно следующая информация:

```
Mail  Для просмотра справки введите ?.
"/usr/mail/amy": 2 сообщения 2 новых>
N  1 george   среда  6 января  9:47  11/257 "Собрание отдела"
N  2 mark     среда  6 января 12:59  17/445 "Проект"
```

Когда Amy введет подкоманду **top** для просмотра сообщений, на экране будет показана часть сообщения, как в следующем примере:

```
top 1
Сообщение 1:
From george среда  6 января 1988 г. 9:47
Received: by zeus
    id AA00549; среда, 6 января 1988 г.  9:47:46
Date: среда  6 января 1988 г.  9:47:46
From: george@zeus
Message-Id: <8709111757.AA00178>
To: amy@zeus
Subject: Собрание отдела
Пожалуйста, не забудь про собрание в пятницу
в 13:30 в конференц-зале. Мы будем
```

Сообщение показано лишь частично, так как опции **toplines** присвоено значение 10. На экране отображаются только строки с 1 (поле **Received:**) по 10 (вторая строка текста сообщения). Первая строка, `From george среда 6 января 1988 г. 9:47`, отображается всегда, независимо от значения опции **toplines**.

Отображение информации в сообщении:

Изменяя файл `.mailrc`, вы можете настраивать объем информации заголовков, отображаемой при просмотре сообщений.

Отображение некоторой информации заголовков может быть уже отключено. Для определения игнорируемых полей заголовков просмотрите файл `/usr/share/lib/Mail.rc`.

`Предварительные требования`

В системе должна быть установлена программа работы с электронной почтой.

Отключение отображения заголовков Date (Дата), From (От кого) и To (Кому):

В верхней части каждого сообщения показано несколько полей заголовков. Эти поля заголовков отображаются на экране при просмотре сообщения. Если вы не хотите, чтобы они были показаны при просмотре сообщения, воспользуйтесь подкомандой **ignore**.

Формат подкоманды **ignore**:

`ignore [`*список-полей*`]`

В параметре *СписокПолей* можно указать одно или несколько имен полей, которые должны быть игнорироваться при просмотре сообщения. Например, если указать в файле `.mailrc` строку

```
ignore date from to
```

и указать в файле `/usr/share/lib/Mail.rc` строку

```
ignore received message-id
```

Результат применения подкоманды **t** следующий:

```
t 1
Сообщение 1:
From george среда  6 января 1988 г. 9:47
Subject: Собрание отдела
Пожалуйста, не забудь про собрание в пятницу
в 13:30 в конференц-зале. Мы будем
обсуждать новые процедуры работы с
программой планирования, предложенные
нашими специалистами.
```

Поля **Received:**, **Date:**, **From:**, **Message-Id:** и **To:** не показаны на экране. Для отображения этих полей вызовите подкоманду **T**, **P** или **top**.

> **Примечание:** В этом примере поле **From** показано на экране. Однако это не поле **From:**, включенное в *список-полей* подкоманды **ignore**.

Просмотр игнорируемых заголовков:

Для просмотра игнорируемых полей заголовка используйте подкоманду **ignore**.

Для просмотра списка игнорируемых полей заголовков введите в приглашении программы работы с почтой следующую команду:

```
ignore
```

Будет показан список игнорируемых полей заголовков. Например:

```
mail-from
message-id
return-path
```

Сброс полей заголовков:

Для сброса опции вывода полей заголовков введите подкоманду **retain**.

Например:

```
retain date
```

Просмотр отключенных полей заголовков:

Для просмотра отключенных полей заголовка используйте подкоманду **retain**.

Для просмотра списка отключенных полей заголовков введите подкоманду **retain** без параметров.

Запрет отображения информационной строки:

Информационная строка отображается в верхней части списка сообщений. При запуске команды **mail** в ней показано имя программы работы с почтой.

Она выглядит примерно так:

```
Почта [5.2 UCB]  [Рабочая станция 3.1] Для просмотра справки введите ?.
```

Для отключения информационной строки при запуске программы добавьте следующую строку в файл `$HOME/.mailrc`:

```
set quiet
```

Другая опция, отключающая отображение информационной строки команды **mail**:

```
set noheader
```

Если указать эту опцию в файле `.mailrc`, то список сообщений, находящихся в почтовом ящике, не отображается. При запуске программы **mail** будет показано только ее командное приглашение. Для просмотра списка сообщений нужно будет ввести подкоманду (**h**)eader.

Объединение команд delete и print:

С помощью опции autoprint можно объединить подкоманды delete и print.

Прочитав сообщение вы можете удалить его с помощью подкоманды **d**. Перейти к следующему сообщению можно с помощью подкоманды **p**. Для объединения этих подкоманд введите в файле `.mailrc` следующую строку:

```
set autoprint
```

Если в файле `.mailrc` задана опция **set autoprint**, то подкоманда **d** удаляет текущее сообщение и показывает следующее.

Создание папок по умолчанию для хранения сообщений:

Папки по умолчанию позволяют хранить в них сообщения.

В системе должна быть установлена программа работы с электронной почтой.

С помощью следующей процедуры можно создать каталоги почтовых ящиков для хранения сообщений в папках:

1. Для того чтобы определить, указана ли опция **set folder** в файле `.mailrc`, введите следующую подкоманду в приглашении программы работы с почтой:

   ```
   set
   ```

   Если опция **set folder** активна, то будет показана примерно такая строка:

   ```
   folder /home/george/letters
   ```

   В данном примере `letters` - это каталог, в котором будут храниться почтовые папки.
2. Если опция **set folder** не указана, то добавьте в файл `.mailrc` запись **set folder**:

   ```
   задайте folder=/home/george/letters
   ```

   В этом примере значение `/home/george` задает домашний каталог пользователя george, а `letters` - каталог, в котором будут храниться почтовые папки. Опция **set folder** позволяет сохранять сообщения в каталоге `letters` с помощью знака плюса (+).
3. Если каталог `letters` не существует, его необходимо создать в домашнем каталоге. Находясь в домашнем каталоге, введите в системной командной строке:

   ```
   mkdir letters
   ```

С помощью следующей процедуры можно сохранять записи отправленных другим пользователям сообщений:

1. Добавьте следующую строку в файл `.mailrc`:

   ```
   set record=letters/mailout
   ```
2. Если каталог `letters` не существует, его необходимо создать в домашнем каталоге. Находясь в домашнем каталоге, введите в системной командной строке:

   ```
   mkdir letters
   ```
3. Для просмотра копий отправленных сообщений введите команду:

   ```
   mail -f +mailout
   ```

   В данном примере копии отправленных сообщений хранятся в файле `mailout`.

Текстовые редакторы для ввода текста сообщений:

С помощью опции **set EDITOR=***полное-имя* можно задать текстовый редактор, применяемый для создания текста сообщений.

В системе должна быть установлена программа работы с электронной почтой.

Элемент	Описание
set EDITOR=*полное-имя*	Эта опция файла `.mailrc` задает редактор, который будет запущен при вводе подкоманды **~e**. Параметр *полное-имя* задает полное имя программы альтернативного редактора. Для перехода к редактору e в программе mail введите следующую команду: `~e` Эта подкоманда запускает редактор e или другой редактор, указанный в файле `.mailrc`. Измените сообщение с помощью редактора.
set VISUAL=*полное-имя*	Эта опция файла `.mailrc` задает редактор, который будет запущен при вводе подкоманды **~v**. Параметр *полное-имя* задает полное имя программы альтернативного редактора. Значение по умолчанию - `/usr/bin/vi`. Для перехода к редактору vi в программе mail введите: `~v` Эта подкоманда запускает редактор vi или другой редактор, указанный в файле `.mailrc`. Измените сообщение с помощью редактора.

Подкоманды команды Mail

Команда **mail** использует различные подкоманды для выполнения различных функций.

Этот раздел используется в качестве справочника для команды **mail** и ее подкоманд.

Команды выполнения почты:

Используйте эти системные команды для выполнения почты.

Элемент	Описание
mail	Показывает системный почтовый ящик.
mail -f	Показывает ваш личный почтовый ящик (**mbox**).
mail -f +*папка*	Показывает почтовую папку.
mail *пользователь@адрес*	Адресует сообщение указанному пользователю.

Подкоманды почтового ящика в почтовой программе:

Когда программа работы с почтой обрабатывает содержимое почтового ящика, она показывает на экране приглашение, указывающее, что программа ожидает ввода информации.

Приглашение программы работы с почтой - это амперсанд (&) в начале новой строки. В этом приглашении можно ввести любую подкоманду программы работы с почтой.

Подкоманды управления почтовой программой:

С помощью этих подкоманд можно управлять почтовой программой.

Элемент	Описание
q	Завершает сеанс и применяет все подкоманды, введенные во время работы с почтовым ящиком.
x	Завершает работу и возвращает почтовый ящик в первоначальное состояние.
!	Запускает оболочку, выполняет указанную команду и вновь показывает приглашение программы работы с почтой.
cd *каталог*	Позволяет перейти в указанный `каталог` или каталог `$HOME`.

Подкоманды просмотра почтовой программы:

С помощью этих подкоманд можно управлять отображением почтовой программы.

Элемент	Описание
t	Показывает сообщения из *списка-сообщений* или текущее сообщение.
n	Показывает следующее сообщение.
f *список-сообщений*	Показывает заголовки сообщений из *списка-сообщений* или заголовок текущего сообщения, если *список* не задан.
h *число*	Показывает заголовки групп, содержащих сообщение с указанным *номером*.
top *число*	Показывает часть сообщения.
set	Показывает список всех активных опций, заданных в файле **.mailrc**.
ignore	Показывает список всех игнорируемых полей заголовков.
folder	Показывает число сообщений в текущей папке и полный путь к этой папке.

Обработка сообщений:

С помощью этих подкоманд можно изменять, удалять, вызывать, добавлять и сохранять сообщения.

Элемент	Описание
e *номер*	Запускает редактор для изменения сообщения с указанным *номером* (редактор по умолчанию - e).
d *список-сообщений*	Удаляет сообщения, перечисленные в *списке-сообщений* или текущее сообщение.
u *список-сообщений*	Восстанавливает удаленные сообщения, перечисленные в *списке-сообщений*.
s *список-сообщений* **+***файл*	Добавляет сообщения (с заголовками) в указанный *файл*.
w *список-сообщений* **+***файл*	Добавляет текст сообщений в указанный *файл*.
pre *список-сообщений*	Сохраняет сообщения в системном почтовом ящике.

Подкоманды создания нового сообщения:

С помощью этих команд можно создавать сообщения.

Элемент	Описание
m*список-адресов*	Создает новое сообщение и рассылает его по адресам, перечисленным в *списке-адресов*.
r*список-сообщений*	Отправляет ответ отправителям и получателям сообщений.
R*список-сообщений*	Отправляет ответ только отправителям сообщений.
a	Показывает список псевдонимов и их адреса.

Подкоманды редактора электронной почты:

Редактор почты во время работы отображает на экране приглашение, указывающее, что редактор ожидает ввода информации.

В этом приглашении можно ввести любую подкоманду редактора.

Подкоманды управления редактором электронной почты:

Для управления редактором электронной почты используются следующие подкоманды.

Элемент	Описание
~q	Завершает работу с редактором без сохранения или отправки текущего сообщения.
~p	Показывает содержимое буфера сообщения.
~: *mcmd*	Выполняет подкоманду почтового ящика (*mcmd*).
EOT	Отправляет сообщение (на большинстве терминалов соответствует комбинации клавиш Ctrl-D).
.	Отправляет текущее сообщение.

Команды для добавления информации к заголовку сообщения:

С помощью этих команд можно добавить различные элементы заголовка к сообщениям.

Элемент	Описание
~h	Добавляет информацию в поля заголовков **To:**, **Subject**:, **Cc:** и **Bcc:**.
~t *список-адресов*	Добавляет адреса пользователей, перечисленных в *списке-адресов*, в поле **To:**.
~s *тема*	Присваивает полю **Subject** значение, заданное в параметре *subject*.
~c *список-адресов*	Добавляет адреса пользователей, перечисленных в *списке-адресов*, в поле **Cc:**.
~b *список-адресов*	Добавляет адреса пользователей, перечисленных в *списке-адресов*, в поле `Bcc:`.

Команды для добавления информации к сообщению:

С помощью этих команд можно добавить информацию к сообщению.

Элемент	Описание
~d	Добавляет к сообщению содержимое файла `dead.letter`.
~r *имя-файла*	Добавляет к сообщению содержимое файла *имя-файла*.
~f *список-номеров*	Дополняет сообщения, перечисленные в *списке-номеров*.
~m*список-номеров*	Дополняет сообщения, перечисленные в *списке-номеров*, с отступом.

Команды редактирования сообщения:

С помощью этих команд можно изменить сообщения.

Элемент	Описание
~e	Позволяет изменить сообщение с помощью редактора e (e - редактор по умолчанию).
~v	Позволяет изменить сообщение с помощью редактора vi (vi - редактор по умолчанию).
~w*имя-файла*	Записывает сообщение в файл *имя-файла*.
Команда **~!**	Запускает оболочку, выполняет указанную *команду* и вновь показывает приглашение редактора.
~|*команда*	Передает сообщение в стандартный поток ввода *команды* и заменяет текст сообщения на стандартный поток вывода этой команды.

Подкоманды секретной почты:

Когда программа работы с секретной почтой обрабатывает содержимое почтового ящика, она показывает на экране, указывающее, что программа ожидает ввода информации.

Приглашение программы работы с секретной почтой - это вопросительный знак (?) в начале новой строки. В этом приглашении можно ввести любую подкоманду работы с секретной почтой.

Подкоманды секретной почты:

Для отправки секретной почты используйте следующие подкоманды.

Элемент	Описание
xsend barbara	Адресует сообщение указанному пользователю.
xget	Показывает содержимое секретного почтового ящика.

Задачи почтового ящика:

Следующие подкоманды служат для выполнения различных задач почтового ящика.

Элемент	Описание
q	Завершает работу, оставляя непрочитанные сообщения.
n	Удаляет текущее сообщение и показывает следующее.
d	Удаляет текущее сообщение и показывает следующее.
Клавиша Return	Удаляет текущее сообщение и показывает следующее.
!	Выполняет команду оболочки.
s	Сохраняет сообщение в указанном файле или в mbox.
w	Сохраняет сообщение в указанном файле или в mbox.

Задачи по управлению почтой

За выполнение этих задач отвечает диспетчер электронной почты.

1. Настройте файл /etc/rc.tcpip чтобы демон **sendmail** запускаля при загрузке с истемы. См. раздел “Настройка файла /etc/rc.tcpip для запуска демона sendmail”.
2. Настройте файл конфигурации /etc/mail/sendmail.cf . По умолчанию файл /etc/mail/sendmail.cf настроен для доставки локальной почты и пакетов TCP/IP. Для доставки почты с помощью BNU необходимо изменить файл /etc/mail/sendmail.cf. Дополнительную информацию см. в описании файла sendmail.cf в книге *Справочник по файлам* .
3. Задайте в файле /etc/mail/aliases псевдонимы, действующие в рамках домена и системы. Дополнительная информация приведена в разделе “Почтовые псевдонимы”.
4. Настройте очередь почты. Дополнительная информация приведена в разделе “Почтовая очередь” на стр. 49.
5. Настройте протокол почтовой программы. Дополнительная информация приведена в разделе “Ведение протоколов почты” на стр. 53.

Настройка файла /etc/rc.tcpip для запуска демона sendmail

Настройте файл /etc/rc.tcpip таким образом, чтобы демон **sendmail** автоматически запускался при загрузке системы.

1. Измените файл /etc/rc.tcpip с помощью любого текстового редактора.
2. Найдите строку, начинающуюся со слов start /usr/lib/sendmail. По умолчанию эта строка не является комментарием, т.е. в ее начале не должен стоять символ # (знак фунта стерлингов). Если же этот символ стоит в начале строки, удалите его.
3. Сохраните этот файл.

Теперь система будет запускать **sendmail** при загрузке.

Почтовые псевдонимы

Псевдонимы устанавливают соответствие между именами и адресами с помощью личных, системных и доменных файлов.

Предусмотрено три различных типа псевдонимов:

Элемент	Описание
личный	Его задает конкретный пользователь в личном файле `$HOME/.mailrc`.
системный	Его задает администратор системы доставки почты в файле `/etc/mail/aliases`. Эти псевдонимы применяются программой **sendmail** при работе с почтовыми сообщениями в локальной системе. Необходимость изменять псевдонимы локальной системы возникает редко.
доменные	По умолчанию для преобразования псевдонимов программа **sendmail** по умолчанию применяет файл `/etc/alias`. Для того чтобы вместо этого файла применялась служба NIS, добавьте в файл `/etc/netsvc.conf` следующую строку (если этот файл не существует, то создайте его): `aliases=nis`

Файл /etc/mail/aliases

Здесь описаны свойства, содержимое и расположение файла `/etc/mail/aliases`.

Файл `/etc/mail/aliases` содержит группы записей следующего формата:

Псевдоним`:` *Имя1*`,`
Имя2`, ...` *ИмяN*

Псевдоним - это любая последовательность алфавитно-цифровых символов (за исключением специальных символов, например @ или !). Значения от*Имя1* до *ИмяX* - одно или несколько имен адресата. Список имен может состоять из нескольких строк. Каждая последующая строка начинается с пробела или символа табуляции. Пустые строки и строки, начинающиеся с # (знак фунта), представляют собой комментарии.

Файл `/etc/mail/aliases` обязательно должен содержать следующие псевдонимы:

Элемент	Описание
MAILER-DAEMON	ИД пользователя, получающего сообщения, адресованные почтовому демону. Изначально этим пользователем является root: `MAILER-DAEMON: root`
postmaster	ИД пользователя-администратора локальной почтовой системы. Псевдоним **postmaster** определяет адрес почтового ящика, который существует в каждой системе. Этот псевдоним позволяет пользователям в любой системе отправлять сообщения обладателю псевдонима **postmaster**, не зная его реального адреса в данной системе. Изначально этим пользователем является root: `postmaster: root`
nobody	Это ИД получателя сообщений, отправленных таким программам, как **news** и **msgs**. Изначально присваивается имя `/dev/null`: `nobody: /dev/null` Для того чтобы получать подобные сообщения, укажите для этого псевдонима имя реального пользователя.

После изменения этого файла его необходимо заново преобразовывать в базу данных, применяемую командой **sendmail**. См. раздел “Создание базы данных псевдонимов” на стр. 48.

Создание локального псевдонима для почты

С помощью локальных почтовых псевдонимов можно создать группы получателей, или списки рассылки, которым будет отправляться почта.

В данном сценарии почтовый псевдоним `testers` будет назначен получателям `geo@medussa`, `mark@zeus`, `ctw@athena` и `dsf@plato`. После создания псевдонима `testers` его владельцем будет назначен пользователь с адресом glenda@hera.

После добавления псевдонима `testers` в файл `/etc/mail/aliases` база данных псевдонимов будет заново скомпилирована с помощью команды **sendmail**. Затем псевдониму `testers` будет отправлено электронное сообщение.

Рассмотрите следующие вопросы:

- Описанная ниже процедура была протестирована в отдельных версиях AIX. Результаты, которые вы можете получить, в значительной степени зависят от конкретных версии и уровня AIX.

Для создания локального почтового псевдонима выполните следующие действия:

1. Откройте файл /etc/mail/aliases в любом текстовом редакторе.
2. На пустой строке введите имя псевдонима, двоеточие и список получателей через запятую. Например, для определения псевдонима testers нужно ввести следующее:

   ```
   testers: geo@medussa, mark@zeus, ctw@athena, dsf@plato
   ```

3. Создайте владельца псевдонима. Если команде **sendmail** не удастся отправить почтовое сообщение указанному псевдониму, она отправит сообщение об ошибке владельцу этого псевдонима.

   Откройте файл /etc/mail/aliases и укажите владельца псевдонима на отдельной строке. Формат этой строки: owner-*groupname*: *owner*, где *groupname* - это псевдоним, а *owner* - адрес электронной почты владельца. Например, для того чтобы назначить пользователя glenda@hera владельцем псевдонима testers нужно ввести следующее:

   ```
   testers:
   geo@medussa, mark@zeus, ctw@athena, dsf@plato owner-testers: glenda@hera
   ```

4. После создания псевдонима вызовите команду **sendmail -bi**, чтобы скомпилировать базу данных псевдонимов. Эту команду нужно применять после внесения любого изменения в файл /etc/mail/aliases.

Теперь можно отправить электронное сообщение псевдониму testers.

Создание базы данных псевдонимов

Команда **sendmail** не использует определения псевдонимов из файла /etc/mail/aliases локальной системы. Вместо этого команда **sendmail** применяет базу данных, созданную на основе файла /etc/mail/aliases.

Создать базу данных вы можете следующими способами:

- Запустите команду /usr/sbin/sendmail с флагом **-bi**.
- Запустите команду **newaliases**. Это приведет к тому, что команда **sendmail** считает файл /etc/mail/aliases локальной системы и создаст новый файл с базой данных псевдонимов. Этот файл создается в более удобном формате Berkeley:

  /etc/mail/aliases.db
- Запустите команду **sendmail** с флагом **Rebuild Aliases**. В дальнейшем реорганизация базы данных псевдонимов может происходить автоматически. Автоматическую реорганизацию не рекомендуется включать на сильно загруженных машинах с большими файлами псевдонимов. Если время, требующееся для реорганизации базы данных, превысит тайм-аут реорганизации (обычно 5 минут), может быть запущено сразу нескольких этих процессов.

Примечание:

1. Если данные файлы отсутствуют, то команда **sendmail** не сможет обработать почту и вернет сообщение об ошибке.
2. Если у вас есть несколько баз данных псевдонимов, укажите флаг **-bi**. При этом sendmail реорганизует все поддерживаемые базы данных (например, базы данных Администратора сетевых баз данных (NDBM), но не базы данных NIS).

В файле /etc/netsvc.conf задается порядок вызова системных служб. Для того чтобы упорядочить службы псевдонимов, введите следующую строку:

```
aliases=служба, служба
```

где служба - это либо files, либо nis. Например:

```
aliases=files, nis
```

Эта строка означает, что команда **sendmail** должна считать локальный файл псевдонимов, а в случае неудачи запросить `nis`. Если `nis` определена как служба, то она уже должна быть запущена.

Дополнительная информация о файле `/etc/netsvc.conf` приведена в разделе *Справочник по файлам* .

Почтовая очередь

Почтовая очередь представляет собой каталог, в котором хранятся и обрабатываются данные и файлы почтовых сообщений, доставляемых командой **sendmail**. По умолчанию почтовая очередь находится в каталоге `/var/spool/mqueue`.

Есть несколько причин, по которым почтовые сообщения могут быть помещены в очередь.

Например:

1. Команду **sendmail** можно настроить так, чтобы обработка очереди выполнялась через определенные промежутки времени, а не немедленно после получения сообщения. При этом почтовые сообщения необходимо временно хранить.
2. Если удаленный почтовый сервер не отвечает на запрос об установлении соединения, то почтовая программа помещает сообщения в очередь с тем, чтобы позднее повторить попытку.

Печать почтовой очереди

Очередь сообщений можно распечатать с помощью команды **mailq** (или команды **sendmail** с флагом **-bp**).

При этом будет показан список ИД сообщений в очереди, их размер, дата помещения сообщения в очередь, имя получателя и отправителя.

Файлы почтовой очереди

С каждым сообщением в почтовой очереди связано несколько файлов.

Имена файлам присваиваются в соответствии со следующими стандартами:

ТипfИД

Здесь *ИД* - это уникальный идентификатор очереди сообщений, а *Тип* - одна из перечисленных ниже букв, определяющих тип файла:

Элемент	Описание
`d`	Файл данных, содержащий тело сообщения без заголовков.
`q`	Файл управления очередью. Он содержит данные, необходимые для обработки задания.
`t`	Временный файл. Представляет собой копию файла `q` и создается на время повторного создания последнего. Сразу после этого он переименовывается в файл `q`.
`x`	Это файл протокола, который существует в течение сеанса, и в который записывается все происходящее во время данного сеанса.

Например, если с сообщением связан идентификатор очереди `AA00269`, то команда **sendmail** при доставке письма сначала создаст, а затем удалит из каталога очереди почты следующие файлы:

Элемент	Описание
`dfAA00269`	Файл данных
`qfAA00269`	Управляющий файл
`tfAA00269`	Временный файл
`xfAA00269`	Файл протокола

Управляющий файл q:

Управляющий файл `q` состоит из серии строк, начинающихся с кодовой буквы.

Элемент	Описание
B	Задает `тип содержимого` сообщения. Оставшаяся часть - это текст, определяющий `тип содержимого` сообщения. Если данное поле отсутствует, то предполагается, что `тип содержимого` 7-разрядный, и какая-либо специальная обработка не производится. Возможные значения - **7BIT** и **8BITMIME**.
C	Содержит ИД управляющего пользователя. Если адрес получателя указывает на файл или программу, программа **sendmail** выполняет доставку от имени владельца файла или программы. Управляющим пользователем становится владелец файла или программы. То же значение задается и для адресов получателей, полученных из файлов **.forward** и **:include:**. Данным получателям программа **sendmail** доставляет почту от имени управляющего пользователя, а затем снова переключается на профайл пользователя root.
F	Содержит флаги конверта. Флаги могут состоять из различных комбинаций символов **w**, устанавливающего флаг **EF_WARNING**; **r**, устанавливающего флаг **EF_RESPONSE**; **8**, устанавливающего флаг **EF_HAS8BIT**; и **b**, устанавливающего флаг **EF_DELETE_BCC**. Другие буквы игнорируются.
H	Содержит определение заголовка. Число таких строк не ограничено. Порядок строк **H** определяет и их расположение в итоговом сообщении. Эти строки задаются в том же формате, что и определения заголовков в файле конфигурации `/etc/mail/sendmail.cf`.
I	Указывает i-узел и устройство файла `df`. Эта информация применяется для восстановления очереди сообщений после поломки жесткого диска.
K	Содержит время последней попытки доставки (продолжительность в секундах).
M	При помещении сообщения в очередь из-за ошибки при его доставке, информация об ошибке заносится в строку **M**.
N	Показывает общее число попыток доставки.
O	Указывает исходное значение MTS (Система передачи сообщений) для ESMTP. Оно применяется только в извещениях о доставке.
P	Указывает приоритет текущего сообщения. В соответствии со значениями приоритета происходит упорядочивание почтовой очереди. Большее число означает меньший приоритет. Чем дольше находится в очереди сообщение, тем выше становится его приоритет. Исходный приоритет зависит от класса сообщения и его размера.
Q	Указывает исходного получателя, который задан в поле `ORCPT=` транзакции ESMTP. Используется исключительно для извещений о доставке. Данная строка относится только к следующей за ней строке **R**.
R	Содержит имя получателя. Для каждого получателя существует отдельная строка.
S	Указывает адрес отправителя. Существует только одна такая строка.
T	Здесь указано время создания сообщения, необходимое для расчета времени его отмены.
V	Содержит номер версии формата файла почтовой очереди. Эта информация необходима для обеспечения совместимости новых версий команды **sendmail** с ранее созданными файлами очереди. По умолчанию применяется значение **ноль**. Если оно указано, значение должно находиться в первой строке данного файла.
Z	Указывает исходный ИД сообщения, указанный в транзакции ESMTP. Используется исключительно для извещений о доставке.
$	Содержит макрокоманду. Некоторые макрокоманды (**$r** и **$s**) используются на стадии передачи очереди сообщений.

Файл q сообщения, отправленного пользователю `amy@zeus`, будет выглядеть примерно так:

```
P217031
T566755281
MDeferred: Connection timed out during user open with zeus
Sgeo
Ramy@zeus
H?P?return-path: <geo>
Hreceived: by george (0.13 (NL support)/0.01)
        id AA00269; Thu, 17 Dec 87 10:01:21 CST
H?D?date: Thu, 17 Dec 87 10:01:21 CST
H?F?From: geo
Hmessage-id: <8712171601.AA00269@george>
HTo: amy@zeus
Hsubject: test
```

Здесь:

Элемент	Описание
P217031	Приоритет сообщения
T566755281	Время отправки (в секундах)
MDeferred: Connection timed out during user open with zeus	Состояние сообщения
Sgeo	ИД отправителя
Ramy@zeus	ИД получателя
H *lines*	Данные заголовка сообщения

Единицы времени для sendmail

Для задания тайм-аута сообщений и интервала обработки используется специальный формат времени.

Вот пример представления значения времени в данном формате:

-q*ЧислоЕдиницы*

- где *Число* - это целое значение, а *Единицы* - символ единицы измерения времени. Поле *Единицы* может принимать одно из следующих значений:

Элемент	Описание
s	Секунды
m	Минуты
h	Часы
d	Дни
w	Недели

Если поле *Единицы* не задано, то демон **sendmail** по умолчанию исчисляет вермя в минутах (**m**). Ниже приведены три примера того, как задаются значения времени:

```
/usr/sbin/sendmail -q15d
```

Это команда указывает, что демон **sendmail** должен обрабатывать почту каждые 15 дней.

```
/usr/sbin/sendmail -q15h
```

Это команда указывает, что демон **sendmail** должен обрабатывать почту каждые 15 часов.

```
/usr/sbin/sendmail -q15
```

Это команда указывает, что демон **sendmail** должен обрабатывать почту каждые 15 минут.

Заблокированные почтовые очереди

Иногда почтовая очередь по каким-либо причинам оказывается заблокированной. Вы можете принудительно снять блокировку, указав флаг **-q** (без значения).

Для отслеживания происходящих событий можно установить флаг **-v** (подробный режим):

```
/usr/sbin/sendmail -q -v
```

Кроме того, с помощью одного из этих параметров можно настроить команду так, что будут обработаны только сообщения с конкретным идентификатором очереди, сообщения конкретного отправителя или получателя. Например, команда **-qRsally** ограничивает обработку сообщений только теми, у которых в одном из адресов получателя есть последовательность `sally`. Аналогично, команда **-qS***строка* разрешает обрабатывать только сообщения определенных отправителей, а команда **-qI** *строка* - только сообщения с определенным идентификатором очереди.

Настройка интервала обработки очереди

Очередь сообщений обрабатывается демоном **sendmail** через некоторый интервал времени, который задается с флагом **-q** при запуске демона.

Демон **sendmail** обычно запускается файлом `/etc/rc.tcpip` при запуске системы. В файле `/etc/rc.tcpip` предусмотрена переменная Интервал обработки очереди (QPI), которая задает значение флага **-q** в команде запуска демона **sendmail**. По умолчанию значение **qpi** равно 30 минутам. Для того чтобы задать другой интервал обработки очереди, выполните следующие действия:

1. Измените файл `/etc/rc.tcpip` с помощью любого текстового редактора.
2. Найдите строку, в которой задается значение переменной *qpi*, например:
   ```
   qpi=30m
   ```
3. Измените значение переменной *qpi*.

Внесенные изменения вступят в силу после перезапуска системы. Если необходимо, чтобы изменения вступили в силу немедленно, завершите работу программы-демона **sendmail** и перезапустите его сразу после того, как измените значение флага **-q**. Дополнительная информация приведена в разделах “Завершение работы демона sendmail” на стр. 53 и “Запуск демона sendmail”.

Перемещение почтовой очереди

При выключении хоста на длительное время в очереди скапливается большое число сообщений, предназначенных для хоста или пересылаемых через этот хост. В результате команда sendmail очень долго сортирует сообщения в очереди, что приводит к значительному снижению производительности системы. Если переместить очередь во временный каталог и создать новую очередь, то обработку очереди можно будет выполнить позже, когда хост вновь заработает.

Для этого выполните следующие действия:

1. Завершите работу демона **sendmail**, следуя инструкциям из раздела “Завершение работы демона sendmail” на стр. 53.
2. Переместите весь каталог очереди, введя следующую команду:
   ```
   cd /var/spool
   mv mqueue omqueue
   ```
3. Перезапустите демон **sendmail**, следуя инструкциям из раздела “Запуск демона sendmail”.
4. Обработайте старую очередь сообщений с помощью следующей команды:
   ```
   /usr/sbin/sendmail -oQ/var/spool/omqueue -q
   ```

   Флаг **-oQ** позволяет задать альтернативный каталог очереди. Флаг **-q** задает обработку всех заданий в очереди. Если вы хотите получать информацию о ходе выполнения операции, укажите флаг **-v**.

   Примечание: Выполнение этой операции может занять достаточно много времени.
5. После очистки очереди удалите файлы протокола и временный каталог с помощью следующей команды:
   ```
   rm /var/spool/omqueue/*
   rmdir /var/spool/omqueue
   ```

Запуск демона sendmail

Для запуска демона **sendmail** применяется две команды.

Запустить демон **sendmail** можно с помощью любой из следующих команд:

```
startsrc -s sendmail -a "-bd -q15"
/usr/lib/sendmail -bd -q15
```

Если одна из этих команд будет вызвана, когда демон **sendmail** активен, то появится следующее сообщение:

```
Подсистема sendmail уже активна. Запуск нескольких экземпляров не поддерживается.
```

Если демон **sendmail** еще не активен, то появится сообщение о том, что он запущен.

Завершение работы демона sendmail

Для завершения работы демона **sendmail** введите команду **stopsrc -s sendmail**.

Если демон **sendmail** был запущен без помощи команды **startsrc**, то выполните следующие действия:

- Определите идентификатор процесса **sendmail**.
- Введите команду **kill** *sendmail_pid* (где *sendmail_pid* - идентификатор процесса **sendmail**).

Ведение протоколов почты

Команда **sendmail** ведет протокол системы доставки почты с помощью демона **syslogd**.

Следовательно, для ведения протокола необходимо настроить и запустить демон **syslogd**. Файл `/etc/syslog.conf` должен содержать следующую строку:

```
mail.debug          /var/spool/mqueue/log
```

Если она отсутствует, добавьте эту строку в любом текстовом редакторе; убедитесь в правильности указанного пути. Если вы изменили файл `/etc/syslog.conf` после запуска демона **syslogd**, обновите демон **syslogd** с помощью следующей команды:

```
refresh -s syslogd
```

Если файл `/var/spool/mqueue/log` отсутствует, создайте его с помощью следующей команды:

```
touch /var/spool/mqueue/log
```

Сообщения в файле протокола хранятся в следующем формате:

Запись системного протокола содержит системное время, имя компьютера, создавшего запись (в случае ведения общего протокола для нескольких компьютеров локальной сети), слово `sendmail:` и текст сообщения. Большинство сообщений представляют собой последовательность пар *имя=значение*.

При обработке сообщения в протокол обычно заносятся строки **receipt** и **delivery attempt**. Строка **receipt** регистрирует получение сообщения; для каждого сообщения создается одна такая запись. Некоторые поля могут отсутствовать. К таким полям относятся:

Элемент	Описание
`from`	Указывает адрес отправителя сообщения.
`размер`	Указывает размер сообщения в байтах.
`класс`	Указывает класс (числовой приоритет) сообщения.
`pri`	Указывает исходный приоритет (необходим для сортировки очереди сообщений)
`nrcpts`	Указывает количество получателей данного сообщения (после присвоения псевдонима и пересылки).
`proto`	Задает протокол, применявшийся для получения сообщения, например, ESMTP или Программа копирования UNIX-UNIX (UUCP).
`relay`	Указывает, от какого компьютера было получено данное сообщение.

Кроме того, в протокол заносятся строки **delivery attempt**. Их количество совпадает с числом попыток доставки сообщения (таким образом, если доставка была отложена, или сообщение должно быть доставлено нескольким получателям, то может быть создано несколько таких строк). К таким полям относятся следующие:

Элемент	Описание
to	Здесь через запятую перечислены все получатели данной рассылки.
ctladdr	Указывает *управляющего пользователя*, то есть имя пользователя, которое используется для доставки.
delay	Указывает общую продолжительность задержки (промежуток между доставкой и получением сообщения).
xdelay	Указывает время, затраченное при данной попытке доставки.
mailer	Здесь указано имя почтовой программы, через которую сообщение было доставлено данному получателю.
relay	Указывает имя хоста, который принял (или отказался принять) почту для данного получателя.
stat	Здесь указано состояние доставки.

Поскольку в протокол может заноситься большой объем различной информации, существует несколько уровней ведения протокола. На первом (или низшем) уровне в протокол заносится информация только о наиболее нестандартных ситуациях. Высший уровень регистрирует даже самые незначительные события. Наиболее полезная информация заносится в протокол на уровнях 10 и ниже. Протоколы уровня 64 и выше предназначены для задач отладки. Уровни от 11 до 64 зарезервированы для подробных данных.

Типы действий, информацию о которых будет заносить в протокол команда **sendmail**, задаются с помощью опции **L** в файле /etc/mail/sendmail.cf.

Управление протоколами

Поскольку в файл протокола постоянно добавляется информация, его размер быстро растет. Кроме того, в случае ошибок в почтовую очередь могут заноситься самые неожиданные записи. Для того чтобы размер файла протокола и очереди почтовых сообщений не становился слишком большим, периодически запускайте сценарий оболочки /usr/lib/smdemon.cleanu.

Этот сценарий принудительно обрабатывает очередь сообщений с помощью команды **sendmail** и поддерживает четыре копии протокола в файлах log.0, log.1, log.2 и log.3, каждая из которых старше предыдущей. При запуске сценария выполняются следующие операции копирования:

- log.2 в log.3
- log.1 в log.2
- log.0 в log.1
- log в log.0

После выполнения этого сценария записи протокола начинают записываться в новый файл. Периодически вызывайте этот сценарий вручную, либо задайте интервал запуска сценария в файле конфигурации демона **cron**.

Протоколы потоков данных

С помощью флага **-X** команды **sendmail** можно включить запись протоколов потока данных.

Многие реализации **SMTP** являются неполными. Например, некоторые протоколы **SMTP** для персональных компьютеров не поддерживают строки продолжения в кодах ответа. Поэтому их очень сложно отслеживать. В этом случае можно включить ведение протокола потока данных с помощью флага **-X**. Например:

```
/usr/sbin/sendmail -X /tmp/traffic -bd
```

Эта команда создает протокол потока данных в файле /tmp/traffic.

Поскольку такая команда заносит в протокол очень большой объем информации, ее не следует вызывать во время обычной работы системы. После запуска этой команды отправьте сообщение на свой хост с помощью программы **errant**. В файле будет зарегистрирован весь входящий и исходящий поток данных, обрабатываемых командой **sendmail**, включая и данные, передаваемые по протоколу **SMTP**.

С помощью команды **sendmail** в протокол можно записать дамп открытых файлов и кэша соединений. Для этого ей нужно отправить сигнал **SIGUSR1**. Данные будут записаны в протокол с приоритетом **LOG_DEBUG**.

Протокол статистики почты

Команда **sendmail** отслеживает объем почты, обрабатываемой каждой из почтовых программ, с которыми она взаимодействует.

Почтовые программы заданы в файле `/etc/mail/sendmail.cf`.

На рисунке приведена схема, в вершине которой расположены записи Почта и MH. Ниже расположены

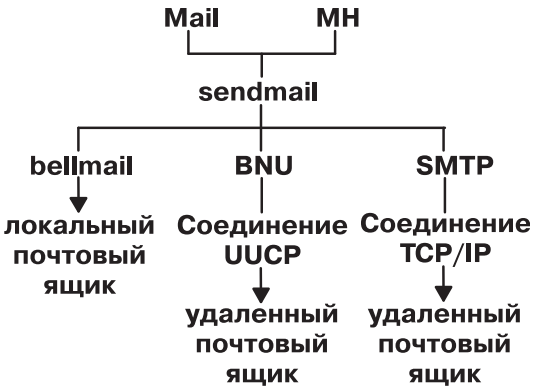


Рисунок 3. Почтовые программы, применяемые командой sendmail

записи bellmail, BNU и SMTP. Под предыдущим уровнем расположены локальный почтовый ящик, соединение UUCP и соединение TCP/IP, соответственно. Под соединениями UUCP и TCP/IP расположены удаленные почтовые ящики.

Для сбора статистических данных о работе почтовой программы создайте файл `/etc/mail/statistics` с помощью следующей команды:

```
touch /etc/mail/statistics
```

Если во время записи статистических данных команда **sendmail** сталкивается с ошибкой, она записывает сообщение с помощью подпрограммы **syslog**. Эти ошибки никак не влияют на остальные операции команды **sendmail**.

Команда **sendmail** обновляет информацию каждый раз, когда она обрабатывает почту. При этом увеличивается не размер файла, а число файлов. Файл содержит данные об объеме обработанной почты со времени создания или очистки файла `/etc/mail/statistics`.

Просмотр сведений программы почты

Статистические данные хранятся в файле `/etc/mail/statistics` в виде базы данных, поэтому их невозможно просмотреть с помощью средств работы с текстовыми файлами.

Для просмотра статистических данных о работе почтовой программы введите следующую команду:

```
/usr/sbin/mailstats
```

Эта команда считывает данные из файла `/etc/mail/statistics.st`, форматирует их и записывает в стандартный вывод. Формат вывода команды **/usr/sbin/mailstats** описан в книге *Справочник по командам, том 3*.

API почтового фильтра команда sendmail

API почтового фильтра команды sendmail (называемые *Milter*) позволяют программам других фирм обращаться к обрабатываемым сообщениям электронной почты для фильтрации мета-информации и содержимого.

Требования к фильтрам sendmail

Так как фильтры используют нити, они должны быть безопасными для нитей. Чтобы обеспечить совместимость фильтров в нитями можно настроить их.

Многие операционные системы поддерживают нити POSIX в стандартных библиотеках C. Флаг компилятора для ссылки для поддержки нитей может различаться в зависимости от используемого компилятора и компоновщика. Если вы не уверены в используемом локальном флаге, проверьте `Makefile` в соответствующем подкаталоге `obj.*/libmilter`.

Примечание: Так как фильтры используют нити, может потребоваться изменить ограничения процесса для фильтра. Например, вы можете использовать setrlimit для повышения числа открытых дескрипторов файлов если фильтр будет занят; иначе почта будет отклонена.

Настройка фильтра sendmail

Воспользуйтесь этими инструкциями чтобы задать фильтры при настройке **sendmail**.

Укажите фильтры с помощью ключевой клавиши X (внешний). В следующем примере заданы три фильтра:

```
Xfilter1, S=local:/var/run/f1.sock, F=R
Xfilter2, S=inet6:999@localhost, F=T, T=C:10m;S:1s;R:1s;E:5m
Xfilter3, S=inet:3333@localhost
```

Можно также указать фильтры в файле .mc с помощью следующего синтаксиса:

```
INPUT_MAIL_FILTER(`filter1', `S=local:/var/run/f1.sock, F=R')
INPUT_MAIL_FILTER(`filter2', `S=inet6:999@localhost, F=T, T=C:10m;S:1s;R:1s;E:5m')
INPUT_MAIL_FILTER(`filter3', `S=inet:3333@localhost')
```

где filter(*номер*) - это имя фильтра. В первой строке указывается, что фильтр для подключения к сокету в домене UNIX находится в каталоге `/var/run`. Во второй строке указывается, что фильтр использует сокет IPv6 и порт 999 локального хоста. В третьей строке указывается, что фильтр использует сокет IPv4 и порт 3333 локального хоста.

`F=` указывает какой из флагов применяется:

Элемент	Описание
R	Отклоняет соединение при недоступности фильтра.
T	Временно отменяет соединение при недоступности фильтра.

Если флаг не указан, сообщение передается через **sendmail** как будто фильтр отсутствует.

Указав значение для `T=`, вы сможете использовать фильтры для переопределения тайм-аутов по умолчанию, используемых **sendmail**. `T=` использует следующие поля:

Элемент	Описание
C	Тайм-аут соединения с фильтром (если 0, используется системный тайм-аут).
S	Тайм-аут отправки сведений от MTA в фильтр.
R	Тайм-аут чтения ответов от фильтра.
E	Общий тайм-аут между отправкой уведомлений окончания сообщений в фильтр и ожидания окончательного подтверждения.

Как следует из приведенного выше примера, разделителями между тайм-аутами являются точки с запятой (`;`), а разделителями между псевдонимами являются запятые (`,`).

Значениями по умолчанию для тайм-аутов являются:

```
T=C:0m;S:10s;R:10s;E:5m
```

где `s` означает секунду, а `m` - минуту.

Опция **InputMailFilters** определяет, каке фильтры используются и в какой последовательности.

Примечание: Если опция **InputMailFilters** не задана, фильтры не используются.
Опция **InputMailFilters** задается автоматически исходя из порядка расположения команд INPUT_MAIL_FILTER в файле `.mc`. Вы можете сбросить это значение, указав значение confINPUT_MAIL_FILTERS в файле `.mc`. Например, если опция **InputMailFilters** задана как:

```
InputMailFilters=фильтр1, фильтр2, фильтр3
```

три фильтра будут вызываться в указанном порядке.

Применив MAIL_FILTER() вместо INPUT_MAIL_FILTER() в файле `.mc` можно задать фильтр без добавления его в список входных фильтров.

Функции управления библиотекой

Фильтр sendmail вызывает функции управления библиотекой для настройки параметров **libmilter** перед передачей управления в **libmilter**. Параметры **libmilter** устанавливаются в вызовах функций **smfi_main**. Фильтр также вызывает функцию **smfi_register** для регистрации своих функций обратного вызова. Каждая функция возвращает значение MI_SUCCESS или MI_FAILURE, которое указывает на состояние операции. Эти функции не обмениваются данными с почтовой программой (MTA), но изменяют состояние библиотеки, которое передается в MTA внутри функции **smfi_main**.

Таблица 1. Функции управления библиотекой

Элемент	описание
smfi_opensocket	**Функция smfi_opensocket создает интерфейсный сокет.**
smfi_register	Функция **smfi_register** регистрирует фильтр.
smfi_setconn	Функция **smfi_setconn** указывает, какой сокет следует использовать.
smfi_settimeout	Функция **smfi_settimeout** устанавливает тайм-аут.
smfi_setbacklog	Функция **smfi_setbacklog** задает размер входящего буфера для функции **listen (2)**.
smfi_setdbg	Функция **smfi_setdbg** задает уровень отладки (трассировки) для библиотеки **milter**.
smfi_stop	Функция **smfi_stop** запускает процедуры выключения.
smfi_main	Функция **smfi_main** передает управление в **libmilter**.

Функция smfi_opensocket:
Назначение

Функция **smfi_opensocket** создает интерфейсный сокет, по которому почтовая программа подключается к фильтру.

Синтаксис

```
#include <libmilter/mfapi.h>
int smfi_opensocket(
bool rmsocket
);
```

Описание

Функция **smfi_opensocket** вызывается только из основного кода программы после вызова функций **smfi_setconn** и **smfi_register**, но до вызова функции **smfi_main**. Функция **smfi_opensocket** создает сокет, ранее указанный при вызове функции **smfi_setconn** и работающий как интерфейсный между MTA и фильтром. Функция **smfi_opensocket** позволяет вызывающему приложению создать сокет. Если функция **smfi_opensocket** не вызывается явно, то ее неявно вызывает функция **smfi_main**.

Аргументы

Таблица 2. Аргументы

Элемент	Описание
rmsocket	Этот флаг указывает, что библиотека должна попытаться удалить сокет UNIX, если он существует, прежде чем создавать новый сокет.

Коды возврата

Функция **smfi_opensocket** возвращает значение MI_FAILURE в следующих случаях. В противном случае функция возвращает MI_SUCCESS.

- Интерфейсный сокет не создан.
- Значение *rmsocket* равно true, и либо невозможно проанализировать сокет, либо невозможно удалить существующий сокет.
- Функция **smfi_setconn** или **smfi_register** не вызвана.

Связанная информация

“Функция smfi_register”

“Функция smfi_setconn” на стр. 61

Функция smfi_register:
Назначение

Функция **smfi_register** позволяет зарегистрировать ряд функций обратного вызова фильтра sendmail.

Синтаксис

```
#include <libmilter/mfapi.h>
int smfi_register(
smfiDesc descr)
);
```

Описание

Функция **smfi_register** создает фильтр sendmail с использованием информации, передаваемой в аргументе **smfiDesc**. Функция **smfi_register** может вызываться только до функции **smfi_main**.

Примечание: Несколько вызовов функции **smfi_register** в одном процессе запрещены. Разрешается зарегистрировать только один фильтр sendmail. Обратите внимание, что библиотека не имеет возможности проверить соблюдение этого ограничения.

Поле xxfi_flags должно содержать ноль или побитовую сумму следующих величин, описывающих возможные действия фильтра.

Таблица 3. Значения

Элемент	Описание
SMFIF_ADDHDRS	Функция **smfi_addheader** добавляет заголовки.
SMFIF_CHGHDRS	Функция **smfi_chgheader** модифицирует или удаляет заголовки.
SMFIF_CHGBODY	Функция **smfi_replacebody** заменяет тело сообщения. Этот фильтр существенно влияет на производительность, если после него с телом сообщения работают другие фильтры.
SMFIF_ADDRCPT	Функция **smfi_addrcpt** добавляет получателя в сообщение.
SMFIF_ADDRCPT_PAR	Функция **smfi_addrcpt_par** добавляет получателя и аргументы ESMTP для сообщения.

Таблица 3. Значения (продолжение)

Элемент	Описание
SMFIF_DELRCPT	Функция **smfi_delrcpt** удаляет получателей из сообщения.
SMFIF_QUARANTINE	Функция **smfi_quarantine** позволяет поместить сообщение в карантин.
SMFIF_CHGFROM	Функция **smfi_chgfrom** изменяет оболочечный адрес отправителя (Mail From).
SMFIF_SETSYMLIST	Функция **smfi_setsymlist** отправляет набор обязательных символов (макросов).

Аргументы

Таблица 4. Аргументы

Элемент	Описание

descr	Дескриптор фильтра типа smfiDesc, описывающий функции фильтра. Структура содержит следующие члены:
	`struct smfiDesc`
	`{`
	` char  *xxfi_name; /* имя фильтра */`
	` int  xxfi_version; /* код версии`
	`-- не изменять */`
	` unsigned long xxfi_flags; /* флаги */`
	
	` /* фильтр информации о соединении */`
	` sfsistat (*xxfi_connect)(SMFICTX *,`
	`char`
	`*, _SOCK_ADDR *);`
	` /* фильтр команды SMTP HELO */`
	` sfsistat (*xxfi_helo)(SMFICTX *,`
	`char *);`
	` /* фильтр оболочки отправителя */`
	` sfsistat (*xxfi_envfrom)(SMFICTX *,`
	`char **);`
	` /* фильтр оболочки получателя */`
	` sfsistat (*xxfi_envrcpt)(SMFICTX *,`
	`char **);`
	` /* фильтр заголовка */`
	` sfsistat (*xxfi_header)(SMFICTX *,`
	`char *,`
	` char *);`
	` /* конец заголовка */`
	` sfsistat (*xxfi_eoh)(SMFICTX *);`
	` /* блок тела */`
	` sfsistat (*xxfi_body)(SMFICTX *,`
	`unsigned char`
	`*, size_t);`
	` /* конец сообщения */`
	` sfsistat (*xxfi_eom)(SMFICTX *);`
	` /* сообщение отменено */`
	` sfsistat (*xxfi_abort)(SMFICTX *);`
	` /* процедура очистки соединения */`
	` sfsistat (*xxfi_close)(SMFICTX *);`
	
	` /* фильтр любой нераспознанной или`
	`нереализованной команды */`
	` sfsistat (*xxfi_unknown)(SMFICTX`
	`*, const char *);`
	
	` /* фильтр команды SMTP DATA */`
	` sfsistat (*xxfi_data)(SMFICTX *);`
	
	` /* обратный вызов согласования */`
	` sfsistat (*xxfi_negotiate)(SMFICTX *,`
	`  unsigned long, unsigned long,`
	`unsigned long,`
	` unsigned long,`
	`  unsigned long *, unsigned long *,`
	`unsigned long *,`
	`unsigned long *`
	`);`
	`};`
	
	Значение NULL для любой функции обратного вызова указывает, что фильтр не обрабатывает данный тип информации и возвращает **SMFIS_CONTINUE**.

headerf	Заголовок - это непустая null-терминированная строка.
headerv	Добавляемое значение заголовка. Это может быть непустая null-терминированная строка или пустая строка.

Коды возврата

Функция **smfi_register** возвращает значение MI_FAILURE в следующих случаях. В противном случае функция возвращает MI_SUCCESS.

- Возникает ошибка выделения памяти.
- Несовместимая версия или недопустимые флаги.

Связанная информация

“Функция smfi_addheader” на стр. 71

“Функция smfi_chgheader” на стр. 73

“Функция smfi_replacebody” на стр. 79

“Функция smfi_addrcpt” на стр. 77

“Функция smfi_addrcpt_par” на стр. 77

“Функция smfi_delrcpt” на стр. 78

“Функция smfi_quarantine” на стр. 81

“Функция smfi_chgfrom” на стр. 76

“Функция smfi_setsymlist” на стр. 95

Функция smfi_setconn:
Назначение

Функция **smfi_setconn** задает сокет, посредством которого этот фильтр может обмениваться данными с **sendmail**.

Синтаксис

```
#include <libmilter/mfapi.h>
int smfi_setconn(
char *oconn;
);
```

Описание

Функция **smfi_setconn** может вызываться только до функции **smfi_main**.

При связи по сокетам UNIX или локальным сокетам фильтры нельзя запускать с правами root.

Права для сокетов UNIX или локальных сокетов должны быть `0600` (чтение и запись только для владельца) или `0660` (чтение и запись только для владельца и группы). Эти права доступа применяются, если для **sendmail** задана опция **RunAsUser**.

Права доступа для сокетов UNIX или локальных сокетов определяются по **umask**, для которой необходимо задать значение `007` или `077`. В операционных системах, где права доступа сокета не используются, например, в Solaris, сокет должен находиться в защищенном каталоге.

Аргументы

Таблица 5. Аргументы

Элемент	Описание

oconn	Адрес требуемого сокета. Адрес должен быть задан как null-терминированная строка в формате **протокол:адрес**:
	`* {unix|local}:/путь`
	`/к/файлу -- именованный конвейер.`
	`* inet:port`
	`@{имя-хоста|IP-адрес} -- сокет IPV4.`
	`* inet6:port`
	`@{имя-хоста|IP-адрес} -- сокет IPV6.`

Коды возврата

Функция **smfi_setconn** не возвращает ошибку, если задан недопустимый адрес. Однако функция **smfi_setconn** не создает сокет, если недостаточно памяти. Ошибка выявляется только в функции **smfi_main**.

Связанная информация

“Функция smfi_main” на стр. 64

Функция smfi_settimeout:
Назначение

Функция **smfi_settimeout** устанавливает значение тайм-аутов ввода-вывода фильтров.

Синтаксис

```
#include <libmilter/mfapi.h>
int smfi_settimeout((
int otimeout
);
```

Описание

Функция **smfi_settimeout** может вызываться только из функции **smfi_main**. Функция **smfi_settimeout** задает продолжительность в секундах ожидания параметром **libmilter** данных для чтения или записи от почтовой программы. После этого возникает тайм-аут.

Примечание: Если функция **smfi_settimeout** не вызывается, то используется значение по умолчанию 7210 секунд.

Аргументы

Таблица 6. Аргументы

Элемент	Описание
otimeout	Продолжительность ожидания параметром **libmilter** данных от MTA в секундах. Значение *otimeout* должно быть положительным. Если *otimeout* равен 0, то **libmilter** не ожидает ответа от MTA.

Коды возврата

Функция **smfi_settimeout** всегда возвращает значение MI_SUCCESS.

Связанная информация

smfi_main

Функция smfi_setbacklog:
Назначение

Функция **smfi_setbacklog** задает величину буфера для функции **listen(2)** фильтра.

Синтаксис

```
#include <libmilter/mfapi.h>
int smfi_setbacklog(
int obacklog
);
```

Описание

Функция **smfi_setbacklog** может вызываться только до функции **smfi_main**. Функция **smfi_setbacklog** задает величину входящего буфера для сокета, который используется буфером функции **listen(2)**. Если функция **smfi_setbacklog** не вызывается, то используется значение по умолчанию для операционной системы.

Аргументы

Таблица 7. Аргументы

Элемент	Описание
obacklog	Число входящих соединений, разрешенное для очереди обработки.

Коды возврата

Функция **smfi_setbacklog** возвращает MI_FAILURE, если аргументу *obacklog* присвоено значение, меньшее и равное нулю.

Связанная информация

“Функция smfi_main” на стр. 64

Функция smfi_setdbg:
Назначение

Функция **smfi_setdbg** задает уровень отладки (трассировки) для библиотеки **milter**.

Синтаксис

```
#include <libmilter/mfapi.h>
int smfi_setdbg(
 int level;
);
```

Описание

Функция **smfi_setdbg** задает внутренний уровень отладки (трассировки) для библиотеки **milter**, чтобы можно было трассировать код. Уровень 0 выключает отладку. Чем больше уровень, тем больше отладочной информации. Текущее наивысшее полезное значение равно 6.

Аргументы

Таблица 8. Аргументы

Элемент	Описание
level	Новый уровень отладки.

Коды возврата

Функция **smfi_setdbg** возвращает по умолчанию значение MI_SUCCESS.

Функция smfi_stop:
Назначение

Функция **smfi_stop** выключает **milter**. После этого вызова соединения не принимаются.

Синтаксис

```
#include <libmilter/mfapi.h>
int smfi_stop(void);
);
```

Описание

Функция **smfi_stop** может вызываться из функций обратного вызова или обработки ошибок в любое время. Функция **smfi_stop** запрещает новые соединения. Однако эта функция не ожидает закрытия существующих соединений (потоков). Эта функция приводит к возврату из функции **smfi_main** в вызывающую программу, которая может завершить работу или выполнить мягкий перезапуск.

Аргументы

Таблица 9. Аргументы

Элемент	Описание
void	Этот аргумент не принимает никаких значений.

Коды возврата

Функция **smfi_stop** возвращает значение SMFI_CONTINUE в следующих случаях:

- Внутренняя процедура останавливает работу библиотеки **milter**.
- Иная процедура останавливает работу библиотеки **milter**.
- Не удается остановить процесс, который требуется запустить.

Пример

```
int ret;
SMFICTX *ctx;
...
ret = smfi_addheader(ctx, "Content-Type",
"multipart/mixed;\n\tboundary=\"foobar\"");
```

Связанная информация

Функции обратного вызова

Функция smfi_main:
Назначение

Функция **smfi_main** передает управление в цикл обработки событий **libmilter**.

Синтаксис

```
#include <libmilter/mfapi.h>
int smfi_main(
);
```

Описание

Функция **smfi_main** вызывается по окончании инициализации фильтра.

Коды возврата

Функция **smfi_main** возвращает MI_FAILURE, если не удалось установить соединение. В противном случае функция возвращает MI_SUCCESS.

Ошибка может возникать вследствие различных причин, которые записываются в протокол. Например, передача неверного адреса в функцию **smfi_setconn** приводит к ошибке.

Связанная информация

“Функция smfi_setconn” на стр. 61

Функции доступа к данным

Функции доступа к данным вызываются из функций обратного вызова, определенных в фильтрах. Они применяются для получения информации о текущем соединении или сообщении.

Таблица 10. Функции доступа к данным

Элемент	Описание
smfi_getsymal	Функция **smfi_getsymal** возвращает значение символа.
smfi_getpriv	Функция **smfi_getpriv** получает указатель на частные данные.
smfi_setpriv	Функция **smfi_setpriv** устанавливает указатель на частные данные.
smfi_setreply	Функция **smfi_setreply** задает код ответа, который необходимо использовать.
smfi_setmlreply	Функция **smfi_setmlreply** задает многострочный ответ, который необходимо использовать.

Функция smfi_getsymval:
Назначение

Функция **smfi_getsymval** получает значение макроса **sendmail**.

Синтаксис

```
#include <libmilter/mfapi.h>
char* smfi_getsymval(
SMFICTX *ctx,
char *headerf,
char *symname
);
```

Описание

Функция **smfi_getsymval** вызывается из любой функции обратного вызова **xxfi_*** и добавляет заголовок в сообщение. Определение макроса зависит от вызываемой функции.

По умолчанию поддерживаются следующие макросы:

Таблица 11. Описание

Элемент	Описание
xxfi_connect	**daemon_name, if_name, if_addr, j, _**
xxfi_hello	**tls_version, cipher, cipher_bits, cert_subject, cert_issuer**
xxfi_envfrom	**i, auth_type, auth_authen, auth_ssf, auth_author, mail_mailer, mail_host, mail_addr**
xxfi_envrcpt	**rcpt_mailer, rcpt_host, rcpt_addr**
xxfi_data	Нет
xxfi_eoh	Нет
xxfi_eom	**msg_id**

Все макросы действуют начиная с момента их получения и до завершения соединения для функций **xxfi_connect**, **xxfi_hello**.

Все макросы действуют до конца сообщения для функций **xxfi_envfrom** и **xxfi_eom**.

Все макросы действуют для каждого получателя для функции **xxfi_envrcpt**.

Список макросов может изменить с помощью опций **confMILTER_MACROS_*** в файле **sendmail.mc**. Область действия таких макросов определяется при их настройке командой **sendmail**. Описание значений макросов приведено в документе *Sendmail Installation and Operation Guide*.

Аргументы

Таблица 12. Аргументы

Элемент	Описание
ctx	Скрытая структура контекста хранится в параметре **libmilter**.
symname	Имя макроса **sendmail**. Односимвольные макросы необязательно заключать в фигурные скобки ("{" и "}"), но макросы с большей длиной необходимо заключать в фигурные скобки, как в файле **sendmail.cf**.

Коды возврата

Функция **smfi_getsymval** возвращает значение макроса в виде null-терминированной строки. В противном случае функция **smfi_getsymval** возвращает NULL, если макрос не задан.

Связанная информация

“Функция обратного вызова xxfi_connect” на стр. 83

“Функция обратного вызова xxfi_helo” на стр. 84

“Функция обратного вызова xxfi_envfrom” на стр. 85

“Функция обратного вызова xxfi_envrcpt” на стр. 85

“Функция обратного вызова xxfi_data” на стр. 86

“Функция обратного вызова xxfi_eoh” на стр. 89

“Функция обратного вызова xxfi_eom” на стр. 90

Функция smfi_getpriv:
Назначение

Функция **smfi_getpriv** получает указатель на данные для этого соединения.

Синтаксис

```
#include <libmilter/mfapi.h>
void* smfi_getpriv(
SMFICTX *ctx
);
```

Описание

Функция **smfi_getpriv** может вызываться из любой функции обратного вызова **xxfi_***.

Аргументы

Таблица 13. Аргументы

Элемент	Описание
ctx	Скрытая структура контекста хранится в параметре **libmilter**.

Коды возврата

Функция **smfi_getpriv** возвращает указатель на частные данные, сохраненный вызовом функции **smfi_setpriv**. В противном случае функция **smfi_setpriv** возвращает NULL, если это значение не задано.

Связанная информация

“Функция smfi_setpriv”

Функция smfi_setpriv:
Назначение

Функция **smfi_setpriv** устанавливает указатель на частные данные для этого соединения.

Синтаксис

```
#include <libmilter/mfapi.h>
int smfi_setpriv
SMFICTX *ctx,
void *privatedata
();
```

Описание

Функция **smfi_setpriv** может вызываться из любой функции обратного вызова **xxfi_***. Она устанавливает указатель на частные данные для *ctx*.

Примечание: Для соединения хранится один указатель на частные данные. Если функция **smfi_setpriv** вызывается несколько раз с различными значениями, то предыдущие значения утрачиваются. Перед завершением работы фильтра необходимо освободить память, выделенную для частных данных, и присвоить указателю значение NULL.

Аргументы

Таблица 14. Аргументы

Элемент	Описание
ctx	Скрытая структура контекста хранится в параметре **libmilter**.
privatedata	Аргумент указывает на частные данные. Это значение возвращается в последующих вызовах функции **smfi_getpriv** с использованием *ctx*.

Коды возврата

Функция **smfi_setpriv** возвращает MI_FAILURE, если *ctx* указывает на недопустимый контекст. В противном случае функция возвращает MI_SUCCESS.

Связанная информация

“Функция smfi_setpriv” на стр. 67

Функция smfi_setreply:
Назначение

Функция **smfi_setreply** задает стандартные коды ответа протокола SMTP. Она работает только с кодами 4XX и 5XX.

Синтаксис

```
#include <libmilter/mfapi.h>
int smfi_setreply
SFICTX *ctx,
char *rcode,
char *xcode,
char *message
);
```

Описание

Функция **smfi_setreply** может вызываться из любой функции обратного вызова **xxfi_***, за исключением **xxfi_connect**. Функция **smfi_setreply** задает код ответа SMTP для соединения. Этот код используется для последующих ответов, возникающих вследствие действий, выполненных этим фильтром.

Значения, передаваемые в функцию **smfi_setreply**, не проверяются на соблюдение стандартов.

Аргумент *message* должен содержать только печатаемые символы. В противном случае поведение функции непредсказуемо. Например, символы CR или LF приводят к ошибке вызова, а символы '%' - к тому, что текст игнорируется.

Примечание: Если в параметре требуется использовать символ %, то он представляется строкой '%%', как для функции printf(3).

Коды ответов и их описания приведены в документах RFC 821 или 2821, а также RFC 1893 или 2034.

Если *rcode* равен 4XX, но для сообщения используется значение SMFI_REJECT, то настраиваемый ответ не применяется.

Если *rcode* равен 5XX, но для сообщения используется значение SMFI_TEMPFAIL, то настраиваемый ответ не применяется.

Примечание: В этих случаях в параметр **milter** возвращается ошибка. Параметр **Libmilter** игнорирует этот код ответа.

Если параметр **milter** возвращает значение SMFI_TEMPFAIL и устанавливает код ответа 421, то сервер SMTP завершает сеанс SMTP с кодом 421.

Аргументы

Таблица 15. Аргументы

Элемент	Описание
ctx	Скрытая структура контекста хранится в параметре **libmilter**.
rcode	Трехзначный код ответа SMTP (RFC 821 или 2821) в виде null-терминированной строки. Аргумент *rcode* не может быть равен NULL, это должен быть допустимый код 4XX или 5XX.
xcode	Расширенный код ответа (RFC 1893 или 2034). Если *xcode* равен NULL, то расширенный код не используется. В противном случае *xcode* должен соответствовать требованиям RFC 1893 или 2034.
message	Текстовая часть ответа SMTP. Если message равен NULL, то используется пустое сообщение.

Коды возврата

Функция **smfi_setreply** возвращает значение MI_FAILURE в следующих случаях. В противном случае функция возвращает MI_SUCCESS.

- Недопустимый аргумент *rcode* или *xcode*.
- Ошибка выделения памяти.

Связанная информация

“Функция обратного вызова xxfi_connect” на стр. 83

Функция smfi_setmlreply:
Назначение

Функция **smfi_setmlreply** задает многострочные ответы для стандартных кодов ошибок протокола SMTP. Функция **smfi_setmlreply** работает только с кодами 4XX и 5XX .

Синтаксис

```
#include <libmilter/mfapi.h>
int smfi_setmlreply(
SMFICTX *ctx,
char *rcode,
char *xcode,
...
);
```

Описание

Функция **smfi_setmlreply** может вызываться из любой функции обратного вызова **xxfi_***, за исключением **xxfi_connect** . Функция **smfi_setmlreply** задает код ответа SMTP для соединений, описанных ниже для *xcode*. Список аргументов должен быть null-терминированным. Этот код используется для последующих ответов, возникающих вследствие действий, выполненных этим фильтром.

Значения, передаваемые в функцию **smfi_setmlreply**, не проверяются на соблюдение стандартов.

Параметр message должен содержать только печатаемые символы, в противном случае поведение функции непредсказуемо. Например, символы CR или LF приводят к ошибке вызова, а символы '%' - к тому, что текст игнорируется.

Примечание: Если в параметре `message` требуется использовать символ %, то он представляется строкой '%%', как для функции **`printf(3)`**.

Коды ответов и их описания приведены в документах RFC 821 или 2821, а также RFC 1893 или 2034.

Если *rcode* равен `4XX`, но для сообщения используется значение SMFI_REJECT, то настраиваемый ответ не применяется.

Если *rcode* равен `5XX`, но для сообщения используется значение SMFI_TEMPFAIL, то настраиваемый ответ не применяется.

Примечание: В этих случаях в параметр **`milter`** возвращается ошибка, и параметр **`Libmilter`** игнорирует ошибку.

Если параметр **`milter`** возвращает значение SMFI_TEMPFAIL и устанавливает код ответа `421`, то сервер SMTP завершает сеанс SMTP с кодом `421`.

Аргументы

Таблица 16. Аргументы

Элемент	Описание
ctx	Скрытая структура контекста хранится в параметре **`libmilter`**.
rcode	Трехзначный код ответа SMTP (RFC 821 или 2821) в виде null-терминированной строки. Аргумент *rcode* не может быть равен NULL, это должен быть допустимый код `4XX` или `5XX`.
xcode	Расширенный код ответа (RFC 1893 или 2034). Если *xcode* равен NULL, то расширенный код не используется. В противном случае *xcode* должен соответствовать требованиям RFC 1893 или 2034
...	Остальные аргументы - это одиночные строки. Число аргументов, составляющих текст ответа SMTP, не должно превышать 32. Список должен быть null-терминированным.

Коды возврата

Функция **smfi_setmlreply** возвращает значение MI_FAILURE в следующих случаях. В противном случае функция возвращает MI_SUCCESS.

- Недопустимый аргумент *rcode* или *xcode*.
- Ошибка выделения памяти.
- Текстовая строка содержит возврат каретки или перенос строки.
- Длина какой-либо строки превышает `MAXREPLYLEN(980)`.
- Текст состоит из более чем 32 строк.

Пример

```
ret = smfi_setmlreply(ctx, "550", "5.7.0",
"Spammer access rejected",
"Please see our policy at:",
 "http://www.example.com/spampolicy.html",
NULL);
```

Этот пример приводит к следующему ответу:

```
550-5.7.0 Spammer access rejected
550-5.7.0 Please see our policy at:
550 5.7.0 http://www.example.com/spampolicy.html
```

Связанная информация

“Функция обратного вызова xxfi_connect” на стр. 83

Функции модификации сообщений

Функции модификации сообщений изменяют содержимое и атрибуты сообщений. Эти функции вызываются только функцией **xxfi_eom**. Функции модификации сообщений могут приводить к дополнительному обмену данными с почтовой программой (MTA). Каждая из этих функций возвращает значение MI_SUCCESS или MI_FAILURE, которое указывает на состояние операции.

Примечание: Данные сообщения (отправители, получатели, заголовки и фрагмента тела сообщения), передаваемые в функции модификации сообщений в параметрах, копируются и сохранять их не требуется. Выделенную память можно освободить.

Для вызова функции модификации сообщений фильтр должен установить соответствующий флаг в описании, который передается в функцию **smfi_register**. Если этот флаг не установлен, то MTA считает вызов функции ошибкой фильтра и прерывает соединение.

Примечание: Состояние, возвращаемое функцией, указывает, успешно ли передан фильтр сообщения в MTA. Это состояние не гарантирует выполнения запрошенной операции в MTA. Например, если функция **smfi_header** вызвана с недопустимым заголовком, она может возвратить флаг MI_SUCCESS, несмотря на то, что MTA впоследствии откажется добавить недопустимый заголовок.

Таблица 17. Функции изменения

Элемент	Описание	функция
smfi_addheader	Функция **smfi_addheader** добавляет заголовок в сообщение.	**SMFIF_ADDHDRS**
smfi_chgheader	Функция **smfi_chgheader** модифицирует или удаляет заголовок.	**SMFIF_CHGHDRS**
smfi_insheader	Функция **smfi_insheader** добавляет заголовок в начало текущего сообщения.	**SMFIF_ADDHDRS**
smfi_chgfrom	Функция **smfi_chgfrom** изменяет оболочечный адрес отправителя.	**SMFIF_CHGFROM**
smfi_addrcpt	Функция **smfi_addrcpt** добавляет получателя в оболочку сообщения.	**SMFIF_ADDRCPT**
smfi_addrcpt_par	Функция **smfi_addrcpt_par** добавляет получателя и параметр ESMTP в оболочку.	**SMFIF_ADDRCPT_PAR**
smfi_delrcpt	Функция **smfi_delrcpt** удаляет получателя из оболочки сообщения.	**SMFIF_DELRCPT**
smfi_replacebody	Функция **smfi_replacebody** заменяет тело сообщения.	**SMFIF_CHGBODY**

Функция smfi_addheader:
Назначение

Функция **smfi_addheader** добавляет заголовок в текущее сообщение.

Синтаксис

```
#include <libmilter/mfapi.h>
int smfi_addheader(
SMFICTX *ctx,
char *headerf,
char *headerv
);
```

Описание

Функция **smfi_addheader** вызывается из функции **xxfi_eom** и добавляет заголовок в сообщение.

Функция **smfi_addheader** не модифицирует существующие заголовки сообщения.

Для изменения значения заголовка используется функция **smfi_chgheader**.

Фильтр, вызывающий функцию **smfi_addheader**, должен установить флаг **SMFIF_ADDHDRS** в аргументе `smfiDesc_str`. Затем фильтр передает это значение в функцию **smfi_register**.

Для функции **smfi_addheader** требуется указать порядок фильтров. Просмотреть модификации заголовка можно с помощью ранее созданных фильтров.

Имя или значение заголовка не проверяется на соответствие стандартам. Однако длина каждой строки заголовка не должна превышать 998 символов. Если требуются заголовки большей длины, используйте многострочный заголовок. Для того чтобы создать многострочный заголовок, вставьте символ перевода строки (`ASCII 0x0a` или `\n` на языке программирования C), за которым следует пробельный символ, такой как пробел (`ASCII 0x20`) или табуляция (`ASCII 0x09` или `\t` на языке программирования C). Перед символом перевода строки нельзя вставлять символ возврата каретки (`ASCII 0x0d`). Почтовая программа (MTA) добавляет его автоматически. При написании фильтра необходимо следить за соблюдением стандартов.

MTA добавляет пробел перед добавленной строкой заголовка, если не установлен флаг **SMFIP_HDR_LEADSPC**. Если флаг установлен, то параметр **`milter`** должен самостоятельно вставить необходимые пробелы.

Аргументы

Таблица 18. Аргументы

Элемент	Описание
ctx	Скрытая структура контекста хранится в параметре **`libmilter`**.
headerf	Заголовок - это непустая null-терминированная строка.
headerv	Добавляемое значение заголовка. Это может быть непустая null-терминированная строка или пустая строка.

Коды возврата

Функция **smfi_addheader** возвращает значение MI_FAILURE в следующих случаях. В противном случае функция возвращает MI_SUCCESS.

- Аргумент *headerf* или *headerv* равен NULL.
- Добавление заголовков в данном состоянии соединения недопустимо.
- Возникает ошибка выделения памяти.
- Возникла ошибка сети.
- Флаг **SMFIF_ADDHDRS** не был установлен при вызове функции **smfi_register**.

Пример

```
int ret;
SMFICTX *ctx;
...
ret = smfi_addheader(ctx, "Content-Type",
"multipart/mixed;\n\tboundary=\"foobar\"");
```

Связанная информация

“Функция обратного вызова xxfi_eom” на стр. 90

“Функция smfi_chgheader”

“Функция smfi_register” на стр. 58

Функция smfi_chgheader:
Назначение

Функция **smfi_chgheader** модифицирует или удаляет заголовок сообщения.

Синтаксис

```
#include <libmilter/mfapi.h>
int smfi_chgheader(
SMFICTX *ctx,
char *headerf,
mi_int32 hdridx,
char *headerv
);
```

Описание

Функция **smfi_chgheader** вызывается из функции **xxfi_eom** и изменяет значение заголовка текущего сообщения.

Функцию **smfi_chgheader** можно использовать для добавления новых заголовков. Однако для этого безопаснее использовать функцию **smfi_addheader** .

Фильтр, вызывающий функцию **smfi_chgheader**, должен установить флаг **SMFIF_CHGHDRS** в аргументе `smfiDesc_str`. Затем фильтр передает это значение в функцию **smfi_register**.

Для функции **smfi_chgheader** требуется указать порядок фильтров. Просмотреть модификации заголовка можно с помощью ранее созданных фильтров.

Имя или значение заголовка не проверяется на соответствие стандартам. Однако длина каждой строки заголовка не должна превышать 998 символов. Если требуются заголовки большей длины, используйте многострочный заголовок. Для того чтобы создать многострочный заголовок, вставьте символ перевода строки (`ASCII 0x0a` или `\n` на языке программирования C), за которым следует пробельный символ, такой как пробел (`ASCII 0x20`) или табуляция (`ASCII 0x09` или `\t` на языке программирования C). Перед символом перевода строки нельзя вставлять символ возврата каретки (`ASCII 0x0d`), так как почтовая программа (MTA) добавляет его автоматически. При написании фильтра необходимо следить за соблюдением стандартов.

MTA добавляет пробел перед добавленной строкой заголовка, если не установлен флаг **SMFIP_HDR_LEADSPC**. Если флаг установлен, то параметр **`milter`** должен самостоятельно вставить необходимые пробелы.

Аргументы

Таблица 19. Аргументы

Элемент	Описание
ctx	Скрытая структура контекста хранится в параметре **libmilter**.
headerf	Заголовок - это непустая null-терминированная строка.
hdridx	Значение индекса заголовка (целое число). Если *hdridx* равен 1, то будет изменен первый заголовок с именем *headerf*. Если *hdridx* больше, чем число вхождений *headerf*, то добавляется дополнительная копия *headerf*.
headerv	Добавляемое значение заголовка. Это может быть непустая null-терминированная строка или пустая строка.

Коды возврата

Функция **smfi_chgheader** возвращает значение MI_FAILURE в следующих случаях. В противном случае функция возвращает MI_SUCCESS.

- Аргумент *headerf* равен NULL.
- Изменение заголовков в данном состоянии соединения недопустимо.
- Возникает ошибка выделения памяти.
- Возникла ошибка сети.
- Флаг **SMFIF_CHGHDRS** не был установлен при вызове функции **smfi_register**.

Пример

```
int ret;
SMFICTX *ctx;
...

ret = smfi_chgheader(ctx, "Content-Type", 1,
"multipart/mixed;\n\tboundary=\"foobar\"");
```

Связанная информация

“Функция обратного вызова xxfi_eom” на стр. 90

“Функция smfi_addheader” на стр. 71

Функция smfi_insheader:
Назначение

Функция **smfi_insheader** добавляет заголовок в начало текущего сообщения.

Синтаксис

```
#include <libmilter/mfapi.h>
int smfi_insheader(
SMFICTX ,
int hdridx,
char *headerf,
char *headerv
);
```

Описание

Функция **smfi_insheader** вызывается из функции **xxfi_eom** и добавляет заголовок в начало сообщения.

Функция **smfi_insheader** не модифицирует существующие заголовки сообщения.

Для изменения значения заголовка используется функция **smfi_chgheader**.

Фильтр, вызывающий функцию **smfi_insheader**, должен установить флаг **SMFIF_ADDHDRS** в аргументе `smfiDesc_str`, который передается в функцию **smfi_register**.

Для функции **smfi_insheader** требуется указать порядок фильтров. Просмотреть модификации заголовка можно с помощью ранее созданных фильтров.

Фильтр получает заголовки, отправляемые клиентом SMTP, и заголовки, измененные предыдущими фильтрами. Он не получает заголовки, вставленные командой **sendmail** или самим этим фильтром. Заголовок вставляется в позиции, которая зависит от уже существующих заголовков сообщения и заголовков, вставка которых настроена для команды **sendmail**.

Например, **sendmail** всегда добавляет заголовок **Received:** в начало сообщения. Если *hdridx* равен `0`, то заголовок будет вставляться перед **Received:**. Однако если последующие фильтры получают вставленный заголовок вместо заголовка **Received:**, они могут работать неправильно, и поэтому вставка заголовка в фиксированной позиции затруднена.

Если значение *hdridx* превышает число заголовков в сообщении, заголовок добавляет последним.

Имя или значение заголовка не проверяется на соответствие стандартам. Однако длина каждой строки заголовка не должна превышать 998 символов. Если требуются заголовки большей длины, используйте многострочный заголовок. Для того чтобы создать многострочный заголовок, вставьте символ перевода строки (`ASCII 0x0a` или `\n` на языке программирования C), за которым следует пробельный символ, такой как пробел (`ASCII 0x20`) или табуляция (`ASCII 0x09` или `\t` на языке программирования C). Перед символом перевода строки нельзя вставлять символ возврата каретки (`ASCII 0x0d`). Почтовая программа (MTA) добавляет его автоматически. При написании фильтра необходимо следить за соблюдением стандартов.

MTA добавляет пробел перед вставленной строкой заголовка, если не установлен флаг **SMFIP_HDR_LEADSPC**. Если флаг установлен, то параметр **`milter`** должен самостоятельно вставить необходимые пробелы.

Аргументы

Таблица 20. Аргументы

Элемент	Описание
ctx	Скрытая структура контекста хранится в параметре **`libmilter`**.
headerf	Заголовок - это непустая null-терминированная строка.
headerv	Добавляемое значение заголовка. Это может быть непустая null-терминированная строка или пустая строка.

Коды возврата

Функция **smfi_insheader** возвращает значение MI_FAILURE в следующих случаях. В остальных случаях функция возвращает MI_SUCCESS.

- Аргумент *headerf* или *headerv* равен NULL.
- Добавление заголовков в данном состоянии соединения недопустимо.
- Возникает ошибка выделения памяти.
- Возникла ошибка сети.
- Флаг **SMFIF_ADDHDRS** не был установлен при вызове функции **smfi_register**.

Пример

```
int ret;
SMFICTX *ctx;
...
ret = smfi_insheader( ctx, 0, "First", "See me?");;
```

Связанная информация

“Функция обратного вызова xxfi_eom” на стр. 90

“Функция smfi_register” на стр. 58

“Функция smfi_chgheader” на стр. 73

Функция smfi_chgfrom:
Назначение

Функция **smfi_chgfrom** изменяет оболочечный адрес отправителя (MAIL From) для текущего сообщения.

Синтаксис

```
#include <libmilter/mfapi.h>
int smfi_chgfrom(
SMFICTX *ctx,
const char *mail,
char *args
);
```

Описание

Функция **smfi_chgfrom** вызывается из функции **xxfi_eom** и изменяет оболочечный адрес отправителя и значение MAIL From для текущего сообщения.

Фильтр, вызывающий функцию **smfi_chgfrom**, должен установить флаг **SMFIF_CHGFROM** в аргументе `smfiDesc_str`. Затем фильтр передает это значение в функцию **smfi_register**.

В вызове можно задать все аргументы протокола ESMTP. Однако некоторые аргументы, такие как `SIZE` и `BODY`, могут привести к неполадкам. Поэтому задавать эти аргументы необходимо с осторожностью. Почтовая программа (MTA) не сообщает в **`milter`**, был ли вызов выполнен успешно.

Аргументы

Таблица 21. Аргументы

Элемент	Описание
ctx	Скрытая структура контекста хранится в параметре **`libmilter`**.
mail	Новый адрес отправителя.
args	Аргументы ESMTP.

Коды возврата

Функция **smfi_chgfrom** возвращает значение MI_FAILURE в следующих случаях. В противном случае функция возвращает MI_SUCCESS.

- Аргумент *mail* равен NULL.
- Изменение отправителя в данном состоянии соединения недопустимо.
- Возникла ошибка сети.
- Флаг **SMFIF_CHGFROM** не был установлен при вызове функции **smfi_register**.

Связанная информация

“Функция обратного вызова xxfi_eom” на стр. 90

“Функция smfi_register” на стр. 58

Функция smfi_addrcpt:
Назначение

Функция **smfi_addrcpt** добавляет получателя текущего сообщения.

Синтаксис

```
#include <libmilter/mfapi.h>
int smfi_addrcpt(
SMFICTX *ctx
char *rcpt
);
```

Описание

Функция **smfi_addrcpt** может вызываться только из функции **xxfi_eom**. Она добавляет получателя в оболочку сообщения.

Примечание: Фильтр, вызывающий функцию **smfi_addrcpt**, должен установить флаг **SMFIF_ADDRCPT** в структуре **smfiDesc_str**, которая передается в функцию **smfi_register**.

Аргументы

Таблица 22. Аргументы

Элемент	Описание
ctx	Скрытая структура контекста хранится в параметре **libmilter**.
rcpt	Новый адрес получателя.

Коды возврата

Функция **smfi_addrcpt** возвращает значение MI_FAILURE в следующих случаях. В противном случае функция возвращает MI_SUCCESS.

- Аргумент *rcpt* равен NULL.
- Добавление получателей в данном состоянии соединения недопустимо.
- Возникла ошибка сети.
- Флаг **SMFIF_ADDRCPT** не был установлен при вызове функции **smfi_register**.

Связанная информация

“Функция обратного вызова xxfi_eom” на стр. 90

“Функция smfi_register” на стр. 58

Функция smfi_addrcpt_par:
Назначение

Функция **smfi_addrcpt_par** добавляет получателя и аргументы ESMTP для текущего сообщения.

Синтаксис

```
#include <libmilter/mfapi.h>
int smfi_addrcpt_par(
SMFICTX *ctx,
char *rcpt,
char *args
);
```

Описание

Функция **smfi_addrcpt_par** может вызываться из функции **xxfi_eom**. Она добавляет получателя в оболочку сообщения.

Аргументы

Таблица 23. Аргументы

Элемент	Описание
ctx	Скрытая структура контекста хранится в параметре **libmilter**.
rcpt	Новый адрес получателя.
args	Параметры ESMTP новых получателей.

Коды возврата

Функция **smfi_addrcpt** возвращает значение MI_FAILURE в следующих случаях. В противном случае функция возвращает MI_SUCCESS.

- Аргумент *rcpt* равен NULL.
- Добавление получателей в данном состоянии соединения недопустимо.
- Возникла ошибка сети.
- Флаг **SMFIF_ADDRCPT_PAR** не был установлен при вызове функции **smfi_register**.

Связанная информация

“Функция smfi_addrcpt” на стр. 77

“Функция smfi_register” на стр. 58

Функция smfi_delrcpt:
Назначение

Функция **smfi_delrcpt** удаляет получателя из оболочки текущего сообщения.

Синтаксис

```
#include <libmilter/mfapi.h>
int smfi_delrcpt(
 SMFICTX *ctx;
 char *rcpt;
);
```

Описание

Функция **smfi_delrcpt** вызывается из функции обратного вызова **xxfi_eom** и позволяет удалить указанного получателя из оболочки текущего сообщения.

Примечание: Удаляемые адреса должны точно совпадать с заданными. Так, адрес и его расширенная форма не совпадают друг с другом.

Аргументы

Таблица 24. Аргументы

Элемент	Описание
ctx	Скрытая структура контекста хранится в параметре **libmilter**.
rcpt	Удаляемый адрес получателя. Это должна быть непустая null-терминированная строка.

Коды возврата

Функция **smfi_delrcpt** возвращает значение MI_FAILURE в следующих случаях. В противном случае функция возвращает MI_SUCCESS.

- Аргумент *rcpt* равен NULL.
- Удаление получателей в данном состоянии соединения недопустимо.
- Возникла ошибка сети.
- Флаг SMFIF_DELRCPT не был установлен при вызове функции **smfi_register**.

Связанная информация

smfi_register

xxfi_eom

Функция smfi_replacebody:
Назначение

Функция **smfi_replacebody** заменяет тело сообщения.

Синтаксис

```
#include <libmilter/mfapi.h>
int smfi_replacebody(
 SMFICTX *ctx,
 unsigned char *bodyp,
 int bodylen
);
```

Описание

Функция **smfi_replacebody** заменяет тело текущего сообщения. Если функция вызывается несколько раз, то последующие вызовы приводят к добавлению новых данных к телу сообщения. Функция может вызываться несколько раз.

Тело сообщения может быть большим, поэтому флаг SMFIF_CHGBODY может заметно влиять на быстродействие фильтра.

Если фильтр устанавливает флаг SMFIF_CHGBODY, но функция **smfi_replacebody** не вызывается, то исходное тело сообщения не изменяется.

Для функции **smfi_replacebody** важен порядок фильтров. Новое содержимое тела создается старыми фильтрами в новых файлах фильтров.

Аргументы

Таблица 25. Аргументы

Элемент	Описание
ctx	Скрытая структура контекста хранится в параметре **libmilter**.
bodyp	Указатель на начало новых данных для тела сообщения. Это необязательно должна быть null-терминированная строка. Если *bodyp* равен NULL, то эта ситуация обрабатывается, как для нулевой длины. Данные тела должны быть в формате CR или LF.
bodylen	Длина массива данных в байтах, на который указывает *bodyp*.

Коды возврата

Функция **smfi_replacebody** возвращает значение MI_FAILURE в следующих случаях. В противном случае функция возвращает MI_SUCCESS.

- *bodyp* == NULL и *bodylen* > 0
- Изменение тела в данном состоянии соединения недопустимо.
- Возникла ошибка сети.
- Флаг SMFIF_CHGBODY не был установлен при вызове функции **smfi_register**.

Связанная информация

smfi_register

Функции обработки сообщений

Функции обработки сообщений предоставляют специальные инструкции обработки параметру **milter** почтовой программы (MTA), при этом содержимого или статус сообщения не изменяются. Функции обработки сообщений могут вызываться только из функции **xxfi_eom**. Функция **xxfi_eom** может инициировать дополнительный обмен данными с MTA возвращает значение MI_SUCCESS или MI_FAILURE, которое указывает на состояние операции.

Примечание: Состояние, возвращаемое функцией, указывает, успешно ли фильтр передал сообщение в MTA. Это состояние не гарантирует выполнения запрошенной операции в MTA.

Таблица 26. Функция обработки сообщений

Элемент	Описание
smfi_progress	Функция **smfi_progress** сообщает о ходе выполнения операции.
smfi_quarantine	Функция **smfi_quarantine** позволяет поместить сообщение в карантин.

Функция smfi_progress:
Назначение

Функция **smfi_progress** сообщает о ходе выполнения операции.

Синтаксис

```
#include <libmilter/mfapi.h>
int smfi_progress(
 SMFICTX *ctx;
);
```

Описание

Функция **smfi_progress** вызывается из функции обратного вызова **xxfi_eom**. Она извещает почтовую программу о том, что фильтр все еще обрабатывает сообщение. Эта функция применяется для обновления тайм-аутов почтовой программы.

Аргументы

Таблица 27. Аргументы

Элемент	Описание
ctx	Скрытая структура контекста хранится в параметре **libmilter**.

Коды возврата

Функция **smfi_progress** возвращает MI_FAILURE, если возникла ошибка сети. В противном случае функция возвращает MI_SUCCESS.

Связанная информация

xxfi_eom

Функция smfi_quarantine:
Назначение

Функция **smfi_quarantine** позволяет поместить сообщение в карантин.

Синтаксис

```
#include <libmilter/mfapi.h>
int smfi_quarantine(
 SMFICTX *ctx;
 char *reason;
);
```

Описание

Функция **smfi_quarantine** вызывается из функции обратного вызова **xxfi_eom** и позволяет поместить сообщение в карантин с указанием причины.

Аргументы

Таблица 28. Аргументы

Элемент	Описание
ctx	Скрытая структура контекста хранится в параметре **libmilter**.
reason	Причина помещения в карантин, непустая null-терминированная строка.

Коды возврата

Функция **smfi_quarantine** возвращает значение MI_FAILURE в следующих случаях. В противном случае функция возвращает MI_SUCCESS.

- *reason* равен NULL или пуст.
- Возникла ошибка сети.
- Флаг SMFIF_QUARANTINE не был установлен при вызове функции **smfi_register**.

Связанная информация

smfi_register

xxfi_eom

Функции обратного вызова

Фильтр sendmail должен реализовывать хотя бы одну функцию обратного вызова, которые регистрируются с помощью функции **smfi_register**.

Таблица 29. Функции обратного вызова

Элемент	Описание
xxfi_connect	Функция **xxfi_connect** вызывается один раз в начале каждого соединения SMTP. Функция возвращает значение SMFIS_CONTINUE.
xxfi_hello	Функция **xxfi_hello** вызывается всякий раз, когда клиент отправляет команду HELO/EHLO.
xxfi_envfrom	Функция **xxfi_envfrom** вызывается в начале сообщения.
xxfi_envrcpt	Функция **xxfi_envrcpt** вызывается для каждого получателя.
xxfi_data	Функция **xxfi_data** обрабатывает команду DATA.
xxfi_unknown	Функция **xxfi_unknown** обрабатывает неизвестные команды протокола SMTP.
xxfi_header	Функция **xxfi_header** обрабатывает заголовок сообщения.
xxfi_eoh	Функция **xxfi_eoh** обрабатывает заголовки сообщения.
xxfi_body	Функция **xxfi_body** обрабатывает фрагмент тела сообщения.
xxfi_eom	Функция **xxfi_eom** обрабатывает конец сообщения.
xxfi_abort	Функция **xxfi_abort** обрабатывает сообщения, передача которых прервана.
xxfi_close	Функция **xxfi_close** вызывается для закрытия текущего соединения.
xxfi_negotiate	Функция **xxfi_negotiate** вызывается в начале соединения SMTP.

Функции обратного вызова должны правильно возвращать значение. Если функция обратного вызова возвращает значение, не входящее в число заранее определенных, то возникает ошибка, и команда **sendmail** прерывает соединение с фильтром.

Параметр **`Milter`** позволяет различить процедуры, ориентированные на **получателя**, **сообщение** и **соединение**:

- Функции обратного вызова, **ориентированные на получателя**, влияют на обработку сообщения для одного получателя.
- Функции обратного вызова, **ориентированные на сообщение**, влияют на обработку одного сообщения.
- Функции обратного вызова, **ориентированные на соединение**, действуют в течение всего соединения (в ходе которого может быть доставлено несколько сообщений различным группам получателей).
- Функция **xxfi_envrcpt** ориентирована на получателя. Функции **xxfi_conect**, **xxfi_hello** и **xxfi_close** ориентированы на соединение. Все прочие функции обратного вызова ориентированы на сообщение.

Таблица 30. Функции обратного вызова

Элемент	Описание
SMFIS_CONTINUE	Продолжить обработку для текущего соединения, сообщения или получателя.
SMFIS_REJECT	• Для процедур, **ориентированных на соединение**, прервать это соединение, затем вызвать **xxfi_close**. • Для процедур, **ориентированных на сообщение**, (за исключением функций **xxfi_eom** и **xxfi_abort**), отклонить это сообщение. • Для процедур, **ориентированных на получателя**, отклонить сообщение для данного получателя, но продолжить обработку сообщения.

Таблица 30. Функции обратного вызова (продолжение)

Элемент	Описание
SMFIS_DISCARD	• Для процедур, ориентированных на **сообщение** или **получателя**, принять сообщение и удалить его. • SMFIS_DISCARD не может возвращаться процедурами, ориентированными на **соединение**.
SMFIS_ACCEPT	• Для процедур, **ориентированных на соединение**, принять это соединение без его дальнейшей обработки фильтром, затем вызвать **xxfi_close**. • Для процедур, **ориентированных на сообщение или получателя**, принять это соединение без его дальнейшей обработки фильтром.
SMFIS_TEMPFAIL	Возвратить временную ошибку, то есть соответствующая команда SMTP возвращает код ошибки 4xx. • Для процедур, ориентированных на **сообщение**, за исключением функции **xxfi_envfrom**, ошибка относится к сообщению. • Для процедур, ориентированных на **соединение**, ошибка относится к соединению, после этого вызывается **xxfi_close**. • Для процедур, ориентированных на **получателя**, ошибка относится только к данному получателю, и обработка сообщения продолжается.
SMFIS_SKIP	Пропустить дальнейшие обратные вызовы того же типа в этой транзакции. В данный момент это значение может возвращать только функция **xxfi_body**. Это значение можно использовать, если для параметра **milter** передано достаточно фрагментов тела сообщения для принятия решения. Однако возможны случаи, когда для возвращаемого значения будут вызываться функции модификации сообщений, разрешенные только из функции **xxfi_eom**. **Примечание:** Параметр **milter** должен согласовать это поведение с почтовой программой (MTA). Параметр **milter** проверяет, доступно ли действие протокола SMFIP_SKIP. Если действие SMFIP_SKIP доступно, то параметр **milter** должен его потребовать.
SMFIS_NOREPLY	• Не отправлять ответ почтовой программе (MTA). Параметр **milter** должен согласовать это поведение с почтовой программой (MTA). Параметр **milter** должен проверить, доступно ли соответствующее действие протокола SMFIP_NR_*. Если действие SMFIP_NR_* доступно, то параметр **milter** должен его потребовать. • Если действие протокола SMFIP_NR_* определено с обратным вызовом, то обратный вызов должен всегда возвращать SMFIS_NOREPLY. Любой другой код ответа несовместим с API. Если в какой-либо ситуации, например, из-за недостатка ресурсов, обратный вызов может возвратить другой код, то не следует устанавливать SMFIP_NR_*, и необходимо использовать SMFIS_CONTINUE как код а по умолчанию. Вместо этого также можно отложить сообщение об ошибке для другой функции обратного вызова, для которой не установлен код SMFIP_NR_*.

Функция обратного вызова xxfi_connect: Назначение

Функция обратного вызова **xxfi_connect** предоставляет сведения о соединения.

Синтаксис

```
#include <libmilter/mfapi.h>
sfsistat (*xxfi_connect)(
        SMFICTX *ctx,
        char       *hostname,
        _SOCK_ADDR *hostaddr);
```

Описание

Функция **xxfi_connect** вызывается один раз в начале соединения SMTP. Она возвращает флаг SMFIS_ALL_CONTINUE.

Примечание: Если соединение отклонено одним из ранее действовавших фильтров при вызове функции **xxfi_connect**, то для данного фильтра функция **xxfi_connect** не вызывается.

Аргументы

Таблица 31. Аргументы

Элемент	Описание
ctx	Скрытая структура контекста хранится в **libmilter**.
hostname	Имя хоста отправителя сообщения, определенное при обратном поиске по адресу хоста. Если обратный поиск не возвращает результат или если ни один из IP-адресов найденного хоста не соответствует исходному IP-адресу, то имя хоста заменяется на IP-адрес отправителя сообщения, заключаемый в квадратные скобки, например, [a.b.c.d]. Если соединение SMTP установлено через stdin, то сюда подставляется localhost.
hostaddr	Адрес хоста, определенный функцией **getpeername(2)** для сокета SMTP. Это значение равно NULL, если тип не поддерживается в текущей версии или если соединение соединение установлено через **stdin**.

Функция обратного вызова xxfi_helo:
Назначение

Функция обратного вызова **xxfi_helo** обрабатывает команду **HELO или EHLO**.

Синтаксис

```
#include <libmilter/mfapi.h>
sfsistat (*xxfi_helo)(
 SMFICTX *ctx,
 char *helohost
);
```

Описание

Функция обратного вызова **xxfi_hello** вызывается всякий раз, когда клиент отправляет команду **HELO или EHLO**. Она возвращает флаг SMFIS_CONTINUE. Поэтому эта функция обратного вызова может вызываться несколько раз или вообще не вызываться. Почтовая программа может устанавливать свои ограничения.

Аргументы

Таблица 32. Аргументы

Элемент	Описание
ctx	Скрытая структура контекста хранится в параметре **libmilter**.
helohost	Значение, передаваемое в команду **HELO или EHLO**, должно быть именем домена отправляющего хоста.

Функция обратного вызова xxfi_envfrom:
Назначение

Функция обратного вызова **xxfi_envfrom** обрабатывает оболочечную команду **MAIL**.

Синтаксис

```
#include <libmilter/mfapi.h>
sfsistat (*xxfi_envfrom)(
 SMFICTX *ctx,
 char **argv
);
```

Описание

Функция обратного вызова **xxfi_envfrom** вызывается после команды **DATA** клиента и возвращает флаг SMFIS_CONTINUE.

Примечание: Дополнительная информация о кодах ESMTP приведена в RFC 1869.

Аргументы

Таблица 33. Аргументы

Элемент	Описание
ctx	Скрытая структура контекста хранится в параметре **libmilter**.
argv	null-терминированная строка аргументов команды SMTP. `argv[0]` всегда содержит адрес отправителя. Остальные аргументы - это аргументы протокола ESMTP.

Коды возврата

Таблица 34. Возвращаемые значения

Элемент	Описание
`SMFIS_TEMPFAIL`	Отправитель и сообщение временно не принимаются. Возможно, что впоследствии будет указан новый отправитель, то есть сообщение будет новым, и функция **xxfi_abort** не вызывается.
`SMFIS_REJECT`	Отправитель и сообщение отклоняются. Возможно, что будет указан новый отправитель, то есть сообщение будет новым, и функция **xxfi_abort** не вызывается.
`SMFIS_DISCARD`	Сообщение принимается и удаляется. Функция **xxfi_abort** не вызывается.
`SMFIS_ACCEPT`	Сообщение принимается, и функция **xxfi_abort** не вызывается.

Связанная информация

xxfi_abort

Функция обратного вызова xxfi_envrcpt:

Назначение

Функция **xxfi_envrcpt** обрабатывает оболочечную команду **RCPT**.

Синтаксис

```
#include <libmilter/mfapi.h>
sfsistat (*xxfi_envrcpt)(
 SMFICTX *ctx,
 char **argv
);
```

Описание

Функция обратного вызова **xxfi_envrcpt** вызывается один раз для получателя и один или несколько раз для сообщения сразу после функции **xxfi_envfrom** и возвращает флаг SMFIS_CONTINUE.

Примечание: Дополнительная информация о расширенных кодах ESMTP приведена в RFC 1869.

Аргументы

Таблица 35. Аргументы

Элемент	Описание
ctx	Скрытая структура контекста хранится в параметре **libmilter**.
argv	null-терминированная строка аргументов команды SMTP. `argv[0]` всегда содержит адрес получателя. Остальные аргументы - это аргументы протокола ESMTP.

Коды возврата

Таблица 36. Коды возврата

Элемент	Описание
SMFIS_TEMPFAIL	Временная ошибка получателя. Отправка остальным получателям продолжается, и функция **xxfi_abort** не вызывается.
SMFIS_REJECT	Отказ в приеме для получателя. Отправка остальным получателям продолжается, и функция **xxfi_abort** не вызывается.
SMFIS_DISCARD	Сообщение принимается или удаляется. Функция **xxfi_abort** вызывается.
SMFIS_ACCEPT	Получатель принимается, и функция **xxfi_abort** не вызывается.

Связанная информация

xxfi_envfrom

xxfi_abort

Функция обратного вызова xxfi_data:
Назначение

Функция обратного вызова **xxfi_data** обрабатывает команду **DATA**.

Синтаксис

```
#include <libmilter/mfapi.h>
sfsistat (*xxfi_data)(
 SMFICTX *ctx
);
```

Описание

Функция обратного вызова **xxfi_data** вызывается после команды **DATA** клиента и возвращает флаг SMFIS_CONTINUE.

Примечание: Дополнительная информация о кодах ESMTP приведена в RFC 1869.

Аргументы

Таблица 37. Аргументы

Элемент	Описание
ctx	Скрытая структура контекста хранится в параметре **libmilter**.

Коды возврата

Таблица 38. Коды возврата

Элемент	Описание
SMFIS_TEMPFAIL	Сообщение временно не принимается.
SMFIS_REJECT	Сообщение отклоняется.
SMFIS_DISCARD	Сообщение принимается и удаляется.
SMFIS_ACCEPT	Сообщение принимается.

Функция обратного вызова xxfi_unknown:
Назначение

Функция обратного вызова **xxfi_unknown** обрабатывает неизвестные или нереализованные команды протокола SMTP.

Синтаксис

```
#include <libmilter/mfapi.h>
sfsistat (*xxfi_unknown)(
 SMFICTX *ctx,
 const char *arg
);
```

Описание

Функция обратного вызова **xxfi_unknown** вызывается в том случае, если клиент использует неизвестную или нереализованную почтовой программой команду SMTP. Она возвращает флаг SMFIS_CONTINUE.

Примечание: Сервер всегда отклоняет такую команду SMTP. Можно возвратить только другой код ошибки.

Аргументы

Таблица 39. Аргументы

Элемент	Описание
ctx	Скрытая структура контекста хранится в параметре **libmilter**.
arg	Команда SMTP вместе со всеми аргументами.

Коды возврата

Таблица 40. Возвращаемые значения

Элемент	Описание
SMFIS_TEMPFAIL	Команда временно не принимается.
SMFIS_REJECT	Команда отклоняется.

Функция обратного вызова xxfi_header:
Назначение

Функция **xxfi_header** обрабатывает заголовок сообщения.

Синтаксис

```
#include <libmilter/mfapi.h>
sfsistat (*xxfi_header)(
 SMFICTX *ctx,
 char *headerf,
 char *headerv
);
```

Описание

Функция обратного вызова **xxfi_header** вызывается один раз для каждого заголовка сообщения и возвращает флаг SMFIS_CONTINUE.

Примечание:

- Начиная с sendmail 8.14, пробелы после двоеточия в поле заголовка сохраняются, если этого требует флаг SMFIP_HDR_LEADSPC. Например, заголовок

  ```
  From: sender <f@example.com>
  To:  user <t@example.com>
  Subject:no
  ```

  будет передан в параметр milter в следующем виде:

  ```
  "From", " sender <f@example.com>"
  "To", "  user <t@example.com>"
  "Subject", "no"
  ```

  в то время как ранее (или в отсутствие флага SMFIP_HDR_LEADSPC) он бы имел следующий вид:

  ```
  "From", "sender <f@example.com>"
  "To", "user <t@example.com>"
  "Subject", "no"
  ```

- Старый фильтр изменяет или добавляет заголовки для новых фильтров.
- Дополнительная информация о формате заголовка приведена в RFC 822 и RFC 2822.

Аргументы

Таблица 41. Аргументы

Элемент	Описание
ctx	Скрытая структура контекста хранится в параметре **libmilter**.
headerf	Имя поля заголовка.
headerv	Значение поля заголовка. Содержимое заголовка должно включать встроенный пробел, то есть несколько строк, завершаемых пробелом и разделенных символом LF (а не CR или LF). Символ конца строки (CR или LF) удаляется.

Информация, связанная с данной:

RFC 2822

RFC 822

Функция обратного вызова xxfi_eoh:
Назначение

Функция **xxfi_eoh** обрабатывает конец заголовков сообщения.

Синтаксис

```
#include <libmilter/mfapi.h>
sfsistat (*xxfi_eoh)(
 SMFICTX *ctx
);
```

Описание

Функция обратного вызова **xxfi_eoh** вызывается один раз после обработки всех заголовков сообщения. Она возвращает флаг SMFIS_CONTINUE.

Аргументы

Таблица 42. Аргументы

Элемент	Описание
ctx	Скрытая структура контекста хранится в параметре **libmilter**.

Функция обратного вызова xxfi_body:
Назначение

Функция обратного вызова **xxfi_body** обрабатывает фрагмент тела сообщения.

Синтаксис

```
#include <libmilter/mfapi.h>
sfsistat (*xxfi_body)(
 SMFICTX *ctx,
 unsigned char *bodyp,
 size_t len
);
```

Описание

Функция **xxfi_body** может вызываться несколько раз между функциями **xxfi_eoh** и **xxfi_eom** или вообще не вызываться ни разу. Она возвращает флаг SMFIS_CONTINUE.

Примечание:

- Параметр **bodyp** указывает на последовательность байтов. Это не строка C (последовательность символов, завершающаяся символом '\0'). Поэтому с этой байтовой последовательностью не следует использовать обычные функции C для обработки строк, такие как **strlen(3)**. Внутри последовательности могут быть байты со значением '\0'. Даже если добавить в конец байт '\0', функции C обработки строк могут вести себя непредсказуемо.
- Тело сообщения может иметь большой размер, и функция обратного вызова **xxfi_body** может существенно влиять на производительность.
- Концы строк представляются в том виде, как их передает протокол SMTP (обычно это CR/LF).
- Старые фильтры изменяют тело для новых фильтров.
- Тело сообщения может отправляться несколькими фрагментами, и для каждого фрагмента будет один раз вызываться функция **xxfi_body**.
- Эта функция возвращает флаг SMFIS_SKIP, если milter получил достаточно фрагментов тела сообщения для принятия решения, но все еще требуется вызывать функции функции модификации сообщений, разрешенные только из функции **xxfi_eom**.

- milter должен согласовать такое поведение с почтовой программой. Это означает, что он должен проверить, доступен ли флаг действия протокола SMFIP_SKIP, и если он доступен, то параметр **milter** должен его потребовать.

Аргументы

Таблица 43. Аргументы

Элемент	Описание
ctx	Скрытая структура контекста хранится в параметре **libmilter**.
bodyp	Указатель на начало данного блока данных тела. *bodyp* не может применяться вне этого вызова функции **xxfi_body**.
len	Длина массива данных, на который указывает *bodyp*.

Связанная информация

xxfi_eoh

xxfi_eom

Функция обратного вызова xxfi_eom:
Назначение

Функция **xxfi_eom** обрабатывает конец сообщения.

Синтаксис

```
#include <libmilter/mfapi.h>
sfsistat (*xxfi_eom)(
 SMFICTX *ctx
);
```

Описание

Функция обратного вызова **xxfi_eom** вызывается один раз после всех функций обратного вызова **xxfi_body** для данного сообщения. Она возвращает флаг SMFIS_CONTINUE.

Примечание: Для внесения изменений в заголовки, тело и оболочку сообщения в функции обратного вызова **xxfi_eom** необходим фильтр. Эти изменения осуществляются процедурами **smfi_***.

Аргументы

Таблица 44. Аргументы

Элемент	Описание
ctx	Скрытая структура контекста хранится в параметре **libmilter**.

Связанная информация

xxfi_body

Функция обратного вызова xxfi_abort:
Назначение

Функция обратного вызова **xxfi_abort** отвечает за работу с сообщениями, прием которых будет прерван.

Синтаксис

```
#include <libmilter/mfapi.h>
sfsistat (*xxfi_abort)(
 SMFICTX *ctx
);
```

Описание

Функция обратного вызова **xxfi_abort** может вызываться в любой момент в ходе обработки сообщения, то есть между любой функцией, работающей с сообщением, и функцией обратного вызова **xxfi_eom**. Она возвращает флаг SMFIS_CONTINUE.

Примечание:

- Функция обратного вызова **xxfi_abort** должна освободить все ресурсы, выделенные для сообщения. Она должна допускать вызов между любыми функциями обратного вызова, работающими с сообщениями.
- Вызовы функций **xxfi_abort** и **xxfi_eom** взаимно исключают друг друга.
- Функция обратного вызова **xxfi_abort** не отвечает за восстановление данных, относящихся к соединению, поскольку при закрытии соединения всегда вызывается функция **xxfi_close**.
- Поскольку текущее сообщение будет не принято, код возврата игнорируется.
- Функция обратного вызова **xxfi_abort** вызывается только в том случае, если обработка сообщения прервана вне области управления фильтра, и фильтр не завершил обработку сообщения. Если, например, фильтр уже возвратил флаг SMFIS_ACCEPT, SMFIS_REJECT или SMFIS_DISCARD из функции обработки сообщения, то функция **xxfi_abort** не будет вызываться, даже если обработка сообщения в дальнейшем прерывается вне области действия фильтра.

Аргументы

Таблица 45. Аргументы

Элемент	Описание
ctx	Скрытая структура контекста хранится в **libmilter**.

Связанная информация

xxfi_close

xxfi_eom

Функция обратного вызова xxfi_close:
Назначение

Функция обратного вызова **xxfi_close** закрывает текущее соединение.

Синтаксис

```
#include <libmilter/mfapi.h>
sfsistat (*xxfi_close)(
 SMFICTX *ctx
);
```

Описание

Функция обратного вызова **xxfi_close** всегда вызывается в завершение соединения и возвращает флаг SMFIS_CONTINUE.

Функция обратного вызова **xxfi_close** может вызываться в произвольный момент, то есть даже перед вызовом функции **xxfi_connect**. Если после установления соединения почтовой программы (MTA) с

фильтром MTA решает прервать обмен данными в этом соединении, например, после обращения к access_db, то никакие данные не будут передаваться в фильтр до тех пор, пока клиент не будет закрыт. В этот момент будет вызвана функция **xxfi_close**. Поэтому этот обратный вызов может быть вообще единственным для данного соединения, и при разработке кода функции обратного вызова **xxfi_close** следует учитывать такую возможность. Было бы ошибкой предполагать, что частный указатель контекста этого обратного вызова будет иметь значение, отличное от NULL.

Функция **xxfi_close** вызывается при закрытии, даже если предыдущая почтовая транзакция была прервана.

Функция **xxfi_close** должна освободить все ресурсы, выделенные для соединения.

Поскольку соединение уже закрывается, код возврата игнорируется.

Аргументы

Таблица 46. Аргументы

Элемент	Описание
ctx	Скрытая структура контекста хранится в параметре **libmilter**.

Связанная информация

xxfi_connect

Функция обратного вызова xxfi_negotiate:
Назначение

Функция обратного вызова **xxfi_negotiate** обрабатывает согласование.

Синтаксис

```
#include <libmilter/mfapi.h>
#include <libmilter/mfdef.h>
sfsistat (*xxfi_negotiate)(
        SMFICTX    "Функция обратного вызова xxfi_negotiate",
 unsigned long f0,
 unsigned long f1,
 unsigned long f2,
 unsigned long f3,
 unsigned long *pf0,
 unsigned long *pf1,
 unsigned long *pf2,
 unsigned long *pf3);
```

Описание

Функция **xxfi_negotiate** вызывается в начале соединения SMTP. Она возвращает флаг SMFIS_ALL_OPTS.

С этой функцией параметр **milter** может динамически определять и выполнять операции и действия при запуске. В предыдущих версиях действия (f0) задавались в поле flags структуры **smfiDesc**, и этапы протокола (f1) определялись неявным образом посредством проверки наличия заданной функции обратного вызова. Вследствие расширений в новой версии **milter** такая статическая процедура выбора не будет работать, если параметру **milter** потребуются новые действия, которые недоступны при взаимодействии со старыми почтовыми программами (MTA). Поэтому при согласовании функция обратного вызова может определить, какие операции доступны, и динамически выбрать необходимые функции обратного вызова из числа доступных. Если какие-либо операции недоступны, то параметр **milter** может либо перейти в старый режим работы, либо остановить работу и предложить пользователю выполнить обновление.

Этапы протокола
(f1, *pf1)

:

- SMFIP_RCPT_REJ: с помощью этого бита параметр **milter** может потребовать от MTA отправки команд RCPT, которые ранее были отклонены по причине того, что пользователь неизвестен или по схожей причине, но не из-за синтаксических ошибок. Если **milter** запрашивает этот этап протокола, то он также должен проверить макрос **{rcpt_mailer}:** если он возвращает ошибку, то MTA отклонит получателя. Обычно в этом случае макросы **{rcpt_host}** и **{rcpt_addr}** содержат и расширенный код состояния, и текст ошибки.
- SMFIP_SKIP указывает, что MTA распознает код возврата SMFIS_SKIP.
- SMFIP_NR_* указывает, что MTA распознает код возврата SMFIS_NOREPLY. Для различных этапов протокола предусмотрены следующие флаги:
  - SMFIP_NR_CONN: “Функция обратного вызова xxfi_connect” на стр. 83
  - SMFIP_NR_HELO: “Функция обратного вызова xxfi_helo” на стр. 84
  - SMFIP_NR_MAIL: “Функция обратного вызова xxfi_envfrom” на стр. 85
  - SMFIP_NR_RCPT: “Функция обратного вызова xxfi_envrcpt” на стр. 85
  - SMFIP_NR_DATA: “Функция обратного вызова xxfi_data” на стр. 86
  - SMFIP_NR_UNKN: “Функция обратного вызова xxfi_unknown” на стр. 87
  - SMFIP_NR_EOH: “Функция обратного вызова xxfi_eoh” на стр. 89
  - SMFIP_NR_BODY: “Функция обратного вызова xxfi_body” на стр. 89
  - SMFIP_NR_HDR: “Функция обратного вызова xxfi_header” на стр. 88
- Флаг SMFIP_HDR_LEADSPC указывает, что MTA может передавать значения заголовков, не изменяя пробелов в начале строки. Если запрашивается этот этап протокола, то MTA не добавляет пробел перед заголовками, если они добавляются, вставляются или изменяются.
- MTA можно запретить отправлять информацию о различных этапах SMTP; такие флаги начинаются с SMFIP_NO*.
  - SMFIP_NOCONNECT: “Функция обратного вызова xxfi_connect” на стр. 83
  - SMFIP_NOHELO: “Функция обратного вызова xxfi_header” на стр. 88
  - SMFIP_NOMAIL: “Функция обратного вызова xxfi_envfrom” на стр. 85
  - SMFIP_NORCPT: “Функция обратного вызова xxfi_envrcpt” на стр. 85
  - SMFIP_NOBODY: “Функция обратного вызова xxfi_body” на стр. 89
  - SMFIP_NOHDRS: “Функция обратного вызова xxfi_header” на стр. 88
  - SMFIP_NOEOH: “Функция обратного вызова xxfi_eoh” на стр. 89
  - SMFIP_NOUNKNOWN: “Функция обратного вызова xxfi_unknown” на стр. 87
  - SMFIP_NODATA: “Функция обратного вызова xxfi_data” на стр. 86

  Для каждой из функций обратного вызова **xxfi_***, которые не использует milter, соответствующий флаг необходимо установить в

  *pf1.

Возможные действия
(f0, *pf0)

описаны в **(xxfi_flags)**.

Если milter возвращает флаг SMFIS_CONTINUE, то milter настраивает требуемые действия и этапы протокола посредством выходных параметров pf0 и pf1, которые соответствуют f0 и f1. Для совместимости с будущими версиями выходные параметры pf2 и pf3 должны быть равны 0.

Аргументы

Таблица 47. Аргументы

Элемент	Описание
ctx	Скрытая структура контекста хранится в параметре **libmilter**.
f0	Действия, предлагаемые MTA.
f1	Шаги протокола, предлагаемые MTA.
f2	Для будущих расширений.
f3	Для будущих расширений.
pf0	Действия, запрошенные **milter**
pf1	Шаги протокола, запрошенные **milter**.
pf2	Для будущих расширений.
pf3	Для будущих расширений.

Коды возврата

Таблица 48. Возвращаемые значения

Элемент	Описание
SMFIS_ALL_OPTS	Если для **milter** только требуется перечислить все доступные этапы протокола и действия, то можно возвратить флаг SMFIS_ALL_OPTS, и MTA сделает все этапы протокола и действия доступными **milter**. В этом случае выходным параметрам pf0-pf3 не требуется присваивать никакие значения, так как они игнорируются.
SMFIS_REJECT	При запуске **milter** возникает ошибка, и к нему не следует обращаться более в текущем соединении.
SMFIS_CONTINUE	Продолжить обработку. В этом случае **milter** должен установить все выходные параметры pf0-pf3. Настройка выходных параметров описана в следующем разделе.

Прочие функции и константы

Прочие функции и константы позволяют получить информацию о версии **libmilter**.

Таблица 49. Константы

Элемент	Описание
smfi_version	Функция **smfi_version** получает динамическую информацию о версии параметра **libmilter**.
smfi_setsymlist	**smfi_setsymlist** задает список макросов, который параметр **libmilter** ожидает получить от почтовой программы (MTA) на данном этапе протокола.

Таблица 50. Константы

Элемент	Описание
SMFI_VERSION	**SMFI_VERSION** позволяет получить версию параметра **libmilter**.

Функция smfi_version:
Назначение

Функция **smfi_version** получает динамическую информацию о версии **libmilter**.

Синтаксис

```
#include <libmilter/mfapi.h>
int smfi_version(
 unsigned int *pmajor,
 unsigned int *pminor,
 unsigned int *ppl
);
```

Описание

Функцию обратного вызова **smfi_version** можно вызывать в любой момент.

Версия библиотеки **libmilter**, заданная при компиляции, доступна посредством макроса SMFI_VERSION. Для того чтобы получить номера основной и дополнительной версии и текущий уровень исправления из этого макроса, можно использовать макросы SM_LM_VRS_MAJOR(v), SM_LM_VRS_MINOR(v) и SM_LM_VRS_PLVL(v). Параметр **milter** может применять макрос SMFI_VERSION для определения того, какие функции необходимо использовать (это задается во время компиляции через директивы препроцессора C). Используя этот макрос и функцию **smfi_version**, параметр **milter** может определить во время выполнения, связан ли он динамически с требуемой версией **libmilter**. Такая функция должна сравнивать только основную и дополнительную версии. Это означает, что библиотека **libmilter** будет совместимой независимо от уровня исправлений.

Аргументы

Таблица 51. Аргументы

Элемент	Описание
pmajor	Указатель на переменную типа unsigned int с номером основной версии.
pminor	Указатель на переменную типа unsigned int с номером дополнительной версии.
ppl	Указатель на переменную типа unsigned int с номером уровня исправлений.

Коды возврата

Функция **smfi_version** возвращает значение MI_SUCCESS.

Функция smfi_setsymlist:
Назначение

Функция **smfi_setsymlist** задает список макросов, который параметр **milter** ожидает получить от почтовой программы (MTA) на данном этапе протокола.

Синтаксис

```
#include <libmilter/mfapi.h>
int smfi_setsymlist(
        SMFICTX *ctx,
 int        stage,
 char     *macros
);
```

Описание

Функция **smfi_setsymlist** должна вызываться в ходе вызова функции **xxfi_negotiate**. Она позволяет переопределить список макросов, который параметр **milter** ожидает получить от почтовой программы (MTA).

Примечание: В настоящий момент можно включить в список не более 5 макросов. Однако это ограничение устанавливается не параметром **milter**, а почтовой программой, и информация о возможных нарушениях этого ограничения не передается в **milter**.

Аргументы

Таблица 52. Аргументы

Элемент	Описание
ctx	Скрытая структура контекста хранится в параметре **libmilter**.
stage	Этап протокола, на котором требуется использовать список макросов. Допустимые значения описаны в файле include/libmilter/mfapi.h, см. макросы C с приставкой SMFIM_. Возможные этапы протокола включают начало соединения, команды HELO или EHLO, MAIL, RCPT, DATA, конец заголовка и конец сообщения.
macros	Список макросов (разделенный пробелами). Пример: "**{rcpt_mailer} {rcpt_host}**".

Коды возврата

Функция **smfi_setsymlist** возвращает значение MI_FAILURE в следующих случаях. В противном случае функция возвращает MI_SUCCESS.

- Недостаточно памяти для копирования списка макросов.
- Список *macros* равен NULL или пуст.
- *stage* не является допустимым этапом протокола.
- Список макросов для этапа уже был настроен ранее.

Связанная информация

xxfi_negotiate

Флаги отладки для sendmail

В команде **sendmail** предусмотрено большое число флагов отладки.

Каждому флагу отладки соответствует свой номер и уровень. Чем выше уровень, тем больше данных содержится в выводе функции. Считается, что на уровнях выше 9-го выводится настолько большой объем информации, что они могут применяться только для отладки определенного фрагмента кода. Флаги отладки задаются с помощью опции **-d**, как показано ниже:

```
debug-flag:     -d debug-list
debug-list:     debug-flag[.debug-flag]*
debug-flag:     debug-range[.debug-level]
debug-range:    integer|integer-integer
debug-level:    integer
```

```
-d12            Установить флаг 12 на уровень 1
-d12.3          Установить флаг 12 на уровень 3
-d3-17          Установить флаги с 3 по 17 на уровень 1
-d3-17.4        Установить флаги с 3 по 17 на уровень 4
```

Поддерживаются следующие флаги отладки:

Элемент	Описание
-d0	Общая отладка.
-d1	Показать информацию об отправке.
-d2	Закончить на *finis*( ).
-d3	Распечатать среднюю загрузку.
-d4	Места на диске достаточно.
-d5	Показать события.
-d6	Показать необработанные почтовые сообщения.
-d7	Имя файла очереди.
-d8	Преобразование имен DNS.
-d9	Трассировка запросов RFC1413.
-d9.1	Создать каноническое имя для хоста.
-d10	Показывать уведомление о доставке получателю.
-d11	Трассировать доставку.
-d12	Показать подключение зависимого хоста.
-d13	Показывать уведомление о доставке.
-d14	Показать запятые в полях заголовков.
-d15	Показать действия по получению сетевых запросов.
-d16	Инициируемые соединения.
-d17	Показать список хостов MX.

Примечание: В настоящее время команда **sendmail** поддерживает почти 200 флагов отладки.

Протокол доступа к сообщениям Internet и Почтовый протокол

Для удаленного доступа к почте в AIX применяется два типа серверов электронной почты Internet.

- **Почтовый протокол** (**POP** или **POP3DS**)
- **Протокол доступа к сообщениям Internet** (**IMAP** или **IMAPDS**)

Каждый тип сервера хранит электронные сообщения и предоставляет доступ к этим сообщениям. При использовании протоколов доступа к почте на сервере исчезает необходимость в том, чтобы для приема почты компьютер был все время включен.

Сервер **POP** или **POP3DS** предназначен для просмотра почты в автономном режиме. Удаленные клиенты могут получать почтовые сообщения с сервера с помощью клиента **POP** или **POP3DS**. Программа-клиент может либо загружать сообщения и немедленно удалять их с сервера, либо загружать их и оставлять на сервере **POP** или **POP3DS**. После загрузки сообщений вся последующая обработка почты выполняется на компьютере-клиенте. Сервер **POP** не позволяет работать с почтовым ящиком сразу нескольким клиентам. В версии **POP3DS** применяются библиотеки OpenSSL, для работы с которыми требуются сертификаты безопасности.

Сервер **IMAP** или **IMAPDS** предоставляет большее количество функций, чем **POP**-сервер, и имеет другой интерфейс. Сервер **IMAP** или **IMAPDS** предоставляет автономный доступ, доступ в режиме реального времени и доступ в режиме отсоединения. Применяемый протокол позволяет управлять удаленным почтовым ящиком точно так же, как и локальным. Например, клиент может выполнять поиск по сообщениям и присваивать сообщениям флаги состояния, такие как **удалено** или**отправлен ответ**. Кроме того, можно оставлять сообщения в базе данных сервера до тех пор, пока не будет дана явная команда удалить их. Сервер **IMAP** обеспечивает одновременный доступ к почтовому ящику нескольким клиентам. В версии **IMAPDS** применяются библиотеки OpenSSL, для работы с которыми требуются сертификаты безопасности.

Каждый их этих типов серверов применяется только для доступа к почте. Оба сервера используют для пересылки почты **Простой протокол передачи почты (SMTP)**.

Все применяемые протоколы являются открытыми и основаны на стандартах RFC. Работа серверов **IMAP** основана на стандартах RFC 2060 и 2061, а серверов **POP** - на стандарте RFC 1939. Оба протокола

предназначены для установления соединения через сокеты TCP. Сервер **IMAP** работает через порт 143, а сервер **IMAPDS** - через порт 993. Сервер **POP** работает через порт 110, а сервер **POP3DS** - через порт 995. Всеми серверами управляет демон **inetd**.

Требования: Для работы с версиями OpenSSL необходимо установить OpenSSL. OpenSSL находится на компакт-диске *AIX Toolbox for Linux Applications*.

Настройка серверов IMAP и POP

Выполните эту процедуру для настройки серверов **IMAP** и **POP**.

Для выполнения этой задачи необходимы права доступа root.

1. Удалите символы комментария из строк **imapd** или **imapds** и **pop3d** или **pop3ds** в файле `/etc/inetd.conf`. Ниже приведены примеры строк в файле конфигурации:

```
#imap2  stream  tcp     nowait  root    /usr/sbin/imapd imapd
#pop3   stream  tcp     nowait  root    /usr/sbin/pop3d pop3d
#imaps  stream  tcp     nowait  root    /usr/sbin/imapds imapds
#pop3s   stream  tcp     nowait  root    /usr/sbin/pop3ds pop3ds
```

2. Укажите файлы конфигурации для сервера **imapds** в файле `/etc/imapd.cf` и для сервера **pop3ds** в файле `/etc/pop3d.cf`. По умолчанию для сервера **imapds** и для сервера **pop3ds** включены менее защищенные протоколы квитирования SSLv2 и SSLv3. Но SSLv2 и SSLv3 можно выключить, внеся соответствующие изменения в файлы конфигурации (см. следующий пример). Также можно включить или выключить любой шифр путем изменения строки `SSL_CIPHER_LIST` в файле конфигурации. Этот параметр переопределяет строку шифров по умолчанию, которая жестко прописана в приложениях.

   Файл конфигурации для сервера imapds (`/etc/imapd.cf`):

```
#====================================================================
#
# Пример файла конфигурации сервера IMAP
#
#====================================================================
#====================================================================
# Удалите символ комментария у строки внизу, чтобы выключить SSL v2 для сервера imap.
#
#   Disable SSL V2  --->  SSL_OP_NO_SSLv2         YES
#   Allow SSL V2    --->  SSL_OP_NO_SSLv2         NO
#
#
#SSL_OP_NO_SSLv2         YES  <-------------- удалите символ комментария у этой строки, чтобы выключить sslv2
#====================================================================
# Удалите символ комментария у строки внизу, чтобы выключить SSL v3 для сервера imap.
#
#   Disable SSL V3  --->  SSL_OP_NO_SSLv3         YES
#   Allow SSL V3    --->  SSL_OP_NO_SSLv3         NO
#
#
#SSL_OP_NO_SSLv3         YES  <-------------- удалите символ комментария у этой строки, чтобы выключить sslv3
#====================================================================
# Удалите символ комментария у строки внизу, чтобы использовался список шифров, указанный пользователем,
# для сервера imap. Синтаксический анализатор ожидает строку шифров в кавычках " ".
#
#
#SSL_CIPHER_LIST "ALL:!LOW" <--- удалите символ комментария у этой строки, чтобы изменить строку шифров
#====================================================================
```

   Файл конфигурации для сервера pop3ds (`/etc/pop3d.cf`):

```
#====================================================================
#
# Пример файла конфигурации сервера POP3
#
#====================================================================
#====================================================================
# Удалите символ комментария у строки внизу, чтобы выключить SSL v2 для сервера pop3d.
#
```

```
#   Disable SSL V2  --->  SSL_OP_NO_SSLv2         YES
#   Allow SSL V2    --->  SSL_OP_NO_SSLv2         NO
#
#
#SSL_OP_NO_SSLv2         YES  <----------- удалите символ комментария у этой строки, чтобы выключить sslv2
#=================================================================
# Удалите символ комментария у строки внизу, чтобы выключить SSL v3 для сервера pop3d.
#
#   Disable SSL V3  --->  SSL_OP_NO_SSLv3         YES
#   Allow SSL V3    --->  SSL_OP_NO_SSLv3         NO
#
#
#SSL_OP_NO_SSLv3         YES  <----------- удалите символ комментария у этой строки, чтобы выключить sslv3
#=================================================================
# Удалите символ комментария у строки внизу, чтобы использовался список шифров, указанный пользователем,
# для сервера pop3d. Синтаксический анализатор ожидает строку шифров в кавычках " ".
#
#
#SSL_CIPHER_LIST "ALL:!LOW" <---- удалите символ комментария у этой строки, чтобы изменить строку шифров
#=================================================================
```

3. Обновите демон **inetd** с помощью следующей команды:

   ```
   refresh -s inetd
   ```

Выполнение тестов конфигурации:

Выполните несколько тестов, чтобы убедиться в том, что серверы готовы к работе.

1. Прежде всего убедитесь в том, что серверы работают со своими стандартными портами. Для этого выполните следующие команды, нажимая после ввода каждой из них клавишу Enter:

   ```
   netstat -a | grep imap
   netstat -a | grep pop
   ```

   Ниже приведен вывод команд **netstat**:

   ```
   tcp     0       0     *.imap2         *.*        LISTEN
   tcp     0       0     *.imaps         *.*        LISTEN
   tcp     0       0     *.pop3          *.*        LISTEN
   tcp     0       0     *.pop3s         *.*        LISTEN
   ```

2. Если результаты вывода отличаются от указанных, проверьте записи в файле `/etc/inetd.conf` и повторите команду **refresh -s inetd**.
3. Для проверки конфигурации сервера imapd установите соединение Telnet с этим сервером imap2 через порт 143 (для IMAPDS - через порт Telnet 993). После того как будет установлено соединение Telnet, появится командная строка imapd. В ней можно вводить команды IMAP версии 4, определенные в документе RFC 1730. Для того чтобы запустить эту команду, введите точку (.), пробел, token, имя команды и необходимые параметры. token позволяет упорядочить имена команд. Например:

   ```
   . token команда параметры
   ```

   При подключении к серверу imapd с помощью Telnet ваш пароль будет показан на экране.

   В следующем примере запуска Telnet в строке *пароль* команды **login** необходимо ввести собственный пароль.

   Совет: В случае IMAPDS команда и ее вывод немного отличаются.

   ```
   telnet e-xbelize 143
   Выполнение запроса...
   Установлено соединение с e-xbelize.austin.ibm.com.
   Escape-символ - '^]'.
   * OK e-xbelize.austin.ibm.com IMAP4 server ready
   . 1 login id id_password
   . OK
   . 2 examine /usr/spool/mail/root
   * FLAGS (\Answered \Flagged \Draft \Deleted \Seen)
   * OK [PERMANENTFLAGS (\Answered \Flagged \Draft \Deleted \Seen \*)]
   * 0 EXISTS
   ```

```
* 0 RECENT
* OK [UIDVALIDITY 823888143]
. OK [READ-ONLY] Examine completed
. 3 logout
* BYE Server terminating connection
. OK Logout completed
Соединение закрыто.
```

4. Для проверки конфигурации сервера pop3d установите соединение Telnet с портом POP3, 110 (для POP3DS - через порт Telnet 995). После того как будет установлено соединение Telnet, появится командная строка pop3d. Вы можете вызывать команды протокола POP, определенные стандартом RFC 1725. Для того чтобы запустить одну из этих команд, введите точку (.), пробел, а затем имя команды. Например:

   ```
   . Команда
   ```

   При подключении к серверу imapd с помощью Telnet ваш пароль будет показан на экране.

   В следующем примере запуска Telnet в строке *пароль* команды **pass** необходимо ввести собственный пароль.

   Совет: В случае POP3DS команда и ее вывод немного отличаются.

   ```
   telnet e-xbelize 110
   Выполнение запроса...
   Установлено соединение с e-xbelize.austin.ibm.com.
   Escape-символ - '^]'.
   +OK e-xbelize.austin.ibm.com POP3 server ready
   user id
   +OK Name is a valid mailbox
   pass пароль
   +OK Maildrop locked and ready
   list
   +OK scan listing follows
   .
   stat
   +OK 0 0
   quit
   +OK
   Соединение закрыто.
   ```

Ведение протоколв с помощью утилиты SYSLOG

Программное обеспечение серверов **IMAP** (и **IMAPDS**) и **POP** (и **POP3DS**) отправляет сообщения протокола утилите **SYSLOG**.

1. Для настройки средств ведения протокола серверов **IMAP** и **POP** с помощью **SYSLOG** необходимо работать в системе под именем root. Отредактируйте файл /etc/syslog.conf и внесите следующую запись для ***.debug**:

   ```
   *.debug /usr/adm/imapd.log
   ```

2. Необходимо, чтобы файл usr/adm/imapd.log был создан до того, как демон **syslogd** считает файл конфигурации /etc/syslog.conf. Для того чтобы создать этот файл, введите в командной строке следующую команду и нажмите клавишу Enter:

   ```
   touch /usr/adm/imapd.log
   ```

3. Обносите демон **syslogd**, для того чтобы он заново считал свой файл конфигурации. Введите в командной строке следующую команду и нажмите Enter:

   ```
   refresh -s syslogd
   ```

Команды управления почтой

Здесь собраны сведения о командах управления почтой.

Элемент	Описание
bugfiler	Сохраняет отчеты об ошибках в специальных почтовых каталогах.
comsat	Уведомляет пользователей о поступлении почты (программа-демон).
mailq	Команда печати почтовой очереди.
mailstats	Показывает статистику о передаче почты.
newaliases	Создает новую копию файла базы данных псевдонимов `/etc/mail/aliases`.
rmail	Управляет почтовыми сообщениями, полученными с помощью команды **uucp** из набора Основных сетевых утилит (BNU).
sendbug	Отправляет отчет о системных ошибках по заданному адресу электронной почты.
sendmail	Управляет пересылкой почты в локальной и внешней сетях.
smdemon.cleanu	Периодически выполняет профилактическую "чистку" очереди команды **sendmail**.

Файлы и каталоги почты

Файлы и каталоги почты можно упорядочить по функциям.

Элемент	Описание
`/usr/share/lib/Mail.rc`	Задает значения параметров локальной системы по умолчанию для всех пользователей почтовой программы. Представляет собой текстовый файл, отредактировав который, можно установить параметры по умолчанию для команды **mail**.
`$HOME/.mailrc`	Позволяет изменить значения по умолчанию для параметров почтовой программы в локальной системе.
`$HOME/mbox`	Здесь хранится обработанная почта конкретного пользователя.
`/usr/bin/Mail`, `/usr/bin/mail` или `/usr/bin/mailx`	Задает три имени для одной почтовой программы. Программа электронной почты представляет собой *один* из способов взаимодействия пользователя с почтовым сервером.
`/var/spool/mail`	Задает имя по умолчанию для каталога принятой почты. По умолчанию, все сообщения помещаются в файл `/var/spool/mail/UserName`.
`/usr/bin/bellmail`	Выполняет локальную доставку почты.
`/usr/bin/rmail`	Команда из набора BNU, выполняющая обработку удаленной почты.
`/var/spool/mqueue`	Содержит файл протокола и временные файлы сообщений из почтовой очереди.

Элемент	Описание
`/usr/sbin/sendmail`	Команда **sendmail**.
`/usr/ucb/mailq`	Связана с `/usr/sbin/sendmail`. Команда `mailq` аналогична команде `/usr/sbin/sendmail -bp`.
`/usr/ucb/newaliases`	Связана с файлом `/usr/sbin/sendmail`. Команда `newaliases` аналогична команде `/usr/sbin/sendmail -bi`.
`/etc/netsvc.conf`	Упорядочивает работу с определенными службами преобразования адресов.
`/usr/sbin/mailstats`	Выводит на печать статистические данные о работе команды **sendmail** из файла `/etc/sendmail.st`, если таковой существует. По умолчанию применяется файл `/etc/sendmail.st`, но при необходимости можно задать и другой файл.
`/etc/mail/aliases`	Представляет собой текстовый вариант файла псевдонимов для команды **sendmail**. Изменяя этот файл, можно создавать, изменять и удалять псевдонимы в почтовой системе.
`/etc/aliasesDB`	Задает каталог, содержащий файлы базы данных псевдонимов `DB.dir` и `DB.pag`, которые создаются на основе файла `/etc/mail/aliases` командой **sendmail -bi**.
`/etc/mail/sendmail.cf`	Содержит в текстовой форме данные о конфигурации команды **sendmail**. Отредактировав этот файл, можно изменить конфигурацию.
`/usr/lib/smdemon.cleanu`	Указывает имя файла программы-оболочки, которая управляет почтовой очередью и файлами протокола команды **sendmail** в каталоге `/var/spool/mqueue`.
`/etc/mail/statistics`	Собирает статистику о передаче почты. Размер этого файла постоянен (он не увеличивается). Для просмотра содержимого данного файла предназначена команда `/usr/sbin/mailstats`. Если данная информация вам не нужна, удалите этот файл.

Элемент	Описание
`/var/spool/mqueue`	Задает каталог, в котором располагаются временные файлы, связанные с каждым сообщением очереди. В каждом каталоге можно создать файл протокола.
`/var/spool/cron/crontabs`	Задает каталог, содержащий файлы, к которым обращается программа-демон **cron** для определения того, какое из заданий необходимо запустить. В файле `root` содержится строка запуска сценария `smdemon.cleanu`.

Команды IMAP и POP

Почтовые команды **imapd** и **pop3d** используются для IMAP и POP.

Элемент	Описание
`/usr/sbin/`**imapd**	Запуск сервера Протокола доступа к сообщениям Internet (IMAP).
`/usr/sbin/`**pop3d**	Запуск сервера Почтового протокола версии 3 (POP3).

Протокол TCP/IP

Для упорядочения обмена данными между компьютерами применяются наборы правил, или *протоколы*. В настоящее время наиболее широко распространен набор протоколов под общим названием **TCP/IP**. (Следует помнить, что во многих странах Европы применяется протокол **X.25**). Основные функции семейства протоколов **TCP/IP**: электронная почта, передача файлов между компьютерами и удаленный вход в систему.

Пользовательская команда **mail**, пользовательские команды обработки сообщений (MH) и команда сервера **sendmail** могут применять **TCP/IP** для передачи сообщений между системами, а основные сетевые утилиты (BNU) могут применять **TCP/IP** для передачи файлов и команд между системами.

TCP/IP - это набор протоколов, который задает стандарты связи между компьютерами и содержит подробные соглашения о маршрутизации и межсетевом взаимодействии. TCP/IP широко применяется в Internet, поэтому с его помощью могут общаться пользователи из исследовательских институтов, школ, университетов, правительственных учреждений и промышленных предприятий.

TCP/IP обеспечивает связь подключенных к сети компьютеров, обычно называемых хостами. Любую сеть можно подключить к другой сети и организовать связь с ее хостами. Несмотря на то, что существуют различные сетевые технологии, многие из которых основаны на коммутации пакетов и потоковом режиме передачи, набор протокол **TCP/IP** обладает одним важным преимуществом: он обеспечивает аппаратную независимость.

Так как в протоколах Internet определяется только блок передачи и способ его отправки, **TCP/IP** не зависит от особенностей сетевого аппаратного обеспечения, позволяя организовать обмен информацией между сетями с различной технологией передачи данных. Система IP-адресов позволяет установить соединение между любыми двумя машинами сети. Кроме того, в **TCP/IP** также определены стандарты для многих служб связи, предназначенных для конечных пользователей.

TCP/IP обеспечивает средства, позволяющие вашему компьютеру выступать в роли хоста Internet, который может подключиться к сети и установить соединение с любым другим хостом Internet. В **TCP/IP** предусмотрены команды и средства, которые позволяют выполнять следующие действия:

- Передавать файлы в другую систему
- Входить в удаленную систему
- Выполнять команды в удаленной системе
- Печатать файлы в удаленной системе
- Отправлять электронные сообщения удаленным пользователям
- Вести интерактивный диалог с удаленными пользователями

- Управлять сетью

Примечание: TCP/IP предусмотрены только основные функции управления сетью. По сравнению с TCP/IP, **Простой протокол управления сетью (SNMP)** предоставляет более широкий набор команд и функций управления.

Терминология TCP/IP

Ознакомьтесь с основными понятиями Internet, связанными с TCP/IP.

Элемент	Описание
клиент	Компьютер или процесс, который получает через сеть доступ к данным, службам или ресурсам другого компьютера или процесса.
хост	Компьютер, который подключен к сети Internet и может взаимодействовать с другими хостами Internet. *Локальный хост* конкретного пользователя - это тот компьютер, на котором работает данный пользователь. *Внешний хост* - это любой другой хост сети. С точки зрения сетевого взаимодействия, хосты - это отправители и получатели пакетов данных. Любой хост может быть клиентом, сервером, или и тем, и другим одновременно. В сети Internet хост идентифицируется по своему имени и IP-адресу.
сеть	Как минимум два хоста, соединенных между собой с помощью линии связи. *Физическая сеть* - это аппаратное обеспечение, формирующее сеть. *Логическая сеть* - это абстрактное объединение, которое может соответствовать одной или нескольким физическим сетям, либо их части. Примером логической сети может служить сеть Internet. Преобразование операций, выполняемых в логической сети, в операции физической сети осуществляется с помощью специальных интерфейсов.
пакет	Блок управляющей информации и данных, передаваемый за один раз из хоста в сеть или обратно. Пакеты представляют собой носитель данных, применяется в Internet для обмена информацией между процессами. Пакеты отправляются от *отправителя* к *получателю*.
порт	Логическая точка подключения для процесса. Процессы обмениваются данными через порты (или *сокеты*). У каждого порта есть очередь данных для отправки и очередь полученных данных. В сети, где применяются интерфейсы, с каждым портом, в зависимости от его назначения, связывается *номер порта* Internet. Отдельный порт идентифицируется по *адресу сокета* Internet, который представляет собой пару, состоящую из IP-адреса хоста и номера порта.
процесс	Выполняющаяся программа. Процесс - это активный объект в системе. Терминалы, файлы и устройства ввода/вывода обмениваются друг с другом данными через процессы. Таким образом, сетевое взаимодействие - это *взаимодействия между процессами*.
протокол	Набор правил, определяющих способы управления соединениями на физическом и логическом уровнях. Для выполнения своих функций одни протоколы часто используют другие протоколы. Например, для передачи пакетов, которые обеспечивают установление соединения между двумя хостами, *протокол уровня соединения* применяет *транспортный протокол*.
сервер	Компьютер или процесс, который предоставляет другим компьютерам или процессам сети доступ к данным, службам или ресурсам.

Планирование сети TCP/IP

Стек протоколов **TCP/IP** - это гибкое средство организации сетевого взаимодействия, поэтому каждый пользователь может настроить его с учетом собственных потребностей. При планировании сети обратите внимание не следующие вопросы. Более подробно эти вопросы обсуждаются в других разделах. Данный список следует рассматривать лишь как общий обзор задач.

1. Выберите аппаратную архитектуру сети: Token-Ring, Ethernet версии 2, IEEE 802.3, оптоволоконный интерфейс распределенных данных (FDDI), Последовательный оптический канал (SOC) или Протокол подключения к Internet по последовательной линии (SLIP).
2. Спланируйте топологию сети. Определите, какие функции будет выполнять каждый компьютер. Например, перед началом прокладки кабелей нужно решить, какие компьютеры будут играть роль шлюзов.
3. Определите, какая структура сети - *одноуровневая* или *иерархическая* - лучше соответствует потребностям вашей организации.

   Если ваша сеть невелика, сосредоточена в одном месте и состоит из одной физической сети, то, вероятно, будет достаточно одноуровневой сети. Если же размер сети очень велик, а также если сеть охватывает несколько физических объектов или состоит из нескольких физических сетей, то более эффективной будет иерархическая структура.

4. Если ваша сеть будет подключена к другим сетям, то заранее составьте план установки и настройки шлюзов. Рассмотрите следующие вопросы:
   a. Решите, какие системы будут служить шлюзами.
   b. Выберите тип маршрутизации: статическая, динамическая или их сочетание. При выборе динамической маршрутизации вам необходимо будет определить, какие программы-демоны маршрутизации должны использоваться каждым шлюзом для поддержки выбранных протоколов связи.
5. Выберите схему адресации.

   Если ваша сеть не будет являться частью сети большего размера, то вы можете воспользоваться любой схемой адресации. Если вы хотите подключить сеть к большой сети, например, Internet, то вам необходимо получить набор официальных адресов у провайдера Internet (ISP).
6. Решите, нужно ли разделить вашу сеть на несколько подсетей. Если да, то необходимо выбрать способ задания масок подсетей.
7. Выберите схему присвоения имен. Каждой системе в сети должно быть присвоено уникальное имя хоста.
8. Решите, нужен ли вам сервер имен или будет достаточно файлов `/etc/hosts`.

   Если вы намерены использовать серверы имен, то определите, какие типы серверов вам нужны и какого числа серверов будет достаточно для вашей сети.
9. Решите, какие службы будут предоставляться удаленным пользователям; например: почта, службы печати, средства удаленного входа в систему, удаленное выполнение команд, совместное использование файлов и т.п.

Установка TCP/IP

В этом разделе рассмотрена процедура установки **TCP/IP**.

Дополнительная информация по установке **TCP/IP** приведена в разделе Установка и миграция.

Настройка TCP/IP

Настройку программного обеспечения **TCP/IP** можно начинать сразу после его установки в системе.

Большинство операций по настройке **TCP/IP** можно выполнить различными способами, а именно:

- С помощью программы SMIT (Инструмент управления системой)
- Изменив соответствующий файл конфигурации
- С помощью команд оболочки

Например, сценарий `rc.net` выполняет необходимую минимальную настройку **TCP/IP** во время запуска системы (сценарий `rc.net` запускается программой управления конфигурацией на втором этапе загрузки). При настройке хоста с помощью SMIT файл `rc.net` настраивается автоматически.

С другой стороны, вы можете изменить файл `/etc/rc.bsdnet` с помощью обычного текстового редактора. Таким образом можно ввести традиционные для UNIX команды конфигурации **TCP/IP**, такие как **ifconfig**, **hostname** и **route**. Для того, чтобы изменить файл конфигурации, введите `smit configtcp` и выберите **Настройка rc для BSD**. Дополнительные сведения о файлах и форматах файлов **TCP/IP** приведены в разделе List of TCP/IP Programming References книги *Communications Programming Concepts*.

Некоторые задачи (например, настройку сервера имен) нельзя выполнить с помощью SMIT.

Настройка хоста

Все компьютеры вашей сети должны быть настроены так, чтобы обеспечить оптимальную работу конечных пользователей и сети в целом.

Для каждого хоста в сети необходимо настроить сетевой интерфейс, задать IP-адрес и имя хоста. Кроме того, нужно настроить статические маршруты к шлюзам и другим хостам, указать, какие

программы-демоны должны запускаться по умолчанию, и создать файл /etc/hosts, применяемый для преобразования имен (или указать, что хост должен обращаться к серверу имен).

Настройка хоста для работы в качестве сервера

Если хост будет выполнять какие-либо специальные функции, например, служить шлюзом, файловым сервером или сервером имен, то после настройки основных параметров нужно выполнить настройку этих функций.

Например, для определения IP-адресов по именам хостов с помощью протокола **DNS** в сети с иерархической структурой должен быть настроен хотя бы один сервер имен.

Помните, что хост, на котором установлен сервер, может выполнять и другие задачи. Если нагрузка на сервер имен в вашей сети невелика, то компьютер, на котором он установлен, может использоваться в качестве рабочей станции или файлового сервера.

Примечание: Если в вашей системе установлена NIS, то эти службы также выполняют преобразование имен.

Настройка шлюза

Если вы планируете соединить вашу сеть с другими сетями, вам потребуется хотя бы один шлюз.

Решите, с какими протоколами связи вы хотите работать, а затем выберите демон маршрутизации (**routed** или **gated**), поддерживающий эти протоколы.

Команды настройки и управления TCP/IP

Для настройки и управления сетью **TCP/IP** доступны несколько различных команд. Они описаны в этой таблице.

Элемент	Описание
arp	Показывает или изменяет таблицы соответствия аппаратных и IP-адресов, применяемые протоколом **преобразования адресов**.
finger	Возвращает сведения о пользователях данного хоста.
host	Показывает IP-адрес данного хоста или имя хоста, соответствующее указанному IP-адресу.
hostname	Показывает или задает имя и IP-адрес локального хоста.
ifconfig	Настраивает сетевые интерфейсы и их параметры.
netstat	Показывает локальные и внешние адреса, таблицы маршрутизации, аппаратную статистику и обзорную информацию о переданных пакетах.
no	Задает или показывает параметры сетевого ядра.
ping	Определяет доступность хоста.
route	Позволяет работать с таблицами маршрутизации вручную.
ruptime	Показывает информацию о состоянии хостов, которые подключены к локальной физической сети и на которых запущен сервер **rwhod**.
rwho	Показывает информацию о состоянии пользователей на хостах, которые подключены к локальной физической сети и на которых запущен сервер **rwhod**.
setclock	Читает сообщения сетевой службы времени и в соответствии с ними устанавливает время и дату на локальном хосте.
timedc	Возвращает информацию о демоне **timed**.
trpt	Создает протоколы трассировки сокетов TCP.
whois	Вызывает службу каталогов имен Internet.

Настройка сети TCP/IP

Приведенные здесь инструкции помогут вам выполнить настройку сети. Обязательно прочтите справочную информацию по выполняемым задачам.

Перед началом процедуры убедитесь в выполнении следующих предварительных требований:

1. Должен быть установлен и подключен сетевой адаптер. Дополнительные сведения об установке и прокладке кабелей для аппаратного обеспечения приведены в разделе “Карты сетевых адаптеров локальной сети TCP/IP” на стр. 159.

2. Должно быть установлено программное обеспечение **TCP/IP**. Дополнительная информация о приведена в разделе *Установка и миграция*.

После настройки и запуска сети используйте эту справочную таблицу для устранения неполадок.

Для настройки сети **TCP/IP** выполните следующие действия:

1. Основные принципы работы **TCP/IP** описаны в разделе “Протоколы TCP/IP” на стр. 120. Вы должны ознакомиться со следующими вопросами:
   - Многоуровневая структура **TCP/IP** (работа различных протоколов на разных уровнях)
   - Передача данных между уровнями
2. Выполните минимальную настройку всех компьютеров, подключенных к сети. Для этого добавьте сетевой интерфейс, определите IP-адрес и имя хоста, а также задайте маршрут по умолчанию к вашей сети. Базовые сведения об этих задачах приведены в разделах “Сетевые интерфейсы TCP/IP” на стр. 162, “Адресация TCP/IP” на стр. 168 и “Присвоение имен хостам в сети” на стр. 176.

   Примечание: Базовую конфигурацию необходимо задать на всех хостах сети, независимо от выполняемых ими функций - будь то обычные клиенты, файловые серверы, шлюзы или серверы имен.
3. Настройте и запустите демон **inetd** на всех хостах сети. Ознакомьтесь с разделом “Демоны TCP/IP” на стр. 359 и выполните инструкции из раздела “Настройка демона inetd” на стр. 360.
4. На каждом хосте укажите способ преобразования имен: с помощью локальной таблицы хостов или с помощью сервера имен. В случае иерархической сети настройте хотя бы один хост в качестве сервера имен. Выполните инструкции из раздела “Преобразование имен” на стр. 178.
5. Если ваша сеть будет подключена к другим сетям, настройте по крайней мере один шлюз. Для обмена данными между сетями шлюз может использовать статические маршруты или демон маршрутизации. Выполните инструкции из раздела “Маршрутизация TCP/IP” на стр. 362.
6. Определите, какие службы должны быть предоставлены хостам. По умолчанию доступны все службы. Для того чтобы запретить доступ к какой-либо службе, выполните инструкции из раздела “Сетевые службы клиента” на стр. 361.
7. Решите, какие хосты в сети будут играть роль серверов, и какие функции будет выполнять каждый сервер. Для запуска выбранных демонов сервера выполните инструкции из раздела “Сетевые службы сервера” на стр. 361.
8. При необходимости настройте серверы удаленной печати. Дополнительная информация приведена в разделе Printing administration книги *Принтеры и печать*.
9. **Необязательно:** Один из хостов сети можно сделать главным сервером времени, а остальные настроить таким образом, чтобы они синхронизировали свои часы с этим хостом. Дополнительные сведения приведены в описании демона **timed** в книге *Справочник по командам, том 5*.

Идентификация и защищенные rcmds

Теперь у этих команд появились дополнительные способы идентификации.

Защищенными командами rcmds являются: **rlogin**, **rcp**, **rsh**, **telnet** и **ftp**. По умолчанию эти команды используют стандартный способ идентификации *AIX*. Два дополнительных способа идентификации - Kerberos V.5 и Kerberos V.4.

Если для идентификации применяется Kerberos V.5, то клиент получает паспорт Kerberos V.5 от сервера защиты DCE или стандартного сервера Kerberos. Паспорт - это зашифрованная часть текущей идентификационной информации пользователя DCE или стандартного сервера Kerberos, предназначенная для сервера **TCP/IP**, с которым необходимо установить соединение. Программа-демон сервера **TCP/IP** расшифровывает полученный паспорт. Такой способ позволяет серверу **TCP/IP** абсолютно точно идентифицировать пользователя. Если описанному в паспорте субъекту DCE или стандартного сервера Kerberos разрешен доступ к учетной записи пользователя операционной системы, то соединение устанавливается.

Примечание: Начиная с распределенной вычислительной среды (DCE) версии 2.2, сервер защиты DCE может возвращать паспорта Kerberos версии 5. Защищенные команды rcmds о операционной системе AIX используют библиотеку Kerberos версии 5 и библиотеку GSSAPI, предоставляемую NAS (Служба сетевой идентификации) версии 1.3.

Kerberos V.5 выполняет идентификацию клиента и передает текущую идентификационную информацию пользователя серверу **TCP/IP**. Если для данного пользователя передача идентификационной информации разрешена, то клиент передает ее серверу как особый вид паспорта Kerberos (так называемый паспорт TGT). Программа-демон **TCP/IP**, взаимодействующая с сервером защиты DCE, преобразует TGT в полную идентификационную информацию DCE с помощью команды **k5dcecreds**.

Способ идентификации, применяемый в команде **ftp**, отличается от способов идентификации остальных команд. Для передачи идентификационной информации между командой **ftp** и демоном **ftpd** применяется механизм защиты GSSAPI. Клиент **ftp** шифрует данные с помощью команд **clear**/**safe**/**private**.

Теперь команда **ftp** позволяет клиентам и серверам операционной системы передавать многобайтовые символы по зашифрованному каналу передачи данных. В стандартных реализациях этой команды поддерживается передача только однобайтовых символов. Если соединение с шифрованием данных устанавливается с системами других производителей, то **ftp** поддерживает передачу только однобайтовых символов.

Примечание: Защищенные команды rcmds **rlogin**, **rsh** и **telnet** вместе с методами идентификации **klogin** и **kshell** Kerberos V.5 допускают три попытки входа, после которых соединение с удаленным хостом закрывается.

Настройка системы для защищенных rcmds

Для всех защищенных команд группы rcmds предусмотрен единый механизм настройки системы, позволяющий указать разрешенные способы идентификации. Настройка выполняется как для соединений данной системы с другими хостами, так и для соединений с самой этой системой.

Настройка средств идентификации обеспечивается библиотекой `libauthm.a` и двумя командами **lsauthent** и **chauthent**, которые вызываются из командной строки и обращаются к двум библиотечным процедурам: **get_auth_methods** и **set_auth_methods**.

Система поддерживает три различных способа идентификации: Kerberos V.5, Kerberos V.4 и *стандартный способ идентификации AIX*. Эти способы определяют, как выполняется идентификация пользователей в сети.

- Kerberos V.5 - это основной и наиболее популярный способ идентификации в Распределенной вычислительной среде (DCE). Операционная система либо преобразует полученные паспорта Kerberos V.5 в полные идентификационные данные DCE, либо непосредственно применяет полученные стандартные паспорта Kerberos V.5.
- Kerberos V.4 применяется только двумя защищенными командами из группы rcmds: **rsh** и **rcp**. Поддержка этого способа обеспечивается только в системах SP для обратной совместимости. Паспорт Kerberos V.4 не преобразуется в полную идентификационную информацию DCE.
- *Стандартный способ идентификации AIX* - это способ идентификации, который применялся в AIX.

Если в конфигурации указано одновременно несколько способов идентификации, то они будут применяться в заданном порядке. Если с помощью первого способа соединение установить не удалось, то клиент попытается выполнить идентификацию следующим способом, указанным в конфигурации.

Порядок применения способов идентификации может быть любым. Однако *стандартный способ идентификации AIX* всегда должен быть указан последним, так как в нем не предусмотрена опция перехода к следующему способу. Если в конфигурации не указан *стандартный способ идентификации AIX*, то идентификация с помощью пароля не применяется, а все запросы на установление соединения с применением этого способа идентификации отклоняются.

В конфигурации системы можно вообще не указывать способы идентификации. В этом случае все запросы на установление соединений с другими компьютерами с помощью защищенных команд группы rcmds будут отклоняться. Кроме того, поскольку Kerberos V.4 поддерживается только командами **rsh** и **rcp**, то в системах, в которых применяется только Kerberos V.4, нельзя будет установить соединения с помощью команд **telnet**, **ftp** и **rlogin**.

Информация, связанная с данной:

Функция get_auth_method

Функция set_auth_method

lsauthent, команда

chauthent, команда

Идентификация пользователя Kerberos V.5 для защиты rcmds

Если для идентификации применяется способ Kerberos V.5, то клиент **TCP/IP** получает от службы идентификации зашифрованный паспорт, предназначенный для сервера **TCP/IP**. Сервер расшифровывает паспорт и идентифицирует пользователя (как субъекта DCE или стандартного Kerberos) с помощью специального защищенного метода.

После этого ему необходимо узнать, разрешен ли данному субъекту DCE или стандартного Kerberos доступ к локальной учетной записи пользователя. Сопоставление данного субъекта DCE или стандартного Kerberos с учетной записью пользователя в локальной операционной системе выполняется общей библиотекой, `libvaliduser.a`, в которой предусмотрена единая процедура - **kvalid_user**. Если применяется другой способ идентификации, системный администратор должен заменить библиотеку `libvaliduser.a`.

Настройка DCE для защищенных rcmds

Для установления соединения с каким-либо сетевым интерфейсом с помощью защищенных удаленных команд необходимо создать для него два субъекта DCE.

Значения этих полей приведены ниже:

```
host/полное-имя-интерфейса
ftp/полное-имя-интерфейса
```

, где *полное-имя-интерфейса* - это имя интерфейса и имя домена, соответствующие основному домену *имя-хоста.имя-домена*.

Стандартная конфигурация для защиты rcmds

Для установления соединения с каким-либо сетевым интерфейсом с помощью защищенных удаленных команд необходимо создать для него два субъекта.

Значения этих полей приведены ниже:

```
host/полное-имя-интерфейса@имя-области
ftp/полное-имя-интерфейса@имя-области
```

где *полное-имя-интерфейса* - это имя интерфейса и имя домена, соответствующие основному домену `имя-хоста.имя-домена`. *Имя-области* - это имя области Kerberos V.

Настройка TCP/IP

Для настройки **TCP/IP** создайте файл `.netrc`.

В файле `.netrc` содержится информация для автоматического входа в систему для команд **ftp** и **rexec**. Также можно создать новые макросы **ftp**, задаваемые в файле `$HOME/.netrc`. Ниже приведены инструкции по созданию и редактированию файла `$HOME/.3270keys` для настройки функций клавиш и их сочетаний: Кроме того, в файле `.k5login` перечислены кластеры и субъекты DCE, которым разрешен доступ к учетному файлу пользователя.

Создание файла .netrc

Ниже приведены инструкции по созданию и редактированию файла $HOME/.netrc:

1. В системе должна быть копия файла /usr/samples/tcpip/netrc.
2. В системе не должна работать команда **securetcpip**.

Для создания файла .netrc выполните следующие действия:

1. Скопируйте файл /usr/samples/tcpip/netrc в каталог $HOME с помощью следующей команды:
   ```
   cp /usr/samples/tcpip/netrc $HOME
   ```
2. Отредактируйте в файле $HOME/netrc значения переменных *HostName*, *LoginName* и *Password*. Например:
   ```
   machine host1.austin.century.com login fred password bluebonnet
   ```
3. Для того чтобы задать права доступа в файле $HOME/netrc равным 600 с помощью команды **chmod** введите в командной строке($):
   ```
   chmod 600 $HOME/.netrc
   ```
4. Измените имя файла $HOME/netrc на $HOME/.netrc. Точка (.) в начале имени файла означает, что файл будет скрытым.
   ```
   mv $HOME/netrc $HOME/.netrc
   ```

Файл $HOME/.netrc может содержать несколько определений входа в систему, причем в каждом определении может быть не более 16 макрокоманд.

Создание макрокоманд ftp

Ниже описаны шаги по созданию макрокоманды **ftp**.

Должен быть создан файл $HOME/.netrc.

Для создания макрокоманды **ftp** выполните следующие действия:

1. Добавьте в файл $HOME/.netrc следующие инструкции:
   ```
   macdef init
   put schedule
   ```
   В конце определения макрокоманды **ftp** обязательно должна стоять пустая строка. Пустая строка завершает макрокоманду **ftp**. В приведенном выше примере команда **macdef** определяет макрокоманду init. В следующей строке указывается действие, выполняемое макрокомандой, в данном случае это put *schedule*, где *schedule* - имя файла.
2. После создания макрокоманды **ftp** введите в командной строке:
   ```
   ftp имя-хоста
   ```
   , где *имя-хоста* - это имя хоста, к которому вы хотите подключиться. Команда **ftp** ищет в файле $HOME/.netrc определение входа в систему, совпадающее с именем хоста, и выполняет с его помощью вход в систему.
3. После входа в систему введите в командной строке:
   ```
   ftp init
   ```
   В данном примере **ftp** найдет макрокоманду init и выполнит указанные в ней действия.

   Макрокоманда **ftp** связывается с тем пользователем, имя которого указано в файле непосредственно перед ней. Определения макрокоманд **ftp** не распространяется на весь файл $HOME/.netrc. Макрокоманда init автоматически выполняется при входе в систему. Другие макрокоманды можно выполнять из командного приглашения **ftp** (ftp>) следующим образом:
   ```
   $getit
   ```
   В данном примере $ выполняет макрокоманду **ftp getit**.

Изменение присваивания набора ключей

При оптимизации **TCP/IP** можно использовать эту процедуру для изменения функций и последовательностей ключей.

1. Вы должны уметь работать с редактором **vi**.
2. В системе должен быть установлен редактор vi.

Ниже приведены инструкции по созданию и редактированию файла `$HOME/.3270keys` для настройки функций клавиш и их сочетаний:

1. Скопируйте файл `/etc/3270.keys` в каталог `$HOME` и переименуйте его в `.3270keys`. Для этого выполните следующую команду:

   ```
   cp /etc/3270.keys $HOME/.3270keys
   ```

2. Выполните следующие действия, чтобы изменить операторы связывания в файле `$HOME/.3270keys`, присвоив клавишам нужные функции:
   a. Создайте в редакторе vi новый файл и перейдите в режим вставки.
   b. Нажмите клавиши Ctrl-V, а затем клавишу, действие которой вы хотите задать. Появится значение, соответствующее нажатой клавише.
   c. Введите указанное значение в нужной строке столбца `Sequence` файла `$HOME/.3270keys`.

   Например, открыв окно редактора vi и перейдя в режим вставки, нажмите Ctrl-V и затем Alt-Insert. Появится значение `[[141q`. В столбце `Sequence` первый символ `[` заменяется на `\e`, как показано ниже:

   ```
   Комбинация клавиш функции 3270
   bind   pa1           "\e[141q"  #a_insert
   ```

.k5login, файл:

Файл `.k5login` применяется при выполнении защищенных команд группы rcmds для идентификации с помощью Kerberos V.5. В файле перечислены кластеры и субъекты DCE, которым разрешен доступ к учетному файлу пользователя.

Полное имя файла - `$HOME/.k5login`. Он должен принадлежать локальному пользователю с правами доступа к файлу. Минимальным уровнем доступа для этого файла является 400.

Файл `.k5login` содержит список пар субъект/кластер DCE, которым разрешен доступ к учетному файлу пользователя. Пары субъект/кластер сохраняются в формате Kerberos (отличном от формата DCE). Например, если в файле указана пара

```
UserA@Cell1
```

то субъекту DCE `UserA` из кластера DCE `Cell1` разрешен доступ к учетному файлу пользователя.

Если имя субъекта DCE совпадает с именем учетной записи пользователя, а для учетной записи пользователя не создан файл `$HOME/.k5login`, то субъект DCE получает доступ к ресурсам пользователя (при условии, что настроена идентификация Kerberos V.5).

Дополнительная информация об идентификации с помощью Kerberos V.5 приведена в разделе “Идентификация и защищенные rcmds” на стр. 106.

Способы организации взаимодействия с другой системой или пользователем

Существует несколько способов организации взаимодействия с другой системой или пользователем. В данном разделе описаны два возможных способа. Во-первых, можно установить соединение между локальным и удаленным хостами. Второй способ - это диалог с удаленным пользователем.

Подключение локального хоста к удаленному хосту

Эти команды **TCP/IP** подключения к хосту предназначены для удаленного входа в систему и выполнения в ней команд.

В некоторых случаях может потребоваться доступ к чужому компьютеру. Например, системному администратору может потребоваться изменить права доступа к важным файлам, с которыми вы работаете, или вам нужно получить доступ к собственным файлам с чужой рабочей станции. Можно даже подключиться к своему компьютеру с другой рабочей станции. Функции удаленного входа в систему, например, команды **rlogin**, **rexec** и **telnet**, применяют локальный хост в качестве терминала ввода-вывода. Вводимые с клавиатуры символы передаются удаленному хосту, а результат отображается на локальном мониторе. После завершения сеанса удаленной работы с системой локальный хост переходит в обычный режим работы.

В Протоколе **TCP/IP** для удаленного входа в систему и выполнения в ней команд предусмотрены следующие команды:

Элемент	Описание
rexec	Команда **rexec** позволяет в интерактивном режиме выполнять команды на других хостах, на которых вы зарегистрировались с помощью команды **rlogin**. В сетях с повышенными требованиями к защите системные администраторы обычно запрещают применять эту команду. При выполнении команды **rexec** локальный хост ищет имя пользователя и пароль в файле `$HOME/.netrc` на удаленном хосте. Если они будут найдены, то запрошенная локальным хостом команда будет выполнена. В противном случае, перед обработкой запроса вы должны будете указать имя пользователя и пароль.
rlogin	Команда **rlogin** предназначена для входа в систему на удаленных хостах. В отличие от команды **telnet**, используемой для различных удаленных хостов, команда **rlogin** используется только для хостов UNIX. В сетях с повышенными требованиями к защите системные администраторы обычно запрещают применять эту команду. Команда **rlogin**, как и команда **telnet**, позволяет локальному хосту подключиться к удаленному хосту. Единственное отличие состоит в том, что команда **rlogin** не считается защищенной и может быть запрещена системным администратором, если к защите сети предъявляются повышенные требования. Команда **rlogin** не относится к классу защищенных, поскольку и файл `$HOME/.rhosts`, принадлежащий локальному пользователю, и файл `/etc/hosts.equiv`, принадлежащий администратору системы, содержат списки удаленных хостов, которым разрешен доступ к локальному хосту. Таким образом, если вы оставите рабочую станцию без присмотра, то посторонний пользователь сможет просмотреть имена и пароли, перечисленные в этих файлах. В идеальном случае при вводе команды **rlogin** удаленные пользователи должны вводить пароль, но это требование легко обойти. Если имя хоста, на котором пользователь пытается войти в систему, не указано в файлах `$HOME/.rhosts` и `/etc/hosts.equiv`, то локальный хост предлагает ввести пароль. Для проверки введенного пароля применяется удаленный файл паролей; если пароль указан неверно, то приглашение для ввода пароля появляется еще раз. Нажатие тильды и точки (~.) в приглашении входа в систему прерывает процедуру удаленного входа в систему. Систему можно настроить таким образом, чтобы при выполнении команды **rlogin** для идентификации пользователей применялось средство Kerberos V.5. При этом идентификация пользователей будет выполняться без помощи файла `$HOME/.rhosts` и без отправки пароля по сети. Дополнительная информация о применении команды **rlogin** приведена в разделе “Идентификация и защищенные rcmds” на стр. 106.

Элемент	Описание
rsh и **remsh**	Команды **rsh** и **remsh** предназначены для запуска команд на удаленных хостах, аналогичных локальному. Все необходимые входные данные должны быть предоставлены удаленным хостом. В сетях с повышенными требованиями к защите системные администраторы запрещают применять команды **rsh** и **remsh**. Действие команды **rsh** зависит от заданных параметров: • Если указано имя команды, то эта команда выполняется на удаленном хосте. • Если имя команды не указано, то выполняется команда **rlogin**. При выполнении команды **rsh** локальный хост просматривает файл `/etc/hosts.equiv` на удаленном хосте и определяет, разрешен ли ему вход в систему. Если нет, то просматривается файл `$HOME/.rhosts`. В обоих файлах перечислены удаленные хосты, которым разрешен вход в систему. При запуске команды **rsh** удаленные пользователи должны вводить пароль. В некоторых случаях вводить команду **rlogin** не нужно. Команда **rsh** позволяет выполнять команды на удаленном хосте, но при этом сохраняется защита с помощью паролей. Если для доступа к удаленному хосту необходим пароль, то при запуске команды **rsh** также нужно будет ввести пароль, поскольку обе эти команды обращаются к файлам `$HOME/.rhosts` и `/etc/hosts.equiv`. Систему можно настроить таким образом, чтобы при выполнении команды **rsh** для идентификации пользователей применялось средство Kerberos V.5. При этом идентификация пользователей будет выполняться без помощи файла `$HOME/.rhosts` и без отправки пароля по сети. Дополнительная информация о применении команды **rsh** приведена в разделе “Идентификация и защищенные rcmds” на стр. 106.
telnet, **tn** и **tn3270**	Команда **telnet** предназначена для запуска программы эмуляции терминала, которая реализует протокол TELNET и позволяет работать с различными удаленными хостами. В качестве протокола связи применяется TCP/IP. **Примечание:** Для удобства здесь и далее **telnet** будет обозначать все три команды **telnet**, **tn** и **tn3270**. Команда **telnet** - это один из способов входа в удаленную систему. По сравнению с другими способами у команды **telnet** есть одно важное преимущество - это *защищенная* команда. Команда **rlogin**, которая также обеспечивает удаленный вход в систему, не считается защищенной командой. Для предотвращения доступа пользователей к файлам без соответствующих прав доступа, незаконного копирования важных данных, удаления файлов, а также для защиты от вирусов может потребоваться усиленная защита системы. Для обеспечения такой защиты в TCP/IP предусмотрены специальные функции. Пользователь, выполнивший команду **telnet** для входа в удаленную систему, должен указать имя пользователя и пароль этого пользователя на данном компьютере. Эти действия аналогичны процедуре входа в систему локального хоста. После того, как пользователь успешно вошел в систему удаленного хоста, его терминал начинает действовать так, как будто он непосредственно подключен к хосту. Команда **telnet** поддерживает *согласование типов терминалов*. Если удаленный хост также поддерживает согласование типов терминалов, то команда **telnet** сообщает ему тип локального терминала. Если удаленный хост не поддерживает указанный тип терминала, то команда **telnet** пытается переключиться в режим эмуляции терминала 3270 или DEC VT100. Если вы явно указали тип терминала для эмуляции, то команда **telnet** не согласовывает его. Если локальный и удаленный хосты не могут согласовать тип терминала, то по умолчанию локальный хост применяет значение **none** (нет). Команда **telnet** поддерживает следующие типы терминалов 3270: 3277-1, 3278-1, 3278-2, 3278-3, 3278-4 и 3278-5. Если вы применяете команду **telnet** в режиме 3270 с цветным дисплеем, то по умолчанию цвета и поля отображаются, как на экране 3279. Вы можете установить другие цвета, отредактировав файл раскладки клавиатуры для одного из перечисленных выше типов терминалов. После завершения сеанса **telnet** на экране восстанавливается палитра цветов, которая применялась до начала сеанса. Систему можно настроить таким образом, чтобы при выполнении команды **telnet** для идентификации пользователей применялось средство Kerberos V.5. При этом идентификация пользователей будет выполняться без помощи файла `$HOME/.rhosts` и без отправки пароля по сети. Дополнительная информация о применении команды **telnet** приведена в разделе “Идентификация и защищенные rcmds” на стр. 106.

Примечание: Команды **rsh** и **rexec** предназначены для запуска команд на удаленном хосте, но они не являются защищенными и могут не отвечать всем требованиям, предъявляемым к уровню защиты системы. В связи с этим, в случае повышенных требований к защите применение этих команд может быть запрещено.

Вход в систему на удаленном хосте

Для входа в систему на удаленном хосте введите команду **telnet**.

Для доступа к удаленному хосту необходим ИД пользователя и пароль.

Для того чтобы войти в систему удаленного хоста (в данном примере это хост `host1`), введите следующую команду:

```
telnet host1
```

На экране будет показана примерно следующая информация:

```
Trying . . .
Connected to host1
Escape character is '^T'.

AIX telnet (host1)

AIX Operating System
Версия 7.1
(/dev/pts0)
login:_
```

После входа в систему вы можете выполнять необходимые команды. Для выхода из системы и завершения соединения нажмите клавиши Ctrl-D.

Если войти в систему не удается, то вы можете прервать соединение, нажав клавиши Ctrl-T.

Диалог с удаленным пользователем

Команда **talk** предназначена для диалога с пользователем удаленного хоста в режиме реального времени.

1. На локальном и удаленном хостах должен быть запущен демон **talkd**.
2. Пользователь удаленного хоста должен работать в системе.

В команде **talk** необходимо указать адрес, по которому следует установить связь. Имя хоста удаленной системы должно быть связано с работающим сетевым интерфейсом, с которым могут работать другие сетевые команды, например, команда **ping**. Если у системы нет сетевого интерфейса, т.е. это автономная система, то для выполнения команды **talk** ее имя связывается с циклическим адресом (127.0.0.1).

Электронная почта предназначена для обмена текстовыми сообщениями с другими пользователями сети. Если система настроена правильно и известен адрес получателя, то можно отправить электронное сообщение в другую удаленную систему, находящуюся в любой точке мира.

Для связи с удаленными хостами в **TCP/IP** предусмотрены следующие команды:

Элемент	Описание
mail	Отправляет и получает электронные письма
talk	Позволяет вести интерактивный диалог с пользователем удаленного хоста

1. Для установления диалога с пользователем `dale@host2`, который вошел в систему на удаленном хосте, пользователь `jane@host1` вводит следующую команду:

   ```
   talk dale@host2
   ```

   На экране `dale@host2` появляется примерно следующее сообщение:

   ```
   Сообщение от TalkDaemon@host1 at 15:16...
   talk: запрос на
   подключение от jane@host1.
   talk: ответ от: talk jane@host1
   ```

   Это сообщение информирует `dale@host2` о том, что `jane@host1` пытается начать диалог.

2. Для того чтобы принять предложение, dale@host2 вводит следующую команду:
   ```
   talk jane@host1
   ```

   Теперь пользователи dale@host2 и jane@host1 могут вести диалог.
3. Для завершения диалога любой из пользователей может в любой момент времени нажать Ctrl-C. При этом вновь будет показано окно командной строки.

Передача файлов

Несмотря на то, что сравнительно небольшие файлы можно передавать с помощью электронной почты, для больших файлов существуют более эффективные способы передачи.

Программы электронной почты обычно рассчитаны на обмен сравнительно небольшими текстовыми сообщениями, поэтому для эффективной передачи больших файлов нужны другие средства. Команды **ftp**, **rcp** и **tftp** основаны на **TCP/IP** и устанавливают непосредственное соединение между локальным и удаленным хостом. Основные сетевые утилиты (BNU) также могут устанавливать непосредственные соединения с внешними хостами с помощью **TCP/IP**.

Передача файлов с помощью команд ftp и rcp

Команда **ftp** позволяет скопировать файл с удаленного хоста. При этом команда **ftp** не сохраняет атрибуты файлов и не копирует вложенные каталоги. Копирование атрибутов и вложенных каталогов поддерживается командой **rcp**.

Элемент	Описание
ftp	Передает файлы между хостами с различными операционными системами или с разным представлением символов (например EBCDIC и ASCII) с помощью **Протокола передачи файлов (FTP)**. Эта команда обеспечивает защиту с помощью отправки пароля удаленному хосту, а также поддерживает автоматический вход в систему, передачу файлов и выход из системы.
rcp	Копирует один или несколько файлов. Применяется при обмене файлами между локальным и удаленным хостом, двумя удаленными хостами или для копирования файлов на удаленном хосте. Эта команда аналогична команде **cp**, но она применяется только для операций с удаленными файлами. В сетях с повышенными требованиями к защите системные администраторы могут запретить применять эту команду.

Прежде чем осуществлять передачу файла с помощью команд **ftp** и **rcp**, убедитесь в выполнении следующих условий:

1. Если вы собираетесь применять функцию автоматического входа в систему, то в файле удаленного хоста $HOME/.netrc для вашего хоста должно быть задано разрешение на удаленный вход в систему. Если разрешение не задано, то вы должны знать имя пользователя и пароль для удаленного хоста. Дополнительная информация о файле .netrc приведена в разделе “Создание файла .netrc” на стр. 109.

   В конфигурации системы может быть также задан способ идентификации Kerberos V.5. Он применяется вместо файлов .netrc и $HOME/.rhosts. См. раздел “Идентификация и защищенные rcmds” на стр. 106.
2. Для копирования файла с удаленного хоста у вас должны быть права доступа на чтение этого файла.

   Примечание: Права доступа на чтение и запись файлов и каталогов удаленного хоста зависят от имени, указанного при входе в систему.
3. Для копирования файла с локального хоста на удаленный хост необходимы права доступа на запись в каталог, в котором будет храниться копия файла. Кроме того, если в каталоге удаленного хоста уже есть файл с тем же именем, что и имя копируемого файла, то у вас должны быть права доступа на запись и добавление данных в файл на удаленном хосте.

Вход в удаленный хост напрямую:

При использовании **TCP/IP** для передачи файлов можно воспользоваться этой процедурой для входа в удаленный хост напрямую.

1. С помощью команды **cd** перейдите в каталог, содержащий файл, который вы хотите отправить (при отправке файла), или в каталог, где должен быть расположен переданный файл (при получении файла).

2. Войдите в систему на удаленном хосте с помощью команды:

   ```
   ftp имя-хоста
   ```

   Если вам разрешен автоматический вход в систему, то будет показана приблизительно следующая информация:

   ```
   Connected to canopus.austin.century.com.
   220 canopus.austin.century.com FTP server
   (Version 4.1 Sat Nov 23 12:52:09 CST 1995) ready.
   331 Password required for dee.
   230 User dee logged in.
   ftp>
   ```

   Если разрешения нет, то появится приблизительно следующая информация:

   ```
   Connected to canopus.austin.century.com.
   220 canopus.austin.century.com FTP server
   (Version 4.1 Sat Nov 23 12:52:09 CST 1995) ready.
   Name (canopus:eric): dee
   331 Password required for dee.
   Password:
   230 User dee logged in.
   ftp>
   ```

3. После появления соответствующих приглашений введите имя пользователя и пароль.

Теперь вы можете передавать файлы между хостами.

Вход в удаленный хост не напрямую:

При использовании **TCP/IP** для передачи файлов можно воспользоваться этой процедурой для входа в удаленный хост не напрямую.

1. С помощью команды **cd** перейдите в каталог, содержащий файл, который вы хотите отправить (при отправке файла), или в каталог, где должен быть расположен переданный файл (при получении файла).
2. Войдите в систему на удаленном хосте с помощью команды:

   ```
   ftp
   ```

3. При появлении приглашения `ftp>` введите:

   open *имя-хоста*

   Если вам разрешен автоматический вход в систему, то будет показана приблизительно следующая информация:

   ```
   Connected to canopus.austin.century.com.
   220 canopus.austin.century.com FTP server
   (Version 4.1 Sat Nov 23 12:52:09 CST 1995) ready.
   331 Password required for dee.
   230 User dee logged in.
   ftp>
   ```

   Если разрешения нет, то появится приблизительно следующая информация:

   ```
   Connected to canopus.austin.century.com.
   220 canopus.austin.century.com FTP server
   (Version 4.1 Sat Nov 23 12:52:09 CST 1995) ready.
   Name (canopus:eric): dee
   331 Password required for dee.
   Password:
   230 User dee logged in.
   ftp>
   ```

4. После появления соответствующих приглашений введите имя пользователя и пароль.

Копирование файла с удаленного хоста на локальный:

Команда **ftp** позволяет скопировать файл с удаленного хоста в локальный.

Для копирования файла с удаленного хоста на локальный с помощью команды **ftp** необходимо сначала войти в систему удаленного компьютера. Необходимые инструкции можно найти в разделах Непосредственный вход в систему удаленного хоста и Косвенный вход в систему удаленного хоста.

Примечание: При выполнении команды **ftp** для копирования файлов применяется тип передачи ASCII, заданный по умолчанию.

Для копирования файла с удаленного хоста на локальный выполните следующие действия:

1. С помощью команды **dir** убедитесь, что файл, который необходимо скопировать, находится в текущем каталоге. (Подкоманда **dir**, предусмотренная для команды **ftp**, работает аналогично команде **ls -l**.) Если файла в этом каталоге нет, с помощью подкоманды **cd** перейдите в нужный каталог.
2. Для копирования локального файла в двоичном формате, введите следующую команду:
   ```
   опции-переключатели
   ```
3. Для копирования файла на свой хост введите команду:
   ```
   get FileName
   ```
   Файл будет помещен в каталог, который был текущим на момент выполнения команды **ftp**.
4. Для завершения сеанса нажмите Ctrl-D или введите команду `quit`.

Копирование файла из локального хоста в удаленный:

Команда **ftp** позволяет скопировать файл с локального хоста в удаленный.

Для копирования файла с локального хоста на удаленный с помощью команды **ftp** необходимо войти в систему удаленного компьютера. Необходимые инструкции можно найти в разделах Непосредственный вход в систему удаленного хоста и Косвенный вход в систему удаленного хоста.

Примечание: По умолчанию команда **ftp** копирует файлы в формате ASCII.

Для копирования файла с локального хоста на удаленный выполните следующие действия:

1. Если необходимо поместить файл в каталог, отличный от `$HOME`, то перейдите в необходимый каталог с помощью команды **cd**.
2. Для копирования локального файла в двоичном формате, введите следующую команду:
   ```
   опции-переключатели
   ```
3. Для копирования файла на удаленный хост введите команду:
   ```
   put файл
   ```
4. Для завершения сеанса нажмите Ctrl-D или введите команду `quit`.

Передача файлов с помощью команд tftp и utftp

Команды **tftp** и **utftp** предназначены для обмена файлами с удаленными хостами с помощью **Упрощенного протокола передачи файлов** (**TFTP**).

Так как протокол **TFTP** предусматривает одновременную передачу только одного файла, команды **tftp** и **utftp** поддерживают не все функции команды **ftp**. В сетях с повышенными требованиями к защите системные администраторы могут запретить применять эти команды.

Примечание: Команда **tftp** не применяется в системах с высоким уровнем защиты.

Прежде чем осуществлять передачу файла с помощью команд **tftp** и **utftp**, убедитесь в выполнении следующих условий:

1. Для копирования файла *с* удаленного хоста необходимы права доступа на *чтение* из каталога, в котором находится этот файл.
2. Для копирования файла *на* удаленный хост необходимы права на *запись* в каталог, в котором будет находиться этот файл.

Копирование файла с удаленного хоста:

При использовании **TCP/IP** для копирования файлов можно использовать данную процедуру для копирования файла с удаленного хоста.

1. Для установления соединения с удаленным хостом введите следующую команду:

   ```
   tftp host1
   ```

   В данном примере `host1` - это имя хоста, к которому вы хотите подключиться.

   Появится приглашение `tftp>`.

2. Проверьте наличие соединения с помощью команды:

   ```
   состояние
   ```

   Будет показано примерно следующее сообщение:

   ```
   Connected to host1
   Mode: netascii Verbose: off Tracing: off
   Remxt-interval: 5 seconds, Max-timeout: 25 seconds
   tftp>
   ```

3. Введите подкоманду **get**, имя передаваемого файла и имя, которое будет присвоено файлу после передачи:

   ```
   get /home/alice/update update
   ```

   При этом необходимы права на чтение в каталоге `/home/alice` на удаленном хосте. В данном примере файл `/home/alice/update` перемещается с хоста `host1` в файл `update` в текущем каталоге локальной системы.

4. Для завершения сеанса введите команду:

   ```
   quit
   ```

   или нажмите Ctrl-D.

Копирование файла на удаленный хост:

При использовании **TCP/IP** для копирования файлов можно использовать данную процедуру для копирования файла на удаленный хост.

1. Для установления соединения с удаленным хостом введите следующую команду:

   ```
   tftp host1
   ```

   В данном примере `host1` - это имя хоста, к которому вы хотите подключиться.

   Появится приглашение `tftp>`.

2. Проверьте наличие соединения с помощью команды:

   ```
   состояние
   ```

   Будет показано примерно следующее сообщение:

   ```
   Connected to host1
   Mode: netascii Verbose: off Tracing: off
   Remxt-interval: 5 seconds, Max-timeout: 25 seconds
   tftp>
   ```

3. Введите подкоманду **put**, имя локального файла, который необходимо передать, путь к файлу и имя, которое будет присвоено файлу в удаленной системе:

   ```
   в ut мой-файл /home/alice/ваш-файл
   ```

   При этом необходимы права на запись в каталог `/home/alice` на удаленном хосте. Файл `myfile`, расположенный в текущем рабочем каталоге пользователя, передается на хост `host1`. Путь к файлу необходимо указывать в том случае, если не задано значение по умолчанию. Файл `myfile` будет скопирован в удаленную систему под именем `yourfile`.

4. Для завершения сеанса введите команду:

   ```
   quit
   ```

или нажмите клавиши Ctrl-D.

Печать на удаленном принтере

Если к вашему хосту подключен локальный принтер, то с помощью приведенной в этом разделе информации вы сможете печатать на удаленном принтере. Кроме того, если локального принтера нет, то вы сможете печатать на удаленном принтере, отличном от заданного по умолчанию.

1. Имя вашего хоста должно быть указано в файле `/etc/hosts.lpd` удаленного хоста.

   Примечание: Система постановки в очередь не поддерживает имена хостов, содержащие многобайтовые символы.

   Для того чтобы применить изменения, внесенные в файл `/etc/hosts.lpd`, без перезагрузки системы, воспользуйтесь командой **refresh** Контроллера системных ресурсов (SRC).
2. Вы должны уметь находить имя очереди и имя удаленного принтера в локальном файле `/usr/lib/lpd/qconfig`

Эту задачу можно выполнить с помощью команды **enq** или инструмента управления системой (SMIT).

> **Примечание:** В данной главе описаны лишь основные принципы удаленной печати. Дополнительная информация об удаленной печати приведена в описании команды **enq**.

Помещение задания в удаленную очередь печати

При использовании **TCP/IP** для печати файлов эту процедуру можно применить для размещения задания в очереди удаленной печати.

Для того чтобы поместить задание в удаленную очередь печати, имя локального хоста должно быть указано в файле `/etc/hosts.lpd` на удаленном хосте (система обслуживания очереди не поддерживает имена хостов с многобайтовыми символами). Для того чтобы применить изменения, внесенные в файл `/etc/hosts.lpd`, без перезагрузки системы, воспользуйтесь командой **refresh** Контроллера системных ресурсов (SRC). Вы также должны определить имя очереди и удаленного принтера в локальном файле `/usr/lib/lpd/qconfig`

1. Найдите имя нужной очереди и имя удаленного устройства. Имя очереди обычно начинается с префикса `rp`, за которым следует одна или несколько цифр. Имя удаленного принтера обычно начинается с префикса `drp`, за которым следует одна или несколько цифр.
2. Введите следующую команду:

   `enq -P` *`имя-очереди:имя-принтера имя-файла`*

   где *имя-очереди* - имя очереди, например `rp1`, а *имя-принтера* - это имя принтера (например, `drp1`), указанные в файле `/usr/lib/lpd/qconfig`. Не забудьте указать двоеточие (`:`) между параметрами *очередь* и *принтер*. *файл* - это имя печатаемого файла.

Ниже приведены примеры применения команды **enq**:

- Для печати файла `memo` на принтере по умолчанию введите следующую команду:

  ```
  enq memo
  ```
- Для печати файла `prog.c` с номерами страниц введите следующую команду:

  ```
  pr prog.c | enq
  ```

  Команда **pr** помещает в начало каждой страницы колонтитул с указанием даты последнего изменения файла, именем файла, а также числом страниц. Затем команда **enq** печатает файл.
- Для печати файла `report` на следующем доступном принтере, настроенном в очереди `fred`, введите следующую команду:

  ```
  enq -P fred report
  ```
- Для печати нескольких файлов, имена которых начинаются с символов `sam`, на следующем доступном принтере, настроенном в очереди `fred`, введите следующую команду:

  ```
  enq -P fred sam*
  ```

Будут напечатаны все файлы, имена которых начинаются с префикса `sam`. Обычные команды состояния показывают только заголовок задания печати, который по умолчанию совпадает с именем первого файла в очереди (если с помощью флага **-T** не было указано другое значение). Для просмотра списка имен всех файлов задания печати воспользуйтесь длинной командой состояния **enq -A -L**.

Помещение задачи в очередь с помощью SMIT

При использовании **TCP/IP** для помещение файлов в очередь можно применять команду **smit**.

1. Для того чтобы поместить задание в очередь печати с помощью SMIT, введите следующую команду:
   ```
   smit
   ```
2. Выберите **Управление печатью**, а затем - меню запуска задания печати.
3. В поле **Файл для печати** введите имя файла, который необходимо напечатать.
4. В поле **Очередь печати** укажите имя удаленного принтера, на котором должен быть напечатан файл.

Теперь вы готовы к печати на удаленном принтере.

Печать файлов из удаленной системы

Вам может понадобиться напечатать файл, который расположен на удаленном хосте. В этом случае расположение напечатанного файла зависит от того, какие удаленные принтеры доступны удаленному хосту.

1. Вы должны иметь возможность войти в удаленную систему с помощью команды **rlogin** или **telnet**.
2. У вас должны быть права на чтение удаленного файла, который вы хотите напечатать на локальном принтере.

   Примечание: Данная процедура описывает лишь основные принципы удаленной печати. За дополнительной информацией об удаленной печати обратитесь к описанию команды **enq**.

Для печати из удаленной системы:

1. Войдите в удаленную систему с помощью команды **rlogin** или **telnet**.
2. Найдите имя нужной очереди и имя удаленного устройства. Имя очереди обычно начинается с префикса `rp`, за которым следует одна или несколько цифр. Имя удаленного принтера обычно начинается с префикса `drp`, за которым следует одна или несколько цифр.
3. Введите следующую команду:
   ```
   enq -P имя-очереди:имя-принтера имя-файла
   ```
   где *имя-очереди* - имя очереди, например `rp1`, а *имя-принтера* - это имя принтера (например, `drp1`), указанные в файле `/usr/lib/lpd/qconfig`. Не забудьте указать : (двоеточие) между параметрами *имя-очереди* и *имя-принтера*. *Имя-файла* - имя файла, который требуется распечатать.
4. Завершите соединение с удаленным хостом, нажав клавиши Ctrl-D или введя команду `quit`.

Просмотр сведений о состоянии

С помощью команд **TCP/IP** вы можете получить информацию о состоянии, пользователях и хостах сети. Эта информация может потребоваться для связи с другим хостом или пользователем.

Команды состояния TCP/IP

Для получения информации о состоянии локального или удаленного хоста и сети, к которой он подключен, в **TCP/IP** предусмотрены определенные команды.

Элемент	Описание
finger или **f**	Показывает информацию о текущих пользователях заданного хоста. Эта информация включает: идентификатор пользователя, его полное имя, имя терминала, дату и время входа в систему.
хост	Преобразует имя хоста в его IP-адрес или наоборот.
ping	Помогает определить состояние сети или хоста. Чаще всего эта команда применяется для проверки работоспособности сети или хоста.
rwho	Показывает список пользователей, которые в настоящий момент работают на хостах локальной сети. В списке указывается имя пользователя, имя хоста, а также дата и время входа в систему.
whois	Определяет принадлежность псевдонима или ИД пользователя. Эта команда применяется только в тех локальных сетях, которые подключены к сети Internet.

Просмотр сведений обо всех пользователях, работающих в системе удаленного хоста

С помощью этой процедуры можно просмотреть сведения обо *всех* пользователях, работающих в системе удаленного хоста.

Для просмотра информации обо всех пользователях, работающих в системе удаленного хоста, выполните следующие действия:

1. Войдите в систему на удаленном хосте, с которым необходимо установить соединение.
2. Для просмотра списка текущих пользователей хоста `alcatraz` введите следующую команду:

   ```
   finger @alcatraz
   ```

   На экране будет показана примерно следующая информация:

   ```
   brown    console  15 марта 15 13:19
   smith   pts0    Март 15 13:01
   jones   tty0    Март 15 13:01
   ```

Пользователь `brown` вошел в систему с консоли, пользователь `smith` - с устройства pts0, а пользователь `jones` - с tty0. Системный администратор может настроить систему таким образом, что команда **finger** будет предоставлять другую информацию. Если при работе с командой **finger** возникнут неполадки, обратитесь к системному администратору.

Просмотр информации о пользователе, работающем в удаленной системе

С помощью этой процедуры можно просмотреть сведения об *определенном* пользователе, работающем в системе удаленного хоста.

Для просмотра информации о пользователе, работающем в системе удаленного хоста, выполните следующие действия:

1. Войдите в систему на удаленном хосте, с которым необходимо установить соединение.
2. Для получения информации о пользователе `brown` хоста `alcatraz` введите:

   ```
   finger brown@alcatraz
   ```

   На экране будет показана примерно следующая информация:

   ```
   Имя:  brown
   Каталог:  /home/brown     Оболочка:  /home/bin/xinit -L -n Работает
   с 8 мая 07:13:49 в консоли
   Без плана.
   ```

Системный администратор может настроить систему таким образом, что команда **finger** будет предоставлять другую информацию. Если при работе с командой **finger** возникнут неполадки, обратитесь к системному администратору.

Протоколы TCP/IP

Протоколом называется набор правил, задающих форматы сообщений и процедуры, которые позволяют компьютерам и прикладным программам обмениваться информацией. Эти правила соблюдаются каждым

компьютером в сети, в результате чего любой хост-получатель может понять отправленное ему сообщение. *Набор* протоколов TCP/IP можно рассматривать как многоуровневую структуру.

На этом рисунке показан стек протоколов **TCP/IP**. Он делится на следующие уровни (начиная с верхнего): прикладной, транспортный, сетевой, интерфейсный и аппаратный.

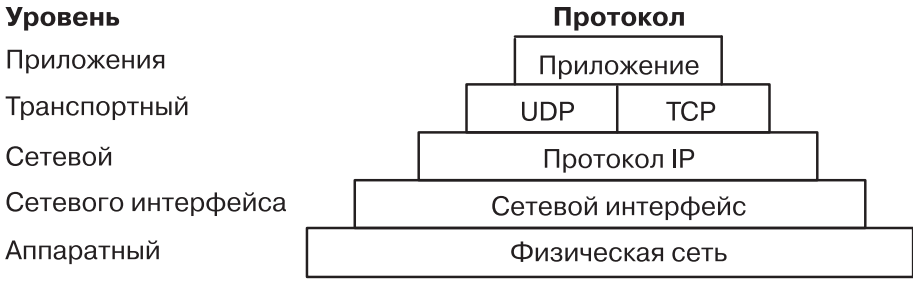


Рисунок 4. Набор протоколов TCP/IP

В протоколе TCP/IP строго зафиксированы правила передачи информации от отправителя к получателю. Сообщение или поток данных приложения отправляется протоколу Internet транспортного уровня, то есть **Протоколу пользовательских дейтаграмм (UDP)** или **Протоколу управления передачей (TCP**). Получив данные от приложения, эти протоколы разделяют всю информацию на небольшие блоки, которые называются *пакетами*. К каждому пакету добавляется адрес назначения, а затем пакет передается на следующий уровень протоколов Internet, то есть сетевой уровень.

На сетевом уровне пакет помещается в дейтаграмму **протокола Internet (IP)**, к которой добавляется заголовок и концевик. Протокол сетевого уровня определяет адрес следующего пункта назначения IP-дейтаграммы (она может быть передана сразу получателю или на промежуточный шлюз) и отправляют ее на уровень сетевого интерфейса.

Уровень сетевого интерфейса принимает **IP**-дейтаграммы и передает их в виде *кадров* с помощью аппаратного обеспечения, такого как адаптер Ethernet или Token-Ring.

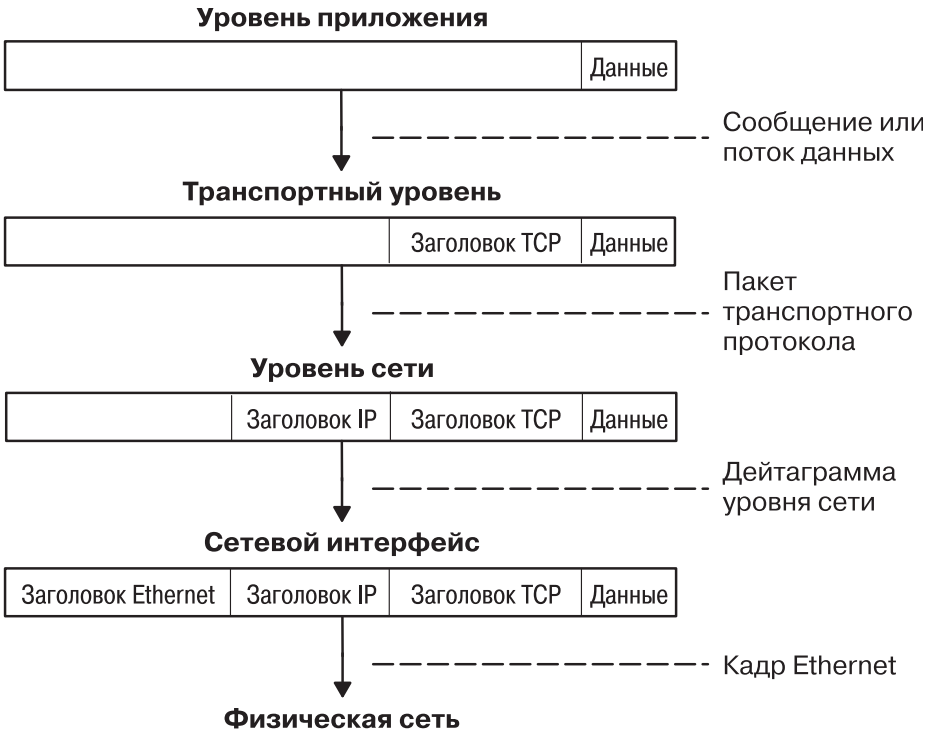


Рисунок 5. Передача информации от приложения-отправителя целевому хосту

На рисунке показана схема передачи информации через стек протоколов TCP/IP от отправителя целевому хосту.

Кадры доставляются на компьютер получателя, после чего они проходят все уровни протоколов в обратном порядке. На каждом уровне удаляются соответствующие этому уровню заголовки, после чего данные передаются на уровень приложения.

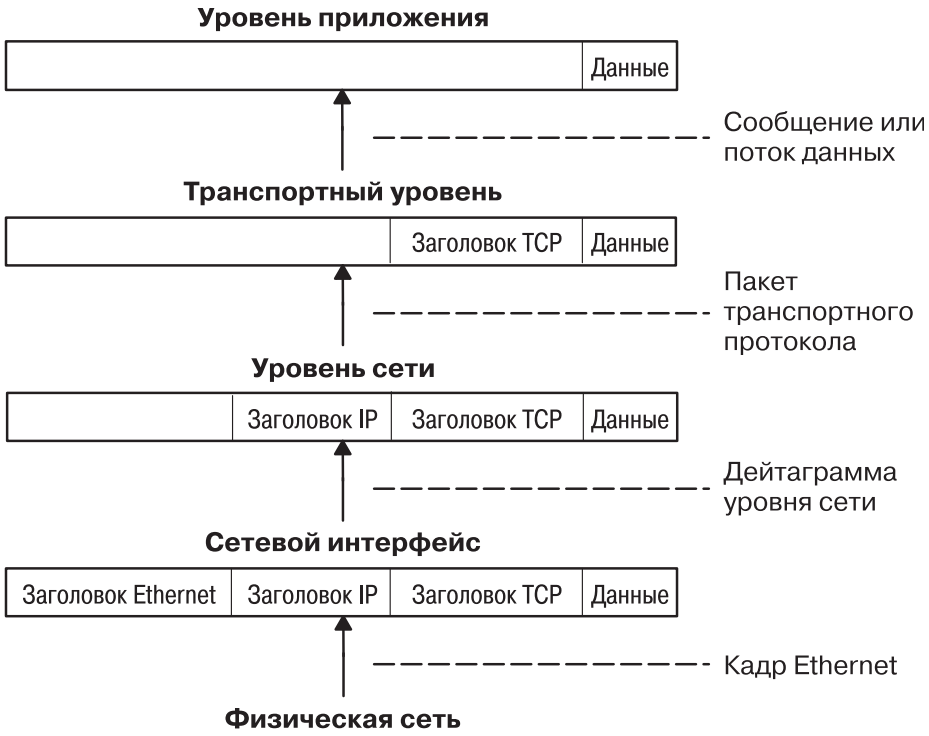
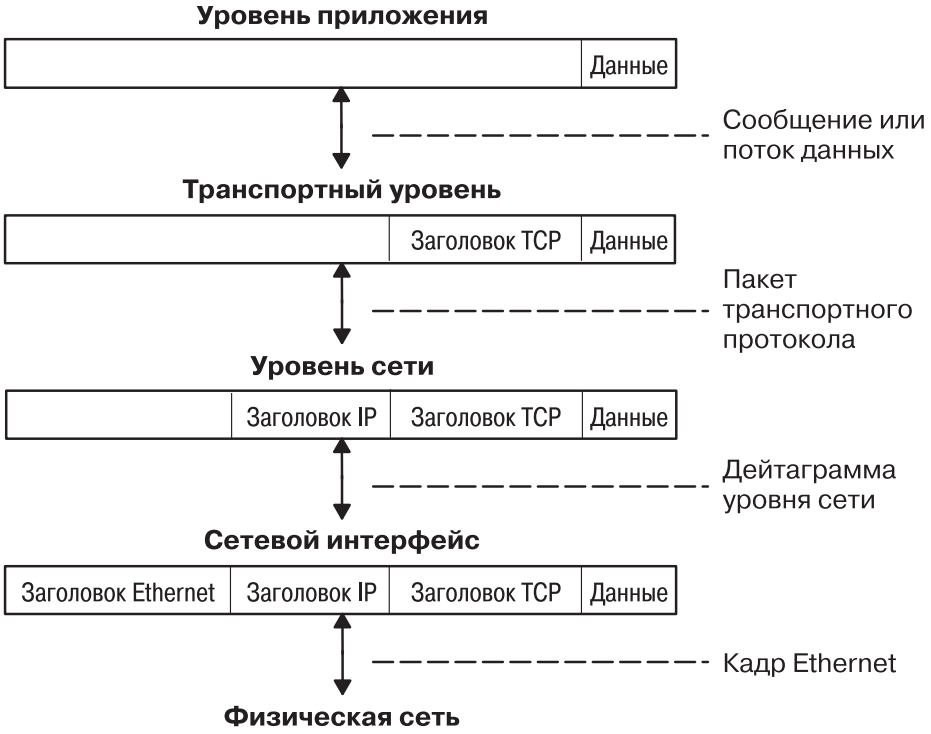


Рисунок 6. Передача информации от хоста приложению

На рисунке показана схема передачи информации через стек протоколов **TCP/IP** от хоста получателю.

Кадры передаются на уровень сетевого интерфейса (в данном случае, адаптеру Ethernet). Уровень сетевого интерфейса отбрасывает заголовок Ethernet и отправляет дейтаграмму на верхний сетевой уровень. На сетевом уровне Протокол Internet отбрасывает IP-заголовок и отправляет пакет на верхний транспортный уровень. Протокол транспортного уровня (в данном случае - **TCP**) удаляет заголовок **TCP** и передает данные на верхний уровень приложения.

Хосты в сети получают и передают информацию одновременно. На рисунке рис. 7 на стр. 124 показан процесс взаимодействия хостов.



Примечание: Заголовки добавляются и удаляются на каждом из уровней при отправке и получении данных хостом.

Рисунок 7. Передача и прием данных

На рисунке показана схема передачи данных в обоих направлениях через стек **TCP/IP**.

Протокол IP версии 6

Протокол Internet (**IP**) версии 6 (**IPv6** или *ng*) - это следующая версия протокола **IP**, которая является значительным шагом вперед по сравнению с **IP** версии 4 (**IPv4**).

Хотя протокол **IPv4** поддерживал развитие глобальной сети Internet, он исчерпал свои возможности, так как у него есть два основных недостатка: ограниченное адресное пространство и сложная схема маршрутизации. Формат 32-разрядных адресов протокола **IPv4** недостаточно гибок для обеспечения маршрутизации в глобальной сети Internet. Внедрение нового стандарта Бесклассовой междоменной маршрутизации (CIDR) продлило время жизни протокола **IPv4** на несколько лет, однако маршрутизация по устаревшей схеме требует слишком много усилий. Даже если масштаб маршрутизации **IPv4** будет увеличен, в Internet в конечном счете перестанут применяться классы сетей.

Инженерный консорциум разработчиков стандартов Internet (IETF) признал, что **IPv4** не способен обеспечить неограниченный рост Internet, поэтому была создана рабочая группа IETF IP*ng*. Из всех предложенных вариантов в качестве следующего протокола IP был выбран **Простой протокол Internet плюс** (**SIPP**) После завершения работы над RFC1883 в декабре 1995 года он был переименован в IP*ng*.

В IPv6 значительно увеличено адресное пространство в соответствии с ростом популярности сети Internet. **IPv6** - новая версия протокола **IPv4**, поэтому в одной и той же сети могут применяться и старый, и новый стеки протоколов. В результате переход от **IPv4** (32-разрядная адресация) к **IPv6** (128-разрядная адресация) в любой сети может осуществляться постепенно.

В данном обзоре описаны общие принципы работы протокола IP*ng*. За дополнительной информацией обратитесь к RFC 2460, 2373, 2465, 1886, 2461, 2462 и 2553.

В *Защита* приведены сведения о защите в наборе протоколов **TCP/IP**, включая **IPv6**. Дополнительные сведения о защите IP для версий 4 и 6 приведены в разделе Защита протокола Internet.

Расширенная маршрутизация и адресация IPv6:

В **IPv6** размер **IP**-адреса увеличен с 32 до 128 разрядов, при этом поддерживается большее число уровней иерархии адресов, значительно большее число адресуемых узлов, а также упрощена процедура автоматической настройки.

В **IPv6** поддерживается три типа адресов:

Элемент	Описание
обычный	Обычный адрес идентифицирует один интерфейс, а пакет с обычным адресом доставляется конкретному адресату. Обычные адреса в свою очередь подразделяются на категории: адреса уровня линии связи, уровня сайта и глобальные. Существует два специальных адреса этого типа: • ::/128 (произвольный адрес) • ::1/128 (циклический адрес)
групповой	Групповой адрес идентифицирует группу интерфейсов, а пакет с таким адресом доставляется всем членам группы. В групповом адресе указывается префикс ff::/8. Групповые адреса также разделяются на области: адреса уровня узла, уровня линии связи, уровня сайта и уровня организации.
нечеткий	Пакет с нечетким адресом отправляется от одного хоста и доставляется только одному интерфейсу из заданной группы (обычно, ближайшему члену этой группы в соответствии с метрикой расстояния протокола маршрутизации). Например, несколько веб-серверов могут быть объединены в группу с нечетким адресом. Запрос с таким адресом будет доставляться только одному веб-серверу из данной группы. Нечеткие адреса ничем не отличаются от обычных адресов. Обычный адрес становится нечетким, если несколько интерфейсов объединяются в группу с таким адресом.

Примечание: В **IPv6** не предусмотрены адреса оповещения. Их функции выполняют групповые адреса.

Автоматическая настройка IPv6:

В **IPv4** для загрузки узла и его последующего взаимодействия с другими узлами сети было предусмотрено три способа: номер интерфейсной платы узла, а также серверы **BOOTP** и **DHCP**.

В **IPv6** было введено понятие *области* **IP**-адресов, одна из которых - область линии связи. Адреса этой области состоят из предопределенного префикса локальной линии связи и локального идентификатора хоста. Локальный идентификатор обычно создается на основе адреса MAC настраиваемого интерфейса. Такие адреса позволяют одному хосту отправлять данные другим хостам в этой же подсети, при этом в изолированных сетях могут применяться адреса только этого типа.

Значащие адреса IPv6:

В протоколе **IPv4** особое значение имели только адреса оповещения (как правило, адреса, состоящие из всех 1 или 0), которые распознавались всеми хостами сети, и классы адресов (например, класс D применялся для многоцелевой рассылки). В адресе **IPv6** предусмотрен префикс, с помощью которого можно быстро определить *область* адреса (например, область локальной линии связи), тип адреса и механизм выделения адреса (по принадлежности к провайдеру или по принадлежности к географической области).

Информация о маршрутизации может быть указана в явном виде в первых битах адреса, но это не было окончательно стандартизовано IETF (для адресов, предоставленных провайдером, информация о маршрутизации содержится в адресе неявно).

Обнаружение дубликатов адресов IPv6:

С помощью автоматической настройки после инициализации или повторной инициализации интерфейса для него выделяется пробный адрес уровня линии связи (при этом адрес не присваивается). После этого интерфейс отправляет сообщение neighbor discovery узлам, адреса которых относятся к той же области линии связи. Узел отправляет сообщение с групповым адресом и узнает, был ли ранее присвоен данный адрес уровня линии связи, и если да, то выбирает другой адрес.

Таким образом исключается возможность присвоения двум интерфейсам, относящимся к одной и той же линии связи, одинаковых адресов. (Если узлы не относятся к одной линии связи, то им могут быть выделены одинаковые глобальные адреса.)

Автоматическая настройка Neighbor discovery/адрес без сохранения состояния:

Протокол Neighbor Discovery (**NDP**) для **IPv6** применяется узлами (хостами и маршрутизаторами) для поиска адресов соседей уровня линии связи, подключенных по конкретной линии связи, а также для создания таблиц маршрутизации до каждого пункта назначения для каждого активного соединения. В **IPv6** предусмотрен механизм автоматической настройки структурных и бесструктурных адресов. Для *автоматической настройки бесструктурных адресов* не требуется настройка хостов вручную или настройка дополнительных серверов. Возможно, потребуется внести небольшие изменения в конфигурацию маршрутизаторов.

Кроме того, протокол **NDP** применяется для поиска соседних маршрутизаторов, которые будут пересылать пакеты от своего имени и отслеживать изменение адресов уровня линии связи. **NDP** применяет **Протокол управляющих сообщений Internet** (**ICMP**) версии 6 и связанные с ним типы сообщений. Протокол **IPv6** Neighbor Discovery представляет собой комбинацию **Протокола преобразования адресов IPv4** (**ARP**), Протокола распространения информации о маршрутизаторах ICMP (RDISC) и протокола **ICMP Redirect** (ICMPv4), однако по сравнению с ними он обладает многими преимуществами.

Такой способ позволяет хостам создавать собственные адреса на основе локальной информации и информации, полученной от маршрутизаторов. Маршрутизаторы выделяют префиксы адресов для подсетей, связанных с данной линией связи, а хосты генерируют маркер интерфейса, однозначно определяющий интерфейс подсети. Адрес состоит из двух частей. Если маршрутизаторы отсутствуют, то хост может создавать только адреса уровня линии связи. Однако этого достаточно для установления соединений между узлами, подключенными к одной и той же линии связи.

Упрощение маршрутизации:

Для упрощения маршрутизации адреса **IPv6** были разделены на две части: префикс и ИД. На первый взгляд может показаться, что это ничем не отличается от разбиения адресов **IPv4** на адрес сети и адрес хоста, однако у этого способа есть два преимущества.

Элемент	Описание
отсутствие классов	Длина префикса и ИД не ограничена, что позволяет избежать проблем при увеличении размера сети.
вложенность	Изменяя длину префикса адреса, можно разделить сеть на любое число подсетей.

Пример 1

128 бит
адрес узла

Пример 2

Элемент	Описание
n бит	128-*n* бит
Префикс подсети	ИД интерфейса

Пример 3:

Элемент	Описание	
n бит	80-*n* бит	48 бит
Префикс абонента	ИД подсети	ИД интерфейса

Пример 4:

Элемент	Описание		
s бит	*n* бит	*m* бит	128-*s*-*n*-*m* бит
Префикс абонента	ИД области	ИД подсети	ИД интерфейса

В общем случае возможности IPv4 ограничиваются первыми тремя примерами, даже если применяется маска подсети переменной длины, VLSM (VLSM - это способ выделения подсетям **IP**-адресов в соответствии с их потребностями, а не в соответствии с правилами, общими для всех сетей). Это является следствием как недостаточной длины адреса, так и отсутствия префикса переменной длины.

Упрощение формата заголовка:

В **IPv6** структура **IP**-заголовка была упрощена за счет удаления или вынесения в дополнительный заголовок некоторых полей заголовка **IPv4**. В результате формат дополнительного заголовка, в котором задается необязательная информация, стал более гибким.

В частности, были удалены такие поля, как:

- длина заголовка (длина постоянна)
- идентификатор
- флаги
- смещение фрагмента (это поле перемещено в заголовок фрагментации)
- контрольная сумма заголовка (целостность данных обеспечивается протоколом верхнего уровня или с помощью заголовка идентификации)

Таблица 53. Заголовок IPv4

Элемент	Описание	Описание	Описание	Описание
Версия	Длина заголовка	Тип сервиса	Суммарная длина	
Идентификатор	Идентификатор	Идентификатор	Флаги	Смещение фрагмента
Время жизни	Время жизни	Протокол	Контрольная сумма заголовка	Контрольная сумма заголовка
Адрес источника	Адрес источника	Адрес источника	Адрес источника	Адрес источника
Адрес получателя	Адрес получателя	Адрес получателя	Адрес получателя	Адрес получателя
Опции	Опции	Опции	Опции	Поле выравнивания

Таблица 54. Заголовок IPv6

Элемент	Описание	Описание	Описание	Описание
Версия	Приоритет		Метка потока	
Длина данных	Длина данных	Длина данных	Следующий заголовок	Ограничение на пересылку
Адрес источника	Адрес источника	Адрес источника	Адрес источника	Адрес источника
Адрес получателя	Адрес получателя	Адрес получателя	Адрес получателя	Адрес получателя

По сравнению с **IPv4** в IP*ng* улучшена структура поля опций. Опции **IPv6** размещаются в отдельных дополнительных заголовках, которые расположены в пакете между заголовком **IPv6** и заголовком транспортного уровня. Большинство дополнительных заголовков не просматриваются и не обрабатываются маршрутизаторами в процессе доставки пакета, пока пакет не будет принят в пункте назначения. Такое разделение поля опций значительно повышает производительность обработки пакетов с этим полем маршрутизаторами. Если в заголовке **IPv4** были указаны какие-либо опции, то маршрутизатор должен был проверять все поле опций.

Другое преимущество структуры поля опций **IPv6** состоит в том, что дополнительные заголовки могут быть произвольной длины, а общий объем опций не ограничен 40 байтами. Эта особенность в сочетании со способом обработки заголовка пакета позволяет разместить в заголовке **IPv6** опции, которые не применялись в **IPv4**, например, опции идентификации и защиты **IPv6**.

Для того чтобы упростить обработку дополнительных заголовков и заголовка транспортного уровня, длина дополнительного заголовка IPv6 всегда кратна 8 байтам.

Применение дополнительных заголовков вместо поля идентификатора протокола и поля опций позволяет легко добавить новые опции.

В настоящий момент определены следующие дополнительные заголовки:

- Заголовок опций транзитного узла, содержащий информацию, которая должна проверяться на каждом узле (маршрутизаторе) по пути следования пакета.
- Заголовок маршрутизации, в котором указывается информация о маршрутизации типа точный/произвольный (применяется редко).
- Заголовок фрагментации содержит сведения, идентифицирующие пакет как фрагмент (маршрутизаторы в **IPv6** не могут фрагментировать дейтаграмму).
- Идентификация (см. раздел "**TCP/IP** security" книги *Защита*
- Шифрование (см. раздел "**TCP/IP** security" книги *Защита*
- Заголовок опций получателя, обрабатываемых в пункте назначения (игнорируются маршрутизаторами).

Контроль над повышение качества обслуживания/трафика:

Хотя качество обслуживания может обеспечивать специальный протокол, например, **RSVP**, в **IPv6** приоритет пакетов указывается явно в поле приоритета **IP**-заголовка.

Значение приоритета устанавливается узлом и отражает относительный приоритет пакета или набора пакетов, в зависимости от которого этот узел, маршрутизатор или хост-получатель решают, сбрасывать этот пакет или нет.

В **IPv6** существует два типа приоритетов: для потоков с управлением перегрузками и без. Считается, что между двумя типами приоритетов взаимосвязь отсутствует.

Поток с управлением перегрузками - это поток, в котором при возникновении перегрузки применяется алгоритм возврата или другой алгоритм, ограничивающий нагрузку линии. Приоритеты для потока с управлением перегрузками:

Элемент	Описание
0	дейтаграмма не принадлежит никакому потоку
1	поток-"заполнитель" (например, новости)
61 см	неконтролируемая передача данных (например, электронная почта)
3	(зарезервирован)
4	контролируемая передача массивов данных (например, **FTP**)
5	(зарезервирован)
6	передача данных по интерактивному соединению (например, Telnet)
7	передача управляющей информации (например, сообщения протоколов маршрутизации)

Поток без управления перегрузками - это поток, из которого при возникновении перегрузки сбрасываются (или не пересылаются) такие пакеты, как видео или аудиоданные, а также другие пакеты реального времени. Для потока с управлением перегрузками применяются примерно следующие правила задания приоритета:

- Самый низкий приоритет устанавливается для дейтаграмм, которые должны в первую очередь отбрасываться при перегруженной линии.
- Самый высокий приоритет должен устанавливаться для дейтаграмм, которые должны отбрасываться в последнюю очередь при перегруженной линии.

Эти правила применимы только для потоков данных от конкретного отправителя. Приоритет потока с управлением перегрузками от одного отправителя не обязательно выше, чем приоритет контролируемой передачи массивов данных от другого отправителя.

Идентификация потоков:

Помимо основных приоритетов потока, в **IPv6** определен механизм для идентификации отдельного потока пакетов. В **IPv6** *поток* - это последовательность пакетов, отправляемых от конкретного отправителя определенному получателю (или группе получателей), на пути к которому пакеты должны пройти определенную обработку.

Метка потока применяется для задания приоритета, а также для других целей.

Метка потока представляет собой случайное число, которое предназначено только для идентификации данного потока. Это означает, что маршрутизатор, анализируя метку потока, не может узнать конкретный тип пакета. Однако он может узнать, что данный пакет принадлежит к той же последовательности пакетов, что и предыдущий пакет с такой же меткой.

Примечание: Метка потока применяется в основном в экспериментальных целях, так как **IPv6** пока не стал общепризнанным стандартом. Применение и управление метками потока еще не было определено или стандартизовано.

Туннелирование IPv6:

Эту задачу можно решить, в частности, с помощью туннелирования.

Пакеты **IPv6** будут эффективно передаваться только в том случае, если они смогут обрабатываться уже существующими хостами и маршрутизаторами **IPv4**. Совместимость **IPv4** с **IPv6** - одна из основных задач при переходе к новому протоколу IP. В течение этого времени для передачи пакетов **IPv6** может применяться существующая инфраструктура **IPv4**.

Хосты и маршрутизаторы **IPv6** передают дейтаграммы **IPv6** через сети **IPv4**, помещая их в пакеты **IPv4**. Существует несколько типов туннелирования:

Элемент	Описание
Маршрутизатор-маршрутизатор	Пакеты **IPv6** могут передаваться через сеть **IPv4** от одного маршрутизатора **IPv6** или **IPv4** к другому такому же маршрутизатору. В этом случае туннель - это один из транзитных участков на пути пакета **IPv6**.
Хост-маршрутизатор	Хост **IPv6** может передавать пакеты **IPv6** маршрутизатору **IPv6** через сеть **IPv4**. Такой туннель представляет собой первый транзитный участок пути пакета.
Хост-хост	Пакеты **IPv6** могут передаваться через сеть **IPv4** от одного хоста **IPv6** или **IPv4** к другому такому же хосту. В этом случае туннель - это весь путь пакета.
Маршрутизатор-хост	Маршрутизаторы **IPv6** и **IPv4** могут передавать пакеты **IPv6** целевым хостам **IPv6** или **IPv4**. Такой туннель представляет собой последний транзитный участок пути пакета.

Типы туннелирования отличаются друг от друга способом, которым узел в начале туннеля определяет адрес узла в конце туннеля. В туннелировании маршрутизатор-маршрутизатор и хост-маршрутизатор пакет **IPv6** передается по туннелю маршрутизатору. В туннелировании хост-хост и маршрутизатор-хост пакет **IPv6** передается хосту-получателю.

Узел в начале туннеля (узел, помещающий пакет **IPv6** в пакет **IPv4**) создает заголовок пакета **IPv4** и передает этот пакет. Узел в конце туннеля (узел, извлекающий пакет **IPv6** из пакета **IPv4**) получает пакет **IPv4** с вложенным пакетом **IPv6**, удаляет заголовок **IPv4**, изменяет заголовок **IPv6** и обрабатывает полученный пакет. Однако для пересылки пакетов **IPv6** узел в начале туннеля должен хранить для каждого туннеля параметры соединения, например, максимальный блок передачи (MTU) туннеля.

В **IPv6** есть два типа туннелей: автоматические и настроенные.

Автоматические туннели
: Автоматические туннели настраиваются с помощью информации адреса **IPv4**, вкладываемой в адрес **IPv6** – адрес **IPv6** целевого хоста включает информацию о том, к какому адресу **IPv4** должен быть направлен пакет по туннелю.

Настроенные туннели
: Эти туннели следует настраивать вручную. Они используются при работе с адресами **IPv6**, не содержащими информацию **IPv4**. Следует указать адреса **IPv6** и **IPv4** конечных точек туннеля.

Информация о создании автоматических и настроенных туннелей приведена в разделе “Настройка туннеля в IPv6” на стр. 138.

Поддержка IPv6 с несколькими адресами уровня линии связи и уровня сайта:

У хоста может быть несколько интерфейсов. Хост с двумя и более интерфейсами называется хостом с несколькими адресами. С каждым интерфейсом связан адрес уровня линии связи.

Адреса уровня линии связи применяются для установления соединений между узлами, подключенными к одной линии связи.

С хостом может быть связано несколько адресов уровня связи. В протоколе **IPv6** дляAIX предусмотрено 4 опции для управления преобразованием адресов уровня линии связи на хостах с несколькими адресами. По

умолчанию применяется опция 1.

Элемент	Описание
Опция 0	На хосте активен только один интерфейс. Данные будут передаваться через интерфейс, с которым связан первый адрес уровня линии связи. Когда протоколу **Neighbor Discovery** (**NDP**) требуется преобразовать адрес, он отправляет сообщение Neighbor Solicitation каждому интерфейсу с адресом уровня линии связи. NDP заносит пакеты в очередь до тех пор, пока не будет получено первое сообщение Neighbor Advertisement. После этого пакеты будут переданы получателю сообщения.
Опция 1	Когда протоколу **NDP** требуется преобразовать адрес (т.е. когда при отправке пакетов в кэше соседних узлов отсутствует адрес уровня канала связи для следующего узла), он рассылает сообщение Neighbor Solicitation всем интерфейсам с адресом уровня линии связи. До тех пор пока **NDP** не получит информацию о соседнем узле, он будет помещать пакеты данных в очередь. **NDP** ждет того момента, когда будут получены ответы от всех интерфейсов. В этом случае гарантируется доставка пакетов нужному интерфейсу. Если бы протокол **NDP** не ожидал получения ответов от всех интерфейсов, а отвечал на первое полученное сообщение Neighbor Advertisement, пакеты данных могли быть отправлены по линии, не связанной с адресом отправителя. Из-за ожидания **NDP** первый пакет отправляется с некоторой задержкой. Следует отметить, что задержка будет и в том случае, если NDP будет дожидаться только первого ответа.
Опция 2	Настройка нескольких интерфейсов разрешена, но отправка пакетов может выполняться только интерфейсом с меткой main_if6. Когда протоколу **NDP** требуется преобразовать адрес, он рассылает сообщение Neighbor Solicitation всем интерфейсам с адресом уровня линии связи. Затем он дожидается получения сообщения Neighbor Advertisement от интерфейса main_if6 (дополнительная информация приведена в описании команды **no**). После получения ответа от этого интерфейса пакеты отправляются по соответствующей линии связи.
Опция 3	Настройка нескольких интерфейсов разрешена, однако отправка пакетов может выполняться только интерфейсом с меткой main_if6, а пакеты с адресами уровня сайта будут отправляться только интерфейсу main_site6 (см. описание команды **no**). Протокол NDP будет работать точно так же, как и в случае Опции 2. Пакеты приложений с адресами уровня сайта будут отправляться только по адресу, связанному с интерфейсом main_site6.

Переход к IPv6 при наличии настроенного IPv4:

Этот сценарий содержит инструкции по переходу от **IPv4** к **IPv6** вручную.

Сеть, описанная в данном примере, состоит из маршрутизатора и двух подсетей. В каждой подсети - два хоста: маршрутизатор и другой хост. Во всех системах этой сети выполняется модернизация до **IPv6**. В конце процедуры маршрутизатор сообщит о префиксе `3ffe:0:0:aaaa::/64` по сетевому интерфейсу `en0` и о префиксе `3ffe:0:0:bbbb::/64` по интерфейсу `en1`. Сначала в системах для тестирования будет настроена временная поддержка **IPv6**. Затем компьютеры будут настроены таким образом, чтобы поддержка **IPv6** включалась во время загрузки.

Если используется операционная система AIX, а параметры **IPv4** не настроены, то обратитесь к разделу “Обновление до IPv6 при отсутствии конфигурации IPv4” на стр. 133.

Рассмотрите следующие вопросы:

- Описанная ниже процедура была протестирована в отдельных версиях AIX. Результаты, которые вы можете получить, в значительной степени зависят от конкретных версии и уровня AIX.

Шаг 1: Настройка хостов для поддержки IPv6

На хостах обеих подсетей выполните следующие действия:

1. Введите следующую команду, чтобы убедиться в наличии конфигурации **IPv4**:

   ```
   netstat -ni
   ```

   Вывод команды будет выглядеть приблизительно следующим образом:

   ```
   Name  Mtu   Network     Address            Ipkts Ierrs    Opkts Oerrs  Coll
   en0   1500  link#2      0.6.29.4.55.ec      279393     0     2510     0     0
   en0   1500  9.3.230.64  9.3.230.117         279393     0     2510     0     0
   ```

```
lo0   16896 link#1                              913     0      919     0     0
lo0   16896 127         127.0.0.1               913     0      919     0     0
lo0   16896 ::1                                 913     0      919     0     0
```

2. Войдите в систему от имени пользователя root и настройте параметры **IPv6** с помощью следующей команды:

   ```
   autoconf6
   ```

3. Еще раз введите следующую команду:

   ```
   netstat -ni
   ```

   Вывод команды будет выглядеть приблизительно следующим образом:

   ```
   Name  Mtu   Network     Address            Ipkts Ierrs    Opkts Oerrs  Coll
   en0   1500  link#2      0.6.29.4.55.ec      279679     0     2658     0     0
   en0   1500  9.3.230.64  9.3.230.117         279679     0     2658     0     0
   en0   1500  fe80::206:29ff:fe04:55ec        279679     0     2658     0     0
   sit0  1480  link#3      9.3.230.117              0     0        0     0     0
   sit0  1480  ::9.3.230.117                        0     0        0     0     0
   lo0   16896 link#1                            2343     0     2350     0     0
   lo0   16896 127         127.0.0.1             2343     0     2350     0     0
   lo0   16896 ::1                               2343     0     2350     0     0
   ```

4. Запустите демон **ndpd-host** с помощью следующей команды:

   ```
   startsrc -s ndpd-host
   ```

Шаг 2: Настройка маршрутизатора для поддержки IPv6

1. Введите следующую команду, чтобы убедиться в наличии параметров **IPv4**:

   ```
   netstat -ni
   ```

2. Войдя в систему от имени пользователя root, введите следующую команду:

   ```
   autoconf6
   ```

3. Вручную настройте глобальные адреса для интерфейсов маршрутизатора, связанных с обеими подсетями. Для этого введите следующие команды:

   ```
   # ifconfig en0 inet6 3ffe:0:0:aaaa::/64 eui64 alias
   # ifconfig en1 inet6 3ffe:0:0:bbbb::/64 eui64 alias
   ```

   Эту операцию необходимо выполнить для каждой подсети, в которую маршрутизатор направляет пакеты.

4. Для активации пересылки IPv6 введите следующую команду:

   ```
   no -o ip6forwarding=1
   ```

5. Запустите демон **ndpd-router** с помощью следующей команды:

   ```
   startsrc -s ndpd-router
   ```

   Демон **ndpd-router** разошлет префиксы, соответствующие заданным для маршрутизатора глобальным адресам. В данном случае демон ndpd-router сообщит о префиксе `3ffe:0:0:aaaa::/64` по сетевому интерфейсу `en0` и о префиксе `3ffe:0:0:bbbb::/64` по сетевому интерфейсу `en1`

Шаг 3. Настройка активации поддержки IPv6 на хосте при загрузке системы

При перезагрузке системы настроенная вами поддержка **IPv6** будет удалена. После выполнения шага 1 в каждой из систем, настроенная вами поддержка **IPv6** будет действовать до перезагрузки системы.

1. Откройте файл `/etc/rc.tcpip` в текстовом редакторе.
2. Удалите символы комментария перед следующими строками:

   ```
   # Start up autoconf6 process
   start /usr/sbin/autoconf6 ""

   # Start up ndpd-host daemon
   start /usr/sbin/ndpd-host "$src_running"
   ```

3. Добавьте флаг **-A** в `start /usr/sbin/autoconf6 ""`:

```
start /usr/sbin/autoconf6 "" -A
```

При перезагрузке системы будет задана конфигурация **IPv6**. Повторите эту процедуру для каждого из хостов.

Шаг 4. Настройка активации поддержки IPv6 на маршрутизаторе при загрузке системы

При перезагрузке настроенная вами поддержка **IPv6** будет удалена. После выполнения шага 2 в вашей системе настроенная вами поддержка **IPv6** будет действовать до перезагрузки системы.

1. Откройте файл `/etc/rc.tcpip` в текстовом редакторе.
2. Удалите символы комментария перед следующей строкой:

   ```
   # Start up autoconf6 process
   start /usr/sbin/autoconf6 ""
   ```

3. Добавьте после этой строки следующие строки:

   ```
   # Настройка глобальных адресов для маршрутизатора
   ifconfig en0 inet6 3ffe:0:0:aaaa::/64 eui64 alias
   ifconfig en1 inet6 3ffe:0:0:bbbb::/64 eui64 alias
   ```

   В данном сценарии сеть состоит из двух подсетей, `en0` и `en1`. Для каждой подсети, в которую маршрутизатор отправляет пакеты, необходимо добавить соответствующую строку в файл.
4. Удалите символы комментария перед следующей строкой:

   ```
   # Запуск
   демона ndpd-router
   start /usr/sbin/ndpd- router "$src_running"
   ```

При перезагрузке поддержка **IPv6** будет запущена автоматически.

Обновление до IPv6 при отсутствии конфигурации IPv4:

В этом сценарии объясняется, как настроить хосты и маршрутизатор для поддержки **IPv6** в отсутствие конфигурации **IPv4**.

Сеть, описанная в данном примере, состоит из маршрутизатора и двух подсетей. В каждой подсети - два хоста: маршрутизатор и другой хост. В конце процедуры маршрутизатор сообщит о префиксе `3ffe:0:0:aaaa::/64` по сетевому интерфейсу `en0` и о префиксе `3ffe:0:0:bbbb::/64` по интерфейсу `en1`. Сначала в системах для тестирования будет настроена временная поддержка **IPv6**. Затем компьютеры будут настроены таким образом, чтобы поддержка IPv6 включалась во время загрузки.

В этом сценарии предполагается, что в системе установлен набор файлов `bos.net.tcp.client`.

Инструкции по переходу к **IPv6** при наличии настроенного протокола **IPv4** приведены в разделе “Переход к IPv6 при наличии настроенного IPv4” на стр. 131.

Рассмотрите следующие вопросы:

- Описанная ниже процедура была протестирована в отдельных версиях AIX. Результаты, которые вы можете получить, в значительной степени зависят от конкретных версии и уровня AIX.

Шаг 1: Настройка хостов для поддержки IPv6

1. Войдя в систему как пользователь root, введите следующую команду в каждой системе подсети:

   ```
   autoconf6 -A
   ```

   Эта команда настраивает все сетевые интерфейсы системы, поддерживающие **IPv6**.

   Примечание: Для того чтобы настроить только часть интерфейсов, укажите флаг **-i**. Например, команда `autoconf6 -i en0 en1` включает интерфейсы `en0` и `en1`.
2. Введите следующую команду, чтобы просмотреть список интерфейсов:

```
netstat -ni
```

Вывод команды будет выглядеть приблизительно следующим образом:

```
Name  Mtu   Network  Address            Ipkts Ierrs    Opkts Oerrs  Coll
en0   1500  link#3      0.4.ac.17.b4.11          7     0       17     0     0
en0   1500  fe80::204:acff:fe17:b411             7     0       17     0     0
lo0   16896 link#1                              436     0      481     0     0
lo0   16896 127         127.0.0.1               436     0      481     0     0
lo0   16896 ::1                                 436     0      481     0     0
```

3. Запустите демон **ndpd-host** с помощью следующей команды:

```
startsrc -s ndpd-host
```

Шаг 2: Настройка маршрутизатора для поддержки IPv6

1. Войдя в систему от имени пользователя root, введите следующую команду в системе маршрутизатора:

```
autoconf6 -A
```

Эта команда настраивает все сетевые интерфейсы системы, поддерживающие **IPv6**.

Примечание: Для того чтобы настроить только часть интерфейсов, укажите флаг `-i`. Например, команда `autoconf6 -i en0 en1` включает интерфейсы `en0` и `en1`.
Вывод команды будет выглядеть приблизительно следующим образом:

```
Name  Mtu   Network  Address            Ipkts Ierrs    Opkts Oerrs  Coll
en1   1500  link#2      0.6.29.dc.15.45          0     0        7     0     0
en1   1500  fe80::206:29ff:fedc:1545             0     0        7     0     0
en0   1500  link#3      0.4.ac.17.b4.11          7     0       17     0     0
en0   1500  fe80::204:acff:fe17:b411             7     0       17     0     0
lo0   16896 link#1                              436     0      481     0     0
lo0   16896 127         127.0.0.1               436     0      481     0     0
lo0   16896 ::1                                 436     0      481     0     0
```

2. Вручную настройте глобальные адреса для интерфейсов маршрутизатора, связанных с обеими подсетями. Для этого введите следующие команды:

```
# ifconfig en0 inet6 3ffe:0:0:aaaa::/64 eui64 alias
# ifconfig en1 inet6 3ffe:0:0:bbbb::/64 eui64 alias
```

Примечание: Эту операцию необходимо выполнить для каждой подсети, в которую маршрутизатор направляет пакеты.

3. Для активации пересылки **IPv6** введите следующую команду:

```
no -o ip6forwarding=1
```

4. Запустите демон **ndpd-router** с помощью следующей команды:

```
startsrc -s ndpd-router
```

Демон **ndpd-router** разошлет префиксы, соответствующие заданным для маршрутизатора глобальным адресам. Демон ndpd-router разошлет префиксы, соответствующие заданным для маршрутизатора глобальным адресам.

5. Для продолжения нажмите клавишу Enter.
6. Нажмите Enter еще раз, чтобы подтвердить запрос на установку данного комплекта программного обеспечения.

Шаг 3. Настройка активации поддержки IPv6 на хосте при загрузке системы

После выполнения шага 1 в каждой из систем, настроенная вами поддержка **IPv6** будет действовать до перезагрузки системы. После выполнения шага 1 в каждой из систем, настроенная вами поддержка **IPv6** будет действовать до перезагрузки системы.

1. Откройте файл `/etc/rc.tcpip` в текстовом редакторе.
2. Удалите символы комментария перед следующими строками:

```
# Start up
autoconf6 process
start /usr/sbin/autoconf6 ""

# Start up ndpd-host daemon
start /usr/sbin/ndpd-host "$src_running"
```

3. Добавьте флаг **-A** в `start /usr/sbin/autoconf6 ""`:

```
start /usr/sbin/autoconf6 "" -A
```

4. Повторите эту процедуру для каждого из хостов.

При перезагрузке поддержка **IPv6** будет запущена автоматически.

Шаг 4. Настройка активации поддержки IPv6 на маршрутизаторе при загрузке системы

После выполнения шага 2 в вашей системе настроенная вами поддержка **IPv6** будет действовать до перезагрузки системы. После выполнения шага 2 в вашей системе настроенная вами поддержка **IPv6** будет действовать до перезагрузки системы.

1. Откройте файл `/etc/rc.tcpip` в текстовом редакторе.
2. Удалите символы комментария перед следующей строкой:

```
# Start up
autoconf6 process
start /usr/sbin/autoconf6 ""
```

3. Добавьте в эту строку флаг **-A**:

```
start /usr/sbin/autoconf6 "" -A
```

4. Добавьте после этой строки следующие строки:

```
# Настройка
глобальных адресов для маршрутизатора
ifconfig en0 inet6 3ffe:0:0:aaaa::/64 eui64 alias
ifconfig en1 inet6 3ffe:0:0:bbbb::/64 eui64 alias
```

   В данном сценарии сеть состоит из двух подсетей, `en0` и `en1`. Для каждой подсети, в которую маршрутизатор отправляет пакеты, необходимо добавить соответствующую строку в файл.

5. Удалите символы комментария перед следующей строкой:

```
# Запуск
демона ndpd-router
start /usr/sbin/ndpd-router "$src_running"
```

6. Введите следующую команду, чтобы пересылка IP включалась при загрузке системы:

```
no -r -o ip6forwarding=1
```

При перезагрузке поддержка **IPv6** будет запущена автоматически.

Статическая конфигурация в динамическом режиме:

Этот сценарий описывает динамическую настройку узла с помощью статических IP-адресов и маршрутов.

В качестве примера рассматривается сеть, состоящая из хоста и маршрутизатора. После завершения сценария на хосте будет настроен интерфейс IPv6. Сначала в системах для тестирования настраивается временная поддержка IPv6. Затем системы настраиваются таким образом, чтобы поддержка IPv6 включалась во время загрузки.

Рассмотрите следующие вопросы:

- Описанная ниже процедура была протестирована в отдельных версиях AIX. Результаты, которые вы можете получить, в значительной степени зависят от конкретных версии и уровня AIX.
- В этом примере предполагается, что **2001:1:2::/48** - это объединенный глобальный обычный адрес для интерфейса IPv6, присвоенный провайдеру комитетом по предоставлению адресов Internet (IANA). **2001:1:2:3:4::/64** - это подсеть, использующая биты 49 - 64, присвоенные администратором сети.

- Описание формата глобального обычного адреса IPv6 приведено в RFC 3587.

Информация, связанная с данной:

Команды динамической конфигурации

Команда autoconf6

Шаг 1. Настройка хостов для IPv6:

Ниже приведены инструкции по настройке хостов для поддержки IPv6.

1. Войдя в систему как пользователь root, настройте параметры IPv6 с помощью следующей команды:
   ```
   # autoconf6
   ```
2. Еще раз введите следующую команду:
   ```
   # netstat -ni
   ```
   Вывод команды должен выглядеть приблизительно следующим образом:
   ```
   Name  Mtu   Network  Address            Ipkts Ierrs    Opkts Oerrs  Coll
   en0   1500  link#2      0.6.29.4.55.ec      279679     0     2658     0     0
   en0   1500  9.3.230.64  9.3.230.117         279679     0     2658     0     0
   en0   1500  fe80::206:29ff:fe04:55ec        279679     0     2658     0     0
   sit0  1480  link#3      9.3.230.117              0     0        0     0     0
   sit0  1480  ::9.3.230.117                        0     0        0     0     0
   lo0   16896 link#1                           2343     0     2350     0     0
   lo0   16896 127         127.0.0.1            2343     0     2350     0     0
   lo0   16896 ::1                             2343     0     2350     0     0
   ```
3. С помощью команды **chdev** добавьте адрес IPv6 для интерфейса хоста. В этом примере младшие 64 бита извлекаются из младших 64 бит IP-адреса уровня линии связи, созданного **autoconf6** в интерфейсе **en0**.
   ```
   # chdev -l en0 -a netaddr6='2001:2:3:4:206:29ff:fe04:55ec' -a prefixlen=64
   ```
4. Удалите существующие маршрушты для следующих префиксов:
   ```
   # route delete -inet6 2001:2:3:4::/64
   ```
5. Настройте статический маршрут префикса на хосте для обеспечения доступа к маршрутизатору, где **fe80::206:29ff:fe04:66e** - это маршрутизатор или шлюз, обладающий доступом к маршрутизатору.
   ```
   # route add -inet6 -net 2001:2:3:4::/64 fe80::206:29ff:fe04:66e -static
   ```

   Примечание: Если требуется изменить маршрут по умолчанию, выполните команду **autoconf6** с параметром **–R**, который запрещает добавление или перезапись маршрутов по умолчанию на узле. Затем повторите шаги 3-5.

Шаг 2. Настройка маршрутизатора для IPv6:

Ниже приведены инструкции по настройке маршрутизатора для поддержки IPv6.

1. Введите следующую команду, чтобы убедиться в наличии параметров IPv4:
   ```
   # netstat -ni
   ```
2. Войдите в систему от имени пользователя root и введите следующую команду:
   ```
   # autoconf6
   ```
3. Для того чтобы включить функцию пересылки IPv6, введите следующую команду:
   ```
   # no -o ip6forwarding=1
   ```
4. Настройте глобальный IP-адрес для интерфейса маршрутизатора. Для этого введите следующую команду:
   ```
   # chdev -l en0 -a netaddr6='2001:4:5:6:207:30ff:fe05:66ec' -a prefixlen=64
   ```
5. Вручную настройте маршруты в маршрутизаторе, чтобы включить точную доставку пакетов. Например, если шлюз **fe80:: 3ca6:70ff:fe00:3004/64** указан для префикса **2001:2:3:4::/64**, добавьте следующий маршрут префикса:
   ```
   # route add -inet6 -net 2001:2:3:4::/64 fe80::3ca6:70ff:fe00:3004 -static
   ```

Шаг 3. Настройка IPv6 для применения на хоста при каждом запуске:

Параметры хостов IPv6, настроенные при выполнении **шага 1. Настройка хостов для поддержки IPv6**, удаляются при перезапуске системы. Для включения поддержки хостов IPv6 при каждой загрузке выполните следующие действия.

1. Откройте файл **/etc/rc.tcpip** в текстовом редакторе.
2. Удалите символы комментария из следующей строки в файле **/etc/rc.tcpip**:

   ```
   # Start up
   autoconf6 process
   start /usr/sbin/autoconf6 ""
   ```

   Примечание: Если предыдущая строка отсутствует в файле **/etc/rc.tcpip**, добавьте ее.
3. Добавьте флаг **-A** в **start /usr/sbin/autoconf6 ""**.

   ```
   start /usr/sbin/autoconf6 "" -A
   ```
4. Добавьте следующую строку в файл **/etc/rc.tcpip** после строки, из которой были удалены символы комментария (или добавленной строки):

   ```
   chdev -l en0 -a netaddr6='2001:2:3:4:206:29ff:fe04:55ec' -a prefixlen=64
   ```
5. Удалите существующие маршруты префиксов с помощью следующей команды:

   ```
   chdev -l inet0 -a delrout6='-net, 2001:2:3:4::/64'
   ```
6. Настройте маршрут. Для этого выполните следующую команду:

   ```
   chdev -l inet0 -a rout6='-net, 2001:2:3:4::/64 ,fe80::206:29ff:fe04:66e,-static'
   ```

В ходе перезапуска системы настраивается конфигурация IPv6.

Примечание: Повторите эту процедуру для каждого хоста.

Шаг 4. Настройка IPv6 для применения на маршрутизаторе при каждом запуске:

Параметры маршрутизаторов IPv6, настроенные при выполнении **шага 2. Настройка маршрутизатора для поддержки IPv6**, удаляются при перезапуске системы. Для включения поддержки IPv6 при каждой загрузке выполните следующие действия.

1. Откройте файл **/etc/rc.tcpip** в текстовом редакторе.
2. Удалите символы комментария из следующей строки в файле **/etc/rc.tcpip**:

   ```
   # Start up
   autoconf6 process
   start /usr/sbin/autoconf6 ""
   ```

   Примечание: Если предыдущая строка отсутствует в файле **/etc/rc.tcpip**, добавьте ее.
3. Добавьте флаг **-A** в **start /usr/sbin/autoconf6 ""**.

   ```
   start /usr/sbin/autoconf6 "" -A
   ```
4. Добавьте следующие строки после строки, из который были удалены (или добавлены) символы комментария на шаге 2, чтобы настроить глобальный IP-адрес на маршрутизаторе и маршрут префикса.

   ```
   chdev -l en0 -a netaddr6='2001:4:5:6:207:30ff:fe05:66ec' -a prefixlen=64
   chdev -l inet0 -a rout6='-net,2001:2:3:4::/64,fe80::3ca6:70ff:fe00:3004,-static'
   ```

   В этом сценарии сеть содержит только одну подсеть **en0**. В этот файл требуется добавить строку для каждой подсети, в которую маршрутизатор отправляет пакеты.

В ходе перезапуска системы автоматически запускается IPv6.

Примечание: В случае применения статических конфигураций вместе с **ndpd-host** при необходимости убедитесь, что флаги в **ndpd-host** настроены для сохранения статических IP-адресов и маршрутов.

Настройка туннеля в IPv6:

Настроить туннель в IPv6 можно одним из двух способов. Первый способ заключается в создании автоматического туннеля. Второй - в создании настроенного туннеля.

Особенности

- Описанная ниже процедура была протестирована в отдельных версиях AIX. Результаты, которые вы можете получить, в значительной степени зависят от конкретных версии и уровня AIX.

Настройка автоматического туннеля в IPv6

В этом сценарии для настройки IPv6 используется команда **autoconf6**, позволяющая настроить автоматический туннель через основной интерфейс `en2`. С помощью команды **autoconf6** также можно настроить туннель через вспомогательный интерфейс `en0`.

Ниже приведены результаты команды **netstat -ni**, показывающие текущую сетевую конфигурацию системы:

```
en0   1500  линия#2     MAC-адрес              0     0       33     0     0
en0   1500  1.1         1.1.1.3                0     0       33     0     0
en2   1500  линия#3     MAC-адрес          79428     0      409     0     0
en2   1500  10.1        10.1.1.1           79428     0      409     0     0
```

- Для включения IPv6 и одного автоматического туннеля введите следующую команду:

  ```
  autoconf6
  ```

  Теперь команда **netstat -ni** выдаст следующие результаты:

  ```
  # netstat -in
  en0   1500  линия#2     MAC-адрес              0     0       33     0     0
  en0   1500  1.1         1.1.1.3                0     0       33     0     0
  en0   1500  fe80::204:acff:fe49:4910            0     0       33     0     0
  en2   1500  линия#3     MAC-адрес          79428     0      409     0     0
  en2   1500  10.1        10.1.1.1           79428     0      409     0     0
  en2   1500  fe80::220:35ff:fe12:3ae8
  sit0  1480  линия#7     10.1.1.1               0     0        0     0     0
  sit0  1480  ::10.1.1.1
  ```

  Если основным интерфейсом является `en2` (IP-адрес 10.1.1.1), то адрес ::10.1.1.1 будет доступен для настройки автоматического туннеля через интерфейс `en2`.

- Для включения автоматического туннеля через интерфейс `en0` введите следующую команду:

  ```
  autoconf6 -s -i en0
  ```

  Теперь команда **netstat -ni** выдаст следующие результаты:

  ```
  # netstat -in
  en0   1500  линия#2     MAC-адрес              0     0       33     0     0
  en0   1500  1.1         1.1.1.3                0     0       33     0     0
  en0   1500  fe80::204:acff:fe49:4910            0     0       33     0     0
  en2   1500  линия#3     MAC-адрес          79428     0      409     0     0
  en2   1500  10.1        10.1.1.1           79428     0      409     0     0
  en2   1500  fe80::220:35ff:fe12:3ae8
  sit0  1480  линия#7     1.1.1.3                0     0        3     0     0
  sit0  1480  ::10.1.1.1                         0     0        3     0     0
  sit0  1480  ::1.1.1.3                          0     0        3     0     0
  ```

  Эти действия добавляют совместимые с IPv4 адреса IPv6 к существующему интерфейсу SIT `sit0`. Кроме того, туннель включается для интерфейса `en0` с использованием адреса ::1.1.1.3. Для обоих туннелей будет использован один и тот же интерфейс `sit0`.

  Примечание: При перезапуске системы автоматические туннели будут удалены. Для того чтобы автоматические туннели сохранялись при загрузке, укажите соответствующие аргументы в команде **autoconf6** в файле `/etc/rc.tcpip`.

Создание настроенного туннеля

В этом сценарии с помощью SMIT будет создан настроенный туннель. Туннель будет доступен даже после перезагрузки системы, так как он будет сохранен в ODM. Туннель будет соединять системы `alpha` и `beta`. Адрес IPv4 `alpha` - 10.1.1.1, адрес IPv4 `beta` - 10.1.1.2.

Для создания настроенных туннелей выполните следующие действия:

1. Для настройки туннеля между системами `alpha` и `beta` введите следующую команду в обеих системах:

   ```
   smit ctinet6
   ```

2. Выберите пункт **Добавить интерфейс туннеля IPV6 в IPV4** в обеих системах.

   ```
   autoconf6
   ```

3. По этому сценарию в системе `alpha` введены следующие значения, основанные на адресах IPv4.

   ```
   * IPV4 адрес источника (десятичный формат с точками) [10.1.1.1]
   * IPV4 адрес получателя (десятичный формат с точками) [10.1.1.2]
     IPV6 адрес источника (с двоеточием)                  []
     IPV6 адрес получателя (с двоеточием)                 []
   ```

   В системе `beta` введены следующие значения:

   ```
   * IPV4 адрес источника (десятичный формат с точками) [10.1.1.2]
   * IPV4 адрес получателя (десятичный формат с точками) [10.1.1.1]
     IPV6 адрес источника (с двоеточием)                  []
     IPV6 адрес получателя (с двоеточием)                 []
   ```

4. Для просмотра настроенных интерфейсов введите следующую команду:

   ```
   ifconfig ctiX
   ```

   где *X* - это номер интерфейса. В этом сценарии получены следующие результаты. В системе `alpha`:

   ```
   cti0: flags=8080051<UP,POINTOPOINT,RUNNING,MULTICAST>
           inet6 fe80::a01:101/128 --> fe80::a01:102
   ```

   В системе `beta`:

   ```
   cti0: flags=8080051 <UP,POINTOPOINT,RUNNING,MULTICAST>
           inet6 fe80::a01:102/128 --> fe80::a01:101
   ```

SMIT автоматически создает адреса IPv6 для обоих концов туннеля следующим способом:

- Младшие 32 бита содержат адрес IPv4
- Старшие 96 битов содержат префикс `fe80::/96`

При желании можно ввести конкретные адреса IPv6.

Трассировка пакетов

Трассировка пакетов - это процесс проверки пути пакета к пункту назначения через уровни протоколов.

Команда **iptrace** выполняет трассировку пакетов на уровне сетевого интерфейса. Команда **ipreport** позволяет просмотреть информацию о трассировке пакетов в шестнадцатеричном и текстовом формате. Команда **trpt** выполняет трассировку пакетов **TCP** на транспортном уровне. Вывод команды **trpt** содержит более подробные сведения, в том числе время, состояние **TCP** и порядок пакетов.

Заголовки пакета сетевого интерфейса

На уровне сетевого интерфейса к передаваемым данным добавляются заголовки пакетов.

На рисунке показана схема передачи данных в обоих направлениях между уровнями структуры сетевого

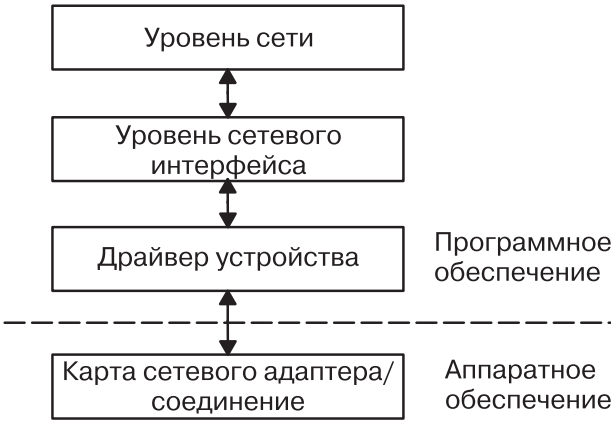


Рисунок 8. Передача пакетов по структуре сетевого интерфейса

интерфейса. С верхнего (программного) уровня данные передаются на сетевой уровень, уровень интерфейса, драйверу устройства, а затем - карте сетевого адаптера или соединению.

После этого пакеты отправляются через сетевой адаптер в нужную сеть. По пути к целевому хосту пакет может пройти через несколько промежуточных шлюзов. В сети получателя заголовки отделяются от пакетов, а полученные данные отправляются хосту-получателю.

Ниже описан формат заголовка пакета для некоторых наиболее распространенных сетевых интерфейсов.

Заголовки кадра адаптера Ethernet:

Заголовок кадра **Протокола Internet** (**IP**) или **Протокола преобразования адресов** (**ARP**) в случае адаптера Ethernet состоит из данных трех полей.

Таблица 55. Заголовок кадра адаптера Ethernet

Поле	Длина	Определение
DA	6 байт	Адрес получателя.
SA	6 байт	Адрес источника. Если нулевой бит этого поля равен 1, то в кадре содержится информация о маршрутизации (RI).
Тип	2 байта	Указывает тип пакета: **IP** или **ARP**. Ниже перечислены значения, соответствующие различным типам.

Значения поля типа:

Элемент	Описание
IP	0800
ARP	0806

Заголовки кадра Token-Ring:

Заголовок MAC (Управления доступом к среде передачи данных) адаптера Token-Ring состоит из пяти полей.

Таблица 56. Заголовок MAC Token-ring

Поле	Длина	Определение
AC	1 байт	Управление доступом. Значение x'00' устанавливает приоритет, равный 0.
FC	1 байт	Управляющее поле. Значение x'40' означает кадр Управления логическим каналом связи.
DA	6 байт	Адрес получателя.
SA	6 байт	Адрес источника. Если нулевой бит этого поля равен 1, то в кадре содержится информация о маршрутизации (RI).
RI	18 байт	Информация о маршрутизации. Ниже описаны допустимые значения этого поля.

Заголовок MAC состоит из двух полей с информацией о маршрутизации размером по 2 байта: RC (управление маршрутизацией) и номера сегментов. Можно указать не более восьми номеров сегментов, идентифицирующих получателей при ограниченном оповещении. Информация RC содержится в нулевом и первом байтах поля RI. Значения первых двух битов поля RC:

Элемент	Описание
бит (0) = 0	Должен применяться обычный маршрут, заданный в поле RI.
бит (0) = 1	Создать поле RI и разослать кадр всем хостам, подключенным к кольцу.
бит (1) = 0	Разослать кадр всем мостам.
бит (1) = 1	Разослать кадр указанным мостам.

Заголовок LLC (Управления логическим каналом связи) состоит из пяти полей, описанных в приведенной ниже таблице.

Таблица 57. Заголовок LLC 802.3

Поле	Длина	Определение
DSAP	1 байт	Целевая служебная точка доступа. Значение этого поля - x'aa'.
SSAP	1 байт	Исходная служебная точка доступа. Значение этого поля - x'aa'.
CONTROL	1 байт	Содержит команды и ответы LLC. Ниже приведены допустимые значения этого поля.
PROT_ID	3 байта	ИД протокола. Это зарезервированное поле. Его значение равно x'0'.
TYPE	2 байта	Указывает тип пакета: **IP** или **ARP**.

Значения управляющих полей:

Управляющие поля token-ring включают в себя ненумерованный информационный кадр, кадр обмена идентификацией и тестовый кадр. Их значения описаны здесь.

Элемент	Описание
x'03'	Информационный кадр без номера (UI). Это обычный способ передачи данных в сети Token-Ring, когда порядок доставки кадров не гарантируется. Данные упорядочиваются на уровне **TCP/IP**.
x'AF'	Кадр идентификационной информации (XID). В этом кадре передаются параметры хоста-отправителя.
x'E3'	Пробный кадр. Этот кадр предназначен для проверки линии связи. Полученные данные отправляются обратно хосту-отправителю.

Заголовки кадра 802.3:

В случае адаптера 802.3 заголовок MAC состоит из двух полей, описанных в следующей таблице.

Таблица 58. Заголовок MAC 802.3

Поле	Длина	Определение
DA	6 байт	Адрес получателя.
SA	6 байт	Адрес источника. Если нулевой бит этого поля равен 1, то в кадре содержится информация о маршрутизации (RI).

Заголовок LLC 802.3 аналогичен заголовку MAC Token-Ring.

Протоколы TCP/IP сетевого уровня

Протоколы Internet сетевого уровня обеспечивают соединение между двумя компьютерами в сети.

Другими словами, этот уровень отвечает за маршрутизацию **TCP/IP**. Эти протоколы принимают запросы на отправку пакетов (содержащие адрес получателя) от транспортного уровня, преобразуют пакеты в дейтаграммы и отправляют их на уровень сетевого интерфейса для дальнейшей обработки.

На рисунке показаны уровни набора протоколов **TCP/IP**. На верхнем (прикладном) уровне находится

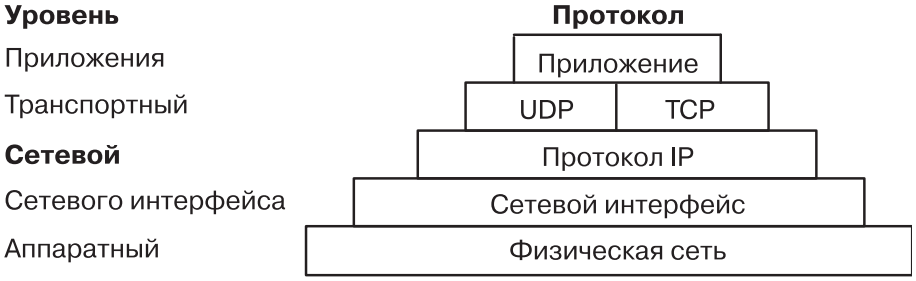


Рисунок 9. Сетевой уровень набора протоколов TCP/IP.

приложение. На транспортном уровне работают **UDP** и **TCP**. Сетевой уровень представлен сетевым (аппаратным) интерфейсом. Аппаратный уровень - это физическая сеть.

В стек **TCP/IP** включены не только протоколы, предусмотренные в RFC 1100 (*Official Internet Protocols*), но и много других популярных протоколов, применяемых хостами Internet.

Примечание: Номера сетей, версий, сокетов, служб и протоколов Internet в **TCP/IP** также соответствуют RFC 1010 (см. *Стандартные номера*).

Протокол преобразования адресов:

Базовый протокол сетевого уровня - это **Протокол преобразования адресов** (**ARP**). **ARP** выполняет динамическое преобразование IP-адресов в уникальные физические адреса хостов локальных сетей.

В качестве примера работы **ARP** рассмотрим два узла - `X` и `Y`. Если узел `X` хочет установить соединение с узлом `Y`, и при этом `X` и `Y` расположены в разных сетях (LAN), то `X` и `Y` взаимодействуют через *мосты*,

маршрутизаторы или *шлюзы*, идентифицируя друг друга по IP-адресам. Внутри LAN узлы обмениваются данными с помощью низкоуровневых аппаратных адресов.

Если узлы подключены к одному и тому же сегменту LAN, то они определяют физические адреса партнеров по протоколу **ARP**. В этом случае узел `X` рассылает запрос **ARP** об аппаратном адресе узла `Y` всем узлам сети. В запросе **ARP** указывает аппаратный и **IP**-адрес узла `X` и **IP**-адрес узла `Y`. При получении запроса **ARP** узел `Y` помещает запись об узле `X` в свой кэш **ARP** (применяемый для быстрого преобразования IP-адресов в аппаратные адреса), а затем отправляет узлу `X` ответ **ARP**, содержащий аппаратный и **IP**-адрес узла `Y`. Когда узел `X` получает от узла `Y` ответ **ARP**, он помещает запись об узле `Y` в свой кэш **ARP**.

После того как в кэш **ARP** узла `X` добавлена запись об узле `Y`, узел `X` сможет отправлять пакеты непосредственно узлу `Y` без обращения к **ARP** (после удаления записи об узле `Y` из кэша **ARP** узел `X` будет вынужден снова обратиться к **ARP**).

В отличие от большинства протоколов, в протоколе **ARP** не зафиксирован формат заголовков. Сообщения ARP поддерживаются в различных локальных сетях:

- Адаптеры Ethernet LAN (поддерживает протоколы Ethernet и 802.3)
- Адаптеры Token-Ring
- Адаптеры FDDI (Оптоволоконного интерфейса распределенных данных)

Протокол ARP неприменим для протоколов **SLIP** (**Протокол подключения к Internet по последовательной линии**) и **SOCC** (**Протокол последовательной оптической линии**), так как эти протоколы предназначены для работы с двухточечными соединениями.

Таблицы преобразования обслуживаются ядром ОС, поэтому у пользователей и приложений нет доступа непосредственно к **ARP**. При отправке IP-пакета одному из драйверов интерфейса, драйвер запрашивает преобразование соответствующего адреса. Если в таблице нет соответствующего аппаратного адреса, **ARP** рассылает пакет с запросом драйвера интерфейса всем хостам локальной сети.

Полные записи хранятся в таблице **ARP** в течение 20 минут, неполные - в течение 3 минут. Для создания постоянной записи в таблице преобразования **ARP** вызовите команду **arp** с параметром *pub* parameter:

```
arp -s 802.3 host2 0:dd:0:a:8s:0 pub
```

Когда хост, поддерживающий протокол **ARP**, получает пакет с запросом **ARP**, он сохраняет аппаратный адрес и **IP**-адрес запрашивающего хоста и при необходимости обновляет свою таблицу ARP. Если **IP**-адрес хоста-получателя не совпадает с адресом, указанным в запросе, то хост удаляет пакет с запросом. Если **IP**-адрес совпадает, хост отправляет запрашивающей системе пакет с ответом. Запрашивающая система сохраняет новую запись преобразования и передает все ожидающие IP-пакеты.

Протокол управляющих сообщений Internet:

Вторым протоколом сетевого уровня является **Протокол управляющих сообщений Internet** (**ICMP**). **ICMP** - обязательная часть любой реализации **IP**. **ICMP** отправляет сообщения об ошибках и управляющие сообщения протоколу **IP**.

С помощью этого протокола шлюзы и хосты отправляют источнику пакетов отчеты о неполадках. **ICMP** выполняет следующие функции:

- Проверяет, что хост-получатель активен и доступен
- Сообщает об неправильных параметрах в заголовке дейтаграммы
- Синхронизует часы и определяет время передачи данных по маршруту
- Определяет IP-адреса и маски подсетей

Примечание: Как и протоколы более высокого уровня, протокол **ICMP** использует базовые функции **IP**. Однако в действительности **ICMP** представляет собой часть протокола **IP** и должен быть реализован в каждом модуле **IP**.

Протокол **ICMP** - это всего лишь средство обмена информацией о неполадках в сети. Он не повышает надежность протокола **IP**. Таким образом, **ICMP** не гарантирует надежной доставки **IP**-пакета, а также доставки сообщения **ICMP** в случае, если **IP**-пакет не был получен или был получен в искаженном виде.

Сообщения **ICMP** отправляются в следующих ситуациях:

- Когда пакет невозможно доставить получателю
- Когда размер буфера шлюза недостаточен для пересылки пакета
- Когда шлюз может предложить хосту более короткий маршрут для доставки пакета

TCP/IP отправляет и принимает сообщения ICMP нескольких типов (см. “Типы сообщений протокола управляющих сообщений Internet”). Протокол **ICMP** встроен в ядро и для него не предусмотрен интерфейс прикладных программ (API).

Типы сообщений протокола управляющих сообщений Internet:

ICMP отправляет и получает сообщения данных типов.

Элемент	Описание
эхо-запрос	Отправляется хостами и маршрутизаторами для проверки работоспособности и достижимости целевого хоста.
информационный запрос	Отправляется хостами и маршрутизаторами для получения IP-адреса сети, к которой они подключены. В сообщениях данного типа указывается **IP**-адрес целевого хоста, сетевая часть которого равна 0.
запрос системного времени	Запрос текущего системного времени целевого хоста.
запрос маски адреса	Отправляется хостом для получения своей маски подсети. Такой запрос отправляется шлюзу, если известен его адрес, или всем хостам в сети.
получатель недостижим	Отправляется шлюзом, если он не может доставить дейтаграмму **IP**.
подавление источника	Отправляется компьютером, если он не успевает обрабатывать поступающие дейтаграммы, чтобы источник пакетов снизил скорость отправки дейтаграмм.
сообщение об изменении маршрута	Отправляется шлюзом, если он обнаруживает, что хост применяет неоптимальный маршрут.
эхо-ответ	Отправляется компьютером в ответ на получение эхо-запроса.
информационный ответ	Отправляется шлюзами в ответ на получение запроса об адресе сети, причем в ответе содержатся адреса отправителя и получателя указанной дейтаграммы **IP**.
ответ на запрос системного времени	Отправляется текущее время.
ответ на запрос маски адреса	Отправляется хостам, запросившим маску адреса.
ошибка в параметрах	Отправляется, когда хост или шлюз обнаруживает ошибку в заголовке дейтаграммы.
превышено максимальное число транзитных участков	Отправляется в том случае, если: • Все **IP**-дейтаграммы содержат счетчик числа транзитных участков, значение которого уменьшается каждым шлюзом. • Если счетчик числа транзитных участков доходит до нуля, шлюз отбрасывает дейтаграмму.
Временные метки Internet	Содержит временные метки прохождения пакета по маршруту.

Протокол Internet:

Третий протокол сетевого уровня в семействе TCP/IP - **Протокол Internet** (**IP**). Он доставляет пакеты по сети Internet без гарантии доставки и без установления соединения.

IP не устанавливает соединение, так как каждый пакет данных передается независимо. Это ненадежный протокол, так как он не гарантирует доставку пакетов: хост-отправитель, хост-получатель и промежуточные хосты не отправляют подтверждения.

Протокол **IP** взаимодействует с протоколами уровня сетевого интерфейса. Данные передаются по физическим соединениям в виде кадров, содержащих заголовок и данные. В заголовке указываются адреса отправителя и получателя. В **IP** данные передаются в виде IP-дейтаграмм, формат которых аналогичен формату кадра. У дейтаграммы также есть заголовок, в котором содержатся IP-адреса отправителя и получателя.

В протоколе **IP** определен формат всех данных, передаваемых в Internet.

На этом рисунке показаны первые 32 бита стандартного заголовка IP-пакета. Более подробно поля

Рисунок 10. Заголовок пакет протокола Internet

заголовка рассмотрены в таблице.

Поля заголовка IP

Элемент	Описание
Версия	В этом поле указывается версия протокола **IP**. В настоящий момент применяется протокол **IP** версии 4.
Длина	Длина заголовка, выраженная в 32-разрядных словах.
Тип сервиса	Содержит пять полей, в которых указываются предпочтительные для данного пакета тип приоритета, задержка, пропускная способность и надежность. (IP не гарантирует, что при передаче пакета будут применяться именно эти параметры.) По умолчанию применяется приоритет процедуры, а также обычные задержка, пропускная способность и надежность. В настоящий момент это поле редко применяется при передаче пакетов в Internet. Данная реализация протокола **IP** соответствует спецификации **IP**, описанной в документе RFC 791, *Internet Protocol*.
Суммарная длина	Длина дейтаграммы в октетах с учетом заголовка. При прохождении пакета через шлюз он может быть фрагментирован, а затем собран на целевом хосте. Общая длина **IP**-пакета независимо настраивается для каждого интерфейса с помощью команды **ifconfig** или команды SMIT `smit chinet`. SMIT заносит значения в базу данных конфигурации на постоянной основе. Если вы хотите задать или изменить значения параметров для активной системы, воспользуйтесь командой **ifconfig**.
Идентификатор	Содержит уникальное целое число, идентифицирующее дейтаграмму.
Флаги	Это поле в сочетании с полем Идентификатор предназначено для управления фрагментацией дейтаграммы. В поле Флаги фрагментации указывается, допустима ли фрагментация дейтаграммы, а также является ли данный фрагмент последним.
Смещение фрагмента	Смещение фрагмента в исходной дейтаграмме, указанное в блоках по 8 октетов.
Время жизни	Определяет интервал времени, в течение которого дейтаграмма может оставаться в сети Internet. Такое ограничение позволяет удалять дейтаграммы с неправильно заданным маршрутом. По умолчанию время жизни пакета равно 255 секундам.
Протокол	В этом поле задается тип протокола более высокого уровня.
Контрольная сумма заголовка	Специальное значение, которое вычисляется для проверки целостности данных.
Адрес источника	IP-адрес хоста-отправителя.
Адрес получателя	IP-адрес хоста-получателя.

Элемент	Описание
Опции	Это поле предназначено для проверки и отладки сетевого соединения. Оно может отсутствовать в дейтаграмме. **Конец списка опций** Обозначает конец списка опций. Это поле указывается в конце всего списка опций, а не после каждой опции. Оно используется только в том случае, если конец списка не совпадает с концом всего заголовка **IP**. Конец списка опций указывается в том случае, если его длина превышает длину дейтаграммы. **Нет операции** Предназначено для выравнивания опций, например, для выравнивания начала следующей опции по 32-разрядной границе. **Запись произвольного маршрута** В этом поле источник дейтаграммы IP указывает информацию о маршрутизации, которая должна применяться шлюзами при передаче дейтаграммы целевому хосту. Здесь указывается *произвольный* маршрут источника: для передачи дейтаграммы следующему узлу маршрута шлюз или хост **IP** могут применять любой маршрут с любым числом промежуточных шлюзов. **Запись точного маршрута** В этом поле источник дейтаграммы IP указывает информацию о маршрутизации, которая должна применяться шлюзами при передаче дейтаграммы целевому хосту. Здесь указывается *точный* маршрут к источнику: шлюз или хост **IP** должны пересылать дейтаграмму непосредственно следующему узлу маршрута с помощью прямого соединения с этим узлом. **Запись о маршруте** В этом поле записывается маршрут IP-дейтаграммы. **Идентификатор потока** Идентификатор потока позволяет смоделировать поток данных, если дейтаграммы пересылаются через сеть, в которой не поддерживается передача потоков данных. **Временные метки Internet** Содержит временные метки прохождения пакета по маршруту.

Заголовок **IP** автоматически добавляется ко всем отправляемым пакетам. В принимаемых пакетах заголовок **IP** удаляется перед передачей данных протоколу более высокого уровня. Протокол **IP** обеспечивает универсальную адресацию хостов в Internet.

Протоколы TCP/IP транспортного уровня

Протоколы **TCP/IP** транспортного уровня позволяют приложениям обмениваться данными с другими приложениями.

На рисунке показаны уровни набора протоколов **TCP/IP**. На верхнем (прикладном) уровне находится

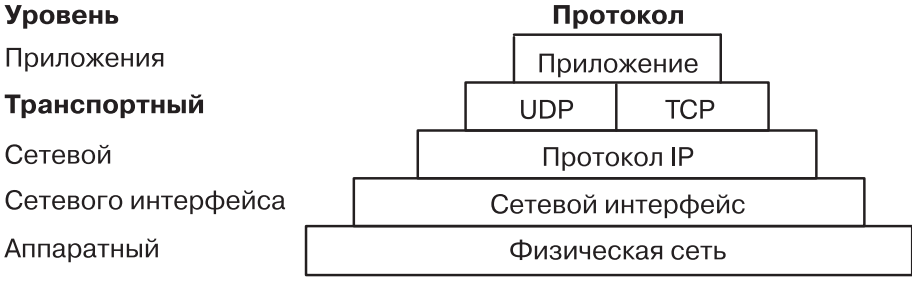


Рисунок 11. Транспортный уровень набора протоколов TCP/IP.

приложение. На транспортном уровне работают **UDP** и **TCP**. Сетевой уровень представлен сетевым (аппаратным) интерфейсом. Аппаратный уровень - это физическая сеть.

Как правило, для установления соединения между хостами Internet на транспортном уровне применяются **Протокол пользовательских дейтаграмм** (**UDP**) и **TCP**. Как **TCP**, так и **UDP** позволяют программам обмениваться сообщениями с приложениями других хостов. Когда приложение передает запрос на отправку сообщения на транспортный уровень, **UDP** или **TCP** разбивает сообщение на пакеты, добавляет заголовок пакета, содержащий адрес получателя, и отправляет пакеты для дальнейшей обработки на сетевой уровень. Для идентификации получателя сообщения в протоколах **TCP** и **UDP** применяются номера портов.

Протоколы более высокого уровня и приложения применяют протокол **UDP** для передачи дейтаграмм, а **TCP** - для передачи данных в потоковом режиме. Эти протоколы представляют собой часть интерфейса сокетов операционной системы.

Протокол пользовательских дейтаграмм:

Иногда возникает необходимость отправить сообщение от одного приложения другому приложению или процессу, выполняемому на другом компьютере, подключенном к сети. **UDP** обеспечивает передачу дейтаграмм между приложениями хостов Internet.

Так как отправитель не знает, какие процессы активны в настоящий момент, для отправки сообщений одному из приложений хоста **UDP** применяет целевой порт протокола (абстрактная точка для приема данных на хосте), представляющий собой положительное целое число. Полученные сообщения помещаются в очередь, связанную с портом протокола, пока приложение не сможет их обработать.

Для отправки дейтаграмм протокол **UDP** применяет протокол **IP**, поэтому **UDP** так же не устанавливает соединения, как и **IP**. Он не гарантирует доставку дейтаграммы и не обеспечивает защиту от дублирования данных. Однако **UDP** позволяет отправителю задать для сообщения исходный и целевой порты и обеспечивает проверку целостности данных и заголовка сообщения с помощью контрольной суммы. Это позволяет отправителю и получателю проверить правильность доставки сообщения.

На рисунке показаны первые 32 бита стандартного заголовка пакета **UDP**. Первые 16 бит содержат номер

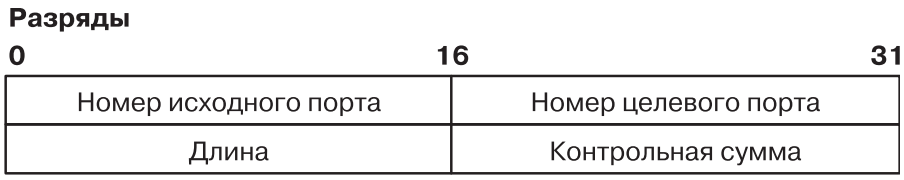


Рисунок 12. Заголовок пакета в протоколе UDP

исходного порта и длину. Вторые 16 бит содержат номер целевого порта и контрольную сумму.

Для надежной доставки дейтаграмм с помощью **UDP** в приложении должны быть предусмотрены процедуры проверки. Для надежной доставки потоков данных предназначен протокол **TCP**.

Поля заголовка дейтаграммы UDP

Элемент	Описание
Порт отправителя	Номер порта отправителя.
Порт получателя	Номер порта получателя.
Длина	Длина дейтаграммы **UDP** в октетах.
Контрольная сумма	Обеспечивает проверку целостности дейтаграммы **UDP** с помощью того же алгоритма, что и в протоколе **IP**.

Интерфейс прикладных программ (API) для работы с **UDP** представляет собой набор библиотечных процедур, основанных на интерфейсе сокетов.

Reliable Datagram Sockets для InfiniBand и RoCE:

Reliable Datagram Sockets (RDS) - это ориентированный на записи протокол без соединений, обеспечивающий упорядоченное обслуживание с помощью InfiniBand без дублирования и RDMA over Converged Ethernet (RoCE). RDS использует подмножество протокола пользовательских дейтаграмм (UDP) из API сокетов.

RDS входит в состав домена **AF_BYPASS**, который применяется для протоколов, которые обходят стек TCP/IP ядра.

Операционная система AIX предлагает две версии RDS: RDSv2 и RDSv3. RDSv3 является последней версией, которая включает поддержку удаленного прямого доступа к памяти (RDMA). RDSv3 в AIX 7.2 в выше поддерживает Open Fabrics Enterprise Distribution (OFED) на основе RDMA over Converged Ethernet (RoCE).

Создание сокета RDS: Для создания сокета RDS выполните системный вызов **socket()** путем добавления следующих строк в программу:

```
#include <sys/bypass.h>
include <net/rds_rdma.h> /* только RDSv3 */
sock = socket (AF_BYPASS, SOCK_SEQPACKET,BYPASSPROTO_RDS);
```

Если в семействе **AF_BYPASS** доступен только один надежный протокол дейтаграмм **BYPASSPROTO_RDS**, то системный вызов **socket()** можно выполнить следующим образом:

```
sock = socket (AF_BYPASS, SOCK_SEQPACKET,0);
```

Системные вызовы

Кроме того, RDS поддерживает следующие системные вызовы:

- blind()
- close()
- getsockopt()
- recvform()
- recvmsg()
- sendmsg()
- sendto()
- setsockopt()

Кроме того, RDSv3 поддерживает следующие системные вызовы:

- connect()
- read()
- recv()
- send()
- write()

Примечание: Несмотря на то, что сокеты RDS не устанавливают соединения, RDSv3 поддерживает системный вызов **connect()**. Однако в этом случае **connect()** не создает объект соединения уровня сокета между двумя конечными точками RDS. Он связывает конечную точку получателя по умолчанию с сокетом. По этой причине системные вызовы **listen()**, **accept()** и **shutdown()** не поддерживаются для сокетов RDS.

Утилита rdsctrl для RDSv2: С помощью утилиты **rdsctrl** (`/usr/sbin/rdsctrl`) можно изменять параметры RDS и собирать статистику. Для RDSv2 утилиту можно использовать после загрузки RDS (**bypassctrl load rds**). Для просмотра дополнительной информации об этой утилите выполните команду rdsctrl без аргументов.

Статистика

Для просмотра различной статистики RDS выполните команду # `rdsctrl stats`.

Для сброса статистики выполните команду # `rdsctrl stats reset`.

Параметры тонкой настройки

Следующие параметры RDS можно настроить после загрузки RDS, но перед запуском приложения RDS:

rds_sendspace
Задает верхний порог буфера отправки для потока. Каждый сокет может содержать несколько потоков. Значение по умолчанию: 524288 байт (512 КБ). Значение указывается с помощью следующей команды: # `rdsctrl set rds_sendspace= <значение в байтах>`.

rds_recvspace
Задает верхний порог буфера приема сокета для потока. Для каждого дополнительного потока сокета **верхний порог приема** увеличивается на это значение. Значение по умолчанию: 524288 байт (512 КБ). Значение указывается с помощью следующей команды: # `rdsctrl set rds_recvspace= <значение в байтах>`.

Примечание: Для высокой производительности потоковой обработки RDS значения параметров **rds_sendspace** и **rds_recvspace** должны быть по крайней мере в четыре раза больше самого большого значения **sendmsg()** RDS. RDS отправляет ACK для каждого набора из четырех входящих сообщений. Если значение **rds_recvspace** не превышает размер сообщения по крайней мере в четыре раза, то пропускная способность будет очень низкой.

rds_mclustsize
Задает размер отдельного кластера памяти, который также является размером фрагмента сообщения. Размер по умолчанию: 16384 байт (16 КБ). Значение, которое должно быть кратно 4096, указывается с помощью следующей команды: # `rdsctrl set rds_mclustsize= <значение, кратное 4096, в байтах>`.

Внимание: Значение параметра **rds_mclustsize** должно совпадать во всех системах (узлах) кластера. Изменение этого значения может повлиять на производительность.

Текущие значения предыдущих параметров можно получить с помощью команды # `rdsctrl get <параметр>`.

Для получения списка всех параметров и их значений выполните команду # `rdsctrl get`.

Утилита rdsctrl для RDSv3: Команда **rdsctrl** для RDSv3 поддерживает следующие параметры:

Элемент	Описание
help [<*имя параметра*>]	Опция **help** позволяет просмотреть описание указанного параметра RDSv3. Если параметр не указан, то эта опция отображает список всех параметров, поддерживаемых для RDSv3, вместе их описаниями.
set [-p] {<*имя параметра*> = <*значение*>}	Опция **set** задает значение указанного параметра RDSv3. Выполняется проверка прав доступа пользователей, чтобы предотвратить несанкционированный доступ к параметрам RDS. Кроме того, выполняется проверка новых значений параметра. Флаг **-p** позволяет сделать присвоение постоянным в случае перезапуска.
get [<*имя параметра*>]	Опция **get** получает текущее значение запрошенного параметра. Если поле имени не указано, то возвращается текущее значение всех доступных параметров RDS.
default [-p] [<*имя параметра*>]	Опция **default** позволяет восстановить значение параметра по умолчанию. Если поле имени указано, то сбрасывается только соответствующий параметр. Если поле имени не указано, то команда применяет для всех параметров значения по умолчанию. Кроме того, эта опция позволяет сделать изменения постоянными в случае перезапуска с помощью флага **-p**.

Элемент	Описание
load [`ofed` | `aixib`]	Опция **load** загружает расширение ядра RDSv3 (если оно еще не загружено). Аргумент `ofed` загружает расширения ядра в RDSv3 для OFED в режиме RoCE. Аргумент `aixib` загружает расширение ядра в RDSv3 в режиме InfiniBand. Для опции **`load`** необязательно указывать аргумент. Если аргумент не указан, то по умолчанию опция **`load`** содержит аргумент `aixib`. По умолчанию утилита **rdsctrl** загружает устройство InfiniBand, если в командной строке не указан новый атрибут (`ofed`).
unload	Опция **unload** применяется для выгрузки расширения ядра RDSv3.
ras [-p] <*minimal* | *normal* | *detail* | *maximal*>	Опция **ras** задает параметры трассировки и проверки ошибок RAS в операционной системе AIX для указанного уровня RDSv3. Эта команда вызывает команды **errctrl** и **ctctrl** операционной системы AIX. Флаг **-p** позволяет сделать параметры постоянными в случае перезапуска.
ras extract	Опция **ras extract** создает дамп буферов трассировки RAS для RDS и отправляет его в стандартный поток вывода.
info [<*флаги*>]	Опция **info** является псевдонимом команды **rds-info**.
ping [<*адрес IP v4*>]	Опция **ping** является псевдонимом команды **rds-ping**.
conn <*restart* | *kill*> <*исходный IP-адрес*> <*целевой IP-адрес*>	Опция **conn** перезапускает (подопция **restart**) или завершает (подопция **kill**) указанное соединение RDS. Перезапускаемое или завершаемое соединение RDS указывается с помощью IP-адресов локального и удаленного узлов соединения. В случае перезапуска соединения прерывается соответствующее соединение InfiniBand и выполняется попытка установить соединение снова. Однако при завершении соединения (подопция kill) прерывается соединение InfiniBand и освобождает ресурсы, связанные с соответствующим соединением RDS.
trace start <*путь к файлу трассировки*> <*максимальный размер собираемых данных для фрагмента RDS*>	Опция **trace start** создает экземпляр сеанса трассировки для регистрации трафика, передаваемого по протоколу RDSv3. Сообщения RDSv3 передаются фрагментами. Каждый передаваемый фрагмент RDS захватывается как пакет трассировки в указанном файле трассировки. Для каждого фрагмента RDS захватывается полезная нагрузка до <*максимальный размер собираемых данных для фрагмента RDS*> байт. Трафик RDS могут отслеживать только пользователи с правами администратора. Кроме того, одновременно может быть активен только один сеанс трассировки.
trace stop	Опция **trace stop** завершает сеанс трассировки, созданный с помощью команды **trace start**. Закрывает файл трассировки, связанный с сеансом трассировки. После этой команды можно выполнить команду **trace report** для создания текстового отчета о файле трассировки.
trace report <*путь к файлу трассировки*>	Опция **trace report** печатает текст отчета в стандартный поток вывода из ранее полученного файла трассировки протокола RDS.
version	Опция **version** печатает текущую версию протокола RDS, загруженную в системе.

Параметры RDSv3: Для просмотра списка параметров, поддерживаемых для RDSv3, выполните команду **rdsctrl help** без аргументов.

API RDMA (только для RDSv3): Программная модель RDMA с сокетами RDS основана на модели клиент/сервер. Клиент RDMA - это приложение, запускающее операцию чтения или записи RDMA с указанного сервера RDMA. Сервер RDMA - это приложение, которое отвечает за передачу данных RDMA. Операция чтения RDMA передает данные из адресного пространства клиента в адресное пространство сервера, а операция записи RDMA передает данные из адресного пространства сервера в адресное пространство клиента. В обоих случаях данные передаются непосредственно между пользовательскими пространствами с обоих сторон без копирования в пространство ядра.

Приложение клиента RDMA может выполнить операцию чтения или записи RDMA путем отправки приложения сервера RDMA запроса уровня приложения вместе с cookie RDMA. В запросе уровня приложения должен быть указан тип операции RDMA (чтение или запись), а также адрес и длина области памяти клиента для удаленного чтения или записи сервером RDMA.

Запрос RDMA можно отправить из клиента RDMA на сервер RDMA двумя способами.

Первый из них предусматривает отправку управляющего сообщения **RDS_CMSG_RDMA_MAP** (со структурой **rds_get_mr_args**) вместе с запросом RDMA уровня приложения с помощью системного вызова **sendmsg()** для сокета RDS. Ядро операционной системы AIX на уровне клиента обрабатывает управляющее сообщение **RDS_CMSG_RDMA_MAP** путем связывания указанной области локальной памяти (из адресного пространства приложения клиента) для доступа DMA и создания cookie RDMA. Затем запрос уровня приложения отправляется серверу вместе с cookie RDMA.

Второй метод состоит из двух шагов. На первом шаге выполняется системный вызов **setsockopt()** с опцией сокета **RDS_GET_MR** и передачей структуры **rds_get_mr_args**. Этот вызов связывает указанную область локальной памяти для доступа DMA и возвращает cookie RDMA. На втором шаге отправляется управляющее сообщение **RDS_CMSG_RDMA_DEST** (с cookie RDMA, который был получен на первом шаге) вместе с запросом RDMA уровня приложения с помощью системного вызова **sendmsg()**.

Первый метод с одним системным вызовом более предпочтителен по сравнению со вторым методом, для которого требуются два системных вызова.

Вместе с запросом на **чтение RDMA** уровня приложения приложение сервера RDMA принимает от клиента управляющее сообщение **RDS_CMSG_RDMA_DEST** (содержащее cookie RDMA). Затем сервер выполняет операцию **чтения RDMA** путем отправки клиенту ответа уровня приложения вместе с управляющим сообщением **RDS_CMSG_RDMA_ARGS** (со структурой **rds_rdma_args**). Ядро операционной системы AIX на уровне сервера обрабатывает управляющее сообщение **RDS_CMSG_RDMA_ARGS** путем связывания указанной области локальной памяти (из адресного пространства приложения сервера) для доступа DMA и динамического запуска операции чтения RDMA. Операция чтения RDMA выполняется адаптером InfiniBand сервера, который взаимодействует с адаптером InfiniBand клиента с целью передачи данных из памяти приложения клиента в память приложения сервера без дополнительного программного вмешательства. После завершения операции чтения RDMA адаптер на сервере отправляет клиенту ответ уровня приложения. Таким образом приложение клиента получает уведомление об успешном выполнении операции чтения RDMA.

Примечание: Операция RDMA запрашивается клиентом с помощью управляющего сообщения **RDS_CMSG_RDMA_MAP** с флагом **RDS_RDMA_USE_ONCE**. В ходе обработки этого запроса область памяти, настроенная для DMA в адресном пространстве клиента, автоматически выключается режим DMA при получении ответа уровня приложения от сервера.

Несмотря на то, что неявное включение и выключение режима DMA упрощает создание приложений RDMA, разработчики должны учитывать, что регистрация памяти для DMA в операционной системе AIX является дорогостоящей операцией. Таким образом, в случае многократного обращения к одной и той же области памяти с помощью RDMA рекомендуется выполнять регистрацию DMA только один раз. Для этого приложение клиента должно использовать управляющее сообщение **RDS_CMSG_RDMA_MAP** без флага **RDS_RDMA_USE_ONCE** при отправке запроса RDMA на сервер. В этом случае при последующих обращениях к той же области памяти клиента на сервер не потребуется отправлять другой запрос. В конце приложению клиента потребуется явным образом выключить режим DMA для области памяти с помощью системного вызова **setsockopt()** с опцией сокета **RDS_FREE_MR**.

Параметры сокета RDS указываются с помощью параметра уровня **SOL_RDS** для системного вызова **setsockopt()** или **getsockopt()**.

Протокол управления передачей:

Протокол **TCP** обеспечивает надежную доставку потока данных между двумя хостами Internet.

Как и **UDP**, **TCP** применяет протокол IP в качестве протокола нижнего уровня для передачи дейтаграмм и поддерживает передачу непрерывного потока дейтаграмм. Однако в отличие от **UDP**, **TCP** обеспечивает надежную доставку сообщений. **TCP** гарантирует, что во время передачи данные не будут искажены, потеряны, скопированы, и не будет изменен их порядок. Это позволяет программистам не встраивать специальные механизмы защиты передачи данных в свои прикладные программы.

Ниже перечислены особенности протокола **TCP**:

Элемент	Описание
Простая передача данных	**TCP** поддерживает двунаправленную передачу непрерывного потока октетов данных между пользователями, разбивая данные на сегменты, которые передаются через Internet. Минимальный размер сегмента **TCP** равен 1024 байтам. В общем случае **TCP** сам решает, когда передать блок пакетов.
Надежность	**TCP** исправляет ошибки, связанные с искажением, потерей, дублированием данных, а также изменением порядка доставки. Для этого **TCP** присваивает всем октетам данных порядковый номер и ожидает поступления подтверждения (ACK) о доставке каждого октета от целевого модуля **TCP**. Если тайм-аут истек до того, как было получено уведомление ACK, то данные передаются повторно. Значение тайм-аута повторной передачи **TCP** для каждого соединения определяется динамически на основе циклического временного счетчика. Порядковые номера позволяют модулю TCP получателя восстановить порядок полученных данных и удалить дубликаты пакетов. Целостность данных проверяется с помощью контрольной суммы, которая подсчитывается отправителем для каждого передаваемого сегмента и проверяется получателем, который отбрасывает поврежденные сегменты.
Управление потоком	Для управления потоком в каждом пакете ACK модуль **TCP** указывает размер окна получателя, то есть допустимый интервал порядковых номеров сообщений, следующих за последним успешно принятым сегментом данных. Размер окна задает число октетов, которое отправитель может передать, пока не будет получено следующее разрешение.
Мультиплексирование	**TCP** позволяет нескольким процессам хоста одновременно работать с функциями **TCP**. На каждом хосте для **TCP** выделяется набор портов. Уникальный идентификатор сокета **TCP** состоит из номера порта, адреса сети и адреса хоста. Каждое соединение однозначно идентифицируется парой сокетов.
Соединения	Для передачи каждого потока данных **TCP** хранит определенную информацию о состоянии. Эта информация составляет описание соединения и содержит такие сведения, как пара сокетов, порядковые номера сегментов и размер окна. Пара сокетов, расположенных на разных концах соединения, уникально идентифицирует это соединение.
Приоритет и защита	Пользователь **TCP** может установить уровень защиты и приоритет отправляемых данных. Если пользователь не указывает собственных значений, то применяются значения по умолчанию.

Эти свойства проиллюстрированы на рисунке **Заголовок пакета TCP**.

На рисунке показано содержимое заголовка пакета **TCP**. Описание полей заголовка приведено ниже.

Рисунок 13. Заголовок пакета TCP

Определения полей заголовков TCP:

Короткие описания полей **Протокол управления передачей** (**TCP**).

Элемент	Описание
Исходный порт	Номер порта исходной прикладной программы.
Порт получателя	Номер порта целевой прикладной программы.
Порядковый номер	Порядковый номер первого байта данных сегмента.
Номер уведомления	Максимальный номер среди номеров полученных байтов.
Смещение данных	Смещение блока данных в сегменте.
Зарезервированное поле	Поле, зарезервированное для будущего применения.
Код	Управляющие флаги, идентифицирующие тип сегмента: **URG** Флаг срочности. **ACK** Флаг пакета, содержащего уведомление о получении. **PSH** Флаг форсированной отправки сегмента (запрос операции PUSH). **RTS** Сброс соединения. **SYN** Синхронизация порядковых номеров. **FIN** Флаг окончания передачи со стороны отправителя.
Размер окна	В этом поле указывается объем данных, который может принять целевое приложение.
Контрольная сумма	Предназначена для проверки целостности заголовка и данных.
Указатель срочных данных	Содержит порядковый номер последнего пакета данных, которые должны быть переданы максимально быстро.
Опции	**Конец списка опций** Обозначает конец списка опций. Это поле указывается в конце всего списка опций, а не после каждой опции. Оно используется только в том случае, если конец списка опций не совпадает с концом заголовка **TCP**. **Нет операции** Обозначает границу между опциями. Может указываться и между другими опциями, например, для выравнивания начала следующей опции по 32-разрядной границе. Отправитель не обязан указывать эту опцию, поэтому у получателя должна быть возможность обработать опцию, даже если ее начало не совпадает с границей слова. **Максимальный размер сегмента** Максимальный размер сегмента, который может быть получен **TCP**. Это поле указывается только в первом запросе на установление соединения.

Интерфейс прикладных программ для работы с **TCP** представляет собой набор библиотечных процедур, основанных на интерфейсе сокетов.

Протоколы TCP/IP прикладного уровня

Верхний уровень **TCP/IP** - это уровень приложений, или прикладной уровень.

На рисунке показаны уровни набора протоколов **TCP/IP**. На верхнем (прикладном) уровне находится

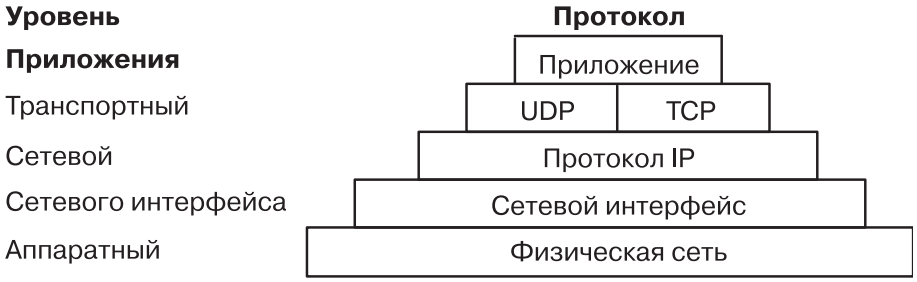


Рисунок 14. Прикладной уровень набора протоколов TCP/IP.

приложение. На транспортном уровне работают **UDP** и **TCP**. Сетевой уровень представлен сетевым (аппаратным) интерфейсом. Аппаратный уровень - это физическая сеть.

При отправке данных от одного приложения приложению другого хоста сети, данные приложения передаются на нижний транспортный уровень, где информация подготавливается к передаче.

Ниже приведен список стандартных протоколов Internet прикладного уровня:

- **Протокол имен доменов**
- **Протокол внешних узлов**
- **Протокол передачи файлов**
- **Протокол Name/Finger**
- **Протокол Telnet**
- **Упрощенный протокол передачи файлов (TFTP)**

Помимо стандартных протоколов, в TCP/IP реализован ряд других популярных протоколов Internet прикладного уровня. Среди них такие протоколы, как:

- Протокол распределенной **локальной сети**
- **Протокол удаленного выполнения команд**
- **Протокол удаленного входа в систему**
- **Протокол удаленной оболочки**
- **Протокол Wake On LAN**
- **Протокол информации о маршрутизации**
- **Протокол сервера времени**

В **TCP/IP** не предусмотрены API для этих протоколов прикладного уровня.

Протокол имен доменов:

Протокол имен доменов (**DOMAIN** позволяет хосту выступать в качестве *сервера имен* для других хостов.

В качестве протокола нижнего уровня протокол **DOMAIN** применяет **UDP** или **TCP**. С помощью этого протокола имена хостов в локальной сети из одного домена могут присваиваться независимо от других доменов. Обычно протокол **DOMAIN** работает на основе **UDP**. Однако если ответные сообщения протокола **UDP** усекаются, то вместо него применяется **TCP**. Протокол **DOMAIN** в **TCP/IP** поддерживает оба транспортных протокола.

В иерархической системе имен **DOMAIN** для преобразования имен и адресов Internet применяются локальные процедуры преобразования, работающие на основе локальной базы данных имен, которая обслуживается демоном **named**. Если указанное хостом имя отсутствует в локальной базе данных, процедура преобразования обращается к удаленному серверу имен **DOMAIN**. Если необходимая информация не найдена и на удаленном сервере, то процедура преобразования просматривает файл `/etc/hosts`.

Примечание: **TCP/IP** разрешает локальным процедурам преобразования применять протокол **DOMAIN**, если на хосте есть файл `/etc/resolv.conf`. Если такого файла нет, **TCP/IP** настраивает локальные процедуры преобразования на применение базы данных `/etc/hosts`.

Протокол **DOMAIN** реализован в **TCP/IP** в виде демона **named** и процедур преобразования. Для него не предусмотрен API.

Протокол внешних шлюзов:

Протокол внешних шлюзов (**EGP**) позволяет внешними шлюзам *автономных систем* обмениваться информацией о маршрутизации.

Автономные системы:

Автономная система - это набор сетей и шлюзов, которые находятся под единым управлением.

Если шлюз расположен в той же самой автономной системе, то он называется *внутренним соседом*, а если в другой автономной системе, то *внешним соседом*. Шлюзы, которые применяют протокол **EGP** для обмена информацией о маршрутизации, называются *равноправными шлюзами* **EGP** или *партнерами*. С помощью **EGP** шлюзы автономных систем предоставляют доступ к информации своим соседям **EGP**.

С помощью протокола **EGP** внешние шлюзы могут запрашивать у других внешних шлюзов согласие на обмен информацией о досягаемости сетей, проверять работоспособность соседей **EGP** и обмениваться сообщениями с информацией об изменении маршрутов.

Протокол **EGP** позволяет внешним шлюзам обмениваться информацией о досягаемости лишь тех сетей, которые достижимы из автономной сети шлюза. Таким образом, внешний шлюз передает с помощью **EGP** информацию своим соседям **EGP** , но не распространяет информацию о соседях **EGP** за пределы автономной системы.

Протокол **EGP** не анализирует метрики расстояний, которые указываются в сообщениях о маршрутизации, передаваемых другими протоколами. В поле расстояния **EGP** указывает, существует ли данный путь (значение 255 означает, что сеть недостижима). Это значение не применяется для выбора наикратчайшего из двух маршрутов, если эти маршруты не относятся к одной автономной системе. Поэтому **EGP** не применяется в качестве алгоритма маршрутизации. В результате существует только один путь от внешнего шлюза до любой сети.

В отличие от **Протокола информации о маршрутизации** (**RIP**), который применяется в автономных системах Internet для динамической настройки маршрутов, маршруты **EGP** фиксированы и заданы в файле **/etc/gated.conf**. **EGP** применяет **IP** в качестве протокола нижнего уровня.

Типы сообщений EGP:

Здесь приведены сведения о различных типах сообщений Протокола внешних шлюзов (EGP).

Элемент	Описание
Neighbor Acquisition Request (Регистрация соседа)	Это сообщение содержит приглашение стать соседями, адресованное другому внешнему шлюзу.
Neighbor Acquisition Reply (Ответ на запрос Регистрация соседа)	Отправляется внешним шлюзом, если он принимает приглашение стать соседями.
Neighbor Acquisition Refusal (Отклонение запроса Регистрация соседа)	Отправляется внешним шлюзом, если он отвергает запрос Регистрация соседа. Это сообщение содержит причины отказа, например, `недостаточно памяти в таблице`.
Neighbor Cease (Отказ от соседа)	Отправляется внешним шлюзом, если он хочет исключить какой-то внешний шлюз из списка соседей. Это сообщение содержит причины отказа, например, `возникла неполадка`.
Neighbor Cease Acknowledgment (Уведомление о получении отказа)	Отправляется внешним шлюзом, чтобы уведомить отправителя о получении сообщения Отказ от соседа.
Neighbor Hello (Приветствие соседа)	Отправляется внешним шлюзом, если он хочет проверить готовность соседа к работе. Шлюз отправляет сообщение `Hello`, а другой хост отвечает ему сообщением `I Heard You`.
I Heard You (Ответ на приветствие)	Отправляется внешним шлюзом в ответ на сообщение `Hello`. Сообщение `I Heard You` содержит информацию о досягаемости сетей отвечающего шлюза. Кроме того, в нем может быть указана причина, по которой запрашивающий шлюз недостижим, например, `Шлюз недостижим из-за неполадок в работе моего сетевого интерфейса`.
NR Poll (Опрос соседей)	Отправляется внешним шлюзом для получения информации о досягаемости сетей из автономных систем, в которых расположены соседи.
Network Reachability (Досягаемость сетей)	Отправляется внешним шлюзом в ответ на сообщение `NR Poll`. В сообщении `Network Reachability` каждый сосед указывает адреса шлюзов, которые достижимы через его соседей.

Элемент	Описание
EGP Error (Ошибка EGP)	Отправляется внешним шлюзом в ответ на сообщение EGP, содержащее неправильную контрольную сумму или поля, в которых заданы неверные значения.

В **TCP/IP** протокол **EGP** реализован в виде сервера **gated**. Для него не предусмотрен API.

Протокол передачи файлов:

Протокол передачи файлов (**FTP**) предназначен для обмена данными между хостами различных типов, а также для обмена файлами между внешними хостами через промежуточные системы.

FTP служит для выполнения таких задач, как просмотр списка удаленных каталогов, переход в другой удаленный каталог, создание и удаление каталогов в другой системе и передача набора файлов по одному запросу. **FTP** обеспечивает защиту данных при передаче, отправляя внешнему хосту имя пользователя и пароль учетного файла пользователя. Хотя в основном **FTP** предназначен для приложений, этот протокол может применяться и для интерактивного диалога между пользователями.

Для передачи файлов протокол **FTP** применяет соединения **TCP/IP**, обеспечивающие надежную доставку данных в режиме потока, а для передачи команд и ответов - соединения Telnet. **FTP** распознает некоторые форматы файлов, в том числе NETASCII, IMAGE и Local 8.

В **TCP/IP** протокол **FTP** реализован в виде команд клиента (**ftp**) и сервера (**ftpd**). Для него не предусмотрен интерфейс прикладных программ (API).

При создании анонимных пользователей и каталогов ftp убедитесь в том, что эти каталоги (например, `/u/ftp`) принадлежат пользователю root и не допускают записи (например, `dr-xr-xr-x`). Для создания учетных файлов таких пользователей, файлов и каталогов предназначен сценарий `/usr/samples/tcpip/anon.ftp`.

Протокол Telnet:

Протокол **Telnet** (**TELNET**) предназначен для взаимодействия терминалов и связанных с ними процессов.

TELNET часто применяется программами эмуляции терминала для входа в удаленную систему. Однако **TELNET** может применяться и для установления соединения между терминалами или процессами. Протокол **TELNET** применяется многими другими протоколами (например, **FTP**) для создания канала, предназначенного для передачи управляющей информации.

В **TCP/IP** протокол **TELNET** реализован в виде команд клиента **tn**, **telnet** или **tn3270**. Демон **telnetd** не предоставляет API для работы с **TELNET**.

В **TCP/IP** предусмотрены следующие параметры **TELNET**, которые применяются по согласованию между клиентом и сервером:

Элемент	Описание
BINARY TRANSMISSION (Применяется в сеансах **tn3270**)	Передача символьной информации в двоичном формате.
SUPPRESS GO_AHEAD (Операционная система подавляет опции GO-AHEAD).	После установления соединения между отправителем и получателем данных отправителю не нужно передавать опцию GO_AHEAD. Если опция GO_AHEAD нежелательна, то участники соединения могут ее подавлять при передаче данных в обоих направлениях. Для каждого направления соединения это действие выполняется независимо.
TIMING MARK (Запрос распознан, но ответ отрицательный)	Уведомляет, что переданные ранее данные были полностью обработаны.
EXTENDED OPTIONS LIST	Увеличивает список опций **TELNET** еще на 256 опций. В стандартном списке опций **TELNET** содержится только 256 опций.
ECHO (Пользовательская команда)	Передает полученные и выведенные на экран данные обратно отправителю.

Элемент	Описание
TERM TYPE	Разрешает серверу определить тип терминала, с которым взаимодействует программа **TELNET** пользователя.
SAK (Ключ защиты)	Настраивает среду, необходимую для установления защищенных соединений между программой TELNET пользователя и удаленным терминалом.
NAWS (Согласование размера окна)	Разрешает клиенту и серверу динамически согласовывать размер окна. Этот параметр применяется приложениями, которые поддерживают изменение размера окна в процессе передачи данных.

Примечание: Для поддержки кодовой страницы ISO 8859 Latin клиент и сервер **TELNET** должны поддерживать передачу 8-разрядных данных в режиме ASCII.

Упрощенный протокол передачи файлов (TFTP):

Упрощенный протокол передачи файлов (**TFTP**) предназначен для обмена файлами с внешними хостами.

Для передачи файлов **TFTP** применяется ненадежный **Протокол пользовательских дейтаграмм**, поэтому он обычно работает быстрее, чем **FTP**. Как и **FTP**, **TFTP** поддерживает передачу файлов в формате NETASCII и в 8-разрядном двоичном формате. В отличие от **FTP**, **TFTP** не поддерживает просмотр каталогов или переход в другой каталог внешнего хоста. Кроме того, в нем не предусмотрена защита с помощью пароля. TFTP позволяет работать только с общими каталогами.

В **TCP/IP** протокол **TFTP** реализован в виде команд клиента (**tftp** и **utftp**) и сервера (**tftpd**). Команда **utftp** применяется в конвейере вместо команды **tftp**. В **TCP/IP** не предусмотрен API для данного протокола.

Дополнительная информация приведена в описании команды **tftp** or **utftp** и в описании демона **tftpd** в книге *Справочник по командам, том 5*.

Протокол Name/Finger:

Протокол Name/Finger (**FINGER**) - это протокол Internet прикладного уровня, предоставляющий интерфейс для передачи данных между командой **finger** и демоном **fingerd**.

Демон **fingerd** предоставляет информацию о пользователях, которые в настоящий момент работают на указанном удаленном хосте. Если в команде finger вы укажете пользователя отдельного хоста, то будет показана информация об этом пользователе. Протокол **FINGER** должен поддерживаться как удаленным, так и локальным хостом. В качестве протокола нижнего уровня **FINGER** применяет **Протокол управления передачей** (“Протокол управления передачей” на стр. 151).

Примечание: В **TCP/IP** не предусмотрен API для данного протокола.

Дополнительная информация приведена в описании команды **finger** и демона **fingerd** в книге *Справочник по командам, том 2*.

Протокол распределенной вычислительной сети:

Протокол **распределенной вычислительной сети (DCN)** - это одна из версий протокола локальной сети. Он реализован в виде сервера **gated**.

Протокол локальной сети (**HELLO**) - это протокол внутренних шлюзов для автономной системы. (За дополнительной информацией обратитесь к разделу “Автономные системы” на стр. 155.) **HELLO** предоставляет информацию о соединениях, маршрутах и времени прохождения пакетов по определенному маршруту. С помощью этой информации любой компьютер сети может определить наикратчайший маршрут к целевому хосту на основе временной задержки, а затем динамически обновить информацию о маршрутах к данному хосту.

Дополнительная информация приведена в описании демона **gated** в книге *Справочник по командам, том 2*.

Протокол удаленного выполнения команд:

Протокол удаленного выполнения команд позволяет запускать команды на удаленных хостах, поддерживающих этот протокол. Он реализован в виде команд клиента (**rexec**) и сервера (**rexecd**).

Дополнительная информация приведена в описании команды **rexec** и демона **rexecd** в книге *Справочник по командам, том 4*.

Протокол удаленного входа в систему:

Протокол удаленного входа в систему позволяет пользователям входить в удаленные системы и работать с их терминалами так, как если бы они были напрямую подключены к этим системам. Этот протокол реализован в виде команд клиента (**rlogin**) и сервера (**rlogind**).

Дополнительная информация приведена в описании команды **rlogin** и демона **rlogind** в книге *Справочник по командам, том 4*.

Протокол удаленной оболочки:

Протокол удаленной оболочки позволяет запускать командную оболочку на удаленных хостах, поддерживающих этот протокол. Он реализован в виде команд клиента (**rshd**) и сервера (**rshd**).

Дополнительная информация приведена в описании команды **rsh** и демона **rshd** в книге *Справочник по командам, том 4*.

Протокол Wake On LAN:

Протокол **Wake On LAN (WOL)** позволяет восстановить работу одного или нескольких хостов, подключенных к сети и находящихся в ждущем режиме. Для восстановления работы применяется особый пакет, отправляемый на заданный адрес или на адреса заданной подсети.

Дополнительная информация о применении протокола **WOL** приведена в описании команды **wol** в книге *Справочник по командам, том 6*.

Протокол информации о маршрутизации:

Протокол информации о маршрутизации (**RIP**) собирает информацию о длине маршрута, измеряемой в транзитных участках, а также обслуживает записи таблиц маршрутизации ядра. Он реализован в виде серверов **routed** и **gated**.

Дополнительная информация приведена в описании демонов **routed** и **gated**.

Протокол сервера времени:

Демон **timed** применяется для синхронизации системного времени хостов.

Он построен по принципу клиент/сервер. Дополнительная информация приведена в описании команды **timedc** и демона **timed** в книге *Справочник по командам, том 5*.

Присвоенные номера

Для обеспечения совместимости с общей сетевой средой большинству протоколов, служб, компонентов и сетей в рамках Internet были присвоены стандартные номера. Кроме того, некоторым компьютерам, сетям, операционным системам, протоколам, службам и терминалам были присвоены стандартные имена.

В **TCP/IP** правила именования и нумерации соответствуют RFC 1010, *Assigned numbers*.

Протокол Internet (**IP**) добавляет к пакету **IP**-заголовок, содержащий поле размером 4 бита, в котором указывается версия общепринятого сетевого протокола, применяемая в настоящее время. Для **IP** десятичное значение номера версии равно 4. Более подробная информация о стандартных номерах и именах **TCP/IP** приведена в файлах `/etc/protocols` и `/etc/services`, поставляемых вместе с **TCP/IP**. За дополнительной информацией о стандартных номерах и именах обратитесь к RFC 1010 и файлу `/etc/services`.

Карты сетевых адаптеров локальной сети TCP/IP

Карта сетевого адаптера - это физическое устройство, которое непосредственно подключается к сетевому кабелю. Она отвечает за прием и передачу данных на физическом уровне.

Карта сетевого адаптера управляется драйвером устройства сетевого адаптера.

Для каждой сети, к которой должен быть подключен компьютер, необходима отдельная карта сетевого адаптера (даже если эти сети одного типа). Например, если система подключена к двум сетям Token-Ring, то в ней должно быть установлено две карты сетевого адаптера.

TCP/IP поддерживает следующие типы карт сетевых адаптеров:

- Ethernet версии 2
- IEEE 802.3
- Token-Ring
- Асинхронные адаптеры и встроенные последовательные порты
- Оптоволоконный интерфейс распределенных данных (FDDI)
- Конвертер последовательного оптического канала (описан в разделе *Kernel Extensions and Device Support Programming Concepts*)
- Fibre-Channel

В сетях Ethernet и IEEE 802.3 применяются адаптеры одного типа.

На каждом компьютере предусмотрено несколько разъемов, в которые можно устанавливать адаптеры связи. Кроме того, каждая машина поддерживает определенное количество адаптеров связи данного типа. В рамках этих ограничений (обусловленных программным обеспечением) вы можете устанавливать любые сочетания адаптеров, общее количество которых не превышает число разъемов расширения в вашей системе (аппаратное ограничение).

Независимо от того, сколько преобразователей последовательных оптических каналов поддерживается системой, можно настроить только один интерфейс **Протокола управления передачей/Протокола Internet (TCP/IP)**. Драйвер последовательного оптического устройства позволяет применять оба преобразователя каналов даже в том случае, когда настроен только один логический интерфейс **TCP/IP**.

Установка сетевого адаптера

Воспользуйтесь данной процедурой для установки сетевого адаптера.

Для установки сетевого адаптера:

1. Выключите компьютер. Информация о том, как выключить систему, приведена в справке по команде **shutdown**.
2. Выключите питание компьютера.
3. Снимите крышку с системного блока.
4. Найдите свободный разъем и вставьте карту адаптера. Убедитесь, что карта адаптера вставлена в разъем полностью.
5. Закройте крышку системного блока.
6. Включите компьютер.

Управление и настройка адаптера

Процедуры настройки и управления адаптерами типа Token-Ring или Ethernet приведены в следующей таблице.

Таблица 59. Задачи управления и настройки адаптеров

Процедура	Команды быстрого доступа SMIT	Команда или файл
Настроить адаптер	`smit chgtok` (token-ring) `smit chgenet` (Ethernet)	1. Определить имя адаптера:[1] `lsdev -C -c adapter -t tokenring -H` or `lsdev -C -c adapter -t ethernet -H` 2. При необходимости измените значение быстродействия (Token-Ring) или типа разъема (Ethernet). Например: `chdev -l tok0 -a ring_speed=16 -P` или `chdev -l ent0 -a bnc_select=dix -P`
Определение аппаратного адреса сетевого адаптера	`smit chgtok` (token-ring) `smit chgenet` (Ethernet)	`lscfg -l tok0 -v` (token-ring)[2] `lscfg -l ent0 -v` (Ethernet)[2]

Задать альтернативный аппаратный адрес	`smit chgtok` (token-ring) `smit chgenet` (Ethernet)	1. Укажите альтернативный аппаратный адрес. Например, для token-ring: [2,3]
		`chdev -l tok0 -a alt_addr=0X10005A4F1B7F`
		Для Ethernet:[2,3]
		`chdev -l ent0 -a alt_addr=0X10005A4F1B7F -p`
		2. Выберите альтернативный адрес для token-ring ring:[4]
		`chdev -l tok0 -a use_alt_addr=yes`
		Для Ethernet:[4]
		`chdev -l ent0 -a use_alt_addr=yes`

Примечание:

1. Имя сетевого адаптера можно изменить при перемещении его из одного разъема в другой или удалении из системы. Во всех подобных случаях выполняйте команду **diag -a**, чтобы обновить базу данных конфигурации.
2. Замените `tok0` и `ent0` на имя вашего адаптера.
3. Вместо `0X10005A4F1B7F` подставьте ваш аппаратный адрес.
4. После выполнения этой процедуры может нарушиться соединение с другими хостами, пока они не обновят содержимое своих кэшей ARP и не получат новый аппаратный адрес данного хоста.

Виртуальные локальные сети

Виртуальные локальные сети (VLAN) представляют собой логические сетевые домены. VLAN разбивает группы пользователей физической сети на сегменты логической сети.

Данная реализация основана на стандарте IEEE 802.1Q VLAN и поддерживает несколько ИД VLAN, применяющих адаптеры Ethernet. Для верхних уровней иерархии протоколов (IP и т.п.) каждый ИД VLAN связан с отдельным интерфейсом Ethernet. Для каждой VLAN создается логический экземпляр адаптера Ethernet, например, `ent1`, `ent2` и т.д.

IEEE 802.1Q VLAN можно настроить для любых поддерживаемых адаптеров Ethernet. Адаптер должен быть подключен к коммутатору, поддерживающему IEEE 802.1Q VLAN.

В одной системе можно настроить несколько логических устройств VLAN. Каждое логическое устройство VLAN представляет собой экземпляр адаптера Ethernet. С помощью этих логических устройств можно настроить те же IP-интерфейсы, что и с помощью физических адаптеров Ethernet. Следовательно, нужно увеличить значение параметра *ifsize* (значение по умолчанию - 8) команды **no** , чтобы помимо интерфейсов Ethernet оно учитывало и логические устройства VLAN. За более подробной информацией обратитесь к документации по команде **no**.

Для устройств VLAN можно задать различные максимальные размеры блока передачи (MTU), даже если они используют один и тот же физический адаптер Ethernet.

Параметры VLAN настраиваются с помощью SMIT. Введите в командной строке `smit vlan` и выберите нужный пункт в главном меню VLAN. Выполнить настройку вам поможет электронная справка.

После сохранения параметров VLAN настройте IP-интерфейс, например, `en1` для Ethernet или `et1` для IEEE 802.3, с помощью SMIT или команд.

AIX 5.3 и более поздние версии поддерживают виртуальный Ethernet, используя виртуальный переключатель ввода-вывода в качестве средства связи внутри памяти между разделами в системе POWER5. Переключатель также поддерживает теги IEEE 802.1Q, что позволяет адаптерам виртуального Ethernet относиться к разным VLAN. Адаптеры виртуального Ethernet создаются и настраиваются в разделах с помощью Консоли аппаратного обеспечения (HMC). После создания раздел обнаружит адаптер виртуального Ethernet в дереве открытого встроенного программного обеспечения во время поиска устройств. После его обнаружения адаптер виртуального Ethernet будет настроен и использован точно так же, как адаптер обычного физического Ethernet. Более подробная информация приведена в документации по аппаратному обеспечению системы POWER5.

Примечание:

1. Если вы попытаетесь настроить ИД VLAN, который уже занят указанным адаптером, то будет показано следующее сообщение об ошибке:

   ```
   Ошибка метода
   (/usr/lib/methods/chgvlan):
     0514-018 Для следующих атрибутов заданы неверные
              значения:
              vlan_tag_id    ИД VLAN
   ```

2. Если логическое устройство VLAN в данный момент используется (например, IP-интерфейсом), удалить это устройство нельзя. При попытке удалить устройство появится примерно следующее сообщение:

   ```
   Ошибка метода (/usr/lib/methods/ucfgcommo):
           0514-062 Невозможно выполнить запрошенную функцию, так как
                    устройство занято.
   ```

   Для удаления логического устройства VLAN сначала отключите пользователя. Например, если устройство занято IP-интерфейсом `en1`, то вызовите следующую команду:

   ```
   ifconfig en1 detach
   ```

   Затем удалите сетевой интерфейс с помощью меню TCP/IP инструмента SMIT.

3. Если логическое устройство VLAN в данный момент используется (например, IP-интерфейсом), то изменить параметры этого устройства (ИД VLAN или базовый адаптер) нельзя. При попытке удалить устройство появится примерно следующее сообщение:

   ```
   Ошибка метода
   (/usr/lib/methods/chgvlan):
           0514-062 Невозможно выполнить запрошенную функцию, так как
                    устройство занято.
   ```

   Для изменения логического устройства VLAN сначала отключите пользователя. Например, если устройство занято IP-интерфейсом `en1`, то вызовите следующую команду:

   ```
   ifconfig en1 detach
   ```

   Затем измените VLAN и снова добавьте сетевой интерфейс с помощью меню TCP/IP инструмента SMIT.

Устранение неполадок VLAN:

Для устранения неполадок VLAN применяются команды **tcpdump** и **trace**.

Ниже указан ИД точки трассировки для различных типов пакетов:

Элемент	Описание
передача пакетов	3FD
прием пакетов	3FE
другие события	3FF

Команда **entstat** позволяет просмотреть общую статистику по физическому адаптеру, для которого настроена VLAN. Она *не* предоставляет статистику по отдельным логическим устройствам VLAN.

Ограничения VLAN:

Создание удаленного дампа по сети VLAN не поддерживается. Кроме того, логические устройства VLAN нельзя применять при создании Cisco Systems Etherchannel.

Сетевые интерфейсы TCP/IP

На уровне сетевого интерфейса **TCP/IP** создает из IP-дейтаграмм пакеты, которые могут интерпретироваться и передаваться с помощью определенных сетевых технологий.

Сетевой интерфейс - это программное обеспечение, взаимодействующее с сетевым драйвером и с уровнем IP. Сетевой интерфейс обеспечивает уровню IP доступ ко всем имеющимся сетевым адаптерам.

Программное обеспечение уровня IP выбирает сетевой интерфейс в соответствии с целевым адресом передаваемого пакета. Каждый сетевой интерфейс имеет свой сетевой адрес. Уровень сетевого интерфейса отвечает за добавление и удаление заголовков протокола уровня передачи, необходимых для доставки сообщения в пункт назначения. Драйвер **сетевого адаптера** управляет картой сетевого адаптера.

Сетевой интерфейс обычно связан с сетевым адаптером, хотя это и не всегда так. Например, циклический интерфейс не связан с картой адаптера. В системе должны быть установлены карты сетевых адаптеров для каждой сети, к которой она подключена (даже если это сети одного типа). Однако для работы со всеми сетевыми адаптерами нужен только один экземпляр программного обеспечения сетевого интерфейса. Например, если система подключена к двум сетям Token-Ring, то в ней должно быть установлено две карты сетевого адаптера. При этом требуется только один экземпляр программного обеспечения сетевого интерфейса **token-ring** и один экземпляр драйвера Token-Ring.

TCP/IP поддерживает следующие типы сетевых интерфейсов:

- Ethernet версии 2 (en)
- IEEE 802.3 (et)
- Token-ring (tr)
- **Протокол подключения к Internet по последовательной линии (SLIP)**
- Циклический интерфейс (lo)
- FDDI
- Последовательный оптический интерфейс (so)
- **Протокол двухточечной связи (PPP)**
- Виртуальный IP (vi)

Интерфейсы Ethernet, 802.3 и Token-Ring используются в локальных сетях (LAN). Интерфейс **SLIP** применяется для работы с последовательными соединениями. Циклический интерфейс позволяет хосту отправлять сообщения самому себе. Последовательный оптический интерфейс предназначен для оптических двухточечных сетей с использованием программы для работы с устройствами Последовательной оптической линии связи. **Двухточечный протокол** чаще всего используется при подключении к другому компьютеру или сети по модему. Интерфейс виртуального IP (часто называемый *виртуальным интерфейсом*) не связан с конкретным сетевым адаптером. На одном хосте можно настроить несколько экземпляров виртуального интерфейса. В этом случае в качестве исходного будет применяться адрес первого виртуального интерфейса, если приложение явно не выберет другой интерфейс. Процессы, использующие виртуальный IP-адрес в

качестве исходного адреса, могут отправлять пакеты через любой сетевой интерфейс, обеспечивающий наилучшую маршрутизацию пакетов. Пакеты, отправленные по виртуальному IP-адресу, доставляются процессу вне зависимости от того, через какой интерфейс они были получены.

Автоматическая настройка сетевых интерфейсов

После установки сетевого адаптера операционная система автоматически добавляет для него соответствующий сетевой интерфейс.

Например, если вы установите адаптер Token-Ring, то операционная система присвоит ему имя `tok0` и добавит сетевой интерфейс Token-Ring с именем `tr0`. Если вы установите адаптер Ethernet, то операционная система присвоит ему имя `ent0` и добавит интерфейсы Ethernet версии 2 и IEEE 802.3 с именами `en0` и `et0`, соответственно.

В большинстве случаев существует однозначное соответствие между именами адаптеров и сетевых интерфейсов. Например, адаптер Token-Ring `tok0` соответствует интерфейсу `tr0`, адаптер tok1 - интерфейсу `tr1` и т.п. Аналогично, адаптер Ethernet `ent0` соответствует интерфейсу `en0` (для Ethernet версии 2) и `et0` (для IEEE 802.3), а адаптер `ent1` - интерфейсу `en1` (для Ethernet версии 2) и `et1` (для IEEE 802.3).

Примечание: Обычно добавлять и удалять сетевые интерфейсы вручную не нужно. Однако это может потребоваться во время выполнения некоторых диагностических процедур при поиске неполадок. В этом случае для удаления или добавления интерфейса можно использовать команду быстрого доступа SMIT **smit inet**.

Значения конфигурации TCP/IP по умолчанию

При каждом запуске системы операционная система автоматически настраивает программное обеспечение сетевого интерфейса на основе информации, хранящейся в базе данных ODM. Первоначально сетевой интерфейс настраивается со значениями по умолчанию.

Для установления соединения с помощью этого интерфейса нужно задать IP-адрес. Это единственный атрибут, который вы должны задать самостоятельно. Для всех остальных необходимых атрибутов могут использоваться значения по умолчанию. Ниже описаны значения по умолчанию для различных типов сетей.

Значения Ethernet TCP/IP по умолчанию:

Значения допустимых атрибутов сетевого адаптера Ethernet можно изменить с помощью меню Выбор сетевого интерфейса в SMIT.

Атрибут	Значение по умолчанию	Возможные значения
netaddr		
state	down	up, down, detach
arp	yes	yes, no
netmask		
для оповещения		

Ниже перечислены атрибуты драйвера сетевого адаптера Ethernet и их значения по умолчанию. Эти значения можно изменить с помощью программы SMIT (меню Драйверы сетевых интерфейсов).

Атрибут	Значение по умолчанию	Возможные значения
mtu	1500	От 60 до 1500

Значения 802.3 TCP/IP по умолчанию:

Значения допустимых атрибутов сетевого адаптера 802.3 можно изменить с помощью меню Выбор сетевого интерфейса в SMIT.

Атрибут	Значение по умолчанию	Возможные значения
netaddr		
state	down	up, down, detach
arp	yes	yes, no
netmask		
для оповещения		

Ниже приведены допустимые значения атрибута драйвера сетевого устройства 802.3 и его значение по умолчанию. Это значение можно изменить с помощью программы SMIT (меню Драйверы сетевых интерфейсов).

Атрибут	Значение по умолчанию	Возможные значения
mtu	1492	От 60 до 1492

Значения Token-Ring TCP/IP по умолчанию:

Значения допустимых атрибутов сетевого адаптера token-ring можно изменить с помощью меню Выбор сетевого интерфейса в SMIT.

Атрибут	Значение по умолчанию	Возможные значения
netaddr		
netmask		
state	down	up, down, detach
arp	yes	yes, no
hwloop	no	yes, no
netmask		
для оповещения		
allcast	нет	yes, no

Ниже перечислены атрибуты драйвера сетевого устройства Token-Ring и их значения по умолчанию. Эти значения можно изменить с помощью программы SMIT (меню Драйверы сетевых интерфейсов).

Атрибут	Значение по умолчанию	Возможные значения
mtu (4 Мбит/с)	1500	От 60 до 4056
mtu (16 Мбит/с)	1500	От 60 до 17960

Примечание: Если соединение устанавливается через мост, то значение MTU (по умолчанию оно равно 1500) должно быть изменено так, чтобы его размер был на 8 байт меньше максимального размера информационного поля (I-кадра) для данного моста. Например, если максимальный размер информационного кадра составляет 1500, то в поле управления маршрутизацией должен быть задан размер MTU, равный 1492. Это относится только к сетевым интерфейсам Token-Ring. Дополнительная информация приведена в разделе “Неполадки TCP/IP при использовании моста Token-Ring/Token-Ring” на стр. 432.

Для адаптеров IBM® 16/4 PowerPC Token-Ring для PowerPC (ISA) значение MTU не должно превышать 2000.

Значения SLIP TCP/IP по умолчанию:

Значения допустимых атрибутов сетевого адаптера SLIP можно изменить с помощью меню Выбор сетевого интерфейса в SMIT.

Атрибут	Значение по умолчанию	Возможные значения
netaddr		
dest		
state	up	up, down, detach
netmask		

Ниже перечислены атрибуты драйвера сетевого адаптера SLIP и их значения по умолчанию. Эти значения можно просмотреть с помощью программы SMIT (меню Драйверы сетевого интерфейса).

Атрибут	Значение по умолчанию	Возможные значения
mtu	1006	От 60 до 4096

Значения последовательного оптического интерфейса TCP/IP по умолчанию:

Допустимые значения преобразователя последовательного оптического сетевого канала можно изменить с помощью меню Выбор сетевого интерфейса в SMIT.

Атрибут	Значение по умолчанию	Возможные значения
netaddr		
state	down	up, down, detach
netmask		

Ниже приведен список допустимых атрибутов и значений по умолчанию для драйвера последовательного оптического сетевого адаптера. Эти значения можно просмотреть с помощью SMIT (меню Драйверы сетевого интерфейса).

Атрибут	Значение по умолчанию	Возможные значения
mtu	61428	От 1 до 61428

Реализация нескольких сетевых интерфейсов в одной сети

Если к одной сети подключено несколько сетевых интерфейсов, то для них должны быть заданы уникальные IP-адреса.

Функция множественной маршрутизации позволяет добавлять в таблицу маршрутизации IP маршруты для интерфейсов с несколькими маршрутами в одной подсети. При этом исходящие данные могут отправляться через различные, а не через один интерфейс.

Управление сетевым интерфейсом

Для управления интерфейсами воспользуйтесь сетью WSM, командой быстрого доступа или выполните задачи из следующей таблицы.

Таблица 60. Задачи управления сетевыми интерфейсами

Процедура	Команды быстрого доступа SMIT	Команда или файл
Просмотреть список всех сетевых устройств	`smit lsinet`	**lsdev -C -c if**
Настроить сетевое устройство	`smit chinet`	См. описание команды **ifconfig** и файла `rc.net`
Изменить информацию о сетевом интерфейсе с удаленно смонтированным каталогом `/usr`	`smit chdev`[1,2]	**chgif**[1,2]
Получение статистической информации о сетевом интерфейсе		**netstat -v**

Примечание:

1. Изменения, внесенные при удаленном монтировании каталога `/usr` сохраняются только в информационной базе данных (ODM). Они применяются после перезапуска сети или выполнения команды **ifconfig**.
2. При работе с удаленным каталогом `/usr` не изменяйте текущий интерфейс, так как в этом каталоге хранятся библиотеки, команды и ядро.

Сетевые опции интерфейсов

Для того чтобы данные передавались по сети с максимальной скоростью (100 Мб/с и более), необходимо правильно настроить сетевые интерфейсы **TCP/IP**. Эта задача осложняется тем, что в одной системе может быть настроено несколько сетевых интерфейсов **TCP/IP**, среди которых могут быть как обычные, так и высокоскоростные интерфейсы.

В операционной системе AIX сетевые параметры интерфейса (ISNO) позволяют администраторам настраивать каждый интерфейс **TCP/IP** для обеспечения максимальной производительности.

Для каждого поддерживаемого интерфейса предусмотрено пять параметров ISNO: **rfc1323**, **tcp_nodelay**, **tcp_sendspace**, **tcp_recvspace** и **tcp_mssdflt**. Если эти параметры заданы, они переопределяют соответствующие системные параметры, заданные командой **no**. Если параметры ISNO не заданы для интерфейса, то применяются системные параметры. Приложение может переопределить параметры ISNO для конкретного сокета, задав необходимые значения с помощью функции **setsockopt**.

Значения параметров ISNO учитываются только в том случае, если атрибут сети **use_isno** равен 1. Этот атрибут можно задать с помощью команды **no**. Значение **use_isno** по умолчанию равно 1.

Для некоторых высокоскоростных адаптеров параметры ISNO заданы в базе данных ODM по умолчанию.

Для интерфейсов Gigabit Ethernet с параметром MTU, равным 9000, опциям ISNO присвоены следующие значения по умолчанию:

Имя	Значение в AIX 4.3.3	Значение в AIX 4.3.3 (4330-08)	Значение в AIX 5.1 (и старше)
tcp_sendspace	131072	262144	262144
tcp_recvspace	92160	131072	131072
rfc1323	1	1	1

Для интерфейсов Gigabit Ethernet с параметром MTU, равным 1500, опциям ISNO присвоены следующие значения по умолчанию:

Имя	Значение в AIX 4.3.3	Значение в AIX 4.3.3 (4330-08)	Значение в AIX 5.1 (и старше)
tcp_sendspace	65536	131072	131072
tcp_recvspace	16384	65536	65536
rfc1323	0	не задано	не задано

Для интерфейсов FDDI с параметром MTU, равным 4352, опциям ISNO присвоены следующие значения по умолчанию:

Имя	Значение
tcp_sendspace	45046
tcp_recvspace	45046

Параметры ISNO нельзя просмотреть или изменить с помощью SMIT. Для их настройки служат команды **chdev** и **ifconfig**. Значения, заданные с помощью команды **ifconfig**, действуют только до следующего перезапуска системы. Команда **chdev** изменяет значения в базе данных ODM, поэтому они будут действовать и после перезагрузки. Для просмотра текущих значений служат команды **lsattr** и **ifconfig**.

Ниже перечислены команды, которые служат для проверки правильности настройки системы и поддерживаемых интерфейсов, а также для настройки и просмотра параметров интерфейсов.

1. Убедитесь, что в системе установлено необходимое программное обеспечение для работы с интерфейсами с помощью команд **no** и **lsattr**.
   - Убедитесь, что включена опция **use_isno**. Для этого вызовите следующую команду:

     ```
     $ no -a | grep isno
               use_isno=1
     ```

   - С помощью команды **lsattr -El** убедитесь, что интерфейс поддерживает новые параметры ISNO:

     ```
     $ lsattr -E -l en0 -H
               атрибут   значение         описание
               rfc1323                    н/д
               tcp_nodelay                н/д
               tcp_sendspace              н/д
               tcp_recvspace              н/д
               tcp_mssdflt                н/д
     ```

2. Задайте параметры интерфейса с помощью команды **ifconfig** или **chdev**. Команда **ifconfig** устанавливает значения временно, поэтому в целях тестирования рекомендуется применять именно ее. Команда **chdev** изменяет базу данных ODM, поэтому новые значения будут действовать и после перезагрузки.
   - Задайте для **tcp_recvspace** и **tcp_sendspace** значение 64 Кб и включите опцию **tcp_nodelay** одним из следующих способов.

     ```
     $ ifconfig en0 tcp_recvspace 65536 tcp_sendspace 65536 tcp_nodelay 1
     $ chdev -l en0 -a tcp_recvspace=65536 -a tcp_sendspace=65536 -a tcp_nodelay=1
     ```

   - Если в выводе команды **no** указано, что глобальный параметр rfc1323 равен единице, то пользователь root может выключить опцию **rfc1323** для всех соединений, установленных через en0:

     ```
     $ ifconfig en0 rfc1323 0
     $ chdev -l en0 -a rfc1323=0
     ```

3. Просмотрите заданные параметры с помощью команды **ifconfig** или **lsattr**:

   ```
   $ ifconfig en0 <UP,BROADCAST,NOTRAILERS,RUNNING,SIMPLEX,MULTICAST,GROUPRT,64BIT>
            en0: flags=e080863
                inet 9.19.161.100 netmask 0xffffff00 broadcast 9.19.161.255
                tcp_sendspace 65536 tcp_recvspace 65536 tcp_nodelay 1 rfc1323 0
   $ lsattr -El en0
            rfc1323         0           н/д              Да
            tcp_nodelay     1           н/д              Да
            tcp_sendspace   65536       н/д              Да
            tcp_recvspace   65536       н/д              Да
            tcp_mssdflt                 н/д              Да
   ```

Адресация TCP/IP

Схема IP-адресации, применяемая в **TCP/IP**, позволяет пользователям и приложениям однозначно идентифицировать сети и хосты, с которыми устанавливаются соединения.

IP-адрес работает так же, как и почтовый адрес, позволяя направлять данные в выбранный пункт назначения. Протокол **TCP/IP** описывает стандарты для присвоения адресов сетям, подсетям, хостам, сокетам, а также для применения специальных адресов оповещения и локальных циклических адресов.

IP-адрес состоит из адреса сети и адреса хоста (или локального адреса). Такой адрес, состоящий из двух частей, позволяет отправителю задавать как сеть, так и конкретный хост в этой сети. Каждой сети присваивается уникальный адрес при подсоединении ее к другим сетям Internet. Однако, если вы не планируете подключать локальную сеть к другим сетям Internet, ей можно присвоить любой сетевой адрес.

Множество адресов Internet состоит из обычных IP-адресов и двух специальных классов адресов: адресов оповещения и циклических адресов.

Адреса Internet

В протоколе Internet (IP) используются адреса длиной 32 разряда, состоящие из двух частей.

32 разряда поделены на четыре *октета*, как показано ниже:

```
01111101   00001101   01001001   00001111
```

Значения октетов обычно записываются в десятичной нотации:

```
125              13              73              15
```

IP-адреса состоят из двух частей - адреса сети и адреса хоста. Такая структура IP-адреса позволяет удаленному хосту при отправке информации задавать как удаленную сеть, так и хост в этой удаленной сети. Хост с нулевым адресом (0) применяется для ссылки на саму сеть.

В TCP/IP предусмотрено три класса IP-адресов: A, B и C. От класса зависит длина адреса сети (и адреса хоста) в IP-адресе. В зависимости от размера сети, она может быть отнесена к тому или иному классу.

Адреса класса A:

Адрес класса A состоит из 8-разрядного адреса сети и 24-разрядного локального адреса или адреса хоста.

Первый бит в сетевом адресе предназначен для указания класса сети, остальные 7 битов - реальный адрес сети. Поскольку максимальное число, которое можно представить в двоичном виде семью битами, равно 128, то класс A содержит 128 адресов сети. Два адреса из этих 128 возможных адресов зарезервированы для специальных случаев: сетевой адрес 127 зарезервирован для локальных циклических адресов, а сетевой адрес, состоящий из одних единиц, означает адрес оповещения.

Таким образом, существует 126 возможных адресов сети класса A и 16 777 214 адресов локальных хостов. В адресе класса A старший разряд равен 0.

На иллюстрации показана типичная структура адреса класса A. Первые 8 бит содержат адрес сети

Примечание: Старший (первый) разряд адреса класса A всегда сброшен.

Рисунок 15. Адрес класса A

(начинающийся с нуля). Оставшиеся 24 бита содержат адрес локального хоста.

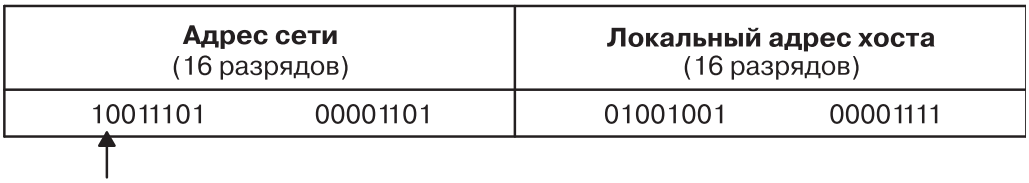
Значения первого октета адресов класса A лежат в диапазоне от 1 до 126.

Адреса класса B:

Адрес класса B состоит из 16-разрядного адреса сети и 16-разрядного адреса хоста.

Первые два бита в адресе сети предназначены для указания класса сети, остальные 14 битов - реальный адрес сети. Существует 16 384 возможных адресов сети и 65 536 адресов локальных хостов. В адресе класса B два старших разряда равны 10.

На иллюстрации показана типичная структура адреса класса B. Первые 16 бит содержат адрес сети. Два



Примечание: Старшие (первые) два разряда адреса класса B всегда равны 1 и 0.

Рисунок 16. Адрес класса B

старших бита всегда равны 10. Оставшиеся 16 бит содержат адрес локального хоста.

Значения первого октета адресов класса B лежат в диапазоне от 128 до 191.

Адреса класса C:

Адрес класса C состоит из 24-разрядного адреса сети и 8-разрядного адреса локального хоста.

Первые три бита в адресе сети задают класс сети, остальные 21 бит - реальный адрес сети. Таким образом, существует 2 097 152 возможных адреса сети и 256 адресов локальных хостов. В адресе класса C два старших разряда равны 1-1-0.

На этом рисунке типичная структура адреса класса C. Первые 24 бита содержат адрес сети (три старших

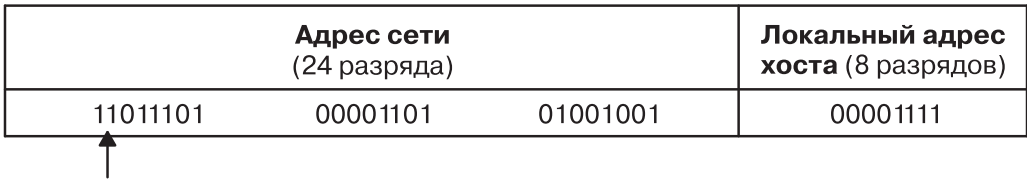


Рисунок 17. Адрес класса C

бита всегда равны 1-1-0). Оставшиеся 8 бит содержат адрес локального хоста.

Другими словами, значения первого октета адресов класса C лежат в диапазонах от 192 до 223.

При выборе класса сети необходимо оценить предполагаемое общее количество локальных хостов в сети и число подсетей в вашей организации. Если организация небольшая, и число хостов не превысит 256, то вполне достаточно адреса класса C. В более крупной организации потребуется адрес класса B или A.

Примечание: Адреса класса D (со значениями 1-1-1-0 в старших битах) зарезервированы для групповых адресов. В этой операционной системе они поддерживаются протоколами UDP и IP.

Компьютеры воспринимают адреса в двоичном коде. Однако обычно IP-адреса указываются в *десятичном формате*, разделяющем 32-разрядный адрес на четыре 8-разрядных поля. Следующее двоичное число:

```
0001010   00000010   00000000   00110100
```

может быть записано в виде

`010.002.000.052` или `10.2.0.52`

В последнем случае значение каждого поля записано в десятичном виде, а поля разделены точками.

Примечание: Команда **hostent** распознает адреса вида: .08 и .008 или .09 и .009. Адреса, начинающиеся с нуля, интерпретируются как восьмеричные, а восьмеричное число не может содержать цифру 8 или 9.

В сети TCP/IP всем сетевым интерфейсам (адаптерам) должны быть присвоены уникальные IP-адреса. Эти адреса задаются в базе данных конфигурации и должны совпадать с адресами, указанными в файле `/etc/hosts` или базе данных **named**, если в сети применяется сервер имен.

IP-адреса, равные нулю:

Если адрес хоста в IP-адресе класса C равен нулю (например, 192.9.200.0), то TCP/IP при отправке такого сообщения использует адрес подстановки.

На запрос должны ответить все компьютеры с адресами класса C, равными 192.9.200.*X* (где *X* - значение от 0 до 254). Это приводит к тому, что сеть наводняется запросами, обращенными к несуществующим хостам.

Аналогичная ситуация возникает при обращении к хосту с адресом 0 в сетях класса B. Например, на запрос, направленный по адресу 129.5.0.0, должны ответить все хосты с адресом класса B вида 129.5.*X*.*X* (где *X* - значение от 0 до 254). Поскольку адреса класса B используются в сетях большего размера, чем адреса класса C, в этом случае будет отправлено гораздо больше запросов к несуществующим хостам.

Адреса подсети

TCP/IP позволяет объединить несколько физических сетей в единую большую логическую сеть. В таком случае физические сети, составляющие большую сеть, называются ее подсетями. Пространства адресов подсетей могут быть организованы произвольным образом, независимо друг от друга и от пространства адресов Internet. Это позволяет при необходимости обойтись одним зарегистрированным IP-адресом для организации доступа к Internet для всех хостов сколь угодно большой внутренней сети.

Способность протокола TCP/IP работать с подсетями также делает возможным разделение одной сети на несколько логических сетей (подсетей). Например, организация, имеющая один IP-адрес, известный внешним пользователям, может создать внутри своей сети несколько подсетей для разных отделов. В таком случае требуется меньшее количество IP-адресов при увеличении потенциальных возможностей локальной маршрутизации.

Стандартный IP-адрес состоит из двух частей: адреса сети и адреса хоста. Для того чтобы иметь возможность работы с подсетью, часть IP-адреса, содержащая локальный адрес, в свою очередь, делится на две части: номер подсети и номер хоста. Подсеть идентифицируется таким образом, чтобы система могла правильно направлять сообщения.

В простых адресах класса A, состоящих из 8-разрядного адреса сети и 24-разрядного локального адреса, локальный адрес идентифицирует конкретный хост в сети.

На иллюстрации показана типичная структура адреса класса A. Первые 8 бит содержат адрес сети

Адрес сети (8 разрядов)	**Локальный адрес хоста** (24 разряда)		
01111101	00001101	01001001	00001111

Рисунок 18. Адрес класса A

(начинающийся с нуля). Оставшиеся 24 бита содержат адрес локального хоста.

Для того чтобы разделить сеть класса A на несколько подсетей, нужно выделить часть разрядов адреса хоста под адрес подсети. Отправители посылают сообщения по данному сетевому адресу, а рассылку по подсетям и по хостам в этих подсетях выполняет сама система. Для того чтобы решить, каким образом выполнить разбиение локального адреса на две части, соответствующие адресу подсети и адресу хоста, необходимо определить число подсетей и количество хостов в этих подсетях.

На следующем рисунке локальный адрес разделен на 12-разрядный адрес подсети и 12-разрядный адрес хоста.

На иллюстрации показана типичная структура адреса класса A. Первые 8 бит содержат адрес сети

Адрес сети (8 разрядов)	**Локальный адрес хоста** (24 разряда)			
Адрес сети	**Адрес подсети**		**Адрес хоста**	
01111101	00001101	0100	1001	00001111

Примечание: Старший (первый) разряд адреса класса A всегда сброшен.

Рисунок 19. Адрес класса A с соответствующим адресом подсети

(начинающийся с нуля). Оставшиеся 24 бита содержат адрес локального хоста, причем первые 8 бит содержат адрес подсети, а последние 8 бит - адрес хоста.

Существует множество способов выбора адресов подсетей и хостов. Биты локального адреса могут подразделяться по-разному, в зависимости от требований и планов расширения организации и структуры ее сети. Единственные ограничения, которые существуют, это:

- `адрес сети` - IP-адрес для сети.
- `адрес подсети` - поле постоянной ширины для данной сети.
- `адрес хоста` - поле размером минимум 1 бит.

Если размер поля `адрес подсети` равен нулю, значит сеть не разделена на подсети, и сеть хоста определяется адресом сети в Internet.

Разряды, относящиеся к адресу подсети, задаются с помощью маски, и поэтому они не обязательно должны быть соседними в адресе. Желательно, однако, чтобы биты подсети были смежными и старшими битами локального адреса.

Маски подсетей:

Когда хост отправляет сообщение в пункт назначения, система должна определить, находится ли получатель в той же сети, что отправитель, и можно ли напрямую связаться с получателем через один из локальных интерфейсов. Система сравнивает адрес пункта назначения с адресом хоста с помощью *маски подсети*.

Если пункт назначения не является локальным, система отправляет сообщение на шлюз. Шлюз выполняет такое же сравнение и определяет, находится ли пункт назначения в локальной сети.

Маска подсети указывает, какие разряды IP-адреса относятся к адресу хоста. Эта битовая маска выделяет из IP-адреса адрес сети и адрес подсети.

На иллюстрации показана типичная структура адреса класса A. Первые 8 бит содержат адрес сети

Адрес сети (8 разрядов)	**Локальный адрес хоста** (24 разряда)	
Адрес сети	**Адрес подсети**	**Адрес хоста**
01111101	00001101 0100	1001 00001111

Адрес класса A с соответствующим адресом подсети

Адрес сети (8 рязрядов)	**Локальный адрес хоста** (24 разряда)	
Адрес сети	Адрес подсети	Адрес хоста
Маска подсети		**Адрес хоста**
01111101	00001101 0100	1001 00001111

Адрес класса A с соответствующей маской подсети

Рисунок 20. Адрес класса A с соответствующим адресом подсети

(начинающийся с нуля). Оставшиеся 24 бита содержат адрес локального хоста, причем первые 8 бит содержат адрес подсети, а последние 8 бит - адрес хоста.

На этом рисунке приведен пример маски подсети класса A со указанной выше схемой разбиения.

Маска подсети, так же, как IP-адрес, состоит из 4 байт. Маска подсети имеет единицу в битах, позиции которых соответствуют позициям битов в адресе сети и подсети, и ноль в битах, позиции которых соответствуют адресу хоста. Маска подсети для этого адреса показана на следующем рисунке.

На иллюстрации показана примерная структура маски подсети. Первые 8 бит содержат адрес сети.

Адрес сети (8 разрядов)	**Локальный адрес хоста** (24 разряда)	
Адрес сети	**Адрес подсети**	**Адрес хоста**
11111111	11111111 1111	0000 00000000

Рисунок 21. Пример маски подсети

Оставшиеся 24 бита содержат адрес локального хоста, причем первые 8 бит содержат адрес подсети, а последние 8 бит - адрес хоста.

Сравнение адресов:

Сравнение адресов пункта назначения и локальной сети выполняется с помощью операции логического умножения или исключающего ИЛИ с маской подсети исходного хоста.

Процедура сравнения описана ниже:

1. Выполняется поразрядная конъюнкция адреса получателя и маски локальной подсети.
2. Выполняется операция исключающего ИЛИ между результатом предыдущей операции и адресом локальной сети. Если все разряды результата будут равны нулю, то предполагается, что с получателем можно напрямую связаться через один из локальных интерфейсов.
3. Если в автономной системе установлено несколько сетевых интерфейсов (ей выделено несколько IP-адресов), то процедура сравнения выполняется для всех локальных интерфейсов.

Предположим, что на некотором хосте установлено два сетевых интерфейса. Их IP-адреса, а также двоичные представления этих адресов приведены в следующем примере:

```
CLASS A   73.1.5.2    =  01001001 00000001 00000101 00000010


CLASS B   145.21.6.3  =  10010001 00010101 00000110 00000011
```

Предположим, что в сетях, к которым подключены эти интерфейсы, применяются следующие маски подсетей:

```
CLASS A   73.1.5.2    =  11111111 11111111 11100000 00000000


CLASS B   145.21.6.3  =  11111111 11111111 11111111 11000000
```

Если из исходной сети T125 необходимо отправить хосту 114.16.23.8 (или, в двоичном виде, 01110010 00010000 00010111 00001000), то сначала система проверяет, можно ли установить соединение с этим хостом напрямую через локальный интерфейс.

Примечание: Для поддержки подсетей в базу данных конфигурации каждого хоста необходимо включить ключевое слово **subnetmask**. Во всех хостах сети должна быть настроена поддержка режима работы с подсетями. Задайте постоянно действующую маску подсети в базе данных конфигурации с помощью меню Выбор сетевого интерфейса в SMIT. Маску подсети также можно задать и в процессе работы системы с помощью команды **ifconfig**. Команда **ifconfig** позволяет задать лишь временную маску подсети.

Адреса для оповещения
TCP/IP позволяет рассылать информацию одновременно всем хостам локальной сети или всем хостам в сетях, доступных напрямую из данного хоста. Такая рассылка называется *оповещением*.

Например, демон маршрутизации **routed** использует оповещающие сообщения для рассылки запросов и получения ответов на них.

Если необходимо разослать данные всем хостам во всех сетях, доступных напрямую, то данные передаются по протоколам UDP и IP, и при этом во всех разрядах адреса хоста-получателя указываются единицы. Если данные рассылаются на все хосты в какой-либо одной сети, то во всех разрядах адреса хоста-получателя указываются нули. В AIX нет пользовательских команд, рассылающих оповещения, но при необходимости такие команды можно создать.

Адрес оповещения можно временно изменять с помощью параметра *broadcast* команды **ifconfig**. Зафиксировать изменение адреса оповещения можно с помощью команды SMIT `smit chinet`. Возможность изменения адреса оповещения может быть полезна в тех случаях, когда необходима совместимость с более ранними версиями программного обеспечения, в которых использовался другой адрес; например, когда для всех ИД хостов задано значение 0.

Локальные циклические адреса
В протоколе IP адрес 127.0.0.1 зарезервирован как локальный циклический адрес.

Хосты используют локальные циклические адреса для отправки сообщений самим себе. Локальный циклический адрес задается диспетчером настройки во время запуска системы. Локальный контур реализован в ядре, но также может устанавливаться с помощью команды **ifconfig**. Локальный контур активизируется при запуске системы.

Преобразование имен TCP/IP
Несмотря на то, что 32-разрядные IP-адреса позволяют однозначно идентифицировать все хосты в сети Internet, пользователям гораздо удобнее работать с осмысленными, легко запоминающимися именами хостов. В **Протоколе управления передачей/Протоколе Internet (TCP/IP)** предусмотрена система имен, поддерживающая как одноуровневую, так и иерархическую структуру сети.

Схема присвоения имен в одноуровневой сети проста. Имена хостов состоят из простого набора символов, а удаленное управление хостами в этом случае, как правило, не используется. В каждой системе одноуровневой сети **TCP/IP** есть файл `/etc/hosts`, содержащий таблицу преобразования имен всех имеющихся хостов в IP-адреса. В больших сетях **TCP/IP** для поддержания в каждой системе файла с таблицей преобразования имен администратору требуется прикладывать очень много усилий. Если сеть **TCP/IP** становится очень большой (например, Internet), то необходимо переходить к иерархической структуре присвоения имен. Обычно иерархия имен соответствует структуре сети. В **TCP/IP** иерархическая структура имен называется *системой имен доменов* (DNS). Для ее поддержания применяется протокол DOMAIN. Протокол DOMAIN реализован в **TCP/IP** демоном **named**.

Как и в случае с одноуровневыми сетями, иерархия имен доменов позволяет присваивать сетям и хостам осмысленные и легко запоминающиеся символьные имена. Однако, вместо того, чтобы хранить файл с таблицей преобразования имен в IP-адреса на каждом хосте, выбирается один или несколько хостов, которые применяются в качестве *серверов имен*. Серверы имен преобразуют символьные имена сетей и хостов в IP-адреса. На сервере имен хранится полная информация об определенной части домена, называемой *областью*. Кроме того, сервер несет *ответственность* за свою область.

Ответственность за присвоение имен
В одноуровневой сети администрирование всех входящих в нее хостов осуществляется централизованно. Такая структура сети требует присвоения всем хостам сети уникальных имен. Если сеть большая, то это требование создает большие трудности для администратора сети.

В доменной сети администрирование выполняется отдельно для каждой группы хостов в соответствии с иерархией доменов и субдоменов. В этом случае имена хостов должны быть уникальными только в пределах локального домена, а централизованное администрирование осуществляется только для *корневого домена*. Такая структура допускает локальное администрирование субдоменов и сокращает нагрузку на организацию, осуществляющую централизованное управление. Например, корневой домен сети Internet состоит из доменов `com` (коммерческие организации), `edu` (образовательные учреждения), `gov` (правительственные организации) и `mil` (военные учреждения). Новые домены верхнего уровня могут добавляться только централизованно. Присваивать имена на втором уровне разрешено уполномоченным представителям внутри соответствующих доменов. Например, уполномоченная организация на уровне домена `com` может присваивать имена всем входящим в него субдоменам коммерческих организаций. Аналогично, присвоение имен на третьем и последующих уровнях разрешено соответствующим организациям. Например, на рисунке "Структура имен доменов в Internet" домен Century уполномочен распределять имена в субдоменах Austin, Hopkins и Charlotte.

На рисунке показана иерархическая структура сети Internet. В вершине дерева находится корневой домен,

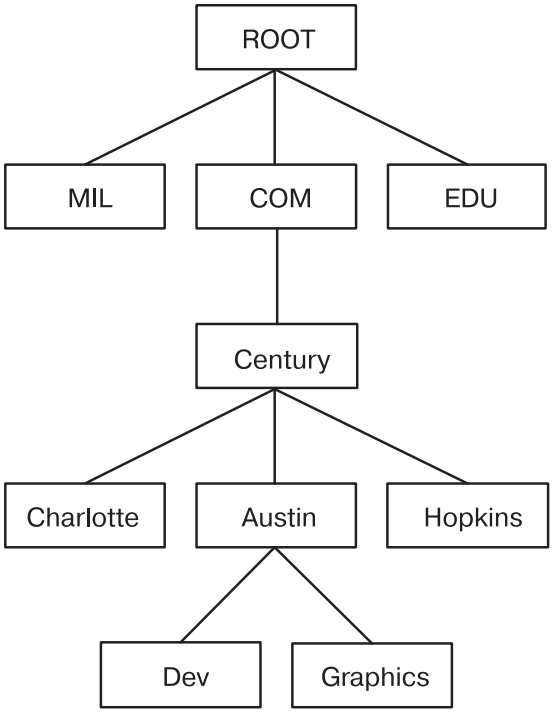


Рисунок 22. Структура имен доменов в Internet

который разделяется на домены mil, com и edu. На один уровень ниже домена com находятся домены Charlotte, Austin и Hopkins. Домен Austin делится на субдомены Dev и Graphics.

Субдомен Austin домена Century может быть, в свою очередь, разделен на две области, например, Dev и Graphics. В этом случае область `austin.century.com` должна включать все данные, относящиеся к домену `austin.century.com`, за исключением тех, которые относятся к областям Dev и Graphics. Область `dev.century.com` должна содержать только данные, относящиеся к Dev; она никак не связана, например, с областью Graphics. Область `austin.century.com` (в отличие от домена с тем же именем) должна содержать только те данные, которые не относятся к другим областям.

Соглашения об именах

В иерархической системе имен доменов каждое имя представляет собой последовательность имен (без учета регистра символов), разделенных точками без промежуточных пробелов.

В протоколе DOMAIN длина имени локального домена ограничена 64 символами, а длина имени хоста - 32 символами. После имени хоста ставится точка (.), а затем указываются имена всех вложенных доменов (также через точку) вплоть до корневого домена. Полное имя хоста, включая точки, должно содержать не более 255 символов и иметь следующий вид:

```
хост.субдомен1.[субдомен2 . . . субдомен].корневой_домен
```

Так как в пределах домена хосты имеют уникальные имена, то при отправке сообщений на хост, находящийся внутри того же домена, можно использовать сокращенное имя. Например, при отправке сообщения с хоста, находящегося в домене `eng`, вместо имени `smith.eng.lsu.edu` можно указать имя `smith`. Кроме того, у каждого хоста может быть несколько псевдонимов, и другие хосты могут использовать их при отправке сообщений.

Присвоение имен хостам в сети

Основная цель присвоения имен хостам - это обеспечение возможности быстро, легко и безошибочно обращаться к компьютерам в вашей сети. Системные администраторы Internet обнаружили, что есть как хорошие, так и плохие способы выбора имен хостов. Эти рекомендации помогут вам избежать ловушек при выборе имен хостов.

Вот несколько рекомендаций по выбору простых и легких для запоминания имен:

- Выбирайте редко используемые слова, например, `sphinx` или `eclipse`.
- Используйте тематические наборы имен, например названия химических элементов (такие как `helium`, `argon` или `zinc`), цветов, рыб и т.п.
- Используйте реально существующие слова, а не случайные наборы символов.

Ниже приведено несколько примеров неудачных имен. В общем случае они или трудны для запоминания, или сбивают с толку (и человека, и компьютер):

- Широко используемые термины, например, `up`, `down` или `crash`.
- Имена, состоящие только из чисел.
- Имена, содержащие знаки препинания.
- Имена, различающиеся регистром символов, например, `Orange` и `orange`.
- Имя или инициалы основного пользователя системы.
- Имена длиной более 8 символов.
- Имена с необычным или намеренно неправильным написанием. Например `czek` легко можно перепутать с "check" или "czech."
- Имена, совпадающие с именем домена, например `yale.edu`.

Серверы имен

В одноуровневой сети без сервера имен имена всех хостов хранятся в файле `/etc/hosts` каждого хоста сети. В больших сетях хранение и обновление таких файлов требует слишком большого объема ресурсов. В иерархической сети задача преобразования имен всех хостов в IP-адреса возложена на определенные хосты, называемые *серверами имен*.

Такая структура имеет ряд преимуществ. На преобразование имен не затрачиваются ресурсы хостов сети, а администратор освобождается от обязанности создавать и обновлять в каждой системе файлы с таблицами преобразования имен. Набор имен, которым управляет какой-либо сервер, называется его *областью ответственности*.

Примечание: Хост, выполняющий преобразование имен в области ответственности, обычно называют *сервером имен*, но на самом деле преобразование осуществляется процессом сервера с именем **named**.

Для снижения нагрузки на сеть все серверы имен в течение определенного времени хранят адреса, полученные с других серверов, в *кэше*. Когда клиент обращается к серверу с запросом на преобразование

имени, сервер сначала просматривает свою кэш-память и определяет, не преобразовывал ли он уже это имя. Так как имена доменов и хостов могут изменяться, то каждый элемент остается в кэш-памяти в течение ограниченного времени, которое определяется параметром времени хранения в кэше (TTL). Таким образом, в пределах области ответственности можно задать период времени, на протяжении которого выполняется точное преобразование имен.

В автономных системах может применяться несколько серверов имен. Обычно серверы имен образуют иерархию, совпадающую со структурой сети. Например, в каждом домене, указанном на рисунке "Структура имен доменов в Internet", может быть свой сервер имен, ответственный за все субдомены. Сервер имен каждого субдомена обменивается информацией с сервером имен домена более высокого уровня (с *родительским* сервером имен), а также с серверами имен других субдоменов.

На рисунке показана иерархическая структура сети Internet. В вершине дерева находится корневой домен,

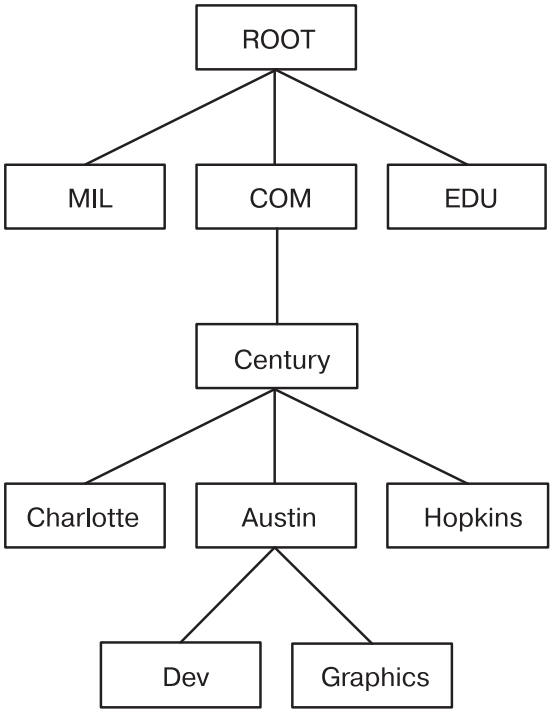


Рисунок 23. Структура имен доменов в Internet

который разделяется на домены mil, com и edu. На один уровень ниже домена com находятся домены Charlotte, Austin и Hopkins. Домен Austin делится на субдомены Dev и Graphics.

Например, на рисунке "Структура имен доменов в Internet" субдомены Austin, Hopkins и Charlotte входят в состав домена Century. Если следовать иерархии организации сети, то сервер имен Austin должен взаимодействовать с серверами имен Charlotte и Hopkins, а также с родительским сервером имен Century. Кроме того, сервер имен Austin будет взаимодействовать с серверами имен, ответственными за его субдомены.

Существует несколько типов серверов имен:

Элемент	Описание
Главный сервер имен	Загружает свои данные из файла или с диска и может передавать полномочия другим серверам в своем домене.
Подчиненный сервер имен	Во время загрузки системы получает с главного сервера информацию о выделенной ему области ответственности, а затем периодически обращается к главному серверу для обновления этой информации. По истечении периода обновления, указанного в записи ресурса начала области ответственности (SOA) на подчиненном сервере имен, а также при получении уведомляющего сообщения от главного сервера, подчиненный сервер заново загружает базу данных с главного сервера, если порядковый номер базы данных на главном сервере больше, чем ее порядковый номер на подчиненном сервере. Если необходимо принудительно передать с главного сервера новую информацию об области ответственности, достаточно просто удалить существующие базы данных с подчиненного сервера и обновить на нем программу-демон **named**.
Ограниченный сервер имен	Данный сервер имен применяет такой же способ копирования базы данных, что и подчиненный сервер, однако он копирует из базы данных главного сервера только записи о серверах имен.
Сервер подсказок	Это сервер имен, который отвечает на запросы только на основании данных, полученных им в результате предыдущих обращений к другим сервером имен. Если в кэш-памяти сервера нет записей преобразования имен в адреса, то для ответа на запросы он обращается к другим серверам.
Клиент или сервер пересылки	Этот сервер пересылает перечисленным серверам те запросы, которые он не может обработать самостоятельно. Серверы, предназначенные только для пересылки, занимаются только получением информации и ее передачей другим клиентам. Они не взаимодействуют с главными серверами имен корневого и других доменов. Для обращения к серверам пересылки используются рекурсивные запросы. Может быть определено несколько серверов пересылки, к которым клиент обращается по очереди до тех пор, пока не исчерпается список. Серверы пересылки обычно применяются в том случае, когда вы не хотите, чтобы все серверы данной сети взаимодействовали с остальными серверами Internet, или когда необходимо создать на серверах имен кэш-память большого объема.
Удаленный сервер	Выполняет все сетевые программы, использующие сервер имен; при этом процесс сервера имен на локальном хосте не запускается. Все запросы обслуживаются сервером имен, работающим в другой системе.

Один и тот же хост может выполнять функции нескольких серверов имен с различными областями ответственности. Например, он может быть главным сервером имен для одной области и подчиненным сервером имен - для другой.

Преобразование имен

Процесс определения IP-адреса по имени хоста называется преобразованием имен и выполняется процедурой **gethostbyname**.

Процесс определения имени хоста по его IP-адресу называется обратным преобразованием имен и выполняется функцией **gethostbyaddr**. Эти процедуры входят в состав библиотеки, содержащей все необходимые *функции преобразования*.

Для определения IP-адресов хостов в сети используются следующие источники:

1. Сервер BIND/DNS (named)
2. Служба информации о сети (NIS)
3. Локальный файл `/etc/hosts`

Для преобразования имен в сети с иерархической структурой функции преобразования сначала обращаются к базе данных сервера имен доменов, которая может быть локальной, если хост сам является сервером имен доменов, или может находиться на удаленном хосте. Серверы имен преобразуют имена доменов в IP-адреса. Группа имен, за которые отвечает сервер имен, называется его областью ответственности. Если функция преобразования применяет удаленный сервер имен, то для отправки запроса на преобразование применяется

протокол имен доменов (DOMAIN). Для преобразования имени в одноуровневой сети функция преобразования просматривает записи в локальном файле /etc/hosts. Если применяется служба NIS, то просматривается файл /etc/hosts главного сервера.

По умолчанию функции преобразования используют средства, перечисленные выше. Сначала функция обращается к серверу BIND/DNS. Если файл /etc/resolv.conf отсутствует, либо серверу BIND/DNS не удалось найти требуемое имя, то функция преобразования обращается к службе NIS (если она активна). Так как служба NIS имеет более высокий приоритет, чем локальный файл /etc/hosts, то на этом поиск завершается. Если служба NIS не запущена, то просматривается локальный файл /etc/hosts. Если имя не удалось найти ни одним из этих способов, то функция преобразования завершает работу с кодом возврата HOST_NOT_FOUND. Если ни одна из служб не доступна, функция преобразования завершает работу с кодом возврата SERVICE_UNAVAILABLE.

Описанный выше порядок действий по умолчанию можно изменить, создав файл /etc/irs.conf и задав в нем требуемую последовательность выполнения процедур. Кроме того, порядок действий по умолчанию и порядок, определенный в файле /etc/irs.conf, можно изменить с помощью переменной среды **NSORDER**. Если создан файл /etc/irs.conf или определена переменная среды **NSORDER**, то необходимо задать хотя бы одно значение параметра.

Задание порядка поиска хостов с помощью файла /etc/irs.conf:

```
hosts значение [ continue ]
```

Порядок поиска задается путем перечисления возможных способов, по одному способу на строке. *Значение* - это один из способов, приведенных в списке, а ключевое слово **continue** указывает, что другой способ преобразования задан в следующей строке.

Задание порядка поиска хостов с помощью переменной среды **NSORDER**:

```
NSORDER=значение,значение,значение
```

Порядок задается в одной строке, значения разделяются запятыми. Между запятыми и знаком равенства можно ставить пробел.

Например, в одноуровневой локальной сети необходим только файл /etc/hosts. В этом случае файл /etc/irs.conf будет содержать следующую строку:

```
hosts local
```

В качестве альтернативы можно задать переменную среды **NSORDER**:

```
NSORDER=local
```

Если локальная сеть состоит из доменов, причем в ней используется сервер имен, а в файле /etc/hosts хранится резервная таблица хостов, то необходимо указать обе службы. В этом случае файл /etc/irs.conf будет содержать следующие строки:

```
hosts dns continue
hosts local
```

Переменная среды **NSORDER** должна быть задана следующим образом:

```
NSORDER=bind,local
```

Примечание: Значения следует вводить в нижнем регистре.

При любой последовательности преобразования переход к следующему способу преобразования выполняется при следующих условиях:

- Текущая служба не запущена, и поэтому недоступна.
- Текущая служба не смогла найти имя и не помечена как "ответственная".

Если файл /etc/resolv.conf не существует, службы BIND/DNS считаются недоступными. Если при запуске функций **getdomainname** и **yp_bind** возникла ошибка, то считается, что служба NIS не настроена или не запущена, и поэтому недоступна. Если не удается открыть файл /etc/hosts , то считается, что служба локального поиска недоступна.

Если служба помечена как *ответственная*, значит достоверность передаваемой ею информации выше, чем у служб, расположенных в списке ниже нее, и она содержит все имена и адреса. Функция преобразования не будет обращаться к последующим службам, так как они содержат только часть информации, предоставляемой ответственной службой. Функция преобразования завершает работу на уровне службы, помеченной как "ответственная", даже если она не нашла запрошенное имя (в этом случае процедура преобразования выдаст сообщение HOST_NOT_FOUND). Если ответственная служба недоступна, то функция обращается к следующей службе.

Ответственная служба задается с помощью ключевого слова =auth, которое указывается после имени службы. Может быть полностью указано слово authoritative, однако используется только часть auth. Например, предположим, что переменной среды **NSORDER** присвоено следующее значение:

```
hosts = nis=auth,dns,local
```

В этом случае при получении ответа от NIS поиск будет завершен, даже если имя не будет найдено. Если служба NIS не работает, то процедура преобразования обратится к DNS.

Для более эффективного поиска имен хостов и сетей серверы имен **TCP/IP** используют кэш-память. Вместо того, чтобы выполнять поиск имени хоста при получении каждого запроса, сервер имен сначала просматривает свою кэш-память и определяет, не преобразовывал ли он уже это имя хоста. Так как имена доменов и хостов могут изменяться, каждый элемент остается в кэш-памяти в течение ограниченного времени, которое определяется параметром времени хранения в кэше (TTL). Таким образом, для сервера имен можно задать период времени, на протяжении которого его ответы могут считаться достоверными.

Потенциальные конфликты имен хоста между серверов имен и sendmail:

В среде DNS имя хоста, которое задается с помощью команды **hostname** в командной строке или в файле rc.net, должно быть официальным именем хоста, известным серверу имен.

В общем случае это полное имя хоста в следующем формате:

```
хост.субдомен.субдомен.корневой-домен
```

Примечание: Для работы функций преобразования необходимо задать имя домена по умолчанию. Если домен по умолчанию не указан в команде **hostname**, он должен быть задан в файле /etc/resolv.conf.

Если имя хоста задано неполностью, а сервер имен доменов применяется в сочетании с программой **sendmail**, то официальное имя хоста необходимо указать в файле конфигурации **sendmail** (/etc/sendmail.cf). Кроме того, для правильной работы программы **sendmail** в этом файле конфигурации нужно задать макроопределение имени домена.

Примечание: Для всех функций программы /etc/sendmail.cf домен, описанный в файле **hostname**, имеет более высокий приоритет, чем домен, заданный командой **sendmail**.

Потенциальные конфликты имен домена между серверов имен и sendmail:

Имена локальных доменов и серверов доменов имен указываются в разных файлах, в зависимости от того, является ли хост сервером имен ДОМЕНА.

Для хоста, не являющего сервером имен, имена локального домена и сервера имен задаются в файле /etc/resolv.conf. Для сервера имен домена локальный домен и другие серверы имен задаются в файлах, которые демон **named** считывает при запуске.

Обратный протокол преобразования адресов

Обратный протокол обратного преобразования адресов (RARP) преобразует уникальные аппаратные адреса адаптеров сети Ethernet в IP-адреса.

Стандартный протокол Ethernet поддерживается со следующими ограничениями:

- Сервер отвечает на запросы **RARP**.
- Сервер применяет только записи постоянной таблицы **ARP**.
- Сервер не применяет записи динамической таблицы **ARP**.
- Сервер не отвечает на собственные запросы автоматически.

Системный администратор должен вручную создать и обновлять таблицу постоянных записей с помощью команды **arp**. В таблицу **ARP** сервера необходимо добавить записи для всех хостов, которые будут обращаться с запросами **RARP** к ответственному источнику.

Задачи преобразования локальных имен (/etc/hosts)

В небольшой одноуровневой сети достаточно настроить файл `/etc/hosts`.

В иерархической сети, применяющей серверы имен, файл `/etc/hosts` может применяться для хранения записей о тех хостах, которые неизвестны серверам имен.

Для настройки локального преобразования имен в системе воспользуйтесь программой SMIT или специальными командами. В последнем случае постарайтесь сохранить формат файла `/etc/hosts`, описанный в разделе Hosts File Format for TCP/IP книги$ *Справочник по файлам* .

Таблица 61. Задачи преобразования локальных имен

Процедура	Команды быстрого доступа SMIT	Команда или файл
Показать список всех хостов	**smit lshostent**	Воспользуйтесь командой **hostent** или **view** `/etc/hosts`
Добавить хост	**smit mkhostent**	Воспользуйтесь командой **hostent** или **edit** `/etc/hosts`
Изменить/показать параметры хоста	**smit chhostent**	Воспользуйтесь командой **hostent** или **edit** `/etc/hosts`
Удалить хост	**smit rmhostent**	Воспользуйтесь командой **hostent** или **edit** `/etc/hosts`

Планирование для преобразования имен DOMAIN

Ниже приведены рекомендации по планированию системы DOMAIN.

Если ваша сеть представляет собой часть большой сети, то настройку серверов имен и доменов нужно будет согласовывать с организацией, обслуживающей зону ответственности, в которую входит ваша сеть.

- Перед утверждением какого-либо плана ознакомьтесь с принципами работы **TCP/IP**, DNS и BIND. Если вы планируете применять службу информации о сети, изучите материалы о службах NFS и NIS. Существует большое количество книг по этой теме.
- Учитывайте будущие потребности.

  Изменить имя *существенно* сложнее, чем сразу выбрать его правильно. Перед изменением файлов конфигурации обязательно согласуйте выбранные имена сетей, шлюзов, серверов имен и хостов с руководством вашей организации.
- Создайте резервные серверы имен.

  Если это невозможно, обязательно создайте подчиненные серверы имен или серверы подсказок, что обеспечит хотя бы минимальную защиту от сбоев.
- При выборе серверов имен учитывайте следующие особенности:
  - Выбирайте компьютеры, которые физически находятся ближе других к внешним системам.

  - Серверы имен должны быть как можно более автономными. Постарайтесь подключить их к отдельным источникам питания и независимым кабельным системам.
  - Храните резервные копии данных вашей службы преобразования имен в другой сети и не отказывайте в аналогичной услуге администраторам других сетей.
- Проверяйте работу серверов.
  - Проверяйте как прямое, так и обратное преобразование имен.
  - Проверяйте передачу информации об областях ответственности от главного к подчиненным серверам имен.
  - Проверяйте каждый сервер имен после сбоя или перезагрузки системы.
- Прежде чем отправлять запросы на преобразование имен на внешние серверы, направляйте их на серверы пересылки. Это позволит серверам имен совместно использовать кэш-память и повысит эффективность работы за счет снижения нагрузки на главные серверы имен.

```
objectclass container
        requires
                objectclass,
                cn
objectclass hosts
        requires
                objectclass,
                hname
        allows
                addr
                halias,
                comment
```

Преобразование серверов имен

В иерархической сети некоторые хосты выполняют функции *серверов имен*. Эти хосты преобразуют имена других хостов в IP-адреса.

Работой сервера имен управляет программа-демон **named**, которая должна быть запущена на хосте - сервере имен.

Перед настройкой определите, какие типы серверов имен лучше всего подходят для обслуживания вашей сети. Существует несколько типов серверов имен.

Главный сервер имен хранит базу данных с таблицами преобразования имен в адреса. Он загружает свои данные из файла или с диска и может передавать полномочия другим серверам своего домена. *Подчиненный сервер имен* и *ограниченный сервер имен* получают информацию об области ответственности от главного сервера имен во время запуска системы, а затем периодически обращаются к главному серверу для обновления этой информации. *Сервер подсказок* отвечает на запросы о преобразовании имени с помощью информации, полученной им от других серверов.

Примечание: В предыдущих версиях сервера имен **named** главный сервер имен определялся как основной, подчиненный сервер имен - как дополнительный, а сервер подсказок - как сервер кэш-памяти.

Помните, что один сервер имен может выполнять разные функции в различных областях ответственности. Например, он может быть главным сервером имен для одной области и подчиненным - для другой. Если в вашей системе установлена NIS, то эти службы также выполняют преобразование имен.

Параметры серверов имен задаются в нескольких файлах.

Элемент	Описание
conf	Этот файл считывается при запуске демона **named**. В файле conf задается тип сервера **named**, список доменов, входящих в его область ответственности, и расположение данных для первоначального заполнения базы данных. По умолчанию этот файл называется /etc/named.conf. Вы можете изменить это имя, задав новое имя и путь в командной строке при запуске программы **named**. Если файл /etc/named.conf не существует, и альтернативный файл conf не был задан в командной строке, то в протокол syslog заносится сообщение, а работа программы **named** завершается. Однако если вы укажете несуществующий альтернативный файл conf, то сообщение об ошибке отправлено не будет, и программа **named** продолжит работу.
cache	Содержит информацию о локальной кэш-памяти. Файл локальной кэш-памяти содержит имена и адреса серверов с наибольшей областью ответственности. Файл кэш-памяти имеет стандартный формат записи ресурса. Имя этого файла указывается в файле conf.
domain data	Существует три типа файлов данных о домене, называемых также файлами данных **named**. *Локальный* файл **named** содержит информацию о преобразовании локальных циклических адресов. *Файл данных* **named** содержит данные о преобразовании имен для всех систем, находящихся в области ответственности сервера имен. *Файл данных для обратного преобразования* **named** содержит данные об обратном преобразовании адресов систем, находящихся в области ответственности сервера. Файлы данных о домене используют стандартный формат записи ресурса. Имена этих файлов задаются пользователем в файле conf. Общепринято, чтобы имена этих файлов содержали имя демона named, а расширение - тип файла и имя домена. Например, сервер имен для домена abc должен иметь следующие файлы: named.abc.data named.abc.rev named.abc.local При изменении файлов данных **named** следует увеличить порядковый номер в записи о ресурсе SOA, для того чтобы подчиненные серверы имен знали об изменении области ответственности.
resolv.conf	Если этот файл существует, то для преобразования имен сервер сначала должен обращаться к серверу имен. Если файл resolv.conf не существует, то для преобразования имен применяется файл /etc/hosts. На сервере имен должен существовать файл resolv.conf, который может содержать адрес локального хоста, циклический адрес (127.0.0.1) или может быть пустым. **Примечание:** Для работы функций преобразования необходимо задать имя домена по умолчанию. Если домен по умолчанию не указан в файле /etc/resolv.conf, он должен быть задан в файле hostname

Время хранения в кэш-памяти (TTL) задается в записях ресурсов. Если в записи ресурса параметр TTL не указан, то по умолчанию он устанавливается равным минимальному значению, заданному для данной области в записи начала области ответственности (SOA). TTL ограничивает время хранения данных в кэш-памяти.

Настройка серверов имен домена:

В этом разделе описан сценарий настройки главного сервера имен, подчиненного сервера имен и сервера подсказок для выполнения преобразования имен. Каждый сервер имен располагается на отдельном компьютере. Для каждого сервера будет настроен особый файл /etc/named.conf. Файл /etc/named.conf считывается при каждом запуске демона **named**. Он задает тип сервера (главный, вспомогательный или сервер подсказок) и источник данных, применяемых для преобразования имен. Все указанные серверы будут поддерживать стандарт BIND 8.

Главный сервер имен будет отвечать за преобразование имен в области abc.aus.century.com. В данном сценарии главному серверу имен присвоен IP-адрес 192.9.201.1 и имя хоста venus.abc.aus.century.com. Этот сервер будет обеспечивать преобразование имен для хостов venus, earth, mars и jupiter. В файле /etc/named.conf будет указано, что файлы данных демона **named** расположены в каталоге /usr/local/domain. Для главного сервера имен будут настроены файлы данных named.ca, named.abc.local, named.abc.data и named.abc.rev.

Затем будет настроен подчиненный сервер имен. Этому серверу присвоено имя хоста earth.abc.aus.century.com и IP-адрес 192.9.201.5. В файле /etc/named.conf подчиненного сервера имен будет задан адрес главного сервера имен, чтобы подчиненный сервер мог скопировать файлы named.abc.data и named.abc.rev главного сервера. Для этого сервера будут настроены файлы данных named.ca и named.abc.local.

После этого будет настроен сервер подсказок. На этом сервере будет расположен локальный кэш имен хостов и записей о преобразовании адресов. Если запрошенный адрес или имя хоста не будут найдены в этом кэше, сервер подсказок обратится к главному серверу, получит информацию о преобразовании и добавит ее в кэш. Для этого сервера будут настроены файлы данных `named.ca` и `named.abc.local`.

Вся информация в файлах данных демона `named` (в их число не входит файл `/etc/named.conf`) будет задана в стандартном формате записей о ресурсах. Дополнительная информация о файлах данных `named` приведена в разделе Стандартный формат записей ресурсов для TCP/IP в *Справочник по файлам* .

В качестве администратора серверов имен будет назначен `gail.zeus.abc.aus.century.com`. Это будет отражено в локальных файлах данных каждого сервера имен. В качестве корневого сервера имен будет использоваться сервер `relay.century.com` с IP-адресом `129.114.1.2`.

В конце этого сценария будет продемонстрирована процедура преобразования имен хостов venus, earth, mars и jupiter. Кроме того, будет продемонстрирована процедура обратного преобразования (IP-адреса в имя хоста). В тех случаях, когда главный сервер имен не может выполнить запрошенное преобразование, он будет обращаться к серверу `relay.century.com` за недостающей информацией.

Рассмотрите следующие вопросы:

- Описанная ниже процедура была протестирована в отдельных версиях AIX. Результаты, которые вы можете получить, в значительной степени зависят от конкретных версии и уровня AIX.

Шаг 1. Настройка главного сервера имен

1. На главном сервере имен откройте файл `/etc/named.conf`. Если в каталоге `/etc` нет файла `/etc/named.conf`, создайте его с помощью следующей команды:

   ```
   touch /etc/named.conf
   ```

   Для настройки файла `/etc/named.conf` выполните следующие действия:

   a. Добавьте предложение directory в раздел options. Будет считаться, что имена файлов данных демона `named` заданы относительно указанного каталога (в данном случае - `/usr/local/domain`). В этом сценарии нужно добавить следующее:

      ```
      options {
          directory "/usr/local/domain";
      };
      ```

      Если вы не укажете имя каталога, то поиск необходимых файлов данных будет выполняться в каталоге `/etc`.

   b. Для того чтобы записи заносились в кэш, расположенный вне указанных областей, задайте имя файла области подсказок. В этом сценарии нужно добавить следующее:

      ```
      zone "." IN {
          type hint;
          file "named.ca";
      };
      ```

   c. Добавьте раздел для каждой области, указав в нем тип настраиваемого сервера имен и файл данных домена сервера имен. В данном сценарии главный сервер будет отвечать за следующие области прямого и обратного преобразования:

      ```
      zone "abc.aus.century.com" in {
          type master;
          file "named.abc.data";
      };
      zone "201.9.192.in-addr.arpa" in {
          type master;
          file "named.abc.rev";
      };
      ```

   d. Задайте имя локального файла `named`. Например:

```
zone "0.0.127.in-addr.arpa" in {
    type master;
    file "named.abc.local";
};
```

После внесения всех необходимых изменений сохраните и закройте файл.

2. Откройте файл /usr/local/domain/named.ca. Добавьте в него адреса корневых серверов имен домена. В данном сценарии нужно добавить следующее:

```
; root name servers.
.          IN    NS    relay.century.com.
relay.century.com.   3600000    IN    A     129.114.1.2
```

После внесения всех необходимых изменений сохраните и закройте файл.

3. Откройте файл /usr/local/domain/named.abc.local. Добавьте в него следующую информацию:
   - Начало области ответственности (SOA) области и время хранения информации в кэше по умолчанию. В данном сценарии нужно добавить следующее:

```
$TTL 3h    ;3 hour

@ IN    SOA     venus.abc.aus.century.com. gail.zeus.abc.aus.century.com.  (

                           1       ;serial
                           3600    ;refresh
                           600     ;retry
                           3600000 ;expire
                           3600    ;negative caching TTL
)
```

   - Запись сервера имен (NS). Вставьте символ табуляции в начале строки; демон **named** заменит символ табуляции на имя области:

```
<tab> IN    NS     venus.abc.aus.century.com.
```

   - Запись указателя (PTR).

```
1      IN    PTR    localhost.
```

После внесения всех необходимых изменений сохраните и закройте файл.

4. Откройте файл /usr/local/domain/named.abc.data. Добавьте в него следующую информацию:
   - Начало области ответственности области и значение по умолчанию для времени хранения информации в кэше для данной области. Эта запись определяет начало области ответственности. Для каждой области должна быть задана только одна такая запись. В этом сценарии нужно добавить следующее:

```
$TTL 3h    ;3 hour

@ IN    SOA     venus.abc.aus.century.com. gail.zeus.abc.aus.century.com. (
                1       ;serial
                3600    ;refresh
                600     ;retry
                3600000 ;expire
                3600    ;negative caching TTL
)
```

   - Записи серверов имен для всех главных серверов имен области. Вставьте символ табуляции в начале строки; демон **named** заменит символ табуляции на имя области:

```
<tab> IN    NS     venus.abc.aus.century.com.
```

   - Информацию о преобразовании имен хостов в адреса для всех хостов, входящих в область ответственности сервера имен:

```
venus        IN    A        192.9.201.1
earth        IN    A        192.9.201.5
mars         IN    A        192.9.201.3
jupiter      IN    A        192.9.201.7
```

Внесите другие необходимые типы записей, например, записи канонических имен или записи систем обмена почтой.

После внесения всех необходимых изменений сохраните и закройте файл.

5. Откройте файл /usr/local/domain/named.abc.rev. Добавьте в него следующую информацию:
   - Начало области ответственности и время хранения информации в кэше по умолчанию. Эта запись определяет начало области ответственности. Для каждой области должна быть задана только одна такая запись:

     ```
     $TTL 3h    ;3 hour

     @ IN    SOA     venus.abc.aus.century.com. gail.zeus.abc.aus.century.com.  (
                                1       ;serial
                                3600    ;refresh
                                600     ;retry
                                3600000 ;expire
                                3600    ;negative caching TTL
     )
     ```

   - Прочие типы записей, например, записи серверов имен. Внося эти записи, вставляйте символ табуляции в начале каждой строки; демон **named** заменит символ табуляции именем области. В этом сценарии нужно добавить следующее:

     ```
     <tab> IN    NS      venus.abc.aus.century.com.
     ```

   - Информацию о преобразовании адресов в имена для всех хостов, находящихся в области ответственности сервера имен.

     ```
     1                       IN    PTR    venus.abc.aus.century.com.
     5                       IN    PTR    earth.abc.aus.century.com.
     3                       IN    PTR    mars.abc.aus.century.com.
     7                       IN    PTR    jupiter.abc.aus.century.com.
     ```

     После внесения всех необходимых изменений сохраните и закройте файл.

6. Создайте файл /etc/resolv.conf с помощью следующей команды:

   ```
   touch /etc/resolv.conf
   ```

   Наличие этого файла означает, что для преобразования имен хост должен использовать сервер имен.

7. Добавьте следующую запись в файл /etc/resolv.conf:

   ```
   nameserver 127.0.0.1
   ```

   127.0.0.1 - это циклический адрес, по которому хост обращается к самому себе, как к серверу имен. Файл /etc/resolv.conf может также содержать примерно следующую запись:

   ```
   domain abc.aus.century.com
   ```

   В данном случае имя домена равно abc.aus.century.com.

   После внесения всех необходимых изменений сохраните и закройте файл.

8. Запустите демон **named** с помощью команды быстрого доступа SMIT smit stnamed. Эта команда указывает, что демон должен инициализироваться при каждом запуске системы. Укажите, нужно ли запустить программу **named** сейчас, при следующем запуске системы или в обоих случаях.

Шаг 2. Настройка подчиненного сервера имен

Для настройки подчиненного сервера имен выполните описанную ниже процедуру. Вам потребуется отредактировать ряд файлов и запустить демон **named** с помощью SMIT.

1. На подчиненном сервере имен откройте файл /etc/named.conf. Если в каталоге /etc нет файла /etc/named.conf, создайте его с помощью следующей команды:

   ```
   touch /etc/named.conf
   ```

   Для настройки файла /etc/named.conf выполните следующие действия:

a. Добавьте предложение directory в раздел options. Будет считаться, что имена файлов данных демона named заданы относительно указанного каталога (в данном случае - /usr/local/domain). В этом сценарии нужно добавить следующее:

```
options {
    directory "/usr/local/domain";
};
```

Если вы не укажете имя каталога, демон **named** будет выполнять поиск необходимых файлов данных в каталоге /etc.

b. Для того чтобы записи заносились в кэш, расположенный вне указанных областей, задайте имя файла области подсказок для сервера имен:

```
zone "." IN {
    type hint;
    file "named.ca";
};
```

c. Задайте области ответственности подчиненного сервера. В каждом разделе нужно задать тип области, имя файла для хранения резервной копии данных сервера имен и IP-адрес главного сервера имен, файлы данных которого должен скопировать подчиненный сервер имен. В данном сценарии нужно добавить следующие разделы областей подчиненного сервера:

```
zone "abc.aus.century.com" IN {
    type slave;
    file "named.abc.data.bak";
    masters { 192.9.201.1; };
};
zone "201.9.192.in-addr.arpa" IN {
    type slave;
    file "named.abc.rev.bak";
    masters { 192.9.201.1; };
};
```

d. Для поддержки преобразования циклического сетевого адреса определите область типа *master* с источником данных named.abc.local, а также домен, за который отвечает сервер имен.

```
zone "0.0.127.in-addr.arpa" in {
    type master;
    file "named.abc.local";
};
```

После внесения всех необходимых изменений сохраните и закройте файл.

2. Внесите необходимые изменения в файл /usr/local/domain/named.ca.

Этот файл содержит адрес корневого сервера имен доменов в сети. В этом сценарии нужно добавить следующее:

```
; root name servers.
.          IN    NS    relay.century.com.
relay.century.com.   3600000    IN    A     129.114.1.2
```

После внесения всех необходимых изменений сохраните и закройте файл.

3. Откройте файл /usr/local/domain/named.abc.local. В этом сценарии нужно добавить следующее:
   - Начало области ответственности (SOA) области и время хранения информации в кэше по умолчанию:

```
$TTL 3h    ;3 hour

@ IN SOA earth.abc.aus.century.com. gail.zeus.abc.aus.century.com.  (

                            1       ;serial
                            3600    ;refresh
                            600     ;retry
                            3600000 ;expire
                            3600    ;negative caching TTL
)
```

- Запись сервера имен (NS). Вставьте символ табуляции в начале каждой строки; демон **named** заменит символ табуляции именем области. Например:

  ```
  <tab> IN    NS     earth.abc.aus.century.com.
  ```

- Запись указателя (PTR).

  ```
  1      IN    PTR    localhost.
  ```

После внесения всех необходимых изменений сохраните и закройте файл.

4. Создайте файл `/etc/resolv.conf` с помощью следующей команды:

   ```
   touch /etc/resolv.conf
   ```

5. Добавьте в него следующую запись:

   ```
   nameserver 127.0.0.1
   domain abc.aus.century.com
   ```

   После внесения всех необходимых изменений сохраните и закройте файл.

6. Запустите демон **named** с помощью команды быстрого доступа SMIT `smit stnamed`. Эта команда указывает, что демон должен инициализироваться при каждом запуске системы. Укажите, нужно ли запустить программу **named** сейчас, при следующем запуске системы или в обоих случаях.

Шаг 3. Настройка сервера подсказок

Для настройки сервера подсказок, или *кэш-сервера*, выполните описанную ниже процедуру. Вам потребуется отредактировать ряд файлов и запустить демон **named** с помощью SMIT или команды.

1. Отредактируйте файл `/etc/named.conf` на сервере подсказок. Если в каталоге `/etc` нет файла `/etc/named.conf`, создайте его с помощью следующей команды:

   ```
   touch /etc/named.conf
   ```

   Для настройки файла `/etc/named.conf` выполните следующие действия:

   a. Добавьте предложение directory в раздел options. Будет считаться, что имена файлов данных демона `named` заданы относительно указанного каталога (в данном случае - `/usr/local/domain`). В этом сценарии нужно добавить следующее:

      ```
      options {
          directory "/usr/local/domain";
      };
      ```

   b. Для поддержки преобразования циклического сетевого адреса определите область типа *master* с источником данных `named.abc.local`, а также домен, за который отвечает сервер имен. В данном примере, ключевое слово опций directory было указано в файле `/etc/named.conf`.

      ```
      zone "0.0.127.in-addr.arpa" IN {
          type master;
         file "named.abc.local";
      };
      ```

   c. Укажите файл кэша для области подсказок. Например:

      ```
      zone "." IN {
          type hint;
          file "named.ca";
      };
      ```

   После внесения всех необходимых изменений сохраните и закройте файл.

2. Внесите необходимые изменения в файл `/usr/local/domain/named.ca`.

   В этом файле содержатся адреса ответственных серверов корневого домена сети. Например:

   ```
   ; root name servers.
   .          IN    NS    relay.century.com.
   relay.century.com.   3600000    IN    A     129.114.1.2
   ```

   После внесения всех необходимых изменений сохраните и закройте файл.

3. Внесите необходимые изменения в файл /usr/local/domain/named.local. В данном сценарии нужно добавить следующую информацию:
   - Начало области ответственности (SOA) области и время хранения информации в кэше по умолчанию:

     ```
     $TTL 3h    ;3 hour

     @ IN    SOA     venus.abc.aus.century.com. gail.zeus.abc.aus.century.com.  (

                               1       ;serial
                               3600    ;refresh
                               600     ;retry
                               3600000 ;expire
                               3600    ;negative caching TTL
     )
     ```

   - Запись сервера имен (NS). Вставьте символ табуляции в начале строки; демон **named** заменит символ табуляции на имя области:

     ```
     <tab> IN    NS     venus.abc.aus.century.com.
     ```

   - Запись указателя (PTR).

     ```
     1      IN    PTR    localhost.
     ```

   После внесения всех необходимых изменений сохраните и закройте файл.
4. Создайте файл /etc/resolv.conf с помощью следующей команды:

   ```
   touch /etc/resolv.conf
   ```

5. Добавьте в него следующую запись:

   ```
   nameserver 127.0.0.1
   domain abc.aus.century.com
   ```

   После внесения всех необходимых изменений сохраните и закройте файл.
6. Запустите демон **named** с помощью команды быстрого доступа SMIT smit stnamed. Эта команда указывает, что демон должен инициализироваться при каждом запуске системы. Укажите, нужно ли запустить программу **named** сейчас, при следующем запуске системы или в обоих случаях.

При перезагрузке системы будет задана конфигурация IPv6. Повторите эту процедуру для каждого из хостов.

Настройка почтового сервера домена:

Почтовый сервер упрощает передачу сообщений пользователям вашей организации из внешней сети. Если такого сервера нет, то в почтовом адресе должен указываться конкретный хост в сети организации.

Например, sam@orange.widget.com, где widget.com - имя домена вашей организации, а orange - хост пользователя sam. При наличии почтового сервера внешним пользователям достаточно указать только имя адресата (без имени хоста) и имя домена вашей организации, например, sam@widget.com.

Для настройки почтового сервера выполните описанную ниже процедуру.

1. Создайте для почтового сервера black.widget.com запись системы обмена почтой (MX) и запись адреса (A):

   ```
   widget.com           IN    MX     10 black.widget.com
   widget.com           IN     A     192.10.143.9
   black.widget.com     IN     A     192.10.143.9
   ```

2. Добавьте файл sendmail.cf на почтовом сервере (black.widget.com) псевдоним домена (класс **w**):

   ```
   Cw $w $?D$w.$D$. widget.com
   ```

3. Укажите почтовым клиентам, куда они должны направлять сообщения, адресованные во внешние сети. Для этого в файле sendmail.cf каждого клиента задайте ссылку на почтовый сервер (макрокоманда **S** macro):

   ```
   DRblack.widget.com
   ```

4. При настройке демона **sendmail** задайте параметр **NameServOpt**, чтобы все клиенты могли применять записи MX сервера имен `brown.widget.com`.
5. Добавьте в файл псевдонимов псевдонимы для пользователей, у которых нет учетных записей на почтовом сервере, например:

   ```
   sam:sam@orange.widget.com
   david:david@green.widget.com
   judy:judy@red.widget.com
   ```

   Примечание: Ту же функцию могут выполнять записи почтового ящика (MB).
6. Так как в базу данных были внесены изменения, необходимо увеличить порядковый номер в записи ресурса SOA.
7. Обновите базу данных сервера имен с помощью команды `refresh -s named`.
8. Для применения изменений на клиентах вызовите команду `refresh -s sendmail`.

Есть и другие способы настройки почтового сервера доменов. В приведенных ниже процедурах используются записи MB (почтовый ящик), MR (переименование почты) и MG (почтовая группы).

Настройка почтового сервера домена с помощью записей почтовых ящиков:

Используйте следующую процедуру для настройки почтового сервера домена с помощью записей почтовых ящиков.

1. Задайте запись почтового ящика (MB) для каждого пользователя домена. Добавьте записи

   ```
   sam IN MB orange.widget.com.
   ```

   в файл `/usr/local/domain/named.abc.data` на хосте `brown.widget.com`. Такие записи сообщают почтовому серверу `black.widget.com`, куда следует направлять почту для каждого пользователя в домене.
2. Укажите, что демон **sendmail** на почтовом сервере `black.widget.com` должен использовать записи MB, заданные на сервере имен `brown.widget.com`. Воспользуйтесь параметром **NameServOpt**.
3. Так как в базу данных были внесены изменения, увеличьте порядковый номер в записи ресурса SOA.
4. Обновите базу данных сервера имен с помощью команды `refresh -s named`.
5. Введите команду `refresh -s sendmail` для применения изменений.

Создание записи переименования почты для пользователя:

Ниже описана процедура создания записи переименования почты.

1. Отредактируйте файл `/usr/local/domain/named.abc.data` на сервере имен доменов.
2. Добавьте запись переименования почты (MR) для каждого псевдонима. Например, если у пользователя `sam` есть псевдоним `sammy`, то запись о переименовании почты будет выглядеть следующим образом:

   ```
   sammy IN MR sam
   ```

   Эта запись говорит о том, что все почтовые сообщения, адресованные `sammy`, должны доставляться пользователю `sam`. Каждая запись MR должна задаваться в отдельной строке.
3. Так как в базу данных были внесены изменения, необходимо увеличить порядковый номер в записи ресурса SOA.
4. Обновите базу данных сервера имен с помощью команды `refresh -s named`.
5. Введите команду `refresh -s sendmail` для применения изменений.

Создание записей членов почтовых групп:

Ниже приведена процедура создания записей членов почтовых групп.

1. Отредактируйте файл `/usr/local/domain/named.abc.data` на сервере имен доменов.
2. Добавьте записи MG для всех почтовых групп. Записи MG работают так же, как файл `/etc/aliases`, но псевдонимы хранятся на сервере имен. Например:

```
users IN HINFO users-request widget.com
users IN MG sam
users IN MG david
users IN MG judy
```

В этом примере все почтовые сообщения, отправленные по адресу users@widget.com, будут доставляться пользователям sam, david и judy. Каждая запись MG должна задаваться в отдельной строке.

Примечание: Для пользователей sam, david и judy должны быть определены записи MB.

3. Так как в базу данных были внесены изменения, необходимо увеличить порядковый номер в записи ресурса SOA.
4. Обновите базу данных сервера имен с помощью команды refresh -s named.
5. Введите команду refresh -s sendmail для применения изменений.

Создание записей почтовых серверов:

Ниже описана процедура создания записей почтовых серверов.

1. Отредактируйте файл /usr/local/domain/named.abc.data на сервере имен доменов.
2. Добавьте записи MX для всех компьютеров, с которыми вы хотели бы обмениваться почтой, и которые не подключены непосредственно к вашей сети. Например, если все почтовые сообщения, отправляемые по адресу purple.widget.com, нужно пересылать по адресу post.office.widget, то укажите следующую запись MX:

   ```
   purple.widget.com IN MX 0 post.office.widget.
   ```

   В записях MX нужно задавать как имя хоста, так и имя компьютера. Каждая запись MG должна задаваться в отдельной строке. Можно использовать символы подстановки, например:

   ```
   *.widget.com IN MX 0 post.office.widget.
   ```

   В этом примере все почтовые сообщения, адресованные неизвестному хосту из домена widget.com (хосту, для которого не задана запись MX), будут пересылаться на адрес post.office.widget.

   Примечание: Записи MX с символами подстановки нельзя использовать в сети Internet.
3. Так как в базу данных были внесены изменения, необходимо увеличить порядковый номер в записи ресурса SOA.
4. Обновите базу данных сервера имен с помощью команды refresh -s named.
5. Введите команду refresh -s sendmail для применения изменений.

Настройка сервера пересылки

Для настройки сервера пересылки выполните описанную ниже процедуру. Потребуется внести изменения в несколько файлов и запустить демон **named** с помощью SMIT или команды.

1. Внесите необходимые изменения в файл /etc/named.conf. Если в каталоге /etc нет файла named.conf, скопируйте файл /usr/samples/tcpip/named.conf в каталог /etc и отредактируйте его. Более подробная информация и пример файла конфигурации приведены в разделе "named.conf File Format for TCP/IP" книги *Справочник по файлам* .
   - Добавьте строку forwarders в раздел опций файла /etc/named.conf. Укажите в этой строке адреса серверов имен, которые будут принимать запросы на пересылку. Например:

     ```
     options {
         ...
         directory "/usr/local/domain";
         forwarders { 192.100.61.1;  129.35.128.222; };
         ...
     };
     ```
   - Укажите циклическую область. Например:

     ```
     zone "0.0.127.in-addr.arpa" in {
         type master;
         file "named.abc.local";
     };
     ```

- Задайте область подсказок. Например:

```
zone "." IN {
    type hint;
    file "named.ca";
};
```

2. Внесите необходимые изменения в файл /usr/local/domain/named.ca. Более подробная информация и пример файла кэша приведены в разделе "DOMAIN Cache File Format for TCP/IP" книги *Справочник по файлам* .

   В этом файле содержатся адреса ответственных серверов корневого домена сети. Например:

```
; root name servers.
.          IN    NS    relay.century.com.
relay.century.com.   3600000    IN    A     129.114.1.2
```

   Примечание: Все строки в этом файле должны быть заданы в стандартном формате записи о ресурсе.

3. Внесите необходимые изменения в файл /usr/local/domain/named.abc.local. Более подробная информация и пример локального файла данных приведен в разделе DOMAIN Local Data File Format for TCP/IP книги *Справочник по файлам* .

   a. Задайте начало области ответственности (SOA) и время хранения информации в кэше по умолчанию. Например:

```
$TTL 3h    ;3 hour

@ IN    SOA     venus.abc.aus.century.com. gail.zeus.abc.aus.century.com.  (

                            1       ;serial
                            3600    ;refresh
                            600     ;retry
                            3600000 ;expire
                            86400   ;negative caching TTL
)
```

   b. Задайте запись сервера имен (NS). Например:

```
<tab> IN    NS     venus.abc.aus.century.com.
```

   c. Задайте запись указателя (PTR).

```
1      IN    PTR    localhost.
```

   Примечание: Все строки в этом файле должны быть заданы в стандартном формате записи о ресурсе.

4. Создайте файл /etc/resolv.conf с помощью следующей команды:

```
touch /etc/resolv.conf
```

   Наличие этого файла говорит хосту о том, что для преобразования имен должен применяться сервер имен, а не файл /etc/hosts.

   В другом варианте файл /etc/resolv.conf может содержать запись:

```
nameserver 127.0.0.1
```

   127.0.0.1 - это циклический адрес, по которому хост обращается к самому себе, как к серверу имен. Файл /etc/resolv.conf может содержать и такие записи:

```
domain  имя_домена
```

   В предыдущем примере в качестве параметра *имя_домена* было указано значение austin.century.com.

5. Выполните одно из следующих действий:

   - Запустите демон **named** с помощью команды SMIT smit stnamed. Эта команда указывает, что демон должен инициализироваться при каждом запуске системы. Укажите, нужно ли запустить программу **named** сейчас, при следующем запуске системы или в обоих случаях.
   - Внесите необходимые изменения в файл /etc/rc.tcpip. Удалите символ комментария в строке запуска демона **named**:

     ```
     #start /etc/named "$src_running"
     ```

     Эта команда указывает, что демон должен инициализироваться при каждом запуске системы.

6. Если вы не хотите инициализировать демон named с помощью SMIT, запустите его для данного сеанса с помощью команды

   ```
   startsrc -s named
   ```

Настройка только сервера пересылки

Для настройки только сервера пересылки выполните описанную ниже процедуру. Потребуется внести изменения в несколько файлов и запустить демон **named** с помощью SMIT или команды.

Примечание: Вы можете получить такую же конфигурацию, не запуская сервер пересылки. Вместо этого создайте файл `/etc/resolv.conf`, содержащий адреса серверов имен, которые будут играть роль серверов пересылки.

1. Внесите необходимые изменения в файл `/etc/named.conf`. Если в каталоге `/etc` нет файла `named.conf`, скопируйте файл `/usr/samples/tcpip/named.conf` в каталог `/etc` и отредактируйте его. Более подробная информация и пример файла `conf` содержится в разделе Формат файла named.conf для TCP/IP книги *Справочник по файлам* .
   - В разделе опций файла `/etc/named.conf` задайте серверы пересылки в строке forwarders, и серверы только для пересылки в строке forward only, указав IP-адреса серверов имен, которым должны пересылаться запросы. Например:

     ```
     options {
         ...
         directory "/usr/local/domain";
         forwarders { 192.100.61.1;  129.35.128.222; };
         forward only;
         ...
     };
     ```

   - Укажите циклическую область. Например:

     ```
     zone "0.0.127.in-addr.arpa" in {
         type master;
         file "named.abc.local";
     };
     ```

   - Задайте область подсказок. Например:

     ```
     zone "." IN {
         type hint;
         file "named.ca";
     };
     ```

2. Внесите необходимые изменения в файл `/usr/local/domain/named.ca`. Например: Более подробная информация и пример файла кэша приведены в разделе DOMAIN Cache File Format for TCP/IP книги *Справочник по файлам* . В этом файле содержатся адреса ответственных серверов корневого домена сети.

   ```
   ; root name servers.
   .          IN    NS    relay.century.com.
   relay.century.com.   3600000    IN    A     129.114.1.2
   ```

   Примечание: Все строки в этом файле должны быть заданы в стандартном формате записи о ресурсе.

3. Внесите необходимые изменения в файл `/usr/local/domain/named.abc.local`. Более подробная информация и пример локального файла данных приведен в разделе DOMAIN Local Data File Format for TCP/IP книги *Справочник по файлам* .
   a. Задайте начало области ответственности (SOA) и время хранения информации в кэше по умолчанию. Например:

      ```
      $TTL 3h    ;3 hour

      @ IN    SOA     venus.abc.aus.century.com. gail.zeus.abc.aus.century.com.  (

                                 1       ;serial
                                 3600    ;refresh
      ```

```
                                600     ;retry
                                3600000 ;expire
                                86400   ;negative caching TTL
    )
```

b. Задайте запись сервера имен (NS). Например:

```
<tab> IN    NS      venus.abc.aus.century.com.
```

c. Задайте запись указателя (PTR).

```
1       IN    PTR     localhost.
```

Примечание: Все строки в этом файле должны быть заданы в стандартном формате записи о ресурсе.

4. Создайте файл /etc/resolv.conf с помощью следующей команды:

   ```
   touch /etc/resolv.conf
   ```

   Наличие этого файла говорит хосту о том, что для преобразования имен должен применяться сервер имен, а не файл /etc/hosts.

   В другом варианте файл /etc/resolv.conf может содержать запись:

   ```
   nameserver 127.0.0.1
   ```

   127.0.0.1 - это циклический адрес, по которому хост обращается к самому себе, как к серверу имен. Файл /etc/resolv.conf может содержать и такие записи:

   ```
   domain  имя_домена
   ```

   В предыдущем примере в качестве параметра *имя_домена* было указано значение austin.century.com.

5. Выполните одно из следующих действий:
   - Запустите демон **named** с помощью команды SMIT smit stnamed. Эта команда указывает, что демон должен инициализироваться при каждом запуске системы. Укажите, нужно ли запустить программу **named** сейчас, при следующем запуске системы или в обоих случаях.
   - Внесите необходимые изменения в файл /etc/rc.tcpip. Удалите символ комментария в строке запуска демона **named**:

     ```
     #start /etc/named "$src_running"
     ```

     Эта команда указывает, что демон должен инициализироваться при каждом запуске системы.

6. Если вы не хотите инициализировать программу **named** с помощью SMIT, запустите ее для данного сеанса с помощью команды

   ```
   startsrc -s named
   ```

Настройка хоста для применения сервера имен

Для того чтобы настроить хост для работы с сервером имен, выполните описанную ниже процедуру.

1. Создайте файл /etc/resolv.conf с помощью следующей команды:

   ```
   touch /etc/resolv.conf
   ```

2. В первой строке файла /etc/resolv.conf укажите слова domain и полное имя домена, в котором находится данный хост. Например:

   ```
   domain abc.aus.century.com
   ```

3. В любой пустой строке ниже строки domain задайте слово nameserver, а затем через один или несколько пробелов укажите IP-адрес сервера имен, который будет применяться данным хостом (сервер имен будет обслуживать домен, указанный в строке domain). Можно указать до 3 серверов имен. Например, файл /etc/resolv.conf может содержать следующие записи:

   ```
   nameserver 192.9.201.1
   nameserver 192.9.201.2
   ```

   Система будет обращаться к серверам имен в порядке их следования в списке.

   ```
   search domainname_list
   ```

   Вместо этого последовательность опроса клиентом списка доменов может быть определена с помощью ключевого слова поиска. В этом случае значениями domainname_list будут abc.aus.century.com и aus.century.com. Количество символьных строк в списке доменов domainname_list не может превышать 1024. Каждая строка отделяется от следующей пробелом.

4. Если сервер имен работает, вы можете проверить соединение с этим сервером с помощью следующей команды:

   ```
   host имя-хоста
   ```

   Чтобы определить, выполняет ли сервер преобразование имен, укажите какое-либо имя хоста. Полученный вывод должен выглядеть следующим образом:

   ```
   brown.abc.aus.century.com is 129.35.145.95
   ```

В таблице приведены другие задачи настройки.

Таблица 62. Задачи настройки хоста для применения сервера имен

Процедура	Команды быстрого доступа SMIT	Команда или файл
Создать файл /etc/resolv.conf	smit stnamerslv2	create и edit /etc/resolv.conf[1]
Получить список всех серверов имен, используемых хостом	smit lsnamerslv	view /etc/resolv.conf
Добавить сервер имен	smit mknamerslv	edit /etc/resolv.conf[2]
Удалить сервер имен	smit rmnamerslv	edit /etc/resolv.conf
Запустить/Перезапустить службу преобразования имен доменов	smit stnamerslv	
Отменить применение службы преобразования имен доменов	smit spnamerslv	
Изменить/Показать домен	smit mkdomain	edit /etc/resolv.conf
Удалить домен	smit rmdomain	edit /etc/resolv.conf

Связанная информация

Файл netsvc.conf

Динамические зоны на сервере имен DNS

Команда **named** поддерживает динамическое обновление информации. Необходимо настроить файлы конфигурации и базы данных named таким образом, чтобы клиенты могли выполнять обновление. Область может быть задана как динамическая или как статическая. По умолчанию применяются статические области.

Для создания динамической области нужно добавить ключевое слово **allow-update** в раздел zone файла /etc/named.conf. Вместе с ключевым словом **allow-update** задается список IP-адресов хостов, которым разрешено отправлять запросы на обновление. Более подробная информация и пример файла conf содержится в разделе Формат файла named.conf для TCP/IP книги *Справочник по файлам* . В приведенном ниже примере обновлять динамическую область разрешено всем хостам:

```
zone "abc.aus.century.com" IN {
    type master;
    file "named.abc.data";
    allow-update { any; };
};
```

После того, как динамическая область описана, можно установить один из следующих трех режимов защиты области:

Элемент	Описание
Unsecured	Всем разрешено в любой момент обновлять любую информацию в области. **Внимание:** Не рекомендуется использовать этот режим. Применение данного режима может привести к потере данных, их перехвату и неполадкам в работе пользователей. Рекомендуется задать четкий список IP-адресов хостов, которым разрешено обновлять незащищенную область.
Controlled	Разрешено создание новой и замена существующей информации. Это, вероятно, самый удобный режим для защищенной среды передачи. В данном режиме все поступающие обновления должны снабжаться меткой времени поступления и зашифрованной подписью.
Presecured	Все обновления существующей информации должны заменяться аналогичной информацией. Не позволяет создавать новую информацию. В данном режиме все поступающие обновления должны снабжаться меткой времени поступления и зашифрованной подписью.

По умолчанию для динамических областей применяется незащищенный режим (unsecured). Для применения другого режима укажите слово `controlled` or `presecured` после ключевого слова **update-security** в разделе zone файла `/etc/named.conf`. Это слово сообщает серверу **named** , какой уровень защиты он должен использовать при работе с этой областью. Например:

```
zone "abc.aus.century.com" IN {
    type master;
    file "named.abc.data";
    allow-update { any; };
    update-security controlled;
};
```

После выбора режима нужно изменить существующие файлы данных в соответствии с текущим уровнем защиты. В незащищенном режиме файлы данных используются "как есть". В управляемом режиме и режиме с предварительной защитой для каждого хоста области необходимо создать пару ключей главного сервера и хоста. Это можно сделать с помощью команды **nsupdate** с параметром **-g**. Данная команда создает пару ключей (личный ключ и общий ключ). Эти ключи используются для создания идентификационных меток обновления. После создания всех ключей для списка имен областей, добавьте их в файл данных. Запись KEY имеет следующий формат:

Индекс ttl Класс Тип Ключевые-флаги Протокол Алгоритм Ключевые-данные

где:

Элемент	Описание
Индекс	Определяет имя, используемое для обращения к данным в области.
ttl	Задает для этих данных время хранения в кэше (TTL). Это необязательное поле.
Класс	Определяет класс данных. Он зависит от области, но обычно имеет значение IN.
Тип	Задает тип записи. В данном случае запись имеет тип KEY.
Флаги	Задает для named информацию о ключе. 0x0000 определяет запись обычного ключа, использующегося для хоста. 0x0100 определяет запись ключа, связанную с именем области.
Протокол	Задает используемый протокол. В данный момент применяется только протокол `0`.
Алгоритм	Задает алгоритм ключа. В данный момент применяется только алгоритм 1. Это способ идентификации MD5 с личным/общим ключом.
Данные	Задает ключ в формате base64. Команда **nsupdate** создает как общие, так и личные ключи в формате base64. Общий ключ указывается в конце файла вывода.

Например, для того чтобы убедиться, что для имени хоста в динамической области установлена защита, необходимо добавить следующую строку в определение области, в которую входит данный хост:

```
bears    4660    IN    KEY    0x0000    0    1    AQOtg......
```

В этом примере для хоста `bears` определена запись KEY. Каждый, кто захочет обновить `bears`, должен будет подписать свое обновление личным ключом, соответствующим общему ключу, хранящемуся в базе данных. Для того чтобы команда **nsupdate** была успешно выполнена, необходимо разместить общий ключ в файле ключей клиента (по умолчанию, `/etc/keyfile`). Записи этого файла имеют следующий формат:

имя-хоста имя-главного-сервера base64 ключ

Аналогичную запись KEY необходимо указать в разделе описания области. *Ключ области должен быть указан и для режима presecured, и для режима controlled. В противном случае будет применяться незащищенный режим.* Это можно сделать так же, как в примере с `bears`, но личный ключ должен быть оставлен администратору для его применения в команде **nsupdate** при работе в административном режиме.

1. Для создания пары ключей с помощью команды **nsupdate** введите:

   ```
   nsupdate -g -h имя-области -p имя-сервера -k файл-ключей
   ```

   При этом создается ключ для области. В этом примере команда **nsupdate** связана с командой **nsupdate4**. Это можно сделать с помощью следующей команды:

   ```
   ln -fs /usr/sbin/nsupdate4 /usr/sbin/nsupdate
   ```

2. Укажите последний ключ из пары в начале раздела описания области следующим образом:

   ```
   IN     KEY     0x0100  0  1  Ключ
   ```

   Запись для файла `named.abc.data` выглядит следующим образом:

   ```
   $TTL 3h    ;3 hour

   @ IN    SOA     venus.abc.aus.century.com. gail.zeus.abc.aus.century.com. (
                   1       ;serial
                   3600    ;refresh
                   600     ;retry
                   3600000 ;expire
                   86400   ;negative caching TTL
   )
           IN      NS      venus.abc.aus.century.com.
           IN      KEY     0x0100  0 1 AQPlwHmIQeZzRk6Q/nQYhs3xwnhfTgF/
                           8YlBVzKSoKxVKPNLINnYW0mB7attTcfhHaZZcZr4u/
                           vDNikKnhnZwgn/
   venus           IN    A         192.9.201.1
   earth           IN    A         192.9.201.5
   mars            IN    A         192.9.201.3
   ```

3. Теперь можно загрузить область, обновив сервер имен. Поместите файл AdminKeyFile в систему клиента или на сервер DHCP, обновляющий область. Ключ области из файла AdminKeyFile может применяться для применения обновлений и выполнения операций обслуживания на сервере имен.

Защита BIND 9

Для обеспечения защиты в BIND 9 реализованы подписи транзакций (TSIG) и подписи (SIG) для команды **named**.

Как и с BIND 8, сервер имен с BIND 9 по умолчанию не разрешает выполнять динамическое обновление ответственных областей.

В BIND 9 для связи между серверами в основном используется TSIG. Это относится к сообщениям передачи области, уведомления и рекурсивного запроса. TSIG также может применяться для динамического изменения. Основной сервер динамической области должен осуществлять управление изменениями с помощью управления доступом, но функций управления доступом на основе IP для этого недостаточно.

Применяя шифрование на основе ключей вместо списков управления доступа, TSIG позволяет осуществлять управление доступом для изменения динамических областей. В отличие от списков управления доступом (ACL), ключ TSIG может передаваться другим агентам, изменяющим область, без необходимости изменения файлов конфигурации на сервере имен. Это означает, что серверу имен не нужно повторно считывать файлы конфигурации.

Следует отметить, что BIND 9 содержит не все ключевые слова, реализованные в BIND 8. В этом примере применяется простая главная конфигурация BIND 8.

Примечание: Для применения демона named 9 необходимо переключить символьную связь с демоном **named** на **named9**, а связь с **nsupdate** - на **nsupdate9** с помощью следующих команд:

1. `ln -fs /usr/sbin/named9 /usr/sbin/named`

2. `ln -fs /usr/sbin/nsupdate9 /usr/sbin/nsupdate`

1. Создайте ключ с помощью команды **dnssec-keygen**:

   ```
   dnssec-keygen -a HMAC-MD5 -b 128 -n HOST keyname
   ```

   - `HMAC-MD5` - применяемый алгоритм шифрования
   - `128` - длина ключа (или число бит)
   - `HOST`: `HOST` - ключевое слово TSIG, применяемое для создания ключей хостов для шифрования с общим ключом.

   Команда

   ```
   dnssec-keygen -a HMAC-MD5 -b 128 -n HOST venus-batman.abc.aus.century.com
   ```

   создает два ключа:

   ```
   Kvenus-batman.abc.aus.century.com.+157+35215.key
   Kvenus-batman.abc.aus.century.com.+157+35215.private
   ```

   - `157` - применяемый алгоритм (HMAC-MD5).
   - `35215` - оттиск ключа. Оттиск нужен в DNNSEC, так как для одной области может существовать несколько ключей.

2. Добавьте следующую запись в файл named.conf главного сервера имен:

   ```
   // TSIG Key
   key venus-batman.abc.aus.century.com. {
           algorithm hmac-md5;
           secret "+UWSvbpxHWFdNwEAdy1Ktw==";
   };
   ```

   Так как предполагается применение HMAC-MD5, то в последней записи обоих файлов ключей находится общий ключ. Выберите безопасные способ копирования общего секретного ключа в систему клиента. Файл ключа копировать не нужно, достаточно скопировать общий секретный ключ.

   Ниже приведена запись для файла `Kvenus-batman.abc.aus.century.com.+157+35215.private`:

   ```
   Private-key-format: v1.2
   Algorithm: 157 (HMAC_MD5)
   Key: +UWSvbpxHWFdNwEAdy1Ktw==
   ```

   Ниже приведен пример файла `named.conf` для главного сервера имен. В области `abc.aus.century.com` передача информации об области и динамические изменения разрешены только серверам с ключом `venus-batman.abc.aus.century.com`. То же самое необходимо сделать в обратной области, в которой для внесения изменений серверам нужен общий ключ.

   ```
   // TSIG Key
   key venus-batman.abc.aus.century.com. {
           algorithm hmac-md5;
           secret "+UWSvbpxHWFdNwEAdy1Ktw==";
   };

   options {
           directory "/usr/local/domain";
   };

   zone "abc.aus.century.com" in {
           type master;
           file "named.abc.data";
           allow-transfer { key venus-batman.abc.aus.century.com.;};
           allow-update{ key venus-batman.abc.aus.century.com.; };
   };
   ```

   Так как передача информации об областях может выполняться не только в областях, для которых необходим ключ, то изменения необходимо внести и в файл `named.conf` подчиненного сервера имен. Все запросы в область 192.9.201.1(venus.abc.aus.century.com) подписаны ключом. Учтите, что имя ключа (venus-batman.abc.aus.century.com.) должно совпадать с именем, указанным на серверах, которые применяют этот ключ.

   Ниже приведен пример файла `named.conf` на подчиненном сервере имен:

```
// TSIG Key
key venus-batman.abc.aus.century.com. {
        algorithm hmac-md5;
        secret "+UWSvbpxHWFdNwEAdy1Ktw==";
};

server 192.9.201.1{
        keys { venus-batman.abc.aus.century.com.;};
};

options {
        directory "/usr/local/domain";
};

zone "abc.aus.century.com" IN {
    type slave;
    file "named.abc.data.bak";
    masters { 192.9.201.1; };
};
```

Подпись транзакций BIND 9:

В BIND 9 для связи между серверами в основном используется TSIG.

Это относится к сообщениям передачи области, уведомления и рекурсивного запроса. TSIG также может применяться для динамического изменения. Основной сервер динамической области должен осуществлять управление изменениями с помощью управления доступом, но функций управления доступом на основе IP для этого недостаточно.

Применяя шифрование на основе ключей вместо списков управления доступа, TSIG позволяет осуществлять управление доступом для изменения динамических областей. В отличие от списков управления доступом (ACL), ключ TSIG может передаваться другим агентам, изменяющим область, без необходимости изменения файлов конфигурации на сервере имен. Это означает, что серверу имен не нужно повторно считывать файлы конфигурации.

Следует отметить, что BIND 9 содержит не все ключевые слова, реализованные в BIND 8. В этом примере применяется простая главная конфигурация BIND 8.

Примечание: Для применения демона named 9 необходимо переключить символьную связь с демоном **named** на **named9**, а связь с **nsupdate** - на **nsupdate9** с помощью следующих команд:

1. `ln -fs /usr/sbin/named9 /usr/sbin/named`
2. `ln -fs /usr/sbin/nsupdate9 /usr/sbin/nsupdate`

1. Создайте ключ с помощью команды **dnssec-keygen**:

   ```
   dnssec-keygen -a HMAC-MD5 -b 128 -n HOST keyname
   ```

   - `HMAC-MD5` - применяемый алгоритм шифрования
   - `128` - длина ключа (или число бит)
   - `HOST`: `HOST` - ключевое слово TSIG, применяемое для создания ключей хостов для шифрования с общим ключом.

   Команда

   ```
   dnssec-keygen -a HMAC-MD5 -b 128 -n HOST venus-batman.abc.aus.century.com
   ```

   создает два ключа:

   ```
   Kvenus-batman.abc.aus.century.com.+157+35215.key
   Kvenus-batman.abc.aus.century.com.+157+35215.private
   ```

   - `157` - применяемый алгоритм (HMAC-MD5).
   - `35215` - оттиск ключа. Оттиск нужен в DNNSEC, так как для одной области может существовать несколько ключей.

2. Добавьте следующую запись в файл named.conf главного сервера имен:

```
// TSIG Key
key venus-batman.abc.aus.century.com. {
         algorithm hmac-md5;
         secret "+UWSvbpxHWFdNwEAdy1Ktw==";
};
```

Так как предполагается применение HMAC-MD5, то в последней записи обоих файлов ключей находится общий ключ. Выберите безопасные способ копирования общего секретного ключа в систему клиента. Файл ключа копировать не нужно, достаточно скопировать общий секретный ключ.

Ниже приведена запись для файла `Kvenus-batman.abc.aus.century.com.+157+35215.private`:

```
Private-key-format: v1.2
Algorithm: 157 (HMAC_MD5)
Key: +UWSvbpxHWFdNwEAdy1Ktw==
```

Ниже приведен пример файла `named.conf` для главного сервера имен. В области `abc.aus.century.com` передача информации об области и динамические изменения разрешены только серверам с ключом `venus-batman.abc.aus.century.com`. То же самое необходимо сделать в обратной области, в которой для внесения изменений серверам нужен общий ключ.

```
// TSIG Key
key venus-batman.abc.aus.century.com. {
         algorithm hmac-md5;
         secret "+UWSvbpxHWFdNwEAdy1Ktw==";
};

options {
         directory "/usr/local/domain";
};
zone "abc.aus.century.com" in {
         type master;
         file "named.abc.data";
         allow-transfer { key venus-batman.abc.aus.century.com.;};
         allow-update{ key venus-batman.abc.aus.century.com.; };
};
```

Так как передача информации об областях может выполняться не только в областях, для которых необходим ключ, то изменения необходимо внести и в файл `named.conf` подчиненного сервера имен. Все запросы в область 192.9.201.1(`venus.abc.aus.century.com`) подписаны ключом. Учтите, что имя ключа (`venus-batman.abc.aus.century.com.`) должно совпадать с именем, указанным на серверах, которые применяют этот ключ.

Ниже приведен пример файла `named.conf` на подчиненном сервере имен:

```
// TSIG Key
key venus-batman.abc.aus.century.com. {
         algorithm hmac-md5;
         secret "+UWSvbpxHWFdNwEAdy1Ktw==";
};

server 192.9.201.1{
         keys { venus-batman.abc.aus.century.com.;};
};

options {
         directory "/usr/local/domain";
};

zone "abc.aus.century.com" IN {
     type slave;
     file "named.abc.data.bak";
     masters { 192.9.201.1; };
};
```

Подпись BIND 9:

BIND 9 частично поддерживает подписи транзакций DNSSEC SIG, в соответствии с RFC 2535.

Функции SIG применяют общие и личные ключи для идентификации сообщений.

С помощью записей SIG администраторы могу подписывать данные области, удостоверяя их подлинность.

Защита корневой области:

Используя данные действия для защиты корневой области, предположим, что другие серверы сети не применяют BIND 9, и необходимо защитить данные области, предоставив другим серверам возможность проверить данные этой области.

Необходимо указать что данная область (в нашем случае - `aus.century.com`) является защищенной корневой областью, и удостоверяет подлинность данных входящих в нее защищенных областей.

1. Создайте ключи с помощью команды **dnssec-keygen**:

   ```
   dnssec-keygen -a RSA -b 512 -r /usr/sbin/named -n ZONE aus.century.com.
   ```

   Примечание: Если установлен компонент OpenSSL, то в качестве алгоритма шифрования для создания ключей может применяться RSA, хотя для этого сначала необходимо связать библиотеку DNS с защищенной библиотекой DNS с помощью следующей команды:

   ```
   ln -fs /usr/lib/libdns_secure.a /usr/lib/libdns.a
   ```

   - ZONE: ZONE - ключевое слово DNSSEC для создания ключей области, предназначенных для шифрования с помощью пары личного и общего ключей.
   - Флаг `r` указывает случайное устройство.
2. Добавьте запись общего ключа, как в файле `named.conf`. Добавленная в этом примере запись указана ниже. Ниже приведено содержимое файла ключей `Kaus.century.com.+001+03254.key`.

   ```
   abc.aus.century.com. IN KEY 256 3 1
   AQOnfGEAg0xpzSdNRe7KePq3Dl4NqQiq7HkwKl6TygUfaw6vz6ldmauB4UQFcGKOyL68/
   Zv5ZnEvyB1fMTAaDLYz
   ```

   Общий ключ из файла `Kzonename.+algor.+fingerprint.key`, или, в данном примере, - `Kaus.century.com.+001+03254.key`. Необходимо удалить класс IN и указать KEY, а также указать тело ключа. После добавления этой запись в файл `/etc/named.conf` и обновления сервера имен область `aus.century.com` станет защищенной корневой областью.

   ```
   trusted-keys {
           aus.century.com. 256 3 1 "AQOnfGEAg0xpzSdNRe7KePq3Dl4NqQiq7HkwKl6Tyg
   Ufaw6vz6ldmauB 4UQFcGKOyL68/Zv5ZnEvyB1fMTAaDLYz";
   };
   options {
           directory "/usr/local/domain";
   };

   zone "abc.aus.century.com" in {
           type master;
           file "named.abc.data.signed";
           allow-update{192.9.201.1;};
   };
   ```

Настройка цепочки доверия:

Защитив корневую область, можно настроить защиту дочерних областей. В данном примере будет настроена защита области `abc.aus.century.com` .

Выполните следующие действия, чтобы защитить оставшиеся дочерние области:

1. Создайте пары ключей с помощью команды **dnssec-keygen**:

```
dnssec-keygen -a RSA -b 512 -r /usr/sbin/named -n ZONE abc.aus.century.com.
```

Флаг `r` указывает произвольный файл ввода.

2. Создайте набор ключей с помощью команды **dnssec-makekeyset**:

   ```
   dnssec-makekeyset -t 172800 Kabc.aus.century.com.+001+11515.key
   ```

   где *Kabc.aus.century.com.+001+11515.key* - ваш общий ключ.

   В результате будет создан файл набора ключей `keyset-abc.aus.century.com`.

3. Отправьте этот файл набора ключей в родительскую область для добавления подписи. В данном примере родительская область - это защищенная корневая область `aus.century.com`.
4. Родительская область подписывает ключ с помощью своего личного ключа.

   ```
   dnssec-signkey keyset-abc.aus.century.com. Kaus.century.com.+001+03254.private
   ```

   В результате будет создан файл `signedkey-abc.aus.century.com`, который родительская область должна вернуть в дочернюю.

5. На дочернем сервере имен области `abc.aus.century.com` добавьте запись `$INCLUDE Kabc.aus.century.com.+001+11515.key` в файл области `named.abc.data`. Поместите файл `signedkey-abc.aus.century.com` в один каталог с файлом области `named.abc.data` . При подписании области на следующем шаге программа включит файл `signedkey-abc.aus.century.com`, полученный от родительской области.

   ```
   $TTL 3h    ;3 hour

   @ IN    SOA     venus.abc.aus.century.com. gail.zeus.abc.aus.century.com.  (
                   1       ;serial
                   3600    ;refresh
                   600     ;retry
                   3600000 ;expire
                   86400   ;negative caching TTL
   )
   $INCLUDE Kabc.aus.century.com.+001+03254.key
   ```

6. Подпишите область с помощью команды **dnssec-signzone**:

   ```
   dnssec-signzone -o abc.aus.century.com. named.abc.data
   ```

7. Укажите в файле **named.conf** дочерней области `abc.aus.century.com` новый файл подписанной области (`named.abc.data.signed`). Например:

   ```
   options {
           directory "/usr/local/domain";
   };

   zone "abc.aus.century.com" in {
           type master;
           file "named.abc.data.signed";
           allow-update{192.9.201.1;};
   };
   ```

8. Обновите сервер имен.

Информация по устранению неполадок приведена в разделе “Неполадки преобразования имен” на стр. 425.

Планирование и настройка преобразования имен LDAP (Схема IBM SecureWay Directory)

Упрощенный протокол доступа к каталогам (LDAP) - это открытый стандартный протокол, регламентирующий способ получения и изменения информации в каталоге.

Схема **LDAP** определяет правила упорядочения данных. Класс объектов **ibm-HostTable**, входящий в схему IBM SecureWay Directory, можно применять для хранения таблиц преобразования имен хостов в адреса.

Класс объектов **ibm-HostTable** определен следующим образом:

```
Имя класса:            ibm-HostTable
Описание:              Запись таблицы хостов, содержащая список имен хостов и
                       соответствующих IP-адресов.
OID:            TBD
RDN:                   ipAddress
Родительский класс:    top
Обязательные атрибуты: host, ipAddress
Необязат. атрибуты:    ibm-hostAlias, ipAddressType, description
```

Ниже приведены определения атрибутов:

```
Имя атрибута:   ipAddress
Описание:       IP-адрес хоста в таблице хостов
OID:            TBD
Синтаксис:      caseIgnoreString
Длина:          256
Одно значение:  Да
Имя атрибута:   ibm-hostAlias
Описание:       Псевдоним хоста в таблице хостов
OID:            TBD
Синтаксис:      caseIgnoreString
Длина:          256
Одно значение:  Нет
Имя атрибута:   ipAddressType
Описание:       Семейство IP-адресов (1=IPv4, 2=IPv6)
OID:            TBD
Синтаксис:      Integer
Длина:          11
Одно значение:  Да
Имя атрибута:   host
Описание:       Имя хоста.
OID:            1.13.18.0.2.4.486
Синтаксис:      caseIgnoreString
Длина:          256
Одно значение:  Нет
Имя атрибута:   description
Описание:       Описание записи в каталоге.
OID:            2.5.4.13
Синтаксис:      caseIgnoreString
Длина:          1024
Одно значение:  Нет
```

Для настройки сервера **LDAP**, соответствующего схеме IBM SecureWay Directory, в качестве хранилища таблицы хостов выполните следующие действия:

1. Добавьте суффикс на сервер **LDAP**. Суффикс - это начальная точка базы данных хостов. Например, "cn=hosts". Это можно сделать с помощью инструмента Администрирование сервера IBM SecureWay Directory с интерфейсом браузера.
2. Создайте файл в формате обмена данными LDAP (LDIF). Это можно сделать вручную или с помощью команды **hosts2ldif**, создающей файл LDIF из файла `/etc/hosts`. Дополнительная информация приведена в описании команды hosts2ldif. Ниже приведен пример файла LDIF:

   ```
   dn: cn=hosts
   objectclass: top
   objectclass: container
   cn: hosts
   dn: ipAddress=1.1.1.1, cn=hosts
   host: test
   ipAddress: 1.1.1.1
   objectclass: ibm-HostTable
   ipAddressType: 1
   ibm-hostAlias: e-test
   ibm-hostAlias: test.austin.ibm.com
   description: первый интерфейс ethernet
   dn: ipAddress=fe80::dead, cn=hosts
   ```

```
host: test
ipAddress: fe80::dead
objectclass: ibm-HostTable
ipAddressType: 2
ibm-hostAlias: test-ll
ibm-hostAlias: test-ll.austin.ibm.com
description: интерфейс v6 на уровне канала
```

3. Импортируйте каталог хостов из файла LDIF на сервер **LDAP**. Это можно сделать с помощью команды **ldif2db** или инструмента Администрирование сервера IBM SecureWay Directory.

Для того чтобы клиент начал использовать базу данных хостов на сервере LDAP с помощью механизма **LDAP** выполните следующие действия:

1. Создайте файл `/etc/resolv.ldap`. Более подробная информация и пример файла `resolv.ldap` приведены в разделе resolv.ldap File Format for TCP/IP в книге *Справочник по файлам* .
2. Измените способ преобразования имен по умолчанию в переменной среды **NSORDER**, файле `/etc/netsvc.conf` или файле `/etc/irs.conf`. Дополнительная информация приведена в описании форматов файлов netsvc.conf File Format for TCP/IP и irs.conf File Format for TCP/IP в книге *Справочник по файлам* .

Хотя применение механизма `ldap` еще поддерживается, эта функция является устаревшей. Этот существующий механизм `ldap` работает со схемой IBM SecureWay Directory, а `nis_ldap` (NIS_LDAP) работает со схемой RFC 2307. Вместо механизма `ldap` рекомендуется применять механизм `nis_ldap`. Информация о преобразовании имен `nis_ldap` приведена в разделе “Планирование и настройка преобразования имен NIS_LDAP (схема RFC 2307)”.

Планирование и настройка преобразования имен NIS_LDAP (схема RFC 2307)

В AIX 5.2 реализован новый механизм преобразования имен NIS_LDAP.

Новый механизм отличается от существующего механизма LDAP схемой LDAP (набором атрибутов и классов объектов, определяющих группировку атрибутов для описания объектов). Существующий механизм LDAP работает с серверами LDAP, совместимыми со схемой IBM SecureWay Directory , и поддерживает только службу присвоения имен хоста. Механизм NIS_LDAP работает с серверами LDAP, совместимыми со схемой RFC 2307, и поддерживает все службы NIS: пользователи и группы, хосты, службы, протоколы, сети и сетевую группу. В RFC 2307 определен набор атрибутов и классов объектов, с помощью которых можно описывать службы информации о сети, включая пользователей и группы.

- Для настройки сервера LDAP необходимо установить этот сервер и перенести на него все необходимые данные.
  1. Настройте сервер с помощью команды **mksecldap**. Механизм `nis_ldap` работает только со схемой RFC 2307. Для настройки сервера LDAP команду **mksecldap** следует вводить с опцией `-S rfc2307` или `-S rfc2307aix` (а не с опцией `-S aix`, которая задает схему IBM SecureWay Directory). По умолчанию команда **mksecldap** переносит на сервер LDAP пользователей и группы, определенные в локальной системе. Для отключения переноса данных укажите опцию `-u NONE`.

     ```
     mksecldap -s -a cn=admin -p adminpwd -S rfc2307aix
     ```

     Эта команда настраивает сервер LDAP с DN администратора `cn=admin` и паролем `adminpwd`. Кроме того, в файл конфигурации `/etc/slapd32.conf` сервера LDAP добавляется суффикс по умолчанию `cn=aixdata`.

     По умолчанию команда **mksecldap** переносит на сервер LDAP пользователей и группы, определенные в локальной системе. Для отключения переноса данных укажите опцию `-u NONE`, которая запрещает перенос локальных пользователей и групп на сервер LDAP, чтобы позже добавить только пользователей и группы NIS.

     ```
     mksecldap -s -a cn=admin -p adminpwd -u NONE
     ```

     Дополнительные сведения о команде **mksecldap** приведены в описании этой команды в книге *Справочник по командам, том 3*.

2. Перенесите данные NIS. Введите на сервере NIS команду **nistoldif**, чтобы перенести отображения NIS на сервер LDAP. С помощью команды **nistoldif** можно также перенести данные из простых файлов. Введите команду **nistoldif** в системе, содержащей данные NIS, которые необходимо перенести на сервер LDAP.

   ```
   nistoldif -h server1.ibm.com -a cn=admin -p adminpwd -d cn=aixdata
   ```

   Эта команда перенесет отображения NIS из локальной системы на сервер LDAP - `server1.ibm.com`. Данные NIS помещаются в DN `cn=aixdata`. С помощью команды **nistoldif** можно также перенести данные из простых файлов в любой системе на сервер LDAP. Данные из простых файлов заменят все отображения, отсутствующие на сервере NIS.

   Дополнительные сведения о команде **nistoldif** приведены в описании этой команды в книге *Справочник по командам, том 4*.

   Примечание: Имена представлены атрибутом `cn` сервера LDAP. В атрибуте `cn`, определенном в RFC 2307, не учитывается регистр символов. Имена, отличающиеся только регистром букв, будут объединены на сервере. При подстановке регистр букв также не учитывается. Поиск `TCP`, `tcp` и `Tcp` возвратит запись протокола TCP.

- Для настройки клиента LDAP для работы с именами на сервере LDAP введите команду **mksecldap** с опциями настройки клиента.

  1. Команда **mksecldap** сохраняет имя сервера LDAP, порт, DN администратора, пароль и базовое DN в файле `/etc/security/ldap/ldap.cfg`, который демон **secldapclntd** считывает при запуске. Команда **mksecldap** автоматически запускает демон **secldapclntd**, если установка выполнена успешно.

     Дополнительная информация приведена в описании файла `/etc/security/ldap/ldap.cfg` file в книге *Справочник по файлам* и в описании демона **secldapclntd** в *Справочник по командам, том 5*.

  2. Команда **mksecldap** добавляет механизм `nis_ldap` в файлы `/etc/netsvc.conf` и `/etc/irs.conf` для направления функций преобразования имен на сервер LDAP. Кроме того, для применения преобразования имен NIS_LDAP можно вручную присвоить переменной среды **NSORDER** значение `nis_ldap`.

     ```
     mksecldap -c -a cn=admin -p adminpwd -h server1.ibm.com
     ```

     После этого локальная система будет применять сервер LDAP `server1.ibm.com`. Для идентификации на сервере LDAP для клиента должны быть заданы пароль и DN администратора сервера LDAP. Для преобразования имен с помощью NIS_LDAP в файлы `/etc/netsvc.conf` и `/etc/irs.conf` вносятся соответствующие изменения.

     Дополнительная информация приведена в описании форматов файлов `/etc/netsvc.conf` и `/etc/irs.conf` в книге *Справочник по файлам* .

  3. На преобразование имен для пользователей и групп файлы `/etc/netsvc.conf` и `/etc/irs.conf` не влияют. Управление преобразованием этих имен осуществляется с помощью файла `/etc/security/user`. Для того чтобы пользователи LDAP имели возможность войти в систему AIX, присвойте переменным пользователя `SYSTEM` и `registry` значение `LDAP` в файле `/etc/security/user` в этой системе клиента. Для этого можно воспользоваться командой **chuser**.

     ```
     chuser -R LDAP SYSTEM=LDAP registry=LDAP foo
     ```

     Систему можно настроить так, чтобы все пользователи LDAP могли входить в эту систему. Для этого необходимо внести изменения в файл `/etc/security/user`. Добавьте в раздел root запись `registry = files`. Затем добавьте в раздел по умолчанию записи `SYSTEM = LDAP` и `registry = LDAP`.

     Дополнительные сведения об идентификации приведены в разделе Упрощенный протокол доступа к каталогам книги *Защита*.

Информация, связанная с данной:

Переход от NIS к службами LDAP с поддержкой RFC 2307

Присвоение адреса и параметров TCP/IP - протокол динамической настройки хостов

Протокол управления передачей/Протокол Internet предназначен для организации связи между компьютерами с определенными адресами. Одной из обязанностей администратора сети является присвоение адресов и задание параметров для всех машин в сети. Обычно администратор информирует пользователей о том, какие адреса выделены их системам, и предоставляет пользователям возможность самим выполнить настройку. Однако ошибки при настройке или неправильное понимание могут вызвать у пользователей вопросы, которые администратор должен будет рассматривать индивидуально. **Протокол динамической настройки хостов (DHCP)** позволяет администратору централизованно настраивать сеть без участия конечных пользователей.

DHCP - это протокол прикладного уровня, который позволяет подключенным к сети системам получать IP-адрес и другие параметры конфигурации с сервера. Для получения информации программа-демон, работающая в системе-клиенте, обменивается пакетами данных с аналогичным демоном, работающим на сервере. В настоящее время клиент **DHCP** включается в базовую часть большинства операционных систем.

Для получения адреса демон клиента **DHCP** (**dhcpcd**) рассылает запрос **DHCP**. Этот запрос принимается и обрабатывается сервером. (Для повышения надежности в сети может быть настроено несколько серверов.) При наличии свободного адреса сервер создает ответное сообщение **DHCP**, в котором указывает предлагаемый IP-адрес и другие параметры настройки клиента. Клиент получает и сохраняет это предложение **DHCP**, ожидая предложений от других серверов. После выбора наилучшего предложения клиент рассылает запрос **DHCP**, в котором указывает выбранное предложение сервера.

Этот запрос получают все работающие в сети серверы **DHCP**. Каждый из них проверяет, ему ли был направлен запрос. Если нет, то сервер освобождает адрес, предложенный им данному клиенту. Если да, то сервер помечает адрес как присвоенный и возвращает подтверждение **DHCP**. На этом обработка запроса завершается, и клиенту на определенное время выделяется адрес.

После того, как пройдет половина времени выделения адреса, клиент отправляет серверу запрос на *обновление*. Если сервер готов продлить время выделения адреса, он отправляет подтверждение **DHCP**. Если клиент не получит ответ от того сервера, который выделил ему текущий адрес, то он рассылает оповещающее сообщение, которое позволяет обнаружить сервер, например, в том случае, если он был переведен в другую сеть. Если до завершения времени выделения адреса клиент не обновит свой адрес, то работа сетевого интерфейса завершается и процесс начинается сначала. Такая методика предотвращает присвоение одинаковых адресов нескольким клиентам.

Сервер **DHCP** присваивает адреса на основе ключей. Существует четыре общих ключа: сеть, класс, вендор и ИД клиента. На основании этих ключей сервер выбирает адрес и параметры конфигурации, которые передаются клиенту.

сеть Указывает, из какого сегмента сети получен пакет. Этот ключ позволяет серверу проверять свою базу данных и присваивать адрес в зависимости от сегмента сети.

class Полностью определяет конфигурацию клиента. Этот ключ может включать адрес и параметры. С его помощью можно указать назначение системы в сети или задать способ объединения систем в группы для упрощения администрирования. Например, администратор сети может создать класс `netbios` для задания опций клиентов NetBIOS или класс `accounting`, объединяющий компьютеры бухгалтерии, которым необходим доступ к определенному принтеру.

vendor Обеспечивает идентификацию аппаратной/программной платформы клиента (например, клиента Microsoft Windows 95 или OS/2 Warp).

ИД клиента
Идентифицирует клиента по имени хоста или по адресу MAC. ИД клиента задается в файле конфигурации демона **dhcpcd**. Кроме того, ИД клиента может применяться сервером для передачи опций данному клиенту или для запрета передачи каких-либо опций клиенту.

Эти ключи могут по одному или в сочетании друг с другом. Если клиент использует несколько ключей, и при этом может быть присвоено несколько адресов, то выбирается только один из них, причем набор опций определяется первым выбранным ключом. Более подробная информация о выборе ключей и адресов приведена в разделе “Настройка DHCP” на стр. 210.

Промежуточный агент обеспечивает передачу оповещающих сообщений клиента за пределы локальной сети. Этот агент называется промежуточным агентом BOOTP. Промежуточные агенты работают как посредники, пересылающие пакеты **DHCP** и **BOOTP**.

Серверы DHCP

В операционной системе AIX сервер **DHCP** был сегментирован на три основных компонента.

Основными компонентами сервера **DHCP** являются база данных, средства поддержки протокола и набор служебных нитей, для каждого из которых задается собственная информация о конфигурации.

Базы данных DHCP:

База данных `db_file.dhcpo` служит для хранения записей о том, каким клиентам были выделены те или иные адреса, а также для управления доступом клиентов (например, можно разрешить работу лишь некоторых клиентов из определенных подсетей или выключить поддержку клиентов **BOOTP** в некоторой сети).

Кроме того, в базе данных хранятся опции, предназначенные для передачи клиентам. База данных реализована в виде динамически загружаемого объекта, что позволяет легко модернизировать и обслуживать сервер.

База данных заполняется и проверяется в соответствии с информацией, заданной в файле конфигурации. Обновление базы данных осуществляется с помощью набора контрольных файлов, что позволяет сократить число операций записи на диск. Кроме того, в этой базе данных хранятся пулы адресов и опций, которые изменяются статически. Они описаны в разделе “Настройка DHCP” на стр. 210.

Файл, хранящийся в оперативной памяти, и его резервная копия - это обычные текстовые файлы, которые можно редактировать. Формат файлов базы данных:

```
DF01
"ИД клиента" "0.0.0.0"
Состояние Начало_времени_выделения_адреса Время_выделения_адреса Конец_времени_выделения_адреса
  "IP-адрес сервера" "ИД класса" "ИД вендора" "Имя хоста" "Имя домена"
"ИД клиента" "0.0.0.0"
Состояние Начало_времени_выделения_адреса Время_выделения_адреса Конец_времени_выделения_адреса
  "IP-адрес сервера" "ИД класса" "ИД вендора" "Имя хоста" "Имя домена"
...
```

В первой строке указывается идентификатор версии файла: `DF01c`. За ней следуют записи, определяющие клиентов. Сервер читает данные, начиная со второй строки, и до конца файла. (Параметры, указанные здесь в кавычках, должны быть заключены в кавычки.)

"*ИД клиента***"**
: Идентификатор клиента.

"0.0.0.0"
: IP-адрес, присвоенный в настоящий момент серверу **DHCP**. Если адрес не присвоен, то это поле будет содержать значение "0.0.0.0".

Состояние
: Текущее состояние клиента. Средства поддержки протокола **DHCP** включают определенный набор состояний, которые сохраняются в базе данных **DHCP**. Номер, следующий за полем *Состояние*, - это значение состояния. Возможны следующие состояния:

(1) СВОБОДЕН

Обозначает доступные адреса. Это состояние может быть указано для клиента только в том случае, если ему не присвоен адрес. В данных, выдаваемых командами **dadmin** и **lssrc**, это состояние обозначается как `Свободен`.

(2) СВЯЗАН

Указывает, что адрес связан с клиентом и что данный адрес выделен клиенту на определенное время. В данных, выдаваемых командами **dadmin** и **lssrc**, это состояние обозначается как `Выделен`.

(3) ИСТЕК

Указывает, что адрес связан с клиентом, но лишь с информационной целью, как и в случае освобожденного адреса. Это состояние свидетельствует о том, что время выделения адреса клиенту истекло. Адрес с истекшим временем выделения могут быть присвоены другим клиентам только после того, как будут присвоены все свободные и освобожденные адреса. В данных, выдаваемых командами **dadmin** и **lssrc**, это состояние обозначается как `Истек`.

(4) ОСВОБОЖДЕН

Указывает, что адрес связан с клиентом, но лишь с информационной целью. В протоколе **DHCP** предполагается, что серверы **DHCP** сохраняют информацию об обслуживаемых клиентах. В основном это делается для того, чтобы по возможности предоставлять клиенту тот же адрес, который уже присваивался ему ранее. Это состояние указывает, что клиент освободил адрес. Этот адрес будет выделяться другим клиентам только в том случае, если других доступных адресов нет. В данных, выдаваемых командами **dadmin** и **lssrc**, это состояние обозначается как `Освобожден`.

(5) ЗАРЕЗЕРВИРОВАН

Указывает, что между клиентом и адресом установлена предварительная связь. Клиент уже разослал поисковое сообщение **DHCP**, а сервер **DHCP** ответил на него, но клиент еще не отправил запрос **DHCP** для окончательного присвоения адреса. В данных, выдаваемых командами **dadmin** и **lssrc**, это состояние обозначается как `Зарезервирован`.

(6) НЕВЕРНЫЙ

Обозначает адрес, который используется в сети, но не был выделен сервером **DHCP**. Кроме того, это состояние указывается для адресов, отклоненных клиентами. Это состояние неприменимо к клиентам. В данных, выдаваемых командами **dadmin** и **lssrc**, это состояние обозначается как `Занят` и `Неверный`, соответственно.

Начало_времени_выделения_адреса

Задает начало текущего времени выделения адреса (в секундах, начиная с 1 января 1970 г).

Время_выделения_адреса

Продолжительность времени выделения адреса (в секундах).

Конец_времени_выделения_адреса

Задается в том же формате, что и *Начало_времени_выделения_адреса*, но определяет конец времени выделения. В некоторых опциях настройки применяются разные значения значения начала и конца времени выделения адреса, которые могут переопределяться значениями, указанными в файле конфигурации. См. раздел “Синтаксис файла сервера DHCP для базы данных db_file” на стр. 227.

"*IP-адрес сервера***"**

IP-адрес сервера DHCP, которому принадлежит эта запись.

"*ИД класса***" "***ИД вендора***" "***Имя хоста***" "***Имя домена***"**

С помощью этих значений сервер определяет, какие опции нужно передать клиенту (эти значения хранятся в виде строк, заключенных в кавычки). Настройка этих параметров позволяет повысить производительность, так как список опций для клиентов может быть создан заранее при запуске сервера **DHCP**.

Контрольные файлы DHCP:

Синтаксис контрольных файлов не определен.

По контрольным файлам и резервным копиям сервер восстанавливает базу данных после сбоя или аварийного завершения работы системы, в ходе которого не удалось закрыть базу данных обычным образом. Контрольный файл, который записывался во время сбоя сервера, будет потерян. По умолчанию это следующие файлы:

/etc/db_file.cr
Применяется при обычной работе базы данных

/etc/db_file.crbk
Резервные копии базы данных

/etc/db_file.chkpt и /etc/db_file.chkpt2
Циклически применяемые контрольные файлы

Сервер **DHCP** поддерживает работу с несколькими нитями. Поддержка нитей обеспечивает высокую скорость работы с базой данных (включая операции сохранения). При получении запроса на сохранение существующий контрольный файл заменяется на следующий, текущая база данных записывается в файл резервной копии и создается новый файл сохранения. Каждая запись клиента заносится в файл и при этом устанавливается бит-переключатель, указывающий, что клиент должен использовать новый контрольный файл. После сохранения всех записей клиентов созданный файл закрывается, а файл резервной копии и старый контрольный файл удаляются. Таким образом, обработка запросов клиентов может продолжаться, причем информация об изменениях базы данных будет записываться в новый файл сохранения или новый контрольный файл в зависимости от того, была ли сохранена запись клиента.

Служба протокола DHCP:

Служба протокола **DHCP** поддерживает RFC 2131 и все еще совместима с RFC 1541. (Сервер может также обрабатывать опции, определенные в RFC 2132.) Отправляемая клиентам информация о конфигурации определяется с помощью базы данных.

Некоторые опции, задаваемые при настройке пулов адресов, влияют на состояние компьютера. Например, перед тем, как выделять адреса, сервер проверяет их с помощью команды ping. В настоящее время ожидания ответа можно задавать отдельно для каждого пула адресов.

Операции с нитями DHCP:

Третья часть сервера **DHCP** - это набор операций, которые, собственно, и обеспечивают его работу. Так как в сервере **DHCP** реализована поддержка нескольких нитей, то эти операции в действительности организованы в виде набора нитей, которые в определенное время выполняют нужные операции и обеспечивают правильную работу.

Первая нить, **main**, обрабатывает запросы SRC (такие как **startsrc**, **stopsrc**, **lssrc**, **traceson** и **refresh**). Кроме того, эта нить согласовывает все операции, влияющие на остальные нити, и обрабатывает сигналы. Пример:

- SIGHUP (-1) обновляет все базы данных в файле конфигурации.
- SIGTERM (-15) завершает работу сервера в нормальном режиме.

Другая нить, **dadmin**, обеспечивает взаимодействие программы клиента **dadmin** и сервера **DHCP**. Утилита **dadmin** служит для определения состояния и изменения базы данных, позволяя избежать редактирования файлов базы данных вручную. В предыдущих версиях сервера **DHCP** во время определения состояния сервера клиенты не могли получать адреса. С появлением нитей **dadmin** и **src** сервер может обрабатывать служебные запросы и запросы клиентов одновременно.

Следующая нить, **garbage**, включает таймеры, которые периодически выполняют очистку и сохранение базы данных, удаляют клиентов, для которых не заданы адреса, а также удаляют адреса, слишком долго находящиеся в зарезервированном состоянии. Значения этих таймеров можно изменить (за дополнительной информацией обратитесь к разделу “Настройка DHCP”). Остальные нити выполняют обработку пакетов. Их число можно настраивать. По умолчанию запускается 10 нитей. Каждая нить может обрабатывать запрос клиента **DHCP**. Число нитей для обработки пакетов зависит от предполагаемой нагрузки и мощности системы. Если помимо **DHCP** в системе работают другие службы, то не стоит запускать в ней 500 нитей.

Планирование DHCP

Для применения данного протокола администратор сети должен установить сервер **DHCP** и настроить промежуточных агентов BOOTP на тех узлах, где нет сервера **DHCP**. Заблаговременное планирование может сократить нагрузку на сеть, вызванную работой сервера **DHCP**.

Например, можно настроить один сервер для обработки запросов всех клиентов, но при этом все пакеты должны будут проходить через этот сервер. Если между двумя большими сетями есть единственный маршрутизатор, то лучше установить два сервера.

Другая особенность **DHCP**, которую следует учитывать - это создаваемая нагрузка на сеть. Например, если по умолчанию адрес выделяется меньше чем на двое суток, и компьютеры выключаются на выходные, то в понедельник утром резко возрастет объем данных, передаваемых по сети сервером **DHCP**. Несмотря на то, что эти данные не вызывают перегрузку сети, их необходимо учитывать при выборе количества серверов **DHCP** и их размещения.

После настройки функции **DHCP** клиенты могут не выполнять никаких дополнительных действий по настройке сети. Клиент **DHCP**, `dhcpcd`, читает файл `dhcpcd.ini`, в котором хранится информация о регистрации и другие параметры, необходимые для начала работы. После установки вам нужно будет выбрать способ настройки **TCP/IP**: несложная настройка вручную или **DHCP**. В случае **DHCP** нужно указать интерфейс и задать некоторые необязательные параметры. Для того чтобы интерфейс выбирался автоматически, укажите ключевое слово **any**. В этом случае `dhcpcd` будет применять тот активный интерфейс, который будет найден первым. Этот способ сокращает число операций, выполняемых пользователем.

Настройка DHCP

По умолчанию сервер **DHCP** считывает информацию из файла `/etc/dhcpsd.cnf`, в котором хранится исходная база данных параметров и адресов.

Команда запуска сервера содержится в файле `/etc/rc.tcpip`. Кроме того, сервер можно запустить с помощью инструмента SMIT или команд SRC. Клиента **DHCP** можно настроить с помощью Инструмента управления системой или путем редактирования файла ASCII.

Настройка сервера **DHCP** - это самая трудная часть настройки **DHCP** в сети. Вначале определите, в каких сетях будут размещаться клиенты **DHCP**. Каждой подсети соответствует пул адресов, который должен быть добавлен в базу данных сервера **DHCP**. Например:

```
database db_file
{

    subnet 9.3.149.0 255.255.255.0
      {  option 3 9.3.149.1 # Шлюз по умолчанию для клиентов данной сети
         option 6 9.3.149.2 # Клиенты сервера имен в данной сети
      }
    ... опции и другие контейнеры, добавляемые позже
}
```

В приведенном выше примере применяется подсеть `9.3.149.0` с маской `255.255.255.0`. В пул включены все адреса этой подсети, от 9.3.149.1 до 9.3.149.254. При необходимости в конце строки можно указать диапазон, а также задать в контейнере subnet оператор range или exclude. Стандартные способы настройки и определения адресов описаны в разделе “Известные опции файла сервера DHCP” на стр. 219.

Оператор базы данных `db_file` задает способ обработки этой части файла конфигурации. Комментарии начинаются с символа # (знак фунта стерлингов). Сервер **DHCP** игнорирует текст, стоящий после символа # до конца строки. С помощью каждой строки `option` сервер задает для клиента какое-либо действие. В разделе “Известные опции файла сервера DHCP” на стр. 219 описаны все поддерживаемые и известные опции. В разделе “Синтаксис файла сервера DHCP для общих операций сервера” на стр. 223 описаны способы задания опций, неизвестных серверу.

Если сервер не знает, как интерпретировать ту или иную опцию, то он передает ее клиенту с помощью методов, предусмотренных по умолчанию. Такая возможность позволяет серверу пересылать специальные параметры, которые не определены в RFC, но могут применяться некоторыми клиентами или в отдельных конфигурациях клиентов.

Файл конфигурации DHCP:

В файле конфигурации есть раздел адресов и раздел определения опций. В этих разделах есть контейнеры, содержащие опции и модификаторы, а также, возможно, другие контейнеры.

Контейнер (по существу, это способ группирования параметров) позволяет объединять клиентов в группы на основании идентификатора. Типы контейнеров: subnet, class, vendor и client. Контейнеры, определяемые пользователем, в настоящее время не поддерживаются. Клиент однозначно определяется своим идентификатором, что позволяет всегда точно обнаруживать его, например, при переносе в другую подсеть. Для описания клиента может применяться несколько контейнеров.

Опции - это идентификаторы, возвращаемые клиенту. Это может быть, например, применяемые по умолчанию адреса шлюза и сервера DNS.

Модификаторы - это одиночные операторы, которые изменяют некоторые параметры контейнера, например, время выделения адреса по умолчанию.

Контейнеры DHCP:

При получении запроса сервер **DHCP** анализирует пакет и на основании ключей идентификации определяет, какие нужно выбрать контейнеры, параметры и адреса.

Пример в разделе “Настройка DHCP” на стр. 210 показывает контейнер subnet. В нем вместо ключа идентификации применяется расположение клиента в сети. Если клиент находится в данной сети, то он попадает в этот контейнер.

Для идентификации клиентов в разных типах контейнеров применяются различные опции:

- Для определения подсети, в которой находится клиент, контейнер subnet использует поле **giaddr** или адрес целевого интерфейса.
- Контейнер class использует значение опции 77 (идентификатор класса пользователя).
- Контейнер vendor использует значение опции 60 (идентификатор класса вендора).
- Контейнер client использует значение опции 61 (идентификатор клиента) для клиентов **DHCP** и поле **chaddr** из пакета **BOOTP** для клиентов **BOOTP**.

Для всех контейнеров, за исключением subnet, можно задать шаблон для сравнения, например, регулярное выражение.

Существует также неявный контейнер *global*. Этот контейнер содержит все опции и модификаторы, которые не запрещены и не переопределены. Большинство контейнеров можно поместить в другие контейнеры в соответствии с областью видимости. С контейнерами могут быть связаны диапазоны адресов. С подсетями всегда связаны диапазоны адресов.

Ниже перечислены основные правила организации контейнеров и подконтейнеров:

- На глобальном уровне допустимы все контейнеры.
- Контейнеры subnet нельзя помещать в другие контейнеры.
- В контейнеры с ограничениями нельзя помещать обычные контейнеры того же типа. (Например, если контейнер содержит опцию, разрешающую только класс `Accounting`, то в него нельзя помещать контейнер, который содержит опцию, разрешающую применение всех классов, имена которых начинаются с буквы "a". Это недопустимо.)
- Контейнеры client с ограничениями не могут содержать вложенные контейнеры.

С помощью этих правил вы можете создавать иерархию контейнеров, в которой опции объединены в наборы, соответствующие конкретным клиентам или группам клиентов.

Каким образом организована обработка адресов и опций, если клиент входит в несколько контейнеров? Сервер **DHCP** получает сообщение, формирует запрос к базе данных (в данном случае - к файлу `db_file`) и получает список контейнеров. Контейнеры перечисляются в списке в порядке их вложенности и приоритета. Приоритет определяется как неявный иерархический уровень контейнера. Контейнеры с ограничениями имеют более высокий приоритет, чем обычные. Сортировка контейнеров выполняется в таком порядке: клиенты, классы, вендоры и подсети. В пределах одного типа контейнеры упорядочиваются по уровню вложенности. Созданный таким образом список упорядочивается от более конкретных объектов к менее конкретным. Например:

```
Subnet 1
--Class 1
--Client 1
Subnet 2
--Class 1
----Vendor 1
----Client 1
--Client 1
```

В примере есть две подсети `Subnet 1` и `Subnet 2`. Кроме того, определен один класс, `Class 1`, один вендор, `Vendor 1`, и один клиент, `Client 1`. `Class 1` и `Client 1` определены в нескольких контейнерах. Поскольку эти определения находятся в разных контейнерах, имена объектов могут совпадать, однако указанные в них значения могут различаться. Если клиент `Client 1` отправит сообщение серверу **DHCP** из подсети `Subnet 1` с указанием класса `Class 1`, определенного в списке опций этого клиента, то сервер **DHCP** создаст следующий список контейнеров:

`Subnet 1, Class 1, Client 1`

Контейнер, определенный наиболее точно, заносится в список последним. Для получения адреса список просматривается в обратном порядке до обнаружения первого доступного адреса. Затем список просматривается в прямом порядке (в соответствии с иерархией) для получения опций. По мере просмотра списка, новые значения опций переопределяют прежние значения, если в контейнере не задана опция **deny**. Поскольку класс `Class 1` и клиент `Client 1` находятся в одной и той же подсети `Subnet 1`, они упорядочиваются в соответствии с приоритетом контейнеров. Если сообщение будет получено от клиента с тем же именем, находящегося в подсети `Subnet 2`, то будет создан следующий список контейнеров:

`Subnet 2, Class 1, Client 1` (на уровне `Subnet 2`), `Client 1` (на уровне `Class 1`)

Первой в списке указывается подсеть `Subnet 2`, затем класс `Class 1`, затем клиент `Client 1` на уровне `Subnet 2` (так как этот клиент находится в иерархии на один уровень ниже). Иерархия подразумевает, что клиент, имя которого указано в первом операторе, определен менее конкретно, чем клиент `Client 1`, определенный в классе `Class 1` подсети `Subnet 2`.

Приоритет, определяемый по уровню вложенности, выше, чем приоритет самих контейнеров. Например, если тот же клиент отправит такое же сообщение, указав идентификатор вендора, то список контейнеров будет следующим:

`Subnet 2, Class 1, Vendor 1, Client 1` (на уровне `Subnet 2`), `Client 1` (на уровне `Class 1`)

Организация поиска на основании приоритета контейнера повышает эффективность, поскольку контейнеры client обеспечивают наиболее точный способ определения одного или нескольких клиентов. В контейнере class адреса определены менее конкретно, чем в контейнере client; в контейнере vendor адреса определены еще менее конкретно, а контейнер subnet содержит самые общие определения адресов.

Адреса и диапазоны адресов DHCP:

С контейнерами любого типа могут быть связаны диапазоны адресов. Контейнеры subnet обязательно имеют связанный диапазон адресов. Диапазон адресов контейнера должен быть подмножеством диапазона адресов родительского контейнера и не должен пересекаться с диапазонами адресов других контейнеров.

Например, если внутри подсети определен класс с диапазоном адресов, то этот диапазон должен быть подмножеством диапазона адресов подсети. Диапазон внутри этого контейнера класса не должен пересекаться с любыми другими диапазонами этого уровня.

Диапазоны адресов могут задаваться в строке контейнера. Для задания несмежных диапазонов адресов можно воспользоваться операторами range и exclude. Таким образом, если в подсети есть два диапазона по десять адресов, то имеет смысл указать эти диапазоны в операторе subnet, чтобы уменьшить объем памяти и избежать конфликтов адресов с другими клиентами.

Когда адрес выбран, все последующие контейнеры, содержащие диапазоны адресов, удаляются из списка вместе со своими дочерними контейнерами. Сетевые опции, заданные в удаленных контейнерах, недопустимы для адресов, не относящихся к данному контейнеру.

Опции файла конфигурации DHCP:

После первого просмотра списка и получения адресов для клиента создается набор опций.

В процессе выбора ранее определенные значения опций переопределяются новыми до тех пор, пока не встретится опция *deny* (запретить); при этом запрещенная опция удаляется из списка опций, отправляемых пользователю. Этот способ разрешает наследование опций родительских контейнеров и сокращает объем данных, которые нужно определять.

Модификаторы DHCP:

Модификаторы - это значения, которые изменяют некоторые параметры данного контейнера, например, права доступа или время выделения адреса.

Перед изменением контейнера определите пулы адресов и опций. Чаще всего используются модификаторы **leasetimedefault**, **supportBootp** и **supportUnlistedclients**.

leasetimedefault
: Задает время выделения адреса для клиента.

supportBootp
: Указывает, должен ли сервер отвечать на запросы клиентов **BOOTP**.

supportUnlistedclients
: Указывает, должен ли клиент для получения адреса быть определен в операторе client. Модификатор **supportUnlistedClients** может иметь значение **none**, **dhcp**, **bootp** или **both**. Он позволяет ограничить доступ к клиентам bootp и разрешает всем клиентам DHCP получать адреса.

Другие модификаторы описаны в разделе “Синтаксис файла сервера DHCP для базы данных db_file” на стр. 227.

Ведение протоколов DHCP:

Следующий шаг настройки после выбора модификаторов - это настройка протоколов.

Параметры протокола указываются в окне диалога, напоминающем окно настройки базы данных, но с ключевым словом **logging_info**. На этапе обучения настройке **DHCP** рекомендуется установить максимально подробное ведение протокола. Кроме того, весьма полезно настроить параметры протокола до начала работы с любыми другими файлами конфигурации, чтобы после инициализации подсистемы протокола информация об ошибках конфигурации заносилась в протокол. Для включения опций протокола укажите ключевое слово **logitem**, а для их отключения удалите это ключевое слово. Другие ключевые слова позволяют задавать имя файла протокола, его размер, а также число взаимозаменяемых файлов протоколов.

Опции сервера DHCP:

Последними нужно указать параметры работы сервера, в том числе число нитей для обработки пакетов, частоту сбора мусора и т.д.

Вот, например, две опции, определяемые сервером:

reservedTime
Указывает, на какое время резервируется адрес после отправки предложения клиенту **DHCP**.

reservedTimeInterval
Указывает, как часто сервер **DHCP** должен просматривать список адресов и определять, для каких адресов истекло время, заданное в параметре **reservedTime**.

Эти опции полезны в том случае, когда в сети есть несколько клиентов, которые рассылают сообщения для обнаружения серверов DHCP, а затем не рассылают запросы, либо отправляемые ими запросы не доходят до серверов. Эти параметры позволяют резервировать адреса только для тех клиентов, которые работают правильно.

Другая полезная опция, **SaveInterval**, указывает, как часто нужно сохранять информацию. В разделе “Синтаксис файла сервера DHCP для общих операций сервера” на стр. 223 описаны все опции сервера и соответствующие ключевые слова.

Замечания о производительности DHCP:

Важно понимать, что от ключевых слов конфигурации и от структуры файла конфигурации зависит распределение памяти и производительность сервера **DHCP**.

Во-первых, понимание механизма наследования опций от контейнеров-предков к потомкам позволяет сократить объем используемой памяти. В среде, в которой не поддерживаются не указанные в списке клиенты, администратор должен явно перечислить всех клиентов в файле. При указании списка опций для каждого отдельного клиента на хранение дерева конфигурации расходуется больше ресурсов памяти сервера, чем при наследовании опций от контейнера-предка к потомку (в роли предка может выступать подсеть, сеть или глобальные контейнеры). Следовательно, администратор должен проверить, не заданы ли некоторые опции многократно для нескольких клиентов, и нельзя ли такие опции определить один раз в родительском контейнере сразу для всех клиентов.

Кроме того, если применяются записи **logItem** INFO и TRACE, то при обработке каждого сообщения клиента **DHCP** в протокол заносится несколько сообщений. Добавление строк в файл протокола может привести к чрезмерному увеличению нагрузки; поэтому ограничение объема заносимой в протокол информации повышает производительность сервера **DHCP**. Вы можете динамически включать ведение протокола при возникновении ошибок в работе сервера **DHCP** с помощью команды SRC **traceson** или **dadmin**.

При выборе значения параметра **numprocessors** следует учесть размер сети сервера **DHCP**, значение параметра **pingTime db_file** и величину задержки распространения информации по сети. Поскольку перед тем как предложить клиенту адрес, входящий в пул сервера, нить обработчика пакетов генерирует сообщение ICMP Echo Request для выяснения состояния этого адреса, время обработки запроса клиента

DISCOVER напрямую зависит от времени ожидания ответа на это сообщение. В сущности, функции нити обработчика пакетов ограничиваются ожиданием либо ответа, либо наступления тайм-аута **pingTime**. Уменьшение значения **numprocessors** сокращает время ответа сервера за счет снижения числа повторных передач пакетов, сохраняя при этом все преимущества применения пробных пакетов на сервере.

Для повышения производительности выберите значение **pingTime**, исходя из значений задержки распространения для всех удаленных сетей, поддерживаемых этим сервером **DHCP**. Значение **numprocessors** следует выбрать на основе значения **pingTime** и размера сети. Выбор слишком маленького значения приведет к остановке всех нитей обработки пакетов. До тех пор пока сервер не получит хотя бы один ответ на эхо-запрос, все сообщения клиентов **DHCP** будут помещаться в очередь порта сервера. В результате сервер будет обрабатывать сообщения клиента большими группами, а не в виде непрерывного потока сообщений.

Если выбранное значение слишком мало, то все нити обработки пакетов будут ожидать ответа.

Для того чтобы такая ситуация не возникала, значение **numprocessors** должно быть больше числа сообщений DISCOVER, которые могут быть приняты за интервал **pingTime** во время пиковой нагрузки на сервер **DHCP**. Однако не следует указывать слишком большое значение **numprocessors**, так как это может резко увеличить расход ресурсов ядра системы на управление нитями.

Например, значения **numprocessors 5** и **pingTime 300** вызовут снижение производительности в среде, в которой на сервер поступает в среднем 10 сообщений DISCOVER за секунду, поскольку скорость их обработки в часы пик не будет превышать 5 сообщений за 3 секунды. В этом случае можно установить значения **numprocessors 20** и **pingTime 80**.

Настройка файла конфигурации DHCP:

При настройке файла конфигурации **DHCP** следует учитывать некоторые факторы.

Во многих сетях есть клиенты нескольких типов; например, к одной сети могут быть подключены компьютеры с разными операционными системами Windows, OS/2, Java™ OS и UNIX. Для каждого из них необходим свой идентификатор вендора (поле, с помощью которого серверу DHCP сообщается тип машины). Для клиентов Java OS clients and IBM Thin Client должны быть заданы дополнительные параметры, например, имя загрузочного файла и другие параметры конфигурации. Однако компьютеры с операционной системой Windows 95 неправильно обрабатывают параметры, заданные для систем Java.

В зависимости от назначения конкретного компьютера, в контейнер vendor могут быть включены различные опции. Например, разработчики приложений могут работать с клиентами данной операционной системы, сотрудники рекламного отдела - с клиентами OS/2, сотрудники отдела продаж - с клиентами Java OS и компьютерами IBM Thin Client, а работники бухгалтерии - с Windows 95. Для каждой группы пользователей необходимо задать свои параметры конфигурации (различные принтеры, серверы имен, веб-серверы по умолчанию и т.п.). В этом случае все перечисленные опции можно включить в контейнер vendor, так как каждая группа пользователей работает со своим типом компьютеров.

Если один и тот же тип компьютеров используется несколькими группами пользователей, то размещение опций в идентификаторе подчиненного класса позволит, например, сотрудникам отдела рекламы печатать на принтерах, недоступных для других отделов.

Примечание: Ниже приведен пример, представляющий собой фрагмент файла конфигурации. Комментарии к каждой строке начинаются со знака фунта стерлингов (#) и описывают назначение данной строки. каким

```
vendor "AIX_CLIENT"
{
# Нет специальных параметров, управляет объектами с помощью класса
}

vendor "OS/2 Client"
{
```

```
# Нет специальных параметров, управляет объектами с помощью класса
}

vendor "Windows 95"
{ option 44 9.3.150.3          # Сервер имен по умолчанию для NetBIOS
}

vendor "Java OS"
{ bootstrapserver 9.3.150.4    # Сервер TFTP по умолчанию для систем с OS Java
  option 67 "javaos.bin"       # Загрузочный файл для систем с OS Java
}

vendor "IBM Thin Client"
{ bootstrapserver 9.3.150.5    # Сервер TFTP по умолчанию для систем Thin Client
  option 67 "thinos.bin"       # Загрузочный файл по умолчанию для систем Thin Client
}

subnet 9.3.149.0 255.255.255.0
{ option 3 9.3.149.1           # Шлюз по умолчанию для подсети
  option 6 9.3.150.2           # Сервер имен для подсети
  class accounting 9.3.149.5-9.3.149.20
  {                            # Диапазон адресов для класса accounting: 9.3.149.5-9.3.149.20
          # Принтер для этой группы также в этом диапазоне, поэтому он исключен.
     exclude 9.3.149.15
     option 9 9.3.149.15       # Сервер LPR (сервер печати)
     vendor "Windows 95"
     {
     option 9 deny              # Windows 95 не поддерживает этот
                                # принтер, поэтому опция запрещена.
     }
  }
 . . .
}
```

DHCP и динамическая система имен доменов

Сервер **DHCP** поддерживает работу в среде **DDNS**.

Для применения сервера **DHCP** в среде DDNS необходимо задать динамическую область сервера DNS.

После настройки сервера DDNS необходимо решить, будет ли сервер **DHCP** обновлять записи типа A, записи PTR, записи обоих типов, или он вообще не должен обновлять записи. Ответ на этот вопрос зависит от того, может ли система клиента выполнять все или часть перечисленных операций.

- Если клиент берет на себя часть работы по обновлению, то можно разрешить ему обновление записей A, а серверу - обновление записей PTR.
- Если клиент может обновлять записи обоих типов, то вы можете отключить обновление записей сервером.
- Если клиент не может обновлять записи, то при настройке сервера укажите, что он должен обновлять записи обоих типов.

С помощью перечисленных ниже ключевых слов сервера **DHCP** можно задавать команды, которые должны выполняться при обновления.

updatedns
: (Не рекомендуется применять.) Задает команду, которая должна выполняться при обновлении всех записей. Она будет выполняться при обновлении как записей PTR, так и записей типа A.

updatednsA
: Задает команду для обновления записей типа A.

updatednsP
: Задает команду для обновления записей типа PTR.

Эти ключевые слова позволяют задать команды, которые должны выполняться сервером **DHCP** для обновления записей. Строки с ключевыми словами должны содержать четыре переменные %s (знак процента, буква s). Первая переменная %s - это имя хоста; вторая - имя домена; третья - IP-адрес; и четвертая - время выделения адреса. Эти переменные используются в качестве первых четырех параметров команды **dhcpaction**. Остальные два параметра команды **dhcpaction** указывают тип обновляемой записи (A, PTR, NONE или BOTH) и задают опцию обновления NIM (NIM или NONIM). Дополнительная информация о взаимодействии NIM и **DHCP** приведена в разделе “Рекомендации по DHCP и управлению сетевой установкой (NIM)” на стр. 278. Например:

```
updatednsA "/usr/sbin/dhcpaction '%s' '%s' '%s' '%s' '%s' A NONIM"
                    # Команда dhcpaction выполняется только для записей типа A
updatednsP "/usr/sbin/dhcpaction '%s' '%s' '%s' '%s' '%s' PTR NONIM"
                    # Команда выполняется только для записей типа PTR
updatedns "/usr/sbin/dhcpaction '%s' '%s' '%s' '%s' '%s' BOTH NIM"
                    # Команда выполняется для записей обоих типов, обновляется NIM
```

Сервер **DHCP** включает также набор ключевых слов для удаления записей DNS после освобождения адреса или завершения времени его выделения. Это следующие ключевые слова:

releasednsA
: Удаляет запись типа A.

releasednsP
: Удаляет запись типа PTR.

removedns
: Удаляет записи обоих типов.

Эти ключевые слова позволяют задать команды, которые должны выполняться сервером **DHCP** при освобождении адреса или истечении времени его выделения. Команда **dhcpremove** действует по тому же принципу, что и команда **dhcpaction**, однако принимает только три параметра:

1. IP-адрес, задаваемый в командной строке с помощью переменной %s
2. Тип удаляемой записи (A, PTR, NONE или BOTH).
3. Опция обновления NIM (NIM или NONIM).

Например:

```
releasednsA "/usr/sbin/dhcpremove '%s' A NONIM"
                    # Команда dhcpremove выполняется только для записей типа A
releasednsP "/usr/sbin/dhcpremove '%s' PTR NONIM"
                    # Команда выполняется только для записей типа PTR
removedns "/usr/sbin/dhcpremove '%s' BOTH NIM"
                    # Команда выполняется для записей обоих типов, обновляется NIM
```

Команды **dhcpaction** и **dhcpremove** проверяют полученные параметры, а затем вызывают команду **nsupdate**, которая в данном выпуске может работать с серверами этой операционной системы и серверами DDNS операционной системы OS/2. Более подробную информацию см. в описании команды **nsupdate**.

Если при обновлении имен **не требуется** обращаться к службе NIM, то в конфигурации сервера DHCP можно включить опцию обмена информацией о сокетах между демоном **DHCP** и командой **nsupdate**. Это повысит производительность и позволит повторно обновлять DNS после сбоя. Для включения этой опции нужно указать "nsupdate_daemon" в качестве первого параметра ключевого слова **updateDNSA**, **updateDNSP**, **releaseDNSA** или **releaseDNSP**. Параметры и флаги в таком режиме обновления те же, что и для команды **nsupdate**. Кроме того, для подстановки можно использовать следующие имена-переменные:

Элемент	Описание
$hostname	Это имя заменяется на имя хоста клиента в команде обновления DNS или на имя хоста, ранее связанное с клиентом, в команде удаления DNS.
$domain	Это имя заменяется на домен DNS в команде обновления записи DNS или на домен, ранее соответствовавший имени хоста клиента, в команде удаления записи DNS.
$ipadress	Это имя заменяется на IP-адрес, который необходимо связать с именем клиента **DHCP** или освободить.
$leasetime	Это имя заменяется на время выделения адреса (в секундах).
$clientid	Это имя заменяется на строку, содержащую идентификатор клиента **DHCP**, или на тип аппаратного обеспечения и аппаратный адрес клиента **BOOTP**.

Например:

```
updateDNSA "nsupdate_daemon -p 9.3.149.2 -h $hostname -d $domain
 -s"d;a;*;a;a;$ipaddress;s;$leasetime;3110400""

updateDNSP "nsupdate_daemon -p 9.3.149.2 -r $ipaddress
 -s"d;ptr;*;a;ptr;$hostname.$domain.;s;$leasetime;3110400""

releaseDNSA "nsupdate_daemon -p 9.3.149.2 -h $hostname -d $domain -s"d;a;*;s;1;3110400""

releaseDNSP "nsupdate_daemon -p 9.3.149.2 -r $ipaddress -s"d;ptr;*;s;1;3110400""
```

Более подробную информацию см. в описании команды **nsupdate**.

Для передачи имени хоста между сервером и клиентом теперь предусмотрены стратегии, задаваемые администратором. По умолчанию, имя хоста, возвращаемое клиенту и используемое для обновления DDNS, задается опцией 12, указанной в файле конфигурации сервера. Кроме того, имя хоста по умолчанию можно задать как предполагаемое имя хоста клиента, указав его с помощью опции 81 (опция DHCPDDNS) или опции 12 (опция HOSTNAME). Однако администратор может переопределять имя хоста по умолчанию с помощью ключевых слов **hostnamepolicy**, **proxyarec** и **appenddomain**. Эти опции и их параметры описаны в разделе “Синтаксис файла сервера DHCP для базы данных db_file” на стр. 227.

Совместимость DHCP с предыдущими версиями

Сервер **DHCP** распознает файлы конфигурации и базы данных более ранних версий, `dhcps.ar` и `dhcps.cr`.

Он анализирует старые файлы конфигурации и создает вместо них новые файлы базы данных. Старые базы данных автоматически преобразуются в новый файл. Сам файл конфигурации не преобразуется.

Модуль работы с базой данных сервера **DHCP**, `db_file`, может читать старый формат. Сервер **DHCP** распознает, когда контейнер базы данных не включен в файл конфигурации, и обрабатывает весь файл как файл параметров конфигурации сервера, параметров протокола и параметров базы данных `db_file`.

Примечание:

1. В данной версии файлов конфигурации не применяются некоторые устаревшие синтаксические конструкции, но они по-прежнему поддерживаются. Это касается следующих конструкций:
2. Контейнер network больше не используется. Для того, чтобы правильно описать сеть, необходимо преобразовать предложение network в соответствующий контейнер subnet с адресом подсети, маской подсети и диапазоном адресов. Если в контейнере network есть вложенные контейнеры subnet, то удалите ключевое слово network и соответствующие ему фигурные скобки, а затем добавьте маску подсети. Для начала работы с контейнером базы данных сгруппируйте все опции, имеющие отношение к сетям и правам доступа клиента, в один контейнер базы данных типа `db_file`.
3. Не используются ключевые слова **updatedns** и **removedns**. Они заменены на команды, работающие с записями типа A и PTR по-отдельности.
4. Вместо ключевых слов **clientrecorddb** и **addressrecorddb** используются **clientrecorddb** и **backupfile**, соответственно.

5. Ключевые слова **option sa** и **option ga** заменены на **bootstrapserver** и **giaddrfield**, соответственно. Дополнительная информация приведена в разделах “Синтаксис файла сервера DHCP для общих операций сервера” на стр. 223 и “Синтаксис файла сервера DHCP для базы данных db_file” на стр. 227.

Известные опции файла сервера DHCP

Здесь описаны известные опции файла сервера **DHCP**.

Примечание: Опции, для которых в приведенной ниже таблице указано, что их нельзя задавать (в колонке Можно указывать стоит Нет), можно указывать в файле конфигурации, но их значения будут заменены на допустимые. Более точные определения опций приведены в RFC 2132.

Номер опции	Тип данных по умолчанию	Можно указывать?	Описание/Использование
0	Нет	Нет	Сервер при необходимости дополняет поле опции.
1	IP-адрес в десятичном формате с точками	Нет	Маска подсети, из которой получен адрес.
61 см	32-разрядное целое число	Да	Задает разницу во времени (в секундах) в подсети клиента по сравнению с Универсальным скоординированным временем (UTC).
3	Один или несколько IP-адресов в десятичном формате с точками	Да	Список IP-адресов шлюзов по умолчанию.
4	Один или несколько IP-адресов в десятичном формате с точками	Да	Список IP-адресов серверов времени.
5	Один или несколько IP-адресов в десятичном формате с точками	Да	Список IP-адресов серверов имен.
6	Один или несколько IP-адресов в десятичном формате с точками	Да	Список IP-адресов серверов DNS.
7	Один или несколько IP-адресов в десятичном формате с точками	Да	Список IP-адресов серверов протоколов.
8	Один или несколько IP-адресов в десятичном формате с точками	Да	Список IP-адресов серверов cookie.
9	Один или несколько IP-адресов в десятичном формате с точками	Да	Список IP-адресов серверов LPR.
10	Один или несколько IP-адресов в десятичном формате с точками	Да	Список IP-адресов серверов Impress.
11	Один или несколько IP-адресов в десятичном формате с точками	Да	Список IP-адресов серверов размещения ресурсов.
12	Строка ASCII	Да	Имя хоста клиента.
13	16-разрядное целое число без знака	Да	Размер загрузочного файла.
14	Строка ASCII	Да	Путь к файлу контрольного дампа.
15	Строка ASCII	Да	Имя домена DNS по умолчанию.
16	IP-адрес	Да	Адрес сервера Swap.
17	Строка ASCII	Да	Путь к корневому каталогу по умолчанию.
18	Строка ASCII	Да	Путь к расширениям для клиента.
19	Yes, No, True, False, 1, 0	Да	Указывает, применяется ли пересылка IP-дейтаграмм.
20	Yes, No, True, False, 1, 0	Да	Указывает, применяется ли маршрутизация удаленного источника.
21	Одна или несколько пар IP-адресов в десятичном формате с точками, в виде *адрес*:*адрес*	Да	Стратегия фильтрации для IP-адресов.
22	16-разрядное целое число без знака	Да	Максимальный размер фрагмента дейтаграммы.
23	8-разрядное целое число без знака	Да	Ограничение на число участков для IP-дейтаграмм (TTL).
24	32-разрядное целое число без знака	Да	Время жизни MTU маршрута в секундах.
25	Список из одного или нескольких 16-разрядных целых чисел без знака	Да	Таблица значений MTU маршрута. Определяет множество значений, задающих размеры MTU, применяемые при определении MTU маршрута.
26	16-разрядное целое число без знака	Да	Размер MTU для интерфейса получателя.
27	Yes, No, True, False, 1, 0	Да	Указывает, все ли подсети локальные.

Номер опции	Тип данных по умолчанию	Можно указывать?	Описание/Использование
28	IP-адрес в десятичном формате с точками	Да	Задает адрес оповещения для интерфейса.
29	Yes, No, True, False, 1, 0	Да	Указывает, следует ли применять функцию определения маски сети ICMP.
30	Yes, No, True, False, 1, 0	Да	Указывает, должен ли клиент предоставлять маску сети ICMP.
31	Yes, No, True, False, 1, 0	Да	Указывает, следует ли применять сообщения ICMP для обнаружения маршрутизаторов.
32	IP-адрес в десятичном формате с точками	Да	Адрес для обращения к маршрутизатору.
33	Одна или несколько пар IP-адресов, в виде *адрес*:*адрес*	Да	Каждая пара адресов соответствует статическому маршруту.
34	Yes/No, True/False, 1/0	Да	Указывает, должна ли применяться инкапсуляция концевиков.
35	32-разрядное целое число без знака	Да	Значение тайм-аута для кэш-памяти ARP.
36	Yes/No, True/False, 1/0	Да	Указывает, должна ли применяться инкапсуляция данных Ethernet.
37	8-разрядное целое число без знака	Да	Ограничение на число участков для пакета TCP (TTL).
38	32-разрядное целое число без знака	Да	Интервал отправки контрольных пакетов **TCP**.
39	Yes/No, True/False, 1/0	Да	Указывает, нужно ли отправлять контрольные пакеты TCP.
40	Строка ASCII	Да	Домен NIS по умолчанию.
41	Один или несколько IP-адресов в десятичном формате с точками	Да	IP-адреса серверов NIS.
42	Один или несколько IP-адресов в десятичном формате с точками	Да	IP-адреса серверов NTP.
43	Шестнадцатеричная строка в формате hex "*цифры*", hex"*цифры*" или 0x*цифры*	Да, но фактически задается только с контейнером vendor	Встроенный контейнер опций для контейнера vendor.
44	Один или несколько IP-адресов в десятичном формате с точками	Да	Список IP-адресов серверов NetBIOS.
45	Один или несколько IP-адресов в десятичном формате с точками	Да	Список IP-адресов серверов рассылки дейтаграмм NetBIOS.
46	8-разрядное целое число без знака	Да	Тип узла NetBIOS.
47	Шестнадцатеричная строка в формате hex "*цифры*", hex"*цифры*" или 0x*цифры*	Да	Область NetBIOS.
48	Один или несколько IP-адресов в десятичном формате с точками	Да	IP-адреса серверов шрифтов X Windows.
49	Один или несколько IP-адресов в десятичном формате с точками	Да	Диспетчер дисплея X Windows.
50	Нет	Нет	IP-адрес, запрашиваемый клиентом.
51	32-разрядное целое число без знака	Да	Время выделения для возвращаемого адреса. По умолчанию сервер **DHCP** использует ключевое слово **leasetimedefault**, однако непосредственное задание опции 51 переопределяет его.
52	Нет	Нет	Переопределение опции. Клиент использует эту опцию, чтобы указать, что у полей **sname** и **file** пакета **BOOTP** могут быть дополнительные опции.
53	Нет	Нет	Используется сервером **DHCP** или клиентом для указания типа сообщения **DHCP**.
54	Нет	Нет	Используются сервером **DHCP** или клиентом для указания адреса сервера или сервера, на который направляется сообщение.
55	Нет	Нет	Используется клиентом **DHCP** для указания требуемых параметров.
56	Строка ASCII	Да	Строка, которую сервер **DHCP** отправляет клиенту. В общем случае может использоваться сервером и клиентом **DHCP** для сообщения о неполадках.
57	Нет	Нет	С помощью этой опции клиент **DHCP** сообщает серверу **DHCP** максимальный размер пакетов, которые он может принимать.

Номер опции	Тип данных по умолчанию	Можно указывать?	Описание/Использование
58	32-разрядное целое число без знака	Да	Интервал времени (в секундах), в течение которого клиент должен отправить запрос на обновление.
59	32-разрядное целое число без знака	Да	Интервал времени (в секундах), в течение которого клиент должен отправить запрос на повторное связывание.
60	Нет	Нет	Эта опция используется клиентом **DHCP** для указания типа вендора. Сервер **DHCP** использует это поле для сравнения с контейнерами vendor.
61	Нет	Нет	Используется для однозначной идентификации клиента **DHCP**. Сервер **DHCP** использует это поле для сравнения с контейнерами client.
66	Строка ASCII	Да	Определяет имя сервера **TFTP**. Это имя хоста, которое используется вместо имени, указанного в поле **siaddr**, в том случае, если клиент поддерживает данную опцию.
67	Строка ASCII	Да	Задает имя загрузочного файла. Может использоваться вместо ключевого слова **bootfile**, которое помещает файл в поле **filename** пакета.
68	Один или несколько IP-адресов в десятичном формате с точками или NONE	Да	Адреса домашних агентов.
69	Один или несколько IP-адресов в десятичном формате с точками	Да	Серверы SMTP по умолчанию.
70	Один или несколько IP-адресов в десятичном формате с точками	Да	Серверы POP3 по умолчанию.
71	Один или несколько IP-адресов в десятичном формате с точками	Да	Серверы NNTP по умолчанию.
72	Один или несколько IP-адресов в десятичном формате с точками	Да	Серверы WWW по умолчанию.
73	Один или несколько IP-адресов в десятичном формате с точками	Да	Серверы Finger по умолчанию.
74	Один или несколько IP-адресов в десятичном формате с точками	Да	Серверы IRC по умолчанию.
75	Один или несколько IP-адресов в десятичном формате с точками	Да	Серверы Street Talk по умолчанию.
76	Один или несколько IP-адресов в десятичном формате с точками	Да	Серверы поддержки каталога Street Talk по умолчанию.
77	Строка ASCII	Да	Идентификатор класса пользователя. Сервер **DHCP** использует это поле для сравнения с контейнерами class.
78	Обязательный байт. Один или несколько IP-адресов в десятичном формате с точками.	Да	Опция агента каталога SLP задает список IP-адресов агентов каталога
79	Обязательный байт и строка ASCII	Да	Строка ASCII задает список разделенных запятыми областей, для применения которых настроен агент SLP
81	Строка ASCII плюс другие элементы.	Нет	С помощью этой опции клиент DHCP определяет стратегию, которую сервер **DHCP** должен применять в отношении DDNS.
85	Один или несколько IP-адресов в десятичном формате с точками	Да	Опция сервера NDS задает один или нескольких серверов NDS, с которыми связывается клиент для обращения к базе данных DNS. Серверы указываются в порядке предпочтения.
86	Строка ASCII	Да	Опция имени дерева NDS задает дерево NDS, к которому обращается клиент.
87	Строка ASCII	Да	Опция контекста NDS задает исходный контекст NDS, применяемый клиентом.
93	Нет	Нет	Архитектура системы клиента DHCP.
94	Нет	Нет	Идентификатор сетевого интерфейса клиента DHCP.

117	Одно или нескольких 16-разрядных целых чисел без знака	Да	Опция поиска службы имен задает предпочтительную последовательность целочисленных кодов опций для служб имен. Например:
			`Name Services value`
			`Domain Name Server Option 6`
			`NIS Option 41`

118	Один IP-адрес в десятичном формате с точками	Нет	Опция выбора подсети отправляет клиентом на сервер dhcp server для выделения IP-адреса из указанной подсети.

Номер опции	Тип данных по умолчанию	Можно указывать?	Описание/Использование
255	Нет	Нет	Эта опция используется сервером и клиентом DHCP для указания на конец списка опций.

Опции контейнера вендора среды исполнения перед загрузкой

При работе с клиентом PXE сервер **DHCP** передает следующую опцию серверу BINLD для настройки этого сервера.

Опция	Тип данных по умолчанию	Можно указывать?	Описание
7	адрес в десятичном формате	Да	IP-адрес для многоцелевой рассылки. IP-адрес, по которому отправляются запросы для обнаружения загрузочного сервера.

Ниже приведен пример применения этой опции:

```
pxeservertype       proxy_on_dhcp_server

Vendor pxeserver
{
     option   7    9.3.4.68
}
```

В этом примере сервер **DHCP** информирует клиента, что сервер proxy работает на том же компьютере и принимает запросы клиентов через порт 4011. В данном примере необходим контейнер vendor, так как сервер BINLD отправляет сообщение INFORM/REQUEST через порт 67 с опцией 60, равной "PXEServer." В ответ сервер **DHCP** отправляет IP-адрес рассылки, на который серверу BINLD будут поступать запросы клиентов PXE.

В приведенном ниже примере сервер **dhcpsd** сообщает клиенту PXE имя загрузочного файла, либо отправляет ему набор подопций с информацией о сервере BINLD. Для создания списка загрузочных файлов для клиентов с определенной архитектурой и версией операционной системы из заданного диапазона применяется ключевое слово **pxeboofile**.

```
pxeservertype     dhcp_pxe_binld

subnet default
{
     vendor pxe
     {
         option 6 2    # Выключить прием многоцелевых пакетов
         option 8 5 4 10.10.10.1 12.1.1.15 12.5.5.5 12.6.6.6\
                  2 2 10.1.1.10 9.3.4.5 1 1 10.5.5.9\
                  1 1 9.3.149.15\
                  4 0
         option 9 5 "WorkSpace On Demand" 2 "Intel"\
                  1 "Microsoft Windows NT" 4 "NEC ESMPRO"
         option 10 2 "Press F8 to View Menu"
     }
     vendor pxeserver
     {
         option 7 239.0.0.239
     }

}

subnet 9.3.149.0 255.255.255.0
{
     option 3   9.3.149.1
     option 6   9.3.149.15
```

```
    vendor pxe
    {
      option    6    4    # в пакете с ответом указывается загрузочный файл
      pxebootfile   1   2   1   os2.one
      pxebootfile   2   2   1   aix.one
    }
}
```

В строках option контейнера PXE указываются действия, которые должен выполнить клиент. В разделе “Опции контейнера вендора PXE” на стр. 306 описаны все поддерживаемые и известные опции PXE.

Синтаксис файла сервера DHCP для общих операций сервера

Здесь приведены сведения о синтаксисе файла **DHCP** для общих операций сервера и допустимые значения для каждого поля.

Примечание: Используемые в таблицы единицы времени (*единицы_времени*) указывать необязательно. Они применяются в качестве модификаторов значений времени. По умолчанию время указывается в минутах. Допустимо указывать время в секундах (1), минутах (60), часах (3600), сутках (86400), неделях (604800), месяцах (2392000) и годах (31536000). В скобках указан коэффициент, на который нужно умножить указанное значение времени, *n*, чтобы перевести его в секунды.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умол-чанию	Описание
database	database *тип*	Да	Нет	Основной контейнер, содержащий определения для пулов адресов, опций и операторов, задающих уровень доступа клиентов. *Тип* - это имя модуля, который должен быть загружен для обработки этой части файла. В текущей версии поддерживается только значение **db_file**.
logging_info	logging_info	Да	Нет	основной контейнер, определяющий параметры ведения протоколов.
logitem	logitem NONE	Нет	По умол-чанию все запре-щены.	Задает уровень ведения протокола. Можно указать несколько строк.
logitem	logitem SYSERR	Нет	По умол-чанию все запре-щены.	Задает уровень ведения протокола. Можно указать несколько строк.
logitem	logitem OBJERR	Нет	По умол-чанию все запре-щены.	Задает уровень ведения протокола. Можно указать несколько строк.
logitem	logitem PROTOCOL	Нет	По умол-чанию все запре-щены.	Задает уровень ведения протокола. Можно указать несколько строк.
logitem	logitem PROTERR	Нет	По умол-чанию все запре-щены.	Задает уровень ведения протокола. Можно указать несколько строк.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
logitem	logitem WARN	Нет	По умолчанию все запрещены.	Задает уровень ведения протокола. Можно указать несколько строк.
logitem	logitem WARNING	Нет	По умолчанию все запрещены.	Задает уровень ведения протокола. Можно указать несколько строк.
logitem	logitem CONFIG	Нет	По умолчанию все запрещены.	Задает уровень ведения протокола. Можно указать несколько строк.
logitem	logitem EVENT	Нет	По умолчанию все запрещены.	Задает уровень ведения протокола. Можно указать несколько строк.
logitem	logitem PARSEERR	Нет	По умолчанию все запрещены.	Задает уровень ведения протокола. Можно указать несколько строк.
logitem	logitem ACTION	Нет	По умолчанию все запрещены.	Задает уровень ведения протокола. Можно указать несколько строк.
logitem	logitem ACNTING	Нет	По умолчанию все запрещены.	Задает уровень ведения протокола. Можно указать несколько строк.
logitem	logitem STAT	Нет	По умолчанию все запрещены.	Задает уровень ведения протокола. Можно указать несколько строк.
logitem	logitem TRACE	Нет	По умолчанию все запрещены.	Задает уровень ведения протокола. Можно указать несколько строк.
logitem	logitem RTRACE	Нет	По умолчанию все запрещены.	Задает уровень ведения протокола. Можно указать несколько строк.
logitem	logitem START	Нет	По умолчанию все запрещены.	Задает уровень ведения протокола. Можно указать несколько строк.
numLogFiles	numLogFiles *n*	Нет	0	Указывает, сколько файлов протоколов нужно создать. Каждый последующий файл протокола создается после заполнения предыдущего. *n* - число создаваемых файлов.
logFileSize	logFileSize *n*	Нет	0	Задает размер каждого файла протокола в блоках по 1024 байта.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умол-чанию	Описание
logFileName	logFileName *путь*	Нет	Нет	Задает путь к первому файлу протокола. Имя файла протокола имеет вид *имя_файла* или *имя_файла.расш*. Имя следующего файла протокола создается на основе базового *имени_файла*, к которому добавляется номер, либо этот номер указывается вместо расширения. Например, если первому файлу присвоено имя `file`, то именем следующего файла будет `file01`. Если имя первого файла - `file.log`, то следующему файлу будет присвоено имя `file.01`.
CharFlag	charflag yes	Нет	true	Не применяется сервером **DHCP** для данной операционной системы; сервер **DHCP** для OS/2 применяет эту опцию для создания окон отладки.
CharFlag	charflag true	Нет	true	Не применяется сервером **DHCP** для данной операционной системы; сервер **DHCP** для OS/2 применяет эту опцию для создания окон отладки.
CharFlag	charflag false	Нет	true	Не применяется сервером **DHCP** для данной операционной системы; сервер **DHCP** для OS/2 применяет эту опцию для создания окон отладки.
CharFlag	charflag no	Нет	true	Не применяется сервером **DHCP** для данной операционной системы; сервер **DHCP** для OS/2 применяет эту опцию для создания окон отладки.
StatisticSnapShot	StatisticSnapShot *n*	Нет	-1, never	Задает интервал времени в секундах между операциями записи статистической информации в файл протокола.
UsedIpAddressExpireInterval	UsedIpAddressExpireInterval *n единицы_времени*	Нет	-1, never	Указывает, с какой периодичностью проверяется правильность адресов, имеющих состояние НЕВЕРНЫЙ.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умол-чанию	Описание
leaseExpireInterval	leaseExpireInterval *n единицы_времени*	Нет	900 seconds	Задает периодичность, с которой будет проверяться завершение времени выделения адресов, находящихся в состоянии СВЯЗАН. Если время выделения адреса закончилось, то адресу присваивается состояние ИСТЕК.
reservedTime	reservedTime *n единицы_времени*	Нет	-1, never	Указывает, как долго адрес может находиться в состоянии ЗАРЕЗЕРВИРОВАН перед тем, как он будет переведен в состояние СВОБОДЕН.
reservedTimeInterval	reservedTimeInterval *n единицы_времени*	Нет	900 seconds	Указывает, с какой периодичностью проверяется адрес, находящийся в состоянии ЗАРЕЗЕРВИРОВАН, перед тем, как он будет переведен в состояние СВОБОДЕН.
saveInterval	saveInterval *n единицы_времени*	Нет	3600 seconds	Указывает, с какой периодичностью сервер **DHCP** должен проводить принудительное сохранение открытых баз данных. Для серверов, работающих с высокой нагрузкой, это значение должно составлять от 60 до 120 секунд.
clientpruneintv	clientpruneintv *n единицы_времени*	Нет	3600 seconds	Указывает, с какой периодичностью сервер **DHCP** удаляет из базы данных клиентов, не связанных с какими-либо адресами (т.е. клиентов с состоянием НЕИЗВЕСТНО). Это сокращает объем памяти, используемой сервером **DHCP**.
numprocessors	numprocessors *n*	Нет	10	Указывает, сколько должно быть создано нитей для обработки пакетов. Минимальное значение равно единице.
userObject	userObject *имя_объекта*	Да	Нет	Указывает, что сервер должен загружать пользовательский общий объект и вызывать программы из этого объекта при каждом взаимодействии с клиентами **DHCP**. Этот объект хранится в каталоге `/usr/sbin` под именем `имя_объекта.dhcpo`. См. описание API пользовательских расширений сервера DHCP.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умол-чанию	Описание
pxeservertype	pxeservertype *тип_сервера*	Нет	dhcp_only	Задает тип сервера **dhcpd**. *Тип_сервера* может принимать следующие значения: **dhcp_pxe_binld** **DHCP** выполняет функции **dhcpsd**, **pxed** и **bindl**. **proxy_on_dhcp_server** **DHCP** сообщает клиенту PXE номер порта сервера proxy на том же компьютере. Значение по умолчанию - `dhcp_only`, то есть **dhcpsd** не поддерживает клиентов PXE.
supportsubnetselection	supportsubnetselection **global** supportsubnetselection **subnetlevel** supportsubnetselection **no**	Нет	Нет	Указывает, поддерживает ли сервер dhcp опцию 118 (опцию выбора подсети) в пакетах клиентов DISCOVER и REQUEST. **global**: все подсети в файле конфигурации поддерживают опцию 118. **subnetlevel**: опция поддерживается подсетями, в конфигурации которых задано ключевое слово supportoption118. **no**: не поддерживает опцию 118.

Синтаксис файла сервера DHCP для базы данных db_file

Синтаксис базы данных `db_file` имеет следующие параметры.

Примечание:

1. Используемые в таблицы единицы времени (*единицы_времени*) указывать необязательно. Они применяются в качестве модификаторов значений времени. По умолчанию время указывается в минутах. Допустимо указывать время в секундах (1), минутах (60), часах (3600), сутках (86400), неделях (604800), месяцах (2392000) и годах (31536000). В скобках указан коэффициент, на который нужно умножить указанное значение времени, *n*, чтобы перевести его в секунды.
2. Элементы, описанные в одном контейнере, могут быть переопределены во вложенном контейнере. Например, можно определить клиентов **BOOTP** на глобальном уровне, но разрешить их работу только в конкретной подсети, указав ключевое слово supportBootp в обоих контейнерах.
3. В контейнерах client, class и vendor поддерживаются регулярные выражения. Если в контейнере class или vendor указана заключенная в кавычки строка, в которой после открывающей кавычки стоит символ !, то остаток строки будет обрабатываться как регулярное выражение. В контейнере client регулярные выражения можно указывать в полях hwtype и hwaddr. В обоих полях можно задавать строку следующего формата:

   ```
   десятичное_число-данные
   ```

Если десятичное_число равно нулю, то данные представляют строку ASCII. Если указано любое другое число, то данные представляют собой набор шестнадцатеричных цифр.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
subnet	subnet default	Да	Нет	Задает подсеть, с которой не связан никакой диапазон адресов. Эта подсеть применяется сервером для ответа на пакеты INFORM/REQUEST, полученные от клиента, адрес которого не относится ни к одному из контейнеров subnet.
subnet	subnet *ид подсети маска*	Да	Нет	Определяет подсеть и пул адресов. Если в строке не задан диапазон адресов и адреса не изменены в контейнере с помощью оператора range или exclude, то предполагается, что пул включает все адреса. Необязательный параметр диапазона представляет собой пару IP-адресов в десятичном формате с точками, разделенных дефисом. Можно (но не обязательно) указывать метку и приоритет. Они используются виртуальными подсетями для идентификации и упорядочения подсетей в виртуальной подсети. Метка и приоритет разделяются двоеточием. Эти контейнеры могут использоваться только на глобальном уровне или на уровне контейнера базы данных.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
subnet	subnet *ид подсети маска диапазон*	Да	Нет	Определяет подсеть и пул адресов. Если в строке не задан диапазон адресов и адреса не изменены в контейнере с помощью оператора range или exclude, то предполагается, что пул включает все адреса. Необязательный параметр диапазона представляет собой пару IP-адресов в десятичном формате с точками, разделенных дефисом. Можно (но не обязательно) указывать метку и приоритет. Они используются виртуальными подсетями для идентификации и упорядочения подсетей в виртуальной подсети. Метка и приоритет разделяются двоеточием. Эти контейнеры могут использоваться только на глобальном уровне или на уровне контейнера базы данных.
subnet	subnet *ид подсети маска метка*:*приоритет*	Да	Нет	Определяет подсеть и пул адресов. Если в строке не задан диапазон адресов и адреса не изменены в контейнере с помощью оператора range или exclude, то предполагается, что пул включает все адреса. Необязательный параметр диапазона представляет собой пару IP-адресов в десятичном формате с точками, разделенных дефисом. Можно (но не обязательно) указывать метку и приоритет. Они используются виртуальными подсетями для идентификации и упорядочения подсетей в виртуальной подсети. Метка и приоритет разделяются двоеточием. Эти контейнеры могут использоваться только на глобальном уровне или на уровне контейнера базы данных.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
subnet	subnet *ид подсети маска диапазон метка*:*приоритет*	Да	Нет	Определяет подсеть и пул адресов. Если в строке не задан диапазон адресов и адреса не изменены в контейнере с помощью оператора range или exclude, то предполагается, что пул включает все адреса. Необязательный параметр диапазона представляет собой пару IP-адресов в десятичном формате с точками, разделенных дефисом. Можно (но не обязательно) указывать метку и приоритет. Они используются виртуальными подсетями для идентификации и упорядочения подсетей в виртуальной подсети. Метка и приоритет разделяются двоеточием. Эти контейнеры могут использоваться только на глобальном уровне или на уровне контейнера базы данных.
subnet	subnet *ид подсети диапазон*	Да	Нет	Определяет подсеть, которая входит в контейнер сети. Задает диапазон адресов. Если не задан диапазон, то считается, что в подсеть входят все адреса. Маска подсети определяется родительским контейнером сети. **Примечание:** Задавать подсеть таким образом не рекомендуется.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
option	option *номер данные* ...	Нет	Нет	Задает опцию для отправки клиенту, или, в случае запрещения, опцию, отправка которой запрещена. Опция option * deny означает, что все параметры, не определенные в данном контейнере, не должны возвращаться клиенту. Опция *номер*deny запрещает только указанную опцию. *Номер* - это 8-разрядное целое число без знака. *Данные* - это данные для опции (см. выше), заключенная в кавычки строка ASCII, либо шестнадцатеричные данные в одном из форматов: 0x*шестн._число* или hex "*шестн._число*" или hex"*шестн._число*". Если опция находится в контейнере vendor, то она будет включена вместе с другими опциями в опцию 43.
option	option *номер*deny	Нет	Нет	Задает опцию для отправки клиенту, или, в случае запрещения, опцию, отправка которой запрещена. Опция option * deny означает, что все параметры, не определенные в данном контейнере, не должны возвращаться клиенту. Опция *номер*deny запрещает только указанную опцию. *Номер* - это 8-разрядное целое число без знака. *Данные* - это данные для опции (см. выше), заключенная в кавычки строка ASCII, либо шестнадцатеричные данные в одном из форматов: 0x*шестн._число* или hex "*шестн._число*" или hex"*шестн._число*". Если опция находится в контейнере vendor, то она будет включена вместе с другими опциями в опцию 43.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
option	option * deny	Нет	Нет	Задает опцию для отправки клиенту, или, в случае запрещения, опцию, отправка которой запрещена. Опция option `*` `deny` означает, что все параметры, не определенные в данном контейнере, не должны возвращаться клиенту. Опция *номер*deny запрещает только указанную опцию. *Номер* - это 8-разрядное целое число без знака. *Данные* - это данные для опции (см. выше), заключенная в кавычки строка ASCII, либо шестнадцатеричные данные в одном из форматов: 0x*шестн._число* или hex "*шестн._число*" или hex"*шестн._число*". Если опция находится в контейнере vendor, то она будет включена вместе с другими опциями в опцию 43.
exclude	exclude *IP-адрес*	Нет	Нет	Изменяет диапазон адресов контейнера, в котором указан оператор exclude. На глобальном уровне и на уровне контейнера базы данных оператор exclude недопустим. Оператор exclude удаляет заданные адреса или диапазон адресов из диапазона, указанного для данного контейнера. Оператор exclude позволяет создавать для подсетей и других контейнеров не смежные диапазоны адресов.
exclude	exclude *адрес-адрес*	Нет	Нет	Изменяет диапазон адресов контейнера, в котором указан оператор exclude. На глобальном уровне и на уровне контейнера базы данных оператор exclude недопустим. Оператор exclude удаляет заданные адреса или диапазон адресов из диапазона, указанного для данного контейнера. Оператор exclude позволяет создавать для подсетей и других контейнеров не смежные диапазоны адресов.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
range	range *IP-адрес*	Нет	Нет	Изменяет диапазон адресов контейнера, в котором указан оператор range. На глобальном уровне и на уровне контейнера базы данных оператор range недопустим. Если в строке описания контейнера не задан диапазон адресов и оператор range стоит в этом контейнере первым, то диапазон адресов контейнера будет определяться именно этим оператором range. Диапазоны, определяемые всеми остальными операторами range, стоящими после первого оператора range, или всеми операторами range, задающими диапазоны в строке описания контейнера, добавляются к текущему диапазону. С помощью оператора range к диапазону можно добавить отдельный адрес или набор адресов. Диапазон должен размещаться внутри группы адресов, определенной для контейнера подсети.
range	range *адрес-адрес*	Нет	Нет	Изменяет диапазон адресов контейнера, в котором указан оператор range. На глобальном уровне и на уровне контейнера базы данных оператор range недопустим. Если в строке описания контейнера не задан диапазон адресов и оператор range стоит в этом контейнере первым, то диапазон адресов контейнера будет определяться именно этим оператором range. Диапазоны, определяемые всеми остальными операторами range, стоящими после первого оператора range, или всеми операторами range, задающими диапазоны в строке описания контейнера, добавляются к текущему диапазону. С помощью оператора range к диапазону можно добавить отдельный адрес или набор адресов. Диапазон должен размещаться внутри группы адресов, определенной для контейнера подсети.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
клиент	client *типаппаратный-адрес* NONE	Да	Нет	Определяет контейнер клиента, запрещающий выделение адреса клиенту с указанным *типом аппаратуры* и *аппаратным адресом*. Если параметр *тип аппаратуры* равен 0, то в параметре *аппаратный адрес* должна быть задана строка ASCII. В остальных случаях *тип аппаратуры* - это тип аппаратного обеспечения клиента, а *аппаратный адрес* - аппаратный адрес клиента. Если *аппаратный адрес* - это строка, то она может быть заключена в кавычки. Если *аппаратный адрес* - это шестнадцатеричная строка, то он может быть записан в виде 0x*шестн._число* или hex *шестн._число*. *Диапазон* разрешает выделять адрес из указанного *диапазона* клиенту с указанным *типом аппаратуры* и *аппаратным адресом*. Для сравнения нескольких клиентов должны использоваться регулярные выражения.
клиент	client *тип аппаратный-адрес* ANY	Да	Нет	Определяет контейнер клиента, запрещающий выделение адреса клиенту с указанным *типом аппаратуры* и *аппаратным адресом*. Если параметр *тип аппаратуры* равен 0, то в параметре *аппаратный адрес* должна быть задана строка ASCII. В остальных случаях *тип аппаратуры* - это тип аппаратного обеспечения клиента, а *аппаратный адрес* - аппаратный адрес клиента. Если *аппаратный адрес* - это строка, то она может быть заключена в кавычки. Если *аппаратный адрес* - это шестнадцатеричная строка, то он может быть записан в виде 0x*шестн._число* или hex *шестн._число*. *Диапазон* разрешает выделять адрес из указанного *диапазона* клиенту с указанным *типом аппаратуры* и *аппаратным адресом*. Для сравнения нескольких клиентов должны использоваться регулярные выражения.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
клиент	client *тип аппаратный адрес адрес*	Да	Нет	Определяет контейнер клиента, запрещающий выделение адреса клиенту с указанным *типом аппаратуры* и *аппаратным адресом*. Если параметр *тип аппаратуры* равен 0, то в параметре *аппаратный адрес* должна быть задана строка ASCII. В остальных случаях *тип аппаратуры* - это тип аппаратного обеспечения клиента, а *аппаратный адрес* - аппаратный адрес клиента. Если *аппаратный адрес* - это строка, то она может быть заключена в кавычки. Если *аппаратный адрес* - это шестнадцатеричная строка, то он может быть записан в виде 0x*шестн._число* или hex *шестн._число*. *Диапазон* разрешает выделять адрес из указанного *диапазона* клиенту с указанным *типом аппаратуры* и *аппаратным адресом*. Для сравнения нескольких клиентов должны использоваться регулярные выражения.
клиент	client *тип аппаратный адрес диапазон*	Да	Нет	Определяет контейнер клиента, запрещающий выделение адреса клиенту с указанным *типом аппаратуры* и *аппаратным адресом*. Если параметр *тип аппаратуры* равен 0, то в параметре *аппаратный адрес* должна быть задана строка ASCII. В остальных случаях *тип аппаратуры* - это тип аппаратного обеспечения клиента, а *аппаратный адрес* - аппаратный адрес клиента. Если *аппаратный адрес* - это строка, то она может быть заключена в кавычки. Если *аппаратный адрес* - это шестнадцатеричная строка, то он может быть записан в виде 0x*шестн._число* или hex *шестн._число*. *Диапазон* разрешает выделять адрес из указанного *диапазона* клиенту с указанным *типом аппаратуры* и *аппаратным адресом*. Для сравнения нескольких клиентов должны использоваться регулярные выражения.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
class	class *строка*	Да	Нет	Определяет контейнер класса с именем *строка*. Строка может быть заключена в кавычки. Если строка заключена в кавычки, то перед сравнением они удаляются. Кавычки необходимы в том случае, если строка содержит пробелы или символы табуляции. Использование контейнера class допустимо на любом уровне. Для задания набора адресов, которые могут быть предложены клиенту с данным классом, рекомендуется пользоваться диапазонами. Диапазон может задаваться как одиночный IP-адрес в десятичном формате с точками или как два IP-адреса, разделенных дефисом.
class	class *строка диапазон*	Да	Нет	Определяет контейнер класса с именем *строка*. Строка может быть заключена в кавычки. Если строка заключена в кавычки, то перед сравнением они удаляются. Кавычки необходимы в том случае, если строка содержит пробелы или символы табуляции. Использование контейнера class допустимо на любом уровне. Для задания набора адресов, которые могут быть предложены клиенту с данным классом, рекомендуется пользоваться диапазонами. Диапазон может задаваться как одиночный IP-адрес в десятичном формате с точками или как два IP-адреса, разделенных дефисом.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
сеть	network *ид сети маска*	Да	Нет	Указывает контейнер, с которым следует сравнивать переданные клиентом опции с данным номером. *Номер* означает номер опции. *Данные* - это ключ для сравнения с этим контейнером в ходе выбора адреса и опции для данного клиента. Для стандартных опций *данные* указываются в явной форме: строка, заключенная в кавычки, IP-адрес или целое число; кроме того, их можно указать в форме шестнадцатеричной строки байтов, начинающейся с символов `0x`. Для опций, неизвестных серверу, допустим только второй вариант. Кроме того, *данные* могут представлять собой выражение, которое следует сравнить с содержимым строки данных опции, переданной клиентом. Такие выражения записываются в кавычках и начинаются с символов `"!` (кавычка и восклицательный знак). Опции, неизвестные серверу, должны указываться в виде шестнадцатеричной строки байтов БЕЗ начальных символов `0x`

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
сеть	network *ид_сети*	Да	Нет	Указывает контейнер, с которым следует сравнивать переданные клиентом опции с данным номером. *Номер* означает номер опции. *Данные* - это ключ для сравнения с этим контейнером в ходе выбора адреса и опции для данного клиента. Для стандартных опций *данные* указываются в явной форме: строка, заключенная в кавычки, IP-адрес или целое число; кроме того, их можно указать в форме шестнадцатеричной строки байтов, начинающейся с символов `0x`. Для опций, неизвестных серверу, допустим только второй вариант. Кроме того, *данные* могут представлять собой выражение, которое следует сравнить с содержимым строки данных опции, переданной клиентом. Такие выражения записываются в кавычках и начинаются с символов `"!` (кавычка и восклицательный знак). Опции, неизвестные серверу, должны указываться в виде шестнадцатеричной строки байтов БЕЗ начальных символов `0x`

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
сеть	network *ид сети диапазон*	Да	Нет	Указывает контейнер, с которым следует сравнивать переданные клиентом опции с данным номером. *Номер* означает номер опции. *Данные* - это ключ для сравнения с этим контейнером в ходе выбора адреса и опции для данного клиента. Для стандартных опций *данные* указываются в явной форме: строка, заключенная в кавычки, IP-адрес или целое число; кроме того, их можно указать в форме шестнадцатеричной строки байтов, начинающейся с символов `0x`. Для опций, неизвестных серверу, допустим только второй вариант. Кроме того, *данные* могут представлять собой выражение, которое следует сравнить с содержимым строки данных опции, переданной клиентом. Такие выражения записываются в кавычках и начинаются с символов `"!` (кавычка и восклицательный знак). Опции, неизвестные серверу, должны указываться в виде шестнадцатеричной строки байтов БЕЗ начальных символов `0x`

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
vendor	vendor *ид_вендора*	Да	Нет	Указывает контейнер, с которым следует сравнивать переданные клиентом опции с данным номером. *Номер* означает номер опции. *Данные* - это ключ для сравнения с этим контейнером в ходе выбора адреса и опции для данного клиента. Для стандартных опций *данные* указываются в явной форме: строка, заключенная в кавычки, IP-адрес или целое число; кроме того, их можно указать в форме шестнадцатеричной строки байтов, начинающейся с символов `0x`. Для опций, неизвестных серверу, допустим только второй вариант. Кроме того, *данные* могут представлять собой выражение, которое следует сравнить с содержимым строки данных опции, переданной клиентом. Такие выражения записываются в кавычках и начинаются с символов `"!` (кавычка и восклицательный знак). Опции, неизвестные серверу, должны указываться в виде шестнадцатеричной строки байтов БЕЗ начальных символов `0x`

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
vendor	vendor *ид_вендора* hex ""	Да	Нет	Указывает контейнер, с которым следует сравнивать переданные клиентом опции с данным номером. *Номер* означает номер опции. *Данные* - это ключ для сравнения с этим контейнером в ходе выбора адреса и опции для данного клиента. Для стандартных опций *данные* указываются в явной форме: строка, заключенная в кавычки, IP-адрес или целое число; кроме того, их можно указать в форме шестнадцатеричной строки байтов, начинающейся с символов `0x`. Для опций, неизвестных серверу, допустим только второй вариант. Кроме того, *данные* могут представлять собой выражение, которое следует сравнить с содержимым строки данных опции, переданной клиентом. Такие выражения записываются в кавычках и начинаются с символов `"!` (кавычка и восклицательный знак). Опции, неизвестные серверу, должны указываться в виде шестнадцатеричной строки байтов БЕЗ начальных символов `0x`

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
vendor	vendor *ид_вендора* hex ""	Да	Нет	Указывает контейнер, с которым следует сравнивать переданные клиентом опции с данным номером. *Номер* означает номер опции. *Данные* - это ключ для сравнения с этим контейнером в ходе выбора адреса и опции для данного клиента. Для стандартных опций *данные* указываются в явной форме: строка, заключенная в кавычки, IP-адрес или целое число; кроме того, их можно указать в форме шестнадцатеричной строки байтов, начинающейся с символов `0x`. Для опций, неизвестных серверу, допустим только второй вариант. Кроме того, *данные* могут представлять собой выражение, которое следует сравнить с содержимым строки данных опции, переданной клиентом. Такие выражения записываются в кавычках и начинаются с символов `"!` (кавычка и восклицательный знак). Опции, неизвестные серверу, должны указываться в виде шестнадцатеричной строки байтов БЕЗ начальных символов `0x`

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
vendor	vendor *ид_вендора* 0x*данные*	Да	Нет	Указывает контейнер, с которым следует сравнивать переданные клиентом опции с данным номером. *Номер* означает номер опции. *Данные* - это ключ для сравнения с этим контейнером в ходе выбора адреса и опции для данного клиента. Для стандартных опций *данные* указываются в явной форме: строка, заключенная в кавычки, IP-адрес или целое число; кроме того, их можно указать в форме шестнадцатеричной строки байтов, начинающейся с символов `0x`. Для опций, неизвестных серверу, допустим только второй вариант. Кроме того, *данные* могут представлять собой выражение, которое следует сравнить с содержимым строки данных опции, переданной клиентом. Такие выражения записываются в кавычках и начинаются с символов `"!` (кавычка и восклицательный знак). Опции, неизвестные серверу, должны указываться в виде шестнадцатеричной строки байтов БЕЗ начальных символов `0x`

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
vendor	vendor *ид_вендора* ""	Да	Нет	Указывает контейнер, с которым следует сравнивать переданные клиентом опции с данным номером. *Номер* означает номер опции. *Данные* - это ключ для сравнения с этим контейнером в ходе выбора адреса и опции для данного клиента. Для стандартных опций *данные* указываются в явной форме: строка, заключенная в кавычки, IP-адрес или целое число; кроме того, их можно указать в форме шестнадцатеричной строки байтов, начинающейся с символов `0x`. Для опций, неизвестных серверу, допустим только второй вариант. Кроме того, *данные* могут представлять собой выражение, которое следует сравнить с содержимым строки данных опции, переданной клиентом. Такие выражения записываются в кавычках и начинаются с символов `"!` (кавычка и восклицательный знак). Опции, неизвестные серверу, должны указываться в виде шестнадцатеричной строки байтов БЕЗ начальных символов `0x`

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
vendor	vendor *ид_вендора диапазон*	Да	Нет	Указывает контейнер, с которым следует сравнивать переданные клиентом опции с данным номером. *Номер* означает номер опции. *Данные* - это ключ для сравнения с этим контейнером в ходе выбора адреса и опции для данного клиента. Для стандартных опций *данные* указываются в явной форме: строка, заключенная в кавычки, IP-адрес или целое число; кроме того, их можно указать в форме шестнадцатеричной строки байтов, начинающейся с символов `0x`. Для опций, неизвестных серверу, допустим только второй вариант. Кроме того, *данные* могут представлять собой выражение, которое следует сравнить с содержимым строки данных опции, переданной клиентом. Такие выражения записываются в кавычках и начинаются с символов `"!` (кавычка и восклицательный знак). Опции, неизвестные серверу, должны указываться в виде шестнадцатеричной строки байтов БЕЗ начальных символов `0x`

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
vendor	vendor *ид_вендорадиапазон* hex ""	Да	Нет	Указывает контейнер, с которым следует сравнивать переданные клиентом опции с данным номером. *Номер* означает номер опции. *Данные* - это ключ для сравнения с этим контейнером в ходе выбора адреса и опции для данного клиента. Для стандартных опций *данные* указываются в явной форме: строка, заключенная в кавычки, IP-адрес или целое число; кроме того, их можно указать в форме шестнадцатеричной строки байтов, начинающейся с символов `0x`. Для опций, неизвестных серверу, допустим только второй вариант. Кроме того, *данные* могут представлять собой выражение, которое следует сравнить с содержимым строки данных опции, переданной клиентом. Такие выражения записываются в кавычках и начинаются с символов `"!` (кавычка и восклицательный знак). Опции, неизвестные серверу, должны указываться в виде шестнадцатеричной строки байтов БЕЗ начальных символов `0x`

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
vendor	vendor *ид_вендорадиапазон* hex ""	Да	Нет	Указывает контейнер, с которым следует сравнивать переданные клиентом опции с данным номером. *Номер* означает номер опции. *Данные* - это ключ для сравнения с этим контейнером в ходе выбора адреса и опции для данного клиента. Для стандартных опций *данные* указываются в явной форме: строка, заключенная в кавычки, IP-адрес или целое число; кроме того, их можно указать в форме шестнадцатеричной строки байтов, начинающейся с символов `0x`. Для опций, неизвестных серверу, допустим только второй вариант. Кроме того, *данные* могут представлять собой выражение, которое следует сравнить с содержимым строки данных опции, переданной клиентом. Такие выражения записываются в кавычках и начинаются с символов `"!` (кавычка и восклицательный знак). Опции, неизвестные серверу, должны указываться в виде шестнадцатеричной строки байтов БЕЗ начальных символов `0x`

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
vendor	vendor *ид_вендора диапазон* 0xзначение	Да	Нет	Указывает контейнер, с которым следует сравнивать переданные клиентом опции с данным номером. *Номер* означает номер опции. *Данные* - это ключ для сравнения с этим контейнером в ходе выбора адреса и опции для данного клиента. Для стандартных опций *данные* указываются в явной форме: строка, заключенная в кавычки, IP-адрес или целое число; кроме того, их можно указать в форме шестнадцатеричной строки байтов, начинающейся с символов `0x`. Для опций, неизвестных серверу, допустим только второй вариант. Кроме того, *данные* могут представлять собой выражение, которое следует сравнить с содержимым строки данных опции, переданной клиентом. Такие выражения записываются в кавычках и начинаются с символов `"!` (кавычка и восклицательный знак). Опции, неизвестные серверу, должны указываться в виде шестнадцатеричной строки байтов БЕЗ начальных символов `0x`

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
vendor	vendor *ид_вендора диапазон* ""	Да	Нет	Указывает контейнер, с которым следует сравнивать переданные клиентом опции с данным номером. *Номер* означает номер опции. *Данные* - это ключ для сравнения с этим контейнером в ходе выбора адреса и опции для данного клиента. Для стандартных опций *данные* указываются в явной форме: строка, заключенная в кавычки, IP-адрес или целое число; кроме того, их можно указать в форме шестнадцатеричной строки байтов, начинающейся с символов `0x`. Для опций, неизвестных серверу, допустим только второй вариант. Кроме того, *данные* могут представлять собой выражение, которое следует сравнить с содержимым строки данных опции, переданной клиентом. Такие выражения записываются в кавычках и начинаются с символов `"!` (кавычка и восклицательный знак). Опции, неизвестные серверу, должны указываться в виде шестнадцатеричной строки байтов БЕЗ начальных символов `0x`

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
входящая опция	входящая опция *номер данные*	Да	Нет	Указывает контейнер, с которым следует сравнивать переданные клиентом опции с данным номером. *Номер* означает номер опции. *Данные* - это ключ для сравнения с этим контейнером в ходе выбора адреса и опции для данного клиента. Для стандартных опций *данные* указываются в явной форме: строка, заключенная в кавычки, IP-адрес или целое число; кроме того, их можно указать в форме шестнадцатеричной строки байтов, начинающейся с символов `0x`. Для опций, неизвестных серверу, допустим только второй вариант. Кроме того, *данные* могут представлять собой выражение, которое следует сравнить с содержимым строки данных опции, переданной клиентом. Такие выражения записываются в кавычках и начинаются с символов `"!` (кавычка и восклицательный знак). Опции, неизвестные серверу, должны указываться в виде шестнадцатеричной строки байтов БЕЗ начальных символов `0x`

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
входящая опция	входящая опция *номер данные диапазон*	Да	Нет	Указывает контейнер, с которым следует сравнивать переданные клиентом опции с данным номером. *Номер* означает номер опции. *Данные* - это ключ для сравнения с этим контейнером в ходе выбора адреса и опции для данного клиента. Для стандартных опций *данные* указываются в явной форме: строка, заключенная в кавычки, IP-адрес или целое число; кроме того, их можно указать в форме шестнадцатеричной строки байтов, начинающейся с символов `0x`. Для опций, неизвестных серверу, допустим только второй вариант. Кроме того, *данные* могут представлять собой выражение, которое следует сравнить с содержимым строки данных опции, переданной клиентом. Такие выражения записываются в кавычках и начинаются с символов `"!` (кавычка и восклицательный знак). Опции, неизвестные серверу, должны указываться в виде шестнадцатеричной строки байтов БЕЗ начальных символов `0x`

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
virtual	virtual fill *ИД ИД* ...	Нет	Нет	Определяет виртуальную подсеть со стратегией. Слово `fill` означает, что перед переходом к следующему контейнеру необходимо использовать все адреса текущего контейнера. Слово `rotate` указывает, что для каждого запроса адрес выбирается из следующего указанного пула. `sfill` и `srotate` означают то же, что `fill` и `rotate`, однако в этом случае производится поиск, чтобы узнать, не соответствует ли клиент каким-либо контейнерам, вендорам или классам в подсети. Если обнаружено совпадение с контейнером, в котором выделяется адрес, то, адрес выбирается из этого контейнера, а не назначается в соответствии со стратегией. Число ИД в списке не ограничено. *ИД* - это либо ИД подсети, либо метка, заданная в описании подсети. Метка нужна при наличии нескольких подсетей с одинаковым идентификатором.
virtual	virtual sfill *ИД ИД* ...	Нет	Нет	Определяет виртуальную подсеть со стратегией. Слово `fill` означает, что перед переходом к следующему контейнеру необходимо использовать все адреса текущего контейнера. Слово `rotate` указывает, что для каждого запроса адрес выбирается из следующего указанного пула. `sfill` и `srotate` означают то же, что `fill` и `rotate`, однако в этом случае производится поиск, чтобы узнать, не соответствует ли клиент каким-либо контейнерам, вендорам или классам в подсети. Если обнаружено совпадение с контейнером, в котором выделяется адрес, то, адрес выбирается из этого контейнера, а не назначается в соответствии со стратегией. Число ИД в списке не ограничено. *ИД* - это либо ИД подсети, либо метка, заданная в описании подсети. Метка нужна при наличии нескольких подсетей с одинаковым идентификатором.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
virtual	virtual rotate *ИД ИД* ...	Нет	Нет	Определяет виртуальную подсеть со стратегией. Слово `fill` означает, что перед переходом к следующему контейнеру необходимо использовать все адреса текущего контейнера. Слово `rotate` указывает, что для каждого запроса адрес выбирается из следующего указанного пула. `sfill` и `srotate` означают то же, что `fill` и `rotate`, однако в этом случае производится поиск, чтобы узнать, не соответствует ли клиент каким-либо контейнерам, вендорам или классам в подсети. Если обнаружено совпадение с контейнером, в котором выделяется адрес, то, адрес выбирается из этого контейнера, а не назначается в соответствии со стратегией. Число ИД в списке не ограничено. *ИД* - это либо ИД подсети, либо метка, заданная в описании подсети. Метка нужна при наличии нескольких подсетей с одинаковым идентификатором.
virtual	virtual srotate *ИД ИД* ...	Нет	Нет	Определяет виртуальную подсеть со стратегией. Слово `fill` означает, что перед переходом к следующему контейнеру необходимо использовать все адреса текущего контейнера. Слово `rotate` указывает, что для каждого запроса адрес выбирается из следующего указанного пула. `sfill` и `srotate` означают то же, что `fill` и `rotate`, однако в этом случае производится поиск, чтобы узнать, не соответствует ли клиент каким-либо контейнерам, вендорам или классам в подсети. Если обнаружено совпадение с контейнером, в котором выделяется адрес, то, адрес выбирается из этого контейнера, а не назначается в соответствии со стратегией. Число ИД в списке не ограничено. *ИД* - это либо ИД подсети, либо метка, заданная в описании подсети. Метка нужна при наличии нескольких подсетей с одинаковым идентификатором.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
inorder:	inorder: *id id* ...	Нет	Нет	Определяет виртуальную подсеть со стратегией заполнения, т.е. перед переходом к следующему контейнеру должны быть использованы все адреса текущего контейнера. Число ИД в списке не ограничено. *ИД* - это либо ИД подсети, либо метка, заданная в описании подсети. Метка нужна при наличии нескольких подсетей с одинаковым идентификатором.
balance:	balance: *ИД ИД* ...	Нет	Нет	Определяет виртуальную подсеть со стратегией смены адресов, при которой каждый следующий адрес выбирается из следующего контейнера. Число ИД в списке не ограничено. *ИД* - это либо ИД подсети, либо метка, заданная в описании подсети. Метка нужна при наличии нескольких подсетей с одинаковым идентификатором.
supportBootp	supportBootp true	Нет	Да	Указывает, должен ли текущий контейнер и все контейнеры следующих уровней (до переопределения) поддерживать клиентов **BOOTP**.
supportBootp	supportBootp 1	Нет	Да	Указывает, должен ли текущий контейнер и все контейнеры следующих уровней (до переопределения) поддерживать клиентов **BOOTP**.
supportBootp	supportBootp yes	Нет	Да	Указывает, должен ли текущий контейнер и все контейнеры следующих уровней (до переопределения) поддерживать клиентов **BOOTP**.
supportBootp	supportBootp false	Нет	Да	Указывает, должен ли текущий контейнер и все контейнеры следующих уровней (до переопределения) поддерживать клиентов **BOOTP**.
supportBootp	supportBootp 0	Нет	Да	Указывает, должен ли текущий контейнер и все контейнеры следующих уровней (до переопределения) поддерживать клиентов **BOOTP**.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
supportBootp	supportBootp no	Нет	Да	Указывает, должен ли текущий контейнер и все контейнеры следующих уровней (до переопределения) поддерживать клиентов **BOOTP**.
supportBootp				Указывает, должен ли текущий контейнер и все контейнеры следующих уровней (до переопределения) поддерживать клиентов **BOOTP**.
supportUnlistedclients	supportUnlistedclients BOTH	Нет	Both	Указывает, должен ли текущий контейнер и все контейнеры следующих уровней (до переопределения) поддерживать клиентов, не перечисленных в списке. Значение указывает, каким клиентам необходимо разрешить доступ (без использования специальных операторов описания клиентов): только клиентам **DHCP**, только клиентам **BOOTP** или запретить доступ всем клиентам. **Примечание:** Значения true и false поддерживаются для совместимости с предыдущими версиями, и применять их не рекомендуется. True соответствует значению BOTH, a false - значению NONE.
supportUnlistedclients	supportUnlistedclients DHCP	Нет	Both	Указывает, должен ли текущий контейнер и все контейнеры следующих уровней (до переопределения) поддерживать клиентов, не перечисленных в списке. Значение указывает, каким клиентам необходимо разрешить доступ (без использования специальных операторов описания клиентов): только клиентам **DHCP**, только клиентам **BOOTP** или запретить доступ всем клиентам. **Примечание:** Значения true и false поддерживаются для совместимости с предыдущими версиями, и применять их не рекомендуется. True соответствует значению BOTH, a false - значению NONE.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
supportUnlistedclients	supportUnlistedclients BOOTP	Нет	Both	Указывает, должен ли текущий контейнер и все контейнеры следующих уровней (до переопределения) поддерживать клиентов, не перечисленных в списке. Значение указывает, каким клиентам необходимо разрешить доступ (без использования специальных операторов описания клиентов): только клиентам **DHCP**, только клиентам **BOOTP** или запретить доступ всем клиентам. **Примечание:** Значения true и false поддерживаются для совместимости с предыдущими версиями, и применять их не рекомендуется. True соответствует значению BOTH, a false - значению NONE.
supportUnlistedclients	supportUnlistedclients NONE	Нет	Both	Указывает, должен ли текущий контейнер и все контейнеры следующих уровней (до переопределения) поддерживать клиентов, не перечисленных в списке. Значение указывает, каким клиентам необходимо разрешить доступ (без использования специальных операторов описания клиентов): только клиентам **DHCP**, только клиентам **BOOTP** или запретить доступ всем клиентам. **Примечание:** Значения true и false поддерживаются для совместимости с предыдущими версиями, и применять их не рекомендуется. True соответствует значению BOTH, a false - значению NONE.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
supportUnlistedclients	supportUnlistedclients true	Нет	Both	Указывает, должен ли текущий контейнер и все контейнеры следующих уровней (до переопределения) поддерживать клиентов, не перечисленных в списке. Значение указывает, каким клиентам необходимо разрешить доступ (без использования специальных операторов описания клиентов): только клиентам **DHCP**, только клиентам **BOOTP** или запретить доступ всем клиентам. **Примечание:** Значения true и false поддерживаются для совместимости с предыдущими версиями, и применять их не рекомендуется. True соответствует значению BOTH, a false - значению NONE.
supportUnlistedclients	supportUnlistedclients yes	Нет	Both	Указывает, должен ли текущий контейнер и все контейнеры следующих уровней (до переопределения) поддерживать клиентов, не перечисленных в списке. Значение указывает, каким клиентам необходимо разрешить доступ (без использования специальных операторов описания клиентов): только клиентам **DHCP**, только клиентам **BOOTP** или запретить доступ всем клиентам. **Примечание:** Значения true и false поддерживаются для совместимости с предыдущими версиями, и применять их не рекомендуется. True соответствует значению BOTH, a false - значению NONE.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
supportUnlistedclients	supportUnlistedclients 1	Нет	Both	Указывает, должен ли текущий контейнер и все контейнеры следующих уровней (до переопределения) поддерживать клиентов, не перечисленных в списке. Значение указывает, каким клиентам необходимо разрешить доступ (без использования специальных операторов описания клиентов): только клиентам **DHCP**, только клиентам **BOOTP** или запретить доступ всем клиентам. **Примечание:** Значения true и false поддерживаются для совместимости с предыдущими версиями, и применять их не рекомендуется. True соответствует значению BOTH, a false - значению NONE.
supportUnlistedclients	supportUnlistedclients false	Нет	Both	Указывает, должен ли текущий контейнер и все контейнеры следующих уровней (до переопределения) поддерживать клиентов, не перечисленных в списке. Значение указывает, каким клиентам необходимо разрешить доступ (без использования специальных операторов описания клиентов): только клиентам **DHCP**, только клиентам **BOOTP** или запретить доступ всем клиентам. **Примечание:** Значения true и false поддерживаются для совместимости с предыдущими версиями, и применять их не рекомендуется. True соответствует значению BOTH, a false - значению NONE.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
supportUnlistedclients	supportUnlistedclients no	Нет	Both	Указывает, должен ли текущий контейнер и все контейнеры следующих уровней (до переопределения) поддерживать клиентов, не перечисленных в списке. Значение указывает, каким клиентам необходимо разрешить доступ (без использования специальных операторов описания клиентов): только клиентам **DHCP**, только клиентам **BOOTP** или запретить доступ всем клиентам. **Примечание:** Значения true и false поддерживаются для совместимости с предыдущими версиями, и применять их не рекомендуется. True соответствует значению BOTH, a false - значению NONE.
supportUnlistedclients	supportUnlistedclients 0	Нет	Both	Указывает, должен ли текущий контейнер и все контейнеры следующих уровней (до переопределения) поддерживать клиентов, не перечисленных в списке. Значение указывает, каким клиентам необходимо разрешить доступ (без использования специальных операторов описания клиентов): только клиентам **DHCP**, только клиентам **BOOTP** или запретить доступ всем клиентам. **Примечание:** Значения true и false поддерживаются для совместимости с предыдущими версиями, и применять их не рекомендуется. True соответствует значению BOTH, a false - значению NONE.
addressrecorddb	addressrecrddb *полное_имя*	Нет	Нет	Работает как ключевое слово **backupfile**. Допустимо только на глобальном уровне или на уровне контейнера базы данных. **Примечание:** Использовать этот метод не рекомендуется.
backupfile	backupfile *полное_имя*	Нет	/etc/db_file.crbk	Определяет файл для хранения резервных копий базы данных. Допустимо только на глобальном уровне или на уровне контейнера базы данных.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
checkpointfile	checkpointfile *полное_имя*	Нет	/etc/db_file.chkpt	Задает контрольные файлы базы данных. Первый контрольный файл задается параметром *полное_имя*. В имени второго файла второй символ*полного имени* заменяется на цифру 2. В связи с этим имена контрольных файлов не должны заканчиваться цифрой 2. Правило действует только для контейнеров глобального уровня или уровня базы данных.
clientrecorddb	clientrecorddb *полное_имя*	Нет	/etc/db_file.cr	Задает файл для сохранения базы данных. В этом файле будут сохраняться записи всех клиентов, которых обслуживает сервер **DHCP**. Допустимо только на глобальном уровне или на уровне контейнера базы данных.
bootstrapserver	bootstrapserver *IP-адрес*	Нет	Нет	Указывает, на каком сервере находятся файлы **TFTP**, которые должны использоваться клиентами после получения ими пакетов **BOOTP** или **DHCP**. Это значение задается в поле **siaddr** пакета. Допустимо на уровне любого контейнера.
giaddrfield	giaddrfield *IP-адрес*	Нет	Нет	Задает поле giaddrfield для ответных сообщений (пакетов). **Примечание:** Данная спецификация недопустима в протоколах **BOOTP** и **DHCP**, однако для работы некоторых клиентов необходимо, чтобы в поле **giaddr** был указан шлюз по умолчанию. Из-за возможных конфликтов ключевое слово giaddrfield должно использоваться только внутри контейнеров клиентов, хотя оно может работать на любом уровне.
pingTime	pingTime *n единицы_времени*	Нет	3 seconds	Задает время ожидания отклика на команду ping перед выделением адреса. По умолчанию время указывается в сотых долях секунды. Значение единиц времени определено в примечаниях, приведенных перед данной таблицей. Допустимо на уровне любого контейнера. Параметр *единицы_времени* указывать не обязательно.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
bootptime	bootptime *n единицы_времени*	Нет	-1, infinite	Определяет время выделения адреса для клиента **BOOTP**. Значение по умолчанию -1, что соответствует бесконечному времени выделения адреса. Применимы обычные единицы времени. Параметр *единицы_времени* указывать не обязательно. Допустимо на уровне любого контейнера.
AllRoutesBroadcast	allroutesbroadcast no	Нет	0	Указывает, должны ли ответные сообщения рассылаться по всем маршрутам, если необходим оповещающий ответ. Допустимо на уровне любого контейнера. Это ключевое слово игнорируется сервером **DHCP** данной операционной системы, поскольку для возврата пакета сохраняется фактический адрес MAC клиента, включая RIF. Допустимо на уровне любого контейнера.
AllRoutesBroadcast	allroutesbroadcast false	Нет	0	Указывает, должны ли ответные сообщения рассылаться по всем маршрутам, если необходим оповещающий ответ. Допустимо на уровне любого контейнера. Это ключевое слово игнорируется сервером **DHCP** данной операционной системы, поскольку для возврата пакета сохраняется фактический адрес MAC клиента, включая RIF. Допустимо на уровне любого контейнера.
AllRoutesBroadcast	allroutesbroadcast 0	Нет	0	Указывает, должны ли ответные сообщения рассылаться по всем маршрутам, если необходим оповещающий ответ. Допустимо на уровне любого контейнера. Это ключевое слово игнорируется сервером **DHCP** данной операционной системы, поскольку для возврата пакета сохраняется фактический адрес MAC клиента, включая RIF. Допустимо на уровне любого контейнера.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
AllRoutesBroadcast	allroutesbroadcast yes	Нет	0	Указывает, должны ли ответные сообщения рассылаться по всем маршрутам, если необходим оповещающий ответ. Допустимо на уровне любого контейнера. Это ключевое слово игнорируется сервером **DHCP** данной операционной системы, поскольку для возврата пакета сохраняется фактический адрес MAC клиента, включая RIF. Допустимо на уровне любого контейнера.
AllRoutesBroadcast	allroutesbroadcast true	Нет	0	Указывает, должны ли ответные сообщения рассылаться по всем маршрутам, если необходим оповещающий ответ. Допустимо на уровне любого контейнера. Это ключевое слово игнорируется сервером **DHCP** данной операционной системы, поскольку для возврата пакета сохраняется фактический адрес MAC клиента, включая RIF. Допустимо на уровне любого контейнера.
AllRoutesBroadcast	allroutesbroadcast 1	Нет	0	Указывает, должны ли ответные сообщения рассылаться по всем маршрутам, если необходим оповещающий ответ. Допустимо на уровне любого контейнера. Это ключевое слово игнорируется сервером **DHCP** данной операционной системы, поскольку для возврата пакета сохраняется фактический адрес MAC клиента, включая RIF. Допустимо на уровне любого контейнера.
addressassigned	addressassigned "*строка*"	Нет	Нет	Определяет заключенную в кавычки строку, которая будет выполняться при присвоении адреса клиенту. Строка должна содержать два параметра подстановки %s. Первый параметр %s - это ИД клиента в виде *тип-строка*. Второй параметр %s - это IP-адрес в десятичном формате с точками. Допустимо на уровне любого контейнера.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
addressreleased	addressreleased "*строка*"	Нет	Нет	Определяет заключенную в кавычки строку, которая будет выполняться при освобождении адреса клиентом. Строка должна содержать один параметр подстановки %s, который задает высвобождаемый IP-адрес в десятичном формате с точками. Допустимо на уровне любого контейнера.
appenddomain	appenddomain 0	Нет	Нет	Указывает, нужно ли добавлять имя домена, определяемое опцией 15, к рекомендуемому для клиента имени хоста в том случае, если клиент не предлагает задавать имя домена. Допустимо на уровне любого контейнера.
appenddomain	appenddomain no	Нет	Нет	Указывает, нужно ли добавлять имя домена, определяемое опцией 15, к рекомендуемому для клиента имени хоста в том случае, если клиент не предлагает задавать имя домена. Допустимо на уровне любого контейнера.
appenddomain	appenddomain false	Нет	Нет	Указывает, нужно ли добавлять имя домена, определяемое опцией 15, к рекомендуемому для клиента имени хоста в том случае, если клиент не предлагает задавать имя домена. Допустимо на уровне любого контейнера.
appenddomain	appenddomain 1	Нет	Нет	Указывает, нужно ли добавлять имя домена, определяемое опцией 15, к рекомендуемому для клиента имени хоста в том случае, если клиент не предлагает задавать имя домена. Допустимо на уровне любого контейнера.
appenddomain	appenddomain yes	Нет	Нет	Указывает, нужно ли добавлять имя домена, определяемое опцией 15, к рекомендуемому для клиента имени хоста в том случае, если клиент не предлагает задавать имя домена. Допустимо на уровне любого контейнера.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
appenddomain	appenddomain true	Нет	Нет	Указывает, нужно ли добавлять имя домена, определяемое опцией 15, к рекомендуемому для клиента имени хоста в том случае, если клиент не предлагает задавать имя домена. Допустимо на уровне любого контейнера.
canonical	canonical 0	Нет	0	Указывает, что ИД клиента имеет канонический формат. Допустимо только в контейнере клиента.
canonical	canonical no	Нет	0	Указывает, что ИД клиента имеет канонический формат. Допустимо только в контейнере клиента.
canonical	canonical false	Нет	0	Указывает, что ИД клиента имеет канонический формат. Допустимо только в контейнере клиента.
canonical	canonical 1	Нет	0	Указывает, что ИД клиента имеет канонический формат. Допустимо только в контейнере клиента.
canonical	canonical yes	Нет	0	Указывает, что ИД клиента имеет канонический формат. Допустимо только в контейнере клиента.
canonical	canonical true	Нет	0	Указывает, что ИД клиента имеет канонический формат. Допустимо только в контейнере клиента.
leaseTimeDefault	leaseTimeDefault *n единицы_времени*	Нет	86400 seconds	Задает время выделения адреса по умолчанию для клиентов. Допустимо на уровне любого контейнера. Параметр *единицы_времени* указывать не обязательно.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
proxyarec	proxyarec never	Нет	usedhcpddnsplus	Указывает опции и способы для обновления записей A в DNS. Значение `never` указывает, что запись A никогда не обновляется. Значение `usedhcpddns` указывает, что должна применяться опция 81, если она задана клиентом. Значение `usedhcpddnsplus` указывает, что должна применяться опция 81 или опции 12 и 15, если они заданы. Значение `always` задает обновление записей A для всех клиентов. Значения `XXXXprotected` изменяют команду **nsupdate** и выполняют проверку допустимости клиента. Значение `standard` аналогично значению `always`. Значение `protected` аналогично значению `alwaysprotected`. Допустимо на уровне любого контейнера.
proxyarec	proxyarec usedhcpddns	Нет	usedhcpddnsplus	Указывает опции и способы для обновления записей A в DNS. Значение `never` указывает, что запись A никогда не обновляется. Значение `usedhcpddns` указывает, что должна применяться опция 81, если она задана клиентом. Значение `usedhcpddnsplus` указывает, что должна применяться опция 81 или опции 12 и 15, если они заданы. Значение `always` задает обновление записей A для всех клиентов. Значения `XXXXprotected` изменяют команду **nsupdate** и выполняют проверку допустимости клиента. Значение `standard` аналогично значению `always`. Значение `protected` аналогично значению `alwaysprotected`. Допустимо на уровне любого контейнера.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
proxyarec	proxyarec usedhcpddnsplus	Нет	usedhcpddnsplus	Указывает опции и способы для обновления записей A в DNS. Значение `never` указывает, что запись A никогда не обновляется. Значение `usedhcpddns` указывает, что должна применяться опция 81, если она задана клиентом. Значение `usedhcpddnsplus` указывает, что должна применяться опция 81 или опции 12 и 15, если они заданы. Значение `always` задает обновление записей A для всех клиентов. Значения `XXXXprotected` изменяют команду **nsupdate** и выполняют проверку допустимости клиента. Значение `standard` аналогично значению `always`. Значение `protected` аналогично значению `alwaysprotected`. Допустимо на уровне любого контейнера.
proxyarec	proxyarec always	Нет	usedhcpddnsplus	Указывает опции и способы для обновления записей A в DNS. Значение `never` указывает, что запись A никогда не обновляется. Значение `usedhcpddns` указывает, что должна применяться опция 81, если она задана клиентом. Значение `usedhcpddnsplus` указывает, что должна применяться опция 81 или опции 12 и 15, если они заданы. Значение `always` задает обновление записей A для всех клиентов. Значения `XXXXprotected` изменяют команду **nsupdate** и выполняют проверку допустимости клиента. Значение `standard` аналогично значению `always`. Значение `protected` аналогично значению `alwaysprotected`. Допустимо на уровне любого контейнера.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
proxyarec	proxyarec usedhcpddnsprotected	Нет	usedhcpddnsplus	Указывает опции и способы для обновления записей A в DNS. Значение `never` указывает, что запись A никогда не обновляется. Значение `usedhcpddns` указывает, что должна применяться опция 81, если она задана клиентом. Значение `usedhcpddnsplus` указывает, что должна применяться опция 81 или опции 12 и 15, если они заданы. Значение `always` задает обновление записей A для всех клиентов. Значения `XXXXprotected` изменяют команду **nsupdate** и выполняют проверку допустимости клиента. Значение `standard` аналогично значению `always`. Значение `protected` аналогично значению `alwaysprotected`. Допустимо на уровне любого контейнера.
proxyarec	proxyarec usedhcpddnsplusprotected	Нет	usedhcpddnsplus	Указывает опции и способы для обновления записей A в DNS. Значение `never` указывает, что запись A никогда не обновляется. Значение `usedhcpddns` указывает, что должна применяться опция 81, если она задана клиентом. Значение `usedhcpddnsplus` указывает, что должна применяться опция 81 или опции 12 и 15, если они заданы. Значение `always` задает обновление записей A для всех клиентов. Значения `XXXXprotected` изменяют команду **nsupdate** и выполняют проверку допустимости клиента. Значение `standard` аналогично значению `always`. Значение `protected` аналогично значению `alwaysprotected`. Допустимо на уровне любого контейнера.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
proxyarec	proxyarec alwaysprotected	Нет	usedhcpddnsplus	Указывает опции и способы для обновления записей A в DNS. Значение `never` указывает, что запись A никогда не обновляется. Значение `usedhcpddns` указывает, что должна применяться опция 81, если она задана клиентом. Значение `usedhcpddnsplus` указывает, что должна применяться опция 81 или опции 12 и 15, если они заданы. Значение `always` задает обновление записей A для всех клиентов. Значения `XXXXprotected` изменяют команду **nsupdate** и выполняют проверку допустимости клиента. Значение `standard` аналогично значению `always`. Значение `protected` аналогично значению `alwaysprotected`. Допустимо на уровне любого контейнера.
proxyarec	proxyarec standard	Нет	usedhcpddnsplus	Указывает опции и способы для обновления записей A в DNS. Значение `never` указывает, что запись A никогда не обновляется. Значение `usedhcpddns` указывает, что должна применяться опция 81, если она задана клиентом. Значение `usedhcpddnsplus` указывает, что должна применяться опция 81 или опции 12 и 15, если они заданы. Значение `always` задает обновление записей A для всех клиентов. Значения `XXXXprotected` изменяют команду **nsupdate** и выполняют проверку допустимости клиента. Значение `standard` аналогично значению `always`. Значение `protected` аналогично значению `alwaysprotected`. Допустимо на уровне любого контейнера.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
	proxyarec protected	Нет	usedhcpddnsplus	Указывает опции и способы для обновления записей A в DNS. Значение `never` указывает, что запись A никогда не обновляется. Значение `usedhcpddns` указывает, что должна применяться опция 81, если она задана клиентом. Значение `usedhcpddnsplus` указывает, что должна применяться опция 81 или опции 12 и 15, если они заданы. Значение `always` задает обновление записей A для всех клиентов. Значения `XXXXprotected` изменяют команду **nsupdate** и выполняют проверку допустимости клиента. Значение `standard` аналогично значению `always`. Значение `protected` аналогично значению `alwaysprotected`. Допустимо на уровне любого контейнера.
releasednsA	releasednsA "*строка*"	Нет	Нет	Задает строку, которая выполняется при освобождении адреса. Строка используется для удаления записи A, связанной с освобождаемым адресом. Допустимо на уровне любого контейнера.
releasednsP	releasednsP "*строка*"	Нет	Нет	Задает строку, которая выполняется при освобождении адреса. Строка используется для удаления записи PTR, связанной с освобождаемым адресом. Допустимо на уровне любого контейнера.
removedns	removedns "*строка*"	Нет	Нет	Задает строку, которая выполняется при освобождении адреса. Строка используется для удаления записей PTR и A, связанных с освобождаемым адресом. Допустимо на уровне любого контейнера. **Примечание:** Вместо данного ключевого слова рекомендуется использовать ключевые слова **releasednsA** и **releasednsP**.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
updatedns	updatedns "*строка*"	Нет	Нет	Задает строку, которая выполняется при связывании адреса. Строка используется для обновления записей PTR и A, связанных с данным адресом. Допустимо на уровне любого контейнера. **Примечание:** Вместо данного ключевого слова рекомендуется использовать ключевые слова **updatednsA** и **updatednsP**.
updatednsA	updatednsA "*строка*"	Нет	Нет	Задает строку, которая выполняется при связывании адреса. Строка используется для обновления записи A, связанной с данным адресом. Допустимо на уровне любого контейнера.
updatednsP	updatednsP "*строка*"	Нет	Нет	Задает строку, которая выполняется при связывании адреса. Строка используется для обновления записи PTR, связанной с данным адресом. Допустимо на уровне любого контейнера.
hostnamepolicy	hostnamepolicy suggested	Нет	default	Задает имя хоста для возвращения клиенту. Стратегия по умолчанию отдает преимущество заданным именам хостов и доменов перед рекомендуемыми. Другие стратегии подразумевают строгое соответствие (например, при указании опции `defined` будет возвращаться заданное имя, а если имя не задано, то оно не будет возвращено). Аналогично, в случае применения стратегии с модификатором `always` сервер будет возвращать параметр имени хоста независимо от того, запросил его клиент в опции списка параметров или нет. Обратите внимание, предложение имени хоста подразумевает его запрос, и имена хостов могут быть предложены с помощью опции 81 или опций 12 и 15. Данное ключевое слово допустимо на уровне любого контейнера.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
hostnamepolicy	hostnamepolicy resolved	Нет	default	Задает имя хоста для возвращения клиенту. Стратегия по умолчанию отдает преимущество заданным именам хостов и доменов перед рекомендуемыми. Другие стратегии подразумевают строгое соответствие (например, при указании опции `defined` будет возвращаться заданное имя, а если имя не задано, то оно не будет возвращено). Аналогично, в случае применения стратегии с модификатором `always` сервер будет возвращать параметр имени хоста независимо от того, запросил его клиент в опции списка параметров или нет. Обратите внимание, предложение имени хоста подразумевает его запрос, и имена хостов могут быть предложены с помощью опции 81 или опций 12 и 15. Данное ключевое слово допустимо на уровне любого контейнера.
hostnamepolicy	hostnamepolicy always_resolved	Нет	default	Задает имя хоста для возвращения клиенту. Стратегия по умолчанию отдает преимущество заданным именам хостов и доменов перед рекомендуемыми. Другие стратегии подразумевают строгое соответствие (например, при указании опции `defined` будет возвращаться заданное имя, а если имя не задано, то оно не будет возвращено). Аналогично, в случае применения стратегии с модификатором `always` сервер будет возвращать параметр имени хоста независимо от того, запросил его клиент в опции списка параметров или нет. Обратите внимание, предложение имени хоста подразумевает его запрос, и имена хостов могут быть предложены с помощью опции 81 или опций 12 и 15. Данное ключевое слово допустимо на уровне любого контейнера.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
hostnamepolicy	hostnamepolicy defined	Нет	default	Задает имя хоста для возвращения клиенту. Стратегия по умолчанию отдает преимущество заданным именам хостов и доменов перед рекомендуемыми. Другие стратегии подразумевают строгое соответствие (например, при указании опции `defined` будет возвращаться заданное имя, а если имя не задано, то оно не будет возвращено). Аналогично, в случае применения стратегии с модификатором `always` сервер будет возвращать параметр имени хоста независимо от того, запросил его клиент в опции списка параметров или нет. Обратите внимание, предложение имени хоста подразумевает его запрос, и имена хостов могут быть предложены с помощью опции 81 или опций 12 и 15. Данное ключевое слово допустимо на уровне любого контейнера.
hostnamepolicy	hostnamepolicy always_defined	Нет	default	Задает имя хоста для возвращения клиенту. Стратегия по умолчанию отдает преимущество заданным именам хостов и доменов перед рекомендуемыми. Другие стратегии подразумевают строгое соответствие (например, при указании опции `defined` будет возвращаться заданное имя, а если имя не задано, то оно не будет возвращено). Аналогично, в случае применения стратегии с модификатором `always` сервер будет возвращать параметр имени хоста независимо от того, запросил его клиент в опции списка параметров или нет. Обратите внимание, предложение имени хоста подразумевает его запрос, и имена хостов могут быть предложены с помощью опции 81 или опций 12 и 15. Данное ключевое слово допустимо на уровне любого контейнера.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
hostnamepolicy	hostnamepolicy default	Нет	default	Задает имя хоста для возвращения клиенту. Стратегия по умолчанию отдает преимущество заданным именам хостов и доменов перед рекомендуемыми. Другие стратегии подразумевают строгое соответствие (например, при указании опции `defined` будет возвращаться заданное имя, а если имя не задано, то оно не будет возвращено). Аналогично, в случае применения стратегии с модификатором `always` сервер будет возвращать параметр имени хоста независимо от того, запросил его клиент в опции списка параметров или нет. Обратите внимание, предложение имени хоста подразумевает его запрос, и имена хостов могут быть предложены с помощью опции 81 или опций 12 и 15. Данное ключевое слово допустимо на уровне любого контейнера.
bootfilepolicy	bootfilepolicy suggested	Нет	suggested	Задает предпочтения при возвращении имени загрузочного файла клиенту. Значение `suggested` означает, что имя загрузочного файла, предложенного клиентом, будет предпочтительней любого имени, указанного в конфигурации сервера. Значение `merge` добавляет имя, предложенное клиентом, к домашнему каталогу, указанному в конфигурации сервера. Значение `defined` означает, что имя загрузочного файла, определенное в конфигурации, будет предпочтительней имени, предложенного клиентом. Значение `always` возвращает заданное имя независимо от того, запрашивает ли клиент параметр загрузочного файла с помощью опции списка параметров.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
bootfilepolicy	bootfilepolicy merge	Нет	suggested	Задает предпочтения при возвращении имени загрузочного файла клиенту. Значение `suggested` означает, что имя загрузочного файла, предложенного клиентом, будет предпочтительней любого имени, указанного в конфигурации сервера. Значение `merge` добавляет имя, предложенное клиентом, к домашнему каталогу, указанному в конфигурации сервера. Значение `defined` означает, что имя загрузочного файла, определенное в конфигурации, будет предпочтительней имени, предложенного клиентом. Значение `always` возвращает заданное имя независимо от того, запрашивает ли клиент параметр загрузочного файла с помощью опции списка параметров.
bootfilepolicy	bootfilepolicy defined	Нет	suggested	Задает предпочтения при возвращении имени загрузочного файла клиенту. Значение `suggested` означает, что имя загрузочного файла, предложенного клиентом, будет предпочтительней любого имени, указанного в конфигурации сервера. Значение `merge` добавляет имя, предложенное клиентом, к домашнему каталогу, указанному в конфигурации сервера. Значение `defined` означает, что имя загрузочного файла, определенное в конфигурации, будет предпочтительней имени, предложенного клиентом. Значение `always` возвращает заданное имя независимо от того, запрашивает ли клиент параметр загрузочного файла с помощью опции списка параметров.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
bootfilepolicy	bootfilepolicy always	Нет	suggested	Задает предпочтения при возвращении имени загрузочного файла клиенту. Значение `suggested` означает, что имя загрузочного файла, предложенного клиентом, будет предпочтительней любого имени, указанного в конфигурации сервера. Значение `merge` добавляет имя, предложенное клиентом, к домашнему каталогу, указанному в конфигурации сервера. Значение `defined` означает, что имя загрузочного файла, определенное в конфигурации, будет предпочтительней имени, предложенного клиентом. Значение `always` возвращает заданное имя независимо от того, запрашивает ли клиент параметр загрузочного файла с помощью опции списка параметров.
stealfromchildren	stealfromchildren true	Нет	Нет	Указывает, должен ли родительский контейнер использовать адреса из дочерних контейнеров в том случае, если его пул адресов исчерпан. Это означает, что если у вас есть подсеть с классом, для которого задан диапазон адресов, то эти адреса зарезервированы для клиентов, соответствующих данному классу. Если для опции `stealfromchildren` указано значение true, то для выполнения запроса будут применяться адреса из дочернего контейнера. По умолчанию адреса из дочерних контейнеров не используются.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
stealfromchildren	stealfromchildren 1	Нет	Нет	Указывает, должен ли родительский контейнер использовать адреса из дочерних контейнеров в том случае, если его пул адресов исчерпан. Это означает, что если у вас есть подсеть с классом, для которого задан диапазон адресов, то эти адреса зарезервированы для клиентов, соответствующих данному классу. Если для опции `stealfromchildren` указано значение true, то для выполнения запроса будут применяться адреса из дочернего контейнера. По умолчанию адреса из дочерних контейнеров не используются.
stealfromchildren	stealfromchildren yes	Нет	Нет	Указывает, должен ли родительский контейнер использовать адреса из дочерних контейнеров в том случае, если его пул адресов исчерпан. Это означает, что если у вас есть подсеть с классом, для которого задан диапазон адресов, то эти адреса зарезервированы для клиентов, соответствующих данному классу. Если для опции `stealfromchildren` указано значение true, то для выполнения запроса будут применяться адреса из дочернего контейнера. По умолчанию адреса из дочерних контейнеров не используются.
stealfromchildren	stealfromchildren false	Нет	Нет	Указывает, должен ли родительский контейнер использовать адреса из дочерних контейнеров в том случае, если его пул адресов исчерпан. Это означает, что если у вас есть подсеть с классом, для которого задан диапазон адресов, то эти адреса зарезервированы для клиентов, соответствующих данному классу. Если для опции `stealfromchildren` указано значение true, то для выполнения запроса будут применяться адреса из дочернего контейнера. По умолчанию адреса из дочерних контейнеров не используются.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
stealfromchildren	stealfromchildren 0	Нет	Нет	Указывает, должен ли родительский контейнер использовать адреса из дочерних контейнеров в том случае, если его пул адресов исчерпан. Это означает, что если у вас есть подсеть с классом, для которого задан диапазон адресов, то эти адреса зарезервированы для клиентов, соответствующих данному классу. Если для опции `stealfromchildren` указано значение true, то для выполнения запроса будут применяться адреса из дочернего контейнера. По умолчанию адреса из дочерних контейнеров не используются.
stealfromchildren	stealfromchildren no	Нет	Нет	Указывает, должен ли родительский контейнер использовать адреса из дочерних контейнеров в том случае, если его пул адресов исчерпан. Это означает, что если у вас есть подсеть с классом, для которого задан диапазон адресов, то эти адреса зарезервированы для клиентов, соответствующих данному классу. Если для опции `stealfromchildren` указано значение true, то для выполнения запроса будут применяться адреса из дочернего контейнера. По умолчанию адреса из дочерних контейнеров не используются.
homedirectory	homedirectory *полное_имя*	Нет	Нет	Определяет домашний каталог, который используется в разделе файла, задающем ответный пакет (сообщение). Допустимо на уровне любого контейнера. Взаимодействие элементов, указанных в поступающем пакете с операторами загрузочного файла и домашнего каталога определяется стратегией загрузочного файла.
bootfile	bootfile *полное_имя*	Нет	Нет	Определяет загрузочный файл, который должен быть указан в ответном пакете. Допустимо на уровне любого контейнера. Взаимодействие элементов, указанных в поступающем пакете с операторами загрузочного файла и домашнего каталога определяется стратегией загрузочного файла.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
pxebootfile	pxebootfile *архитектура старшая_версия младшая_версия загрузочный_файл*	Нет	Нет	Задает загрузочный файл, имя которого будет передано клиенту. Этот параметр применяется только в том случае, если **dhcpsd** поддерживает клиентов PXE (параметр **pxeservertype** равен **dhcp_pxe_binld**). Программа разбора файла конфигурации отравляет сообщение об ошибке, если число параметров после ключевого слова `pxebootfile` меньше четырех, а лишние параметры всегда игнорируются. Ключевое слово `pxebootfile` может применяться только в контейнере.
supportoption118	supportoption118 *no/ yes*	Нет. Может применяться только в контейнере подсети.	Нет	Это ключевое слово указывает, поддерживает ли контейнер опцию 118. Значение yes означает наличие поддержки, а no - ее отсутствие. Для того чтобы эта опция вступила в силу, необходимо также указать ключевое слово **supportsubnetselection**.

Рекомендации по DHCP и управлению сетевой установкой (NIM)

Динамическое распределение IP-адресов - относительно новая технология. Приведенные ниже рекомендации помогут вам организовать взаимодействие **DHCP** и NIM.

1. При настройке объектов в среде NIM, всегда, когда это возможно, указывайте имена хостов. Это позволит вам применять динамический сервер имен, обновляющий IP-адреса при преобразовании имени хоста в IP-адрес в среде NIM.
2. Установите серверы NIM и **DHCP** на одном компьютере. Сервер **DHCP** имеет опцию обновления строки DNS. Если указать эту опцию с параметром `NIM`, то в моменты изменения адресов система будет старается не переключать объекты NIM в состояния, требующие статических IP-адресов.
3. Для клиентов NIM время выделения адресов по умолчанию должно равняться удвоенному времени, требующемуся для установки клиента. Таким образом, выделенный адрес будет допустим на протяжении всего процесса установки. После установки перезапустите клиента. В зависимости от типа установки, сервер **DHCP** будет сразу запущен, или его потребуется предварительно настроить.
4. Сервер `dhcpsd` должен отвечать за обновление записей DNS типа PTR и A. Когда NIM повторно устанавливает машину, файл, содержащий RSA, удаляется, и клиент не может обновить свои записи. Поэтому системные записи обновляет сервер. Для этого измените строку `updatedns` в файле `/etc/dhcpcd.ini` следующим образом:

   ```
   updatedns "/usr/sbin/dhcpaction '%s' '%s' '%s' '%s' '%s' NONE NONIM"
   ```

   В файле `/etc/dhcpsd.cnf` измените строку `updatedns`:

   ```
   updatedns "/usr/sbin/dhcpaction '%s' '%s' '%s' '%s' '%s' BOTH NIM"
   ```

   Примечание: Когда объект NIM ожидает установки BOS, сервер `dhcpsd` может передать аргументы, отличные от установленных изначально. Для того чтобы этого не произошло, постарайтесь максимально сократить время, в течение которого клиент находится в состоянии ожидания.

Следуя данным рекомендациям, вы сможете настроить среду NIM для работы с динамическими клиентами.

Дополнительные сведения об управлении сетевой установкой приведены в разделе *AIX 5L версии 5.3: Руководство и справочник по сетевой установке*.

Протокол динамической настройки хостов версии 6

Протокол динамической настройки хостов (DHCP) позволяет работать с сетевыми конфигурациями из централизованного расположения. Этот раздел посвящен **DHCPv6**; под IP-адресами понимаются адреса IPv6, а под **DHCP** - **DHCPv6** (если не сказано обратное).

Сервер **DHCPv4** может сосуществовать на одной линии с сервером **DHCPv6**. Более подробное описание протокола содержится в RFC 3315.

DHCP - это протокол уровня приложений, который позволяет подключенным к сети системам получать IP-адреса и другие параметры конфигурации с сервера. Эти параметры описаны в разделе *опции*. Для получения опций демон, работающий на клиенте, обменивается пакетами данных с аналогичным демоном, работающим на сервере. Этот обмен сообщениями происходит в форме пакетов **UDP**. Клиент идентифицирует свой исходный адрес на сервере, используя локальный адрес линии - с помощью команды **autoconf6** или другим способом. Сервер принимает сообщения на зарезервированном групповом адресе. Промежуточный агент разрешает связь между клиентом и сервером, если они не расположены на одной линии.

В этом разделе описывается квитирование из четырех сообщений для одного интерфейса с одним IA_NA и одним адресом для этого IA_NA. Для получения IP-адреса демон клиента **DHCP** (**dhcpcd6**) отправляет сообщение `SOLICIT` (требование) по адресу **Все_промежуточные_агенты_и_серверы_DHCP**. Это сообщение принимается и обрабатывается сервером. (Для повышения надежности в сети может быть настроено несколько серверов.) Если для этого клиента имеется свободный адрес, сервер создает и отправляет клиенту сообщение `ADVERTISE` (уведомление). В этом сообщении содержится IP-адрес и другие параметры для клиента. Клиент получает это сообщение `DHCP ADVERTISE` и сохраняет его, ожидая других уведомлений. После того как клиент выберет наиболее подходящее уведомление, он отправляет запрос `DHCP REQUEST` по адресу **Все_промежуточные_агенты_и_серверы_DHCP**, указывая, какое уведомление сервера он выбрал.

Все настроенные серверы **DHCP** получают запрос `REQUEST`. Каждый из них проверяет, ему ли был направлен запрос. Сервер не приступает к обработке пакетов с DUID, отличным от его собственного. Запрошенный сервер помечает адрес как присвоенный и возвращает ответ `DHCP REPLY`. На этом обработка запроса завершается. Клиенту выделяется адрес на время, определенное сервером (`фактический срок действия`).

Когда предпочитаемый срок действия адреса заканчивается, клиент отправляет серверу пакет `RENEW` с запросом на продление этого срока. Если сервер готов продлить срок действия адреса, он отправляет ответ `DHCP REPLY`. Если клиент не получает ответа от сервера, который выделил ему адрес, то он рассылает оповещающее сообщение `DHCP REBIND`, например, в том случае, если сервер был переведен в другую сеть. Если до истечения срока действия адреса клиент не обновит свой адрес, то работа сетевого интерфейса завершится и процесс начнется сначала. Такая процедура позволяет избежать присвоения одинаковых адресов нескольким клиентам.

У клиента может быть несколько опций IA_NA, у каждой из которых может быть несколько адресов. У клиента также может быть несколько опций IA_TA, у каждой из которых тоже может быть несколько адресов:

- **Связь идентификатора для невременных адресов (IA_NA)**: Связь идентификатора для присвоенных невременных адресов.
- **Связь идентификатора для временных адресов (IA_TA)**: Связь идентификатора для временных адресов (см. RFC 3041).
- **DUID**: Уникальный в **DHCP** идентификатор для участников **DHCP**; у каждого клиента и каждого сервера **DHCP** есть уникальный идентификатор DUID, сохраняющийся при перезагрузке.

Сервер **DHCP** присваивает адреса на основе ключей. Существует четыре общих ключа: **класс**, **вендор**, **ИД клиента** и **входящая опция**. На основании этих ключей сервер выбирает адрес и параметры конфигурации, которые передаются клиенту.

class Ключ **класс** полностью настраиваем для клиента. Этот ключ может включать адрес и параметры. С его помощью можно указать назначение системы в сети или задать способ объединения систем в группы для упрощения администрирования. Например, администратор сети может создать класс NetBIOS для задания опций клиентов NetBIOS или класс accounting, объединяющий компьютеры бухгалтерии, которым необходим доступ к определенному принтеру.

vendor Ключ **вендор** помогает идентифицировать клиента по его аппаратному и программному обеспечению.

ИД клиента
Ключ **ИД клиента** идентифицирует клиента посредством DUID. ИД клиента задается в файле `duid` демона **dhcpcd**. Кроме того, ИД клиента может применяться сервером для передачи опций данному клиенту или для запрета передачи каких-либо опций клиенту.

Входящая опция
Ключ **входящая опция** идентифицирует клиента по запрашиваемой им опции.

Эти ключи можно использовать по одному или в сочетании друг с другом. Если клиент использует несколько ключей, и при этом может быть присвоено несколько адресов, то выбирается только один из них, причем набор опций определяется первым выбранным ключом.

Промежуточный агент обеспечивает передачу оповещающих сообщений клиента за пределы локальной сети. Промежуточные агенты работают как посредники, пересылающие пакеты **DHCP**.

Сервер DHCPv6

Сервер **DHCPv6** состоит из трех основных компонентов.

Сервер **DHCP** состоит из трех основных компонентов: базы данных, средств поддержки протокола и набора служебных нитей. Для каждого компонента задается своя информация о конфигурации.

Базы данных DHCPv6:

База данных `db_filev6.dhcpo` используется для отслеживания клиентов и адресов, а также для управления доступом.

Кроме того, в базе данных хранятся опции, предназначенные для передачи клиентам. База данных реализована как динамически загружаемый объект.

База данных заполняется и проверяется в соответствии с информацией, заданной в файле конфигурации. Кроме того, в базе данных хранятся пулы адресов и опций.

Файл, хранящийся в оперативной памяти, и его резервная копия - это обычные текстовые файлы ASCII. Формат файлов базы данных, хранящихся в оперативной памяти, следующий:

Примечание: Не редактируйте вручную эти файлы.

```
DB6-1.0
Client-Info {
duid 1-0006085b68e20004ace491d3
state 7
authinfo {
    protocol 2
    algorithm 1
    rdm 0
    replay 1206567640
}
  Interface 0 {
```

```
    Inoptions {
    interface-id "en1"
    policies 2
    maxopcode 16
    numiana 1
      Ianalist {
      option 3 40 00000001000000320000005000050018deaddeadaaaaaaaa0000000000000006000000064000000c8
      }
    numiata 0
      Optiontable {
      option 6 10 00030004001700180237
      option 8 2 e659
      option 15 14 000369626d000373756e00026870
      option 16 18 000004d2000730783131313131000369626d
      }
    }
    Ianarec {
    IAID 1
    t1 50
    t2 80
      Addrec {
      Address dead:dead:aaaa:aaaa::6
      state 3
      starttime  1087592918
      preferred-lifetime 100
      valid-lifetime 200
      }
    }
  }
}
```

В первой строке указывается идентификатор версии файла: `DB6-1.0`. За ней следуют записи, определяющие клиентов. Сервер читает данные, начиная со второй строки, и до конца файла. (Параметры, указанные здесь в кавычках, должны быть заключены в кавычки.)

duid Идентификатор клиента.

Interface
У клиента может быть несколько интерфейсов. Если у клиента только один интерфейс и клиент создает отдельные сообщения **SOLICIT** для каждого IA_NA или IA_TA, то в файле будут содержаться несколько интерфейсов для этого клиента.

Inoptions
Входящие опции от клиента.

policies
Этот флаг служит для идентификации направленной рассылки, опции повторной настройки и быстрой фиксации.

maxopcode
Наибольший код опции.

numiana
Число IA_NA для этого интерфейса.

Ianalist
Список поступающих от клиента опций IA_NA.

numiata
Число IA_TA для этого интерфейса.

Optiontable
Список опций, запрошенных клиентом, исключая опции IA_NA и IA_TA.

Ianarec
Сохраненный контейнер записей IA_NA из базы данных сервера.

IAID ИД IA_NA.

t1 Предпочтительный срок действия для этого IA_NA.

t2 Фактический срок действия для этого IA_NA.

Addrec
Контейнер записей адресов из базы данных сервера.

Адрес Адрес, выделенный клиенту для этой записи адреса.

state Текущее состояние клиента. Средства поддержки протокола **DHCP** включают определенный набор состояний, которые сохраняются в базе данных **DHCP**. Номер рядом с записью **state** представляет его значение. Возможны следующие состояния:

(1) СВОБОДЕН
Обозначает доступные адреса. Это состояние может быть указано для клиента только в том случае, если ему не присвоен адрес. В выводе команд **dadmin** и **lssrc** это состояние обозначается как `Свободный`.

(2) СВЯЗАН
Указывает, что адрес связан с клиентом и что данный адрес выделен клиенту на определенное время. В данных, выдаваемых командами **dadmin** и **lssrc**, это состояние обозначается как `Выделен`.

(3) ИСТЕК
Указывает, что адрес связан с клиентом, но лишь с информационной целью, как и в случае освобожденного адреса. Это состояние свидетельствует о том, что время выделения адреса клиенту истекло. Адрес с истекшим временем выделения могут быть присвоены другим клиентам только после того, как будут присвоены все свободные и освобожденные адреса. В данных, выдаваемых командами **dadmin** и **lssrc**, это состояние обозначается как `Истек`.

(4) ОСВОБОЖДЕН
Указывает, что адрес связан с клиентом, но лишь с информационной целью. В протоколе **DHCP** предполагается, что серверы **DHCP** сохраняют информацию об обслуживаемых клиентах. В основном это делается для того, чтобы по возможности предоставлять клиенту тот же адрес, который уже присваивался ему ранее. Это состояние указывает, что клиент освободил адрес. Этот адрес будет выделяться другим клиентам только в том случае, если других доступных адресов нет. В данных, выдаваемых командами **dadmin** и **lssrc**, это состояние обозначается как `Освобожден`.

(5) ЗАРЕЗЕРВИРОВАН
Указывает, что между клиентом и адресом установлена предварительная связь. Клиент уже разослал поисковое сообщение **DHCP**, а сервер **DHCP** ответил на него, но клиент еще не отправил запрос **DHCP** для окончательного присвоения адреса. В данных, выдаваемых командами **dadmin** и **lssrc**, это состояние обозначается как `Зарезервирован`.

(6) НЕВЕРНЫЙ
Обозначает адрес, который используется в сети, но не был выделен сервером **DHCP**. Кроме того, это состояние указывается для адресов, отклоненных клиентами. Это состояние неприменимо к клиентам. В данных, выдаваемых командой **dadmin**, это состояние обозначается как `Используется`, а в выводах команды **lssrc** - `Неверный`.

Starttime
Задает время выделения адреса, в секундах, считая от 1 января 2000 года.

preferred-lifetime
Задает срок действия адреса до того, как адрес потребуется обновить, в секундах.

valid-lifetime
Задает срок действия адреса до того, как адрес станет недействительным, в секундах.

протокол
Протокол идентификации, используемый клиентом.

(1) ОТЛОЖЕННАЯ
Клиент использует отложенную идентификацию.

(2) КЛЮЧ ПЕРЕНАСТРОЙКИ
Клиент использует идентификацию по ключу перенастройки.

алгоритм
Алгоритм идентификации, который использует клиент:

(1) HMAC-MD5
Для создания указателей на сообщения клиент использует алгоритм MD5 по ключам.

rdm Метод обнаружения ответа, который использует клиент:

(0) монотонно возрастающий счетчик
Для изменения значения ответа клиент использует монотонно возрастающий счетчик.

ответ Текущее значение в поле ответа.

Синтаксис контрольных файлов не определен. По контрольным файлам и резервным копиям сервер восстанавливает базу данных после сбоя или аварийного завершения работы системы, в ходе которого не удалось закрыть базу данных обычным образом. Все клиенты, не записанные в контрольный файл во время сбоя сервера, будут потеряны. В настоящее время при обработке клиента промежуточные сохранения не выполняются. По умолчанию это следующие файлы:

`/etc/dhcpv6/db_file6.cr`
Применяется при обычной работе базы данных

`/etc/dhcpv6/db_file6.crbk`
Резервные копии базы данных

Операции с нитями DHCP:

Третья часть сервера **DHCP** - это набор операций, которые, собственно, и обеспечивают его работу.

Так как в сервере **DHCP** реализована поддержка нескольких нитей, то эти операции в действительности организованы в виде набора нитей, которые в определенное время выполняют нужные операции и обеспечивают правильную работу.

Главная нить
Эта нить обрабатывает сигналы. Пример:

- SIGHUP (-1) обновляет все базы данных в файле конфигурации.
- SIGTERM (-15) завершает работу сервера в нормальном режиме.
- SIGUSR1 (-30) сервер создаст дамп базы данных конфигурации

Нить src
Нить обрабатывает запросы SRC (такие как **startsrc**, **stopsrc**, **lssrc**, **traceson** и **refresh**).

Нить dadmin
Эта нить обеспечивает взаимодействие программы клиента **dadmin** и сервера **DHCP**. Утилита **dadmin** служит для определения состояния и изменения базы данных, позволяя избежать редактирования файлов базы данных вручную. С появлением нитей **dadmin** и **src** сервер может обрабатывать служебные запросы и запросы клиентов одновременно.

Нить мусора
Эта нить включает таймеры, которые периодически выполняют очистку и сохранение базы данных, удаляют клиенты, для которых не заданы адреса, а также удаляют адреса, слишком долго находящиеся в зарезервированном состоянии. Все эти таймеры можно настроить.

Обработчики пакетов

Каждая из этих нитей может обрабатывать запросы клиента **DHCPv6**. Число обработчиков пакетов зависит от предполагаемой нагрузки и мощности системы. Их число можно настраивать; по умолчанию запускается 1 нить. Максимальное число этих нитей - 50.

Протоколы нитей

В системах со значительным объемом данных, записываемых в файлы протоколов, число нитей ведения протоколов может превышать значение по умолчанию (1) и достигать максимального значения (50).

Нить диспетчера таблиц

Эта нить следит за тем, чтобы демон **dhcpsdv6** не обрабатывал одинаковые пакеты.

Нити обработки

Данные нити обрабатывают пакеты клиента **DHCPv6**.

нить перенастройки

Данная нить управляет перенастройкой при обновлении сервера (например, в результате выполнения команды `dadmin -x 6 -i`).

Настройка DHCPv6

По умолчанию сервер **DHCP** считывает информацию из файла `/etc/dhcpv6/dhcpsdv6.cnf`, в котором хранится исходная база данных параметров и адресов.

Сервер запускается командами SRC. Если **dhcpsdv6** следует запускать при загрузке, добавьте запись в файл `/etc/rc.tcpip`.

Настройка сервера **DHCP** - это самая трудная часть настройки **DHCP** в сети. Вначале определите, в каких сетях будут размещаться клиенты **DHCP**. Каждой подсети соответствует пул адресов, который должен быть добавлен в базу данных сервера **DHCP**. Например:

```
subnet dead:dead:aaaa:: 48 {
        option 23 dead::beef beef:aaaa::bbbb:c aaaa:bbbb::cccc  # список серверов имен
        option 24 austin.ibm.com ibm.com                        # список доменов
 }
```

В приведенном выше примере применяется подсеть `dead:dead:aaaa::` с префиксом из 48 битов. Все адреса этой подсети, от `dead:dead:aaaa::1` до `dead:dead:aaaa:ffff:ffff:ffff:ffff:ff7f`, находятся в пуле. При необходимости в конце строки перед "{" можно указать диапазон, а также задать в контейнере subnet оператор range или exclude.

Комментарии начинаются с символа # (знак фунта стерлингов). Сервер **DHCP** игнорирует текст, стоящий после символа # до конца строки. В строках опций определяются действия, которые должен выполнить клиент.

Если сервер не знает, как интерпретировать ту или иную опцию, то он передает ее клиенту с помощью методов, предусмотренных по умолчанию. Такая возможность позволяет серверу пересылать специальные параметры, которые не определены в RFC, но могут применяться некоторыми клиентами или в отдельных конфигурациях клиентов.

Файл конфигурации DHCPv6:

В файле конфигурации есть раздел адресов и раздел определения опций. В этих разделах есть контейнеры, содержащие опции и модификаторы, а также, возможно, другие контейнеры.

Контейнер (по существу, это способ группирования параметров) позволяет объединять клиентов в группы на основании идентификатора. Типы контейнеров: **subnet**, **class**, **vendor**, **inoption** и **client**. Контейнеры, определяемые пользователем, в настоящее время не поддерживаются. Клиент однозначно определяется

своим идентификатором, что позволяет всегда точно обнаруживать его, например, при переносе в другую подсеть. Для описания клиента может применяться несколько контейнеров.

Опции - это идентификаторы, возвращаемые клиенту. Это могут быть, например, имена доменов или адреса DNS.

Следующий шаг настройки после выбора модификаторов - это настройка протоколов. Параметры протокола указываются в окне диалога, напоминающем окно настройки базы данных, но с ключевым словом **logging_info**. На этапе обучения настройке **DHCP** рекомендуется установить максимально подробное ведение протокола. Кроме того, весьма полезно настроить параметры протокола до начала работы с любыми другими файлами конфигурации, чтобы после инициализации подсистемы протокола информация об ошибках конфигурации заносилась в протокол. Для включения опций протокола укажите ключевое слово **logitem**, а для их отключения удалите это ключевое слово. Другие ключевые слова позволяют задавать имя файла протокола, его размер, а также число взаимозаменяемых файлов протоколов.

Контейнеры DHCPv6:

При получении запроса сервер **DHCP** анализирует пакет и на основании ключей идентификации определяет, какие нужно выбрать контейнеры, параметры и адреса.

Для идентификации клиентов в разных типах контейнеров применяются различные опции:

- Для определения подсети, в которой находится клиент, контейнер **subnet** использует поле **hintlist** или адрес целевого интерфейса.
- Контейнер **class** использует значение опции 15 (идентификатор опции класса пользователя).
- Контейнер **vendor** использует значение опции 16 (опция класса вендора).
- Контейнер **client** использует значение опции 1 (опция ИД клиента) из DUID клиента DHCP.
- Контейнер **inoption** совпадает с запрошенной клиентом опцией.

Для всех контейнеров, за исключением subnet, можно задать шаблон для сравнения, например, регулярное выражение.

Существует также неявный контейнер global. Этот контейнер содержит все опции и модификаторы, которые не запрещены и не переопределены. Большинство контейнеров можно поместить в другие контейнеры в соответствии с областью видимости. С контейнерами могут быть связаны диапазоны адресов. С подсетями всегда связаны диапазоны адресов.

Ниже перечислены основные правила организации контейнеров и подконтейнеров:

- На глобальном уровне допустимы только контейнеры subnet.
- Subnet нельзя поместить в другие контейнеры, включая их самих.
- В контейнеры с ограничениями нельзя помещать обычные контейнеры того же типа. (Например, если контейнер содержит опцию, разрешающую только класс Accounting, то в него нельзя помещать контейнер, который содержит опцию, разрешающую применение всех классов, имена которых начинаются с буквы a.)
- Контейнеры client с ограничениями не могут содержать вложенные контейнеры.
- Контейнеры inoption не могут содержать вложенные контейнеры.

С помощью этих правил вы можете создавать иерархию контейнеров, в которой опции объединены в наборы, соответствующие конкретным клиентам или группам клиентов.

Если клиент соответствует нескольким контейнерам, то сервер **DHCP** формирует запрос к базе данных и получает список контейнеров. Контейнеры перечисляются в списке в порядке их вложенности и приоритета. Приоритет определяется как неявный иерархический уровень контейнера. Контейнеры с ограничениями имеют более высокий приоритет, чем обычные. Сортировка контейнеров выполняется в таком порядке:

клиенты, классы, вендоры и подсети. В пределах одного типа контейнеры упорядочиваются по уровню вложенности. Созданный таким образом список упорядочивается от более конкретных объектов к менее конкретным. Например:

```
Subnet 1
    --Class 1
    --Client 1
Subnet 2
    --Class 1
    ----Vendor 1
    ----Client 1
    --Client 1
```

В примере есть две подсети `Subnet 1` и `Subnet 2`. Кроме того, определен один класс, `Class 1`, один вендор, `Vendor 1`, и один клиент, `Client 1`. `Class 1` и `Client 1` определены в нескольких контейнерах. Поскольку эти определения находятся в разных контейнерах, имена объектов могут совпадать, однако указанные в них значения могут различаться. Если клиент `Client 1` отправит сообщение серверу **DHCP** из подсети `Subnet 1` с указанием класса `Class 1`, определенного в списке опций этого клиента, то сервер **DHCP** создаст следующий список контейнеров:

```
Subnet 1, Class 1, Client 1
```

Контейнер, определенный наиболее точно, заносится в список последним. Для получения адреса список просматривается в обратном порядке до обнаружения первого доступного адреса. Затем список просматривается в прямом порядке (в соответствии с иерархией) для получения опций. По мере просмотра списка новые значения опций переопределяют прежние значения, если в контейнере не содержится опция deny. Поскольку класс `Class 1` и клиент `Client 1` находятся в одной и той же подсети `Subnet 1`, они упорядочиваются в соответствии с приоритетом контейнеров. Если это же сообщение будет получено от клиента с тем же именем, находящегося в подсети Subnet 2, то будет создан следующий список контейнеров:

```
Subnet 2, Class 1, Client 1 (на уровне Subnet 2),
Client 1 (на уровне Class 1)
```

Первой в списке указывается подсеть `Subnet 2`, затем класс `Class 1`, затем клиент `Client 1` на уровне `Subnet 2` (так как этот клиент находится в иерархии на один уровень ниже). Иерархия подразумевает, что клиент, имя которого указано в первом операторе, определен менее конкретно, чем клиент `Client 1`, определенный в классе `Class 1` подсети `Subnet 2`.

Приоритет, определяемый по уровню вложенности, выше, чем приоритет самих контейнеров. Например, если тот же клиент отправит такое же сообщение, указав идентификатор вендора, то список контейнеров будет следующим:

```
Subnet 2, Class 1, Vendor 1, Client 1 (на уровне Subnet 2),
Client 1 (на уровне Class 1)
```

Организация поиска на основании приоритета контейнера повышает эффективность, поскольку контейнеры client обеспечивают наиболее точный способ определения одного или нескольких клиентов. В контейнере class адреса определены менее конкретно, чем в контейнере client; в контейнере vendor адреса определены еще менее конкретно, а контейнер subnet содержит самые общие определения адресов.

Адреса и диапазоны адресов DHCPv6:

С контейнерами любого типа могут быть связаны диапазоны адресов; такие диапазоны обязательно есть у контейнеров subnet.

Диапазон адресов контейнера должен быть подмножеством диапазона адресов родительского контейнера и не должен пересекаться с диапазонами адресов других контейнеров. Например, если внутри подсети определен класс с диапазоном адресов, то этот диапазон должен быть подмножеством диапазона адресов подсети. Диапазон внутри этого контейнера класса не должен пересекаться с любыми другими диапазонами этого уровня.

Диапазоны адресов могут задаваться в строке контейнера. Для задания несмежных диапазонов адресов можно воспользоваться операторами range и exclude. Таким образом, если в подсети есть два диапазона по десять адресов, то имеет смысл указать эти диапазоны в операторе subnet, чтобы уменьшить объем памяти и избежать конфликтов адресов с другими клиентами.

Когда адрес выбран, все последующие контейнеры, содержащие диапазоны адресов, удаляются из списка вместе со своими дочерними контейнерами. Сетевые опции, заданные в удаленных контейнерах, недопустимы для адресов, не относящихся к данному контейнеру.

Опции файла конфигурации DHCPv6:

После первого просмотра списка и получения адресов для клиента создается набор опций.

В процессе выбора ранее определенные значения опций переопределяются новыми до тех пор, пока не встретится опция deny (запретить); при этом запрещенная опция удаляется из списка опций, отправляемых пользователю. Этот способ разрешает наследование опций родительских контейнеров и сокращает объем данных, которые нужно определять.

Опции сервера DHCPv6:

Последними нужно указать параметры работы сервера, в том числе число нитей для обработки пакетов, частоту сбора мусора и т.д.

Вот, например, две опции, определяемые сервером:

reservedTime
: Указывает, на какое время резервируется адрес после отправки сообщения ADVERTISE клиенту **DHCP**.

reservedTimeInterval
: Указывает, как часто сервер **DHCP** должен просматривать список адресов и определять, для каких адресов истекло время, заданное в параметре *reservedTime*.

Эти опции полезны в том случае, когда в сети есть несколько клиентов, которые рассылают сообщения SOLICIT, а затем либо не рассылают сообщения REQUEST, либо отправляемые ими сообщения REQUEST не доходят до серверов. Эти параметры позволяют резервировать адреса только для тех клиентов, которые работают правильно.

Другая полезная опция, *SaveInterval*, указывает, как часто нужно сохранять информацию.

Файл /etc/dhcpv6/dhcpsdv6.cnf:

Настройка сервера **DHCPv6** выполняется путем редактирования файла конфигурации `/etc/dhcpv6/dhcpsdv6.cnf`.

При вводе ключевых слов следует учитывать регистр. Если указан символ '{', то он должен находиться в одной строке с ключевым словом. Пример файла конфигурации находится в `/usr/samples/tcpip/dhcpv6`.

Ниже приведено описание файла `/etc/dhcpv6/dhcpsdv6.cnf`. В этом файле могут быть следующие разделы:

- Ведение протоколов
- Глобальные ключевые слова
- Не вложенные операторы контейнеров
- Вложенные операторы контейнеров
- Опции
- Общие опции

Ведение протоколов в DHCPv6:

Данные ключевые слова сервера **DHCPv6** применяются в разделе ведения протоколов.

Этот раздел необязателен, но если он есть, то он должен располагаться в начале файла конфигурации. Формат раздела следующий:

```
logging_info { опции_протокола
}
```

Значения *опций_протокола* могут быть следующими:

Таблица 63. Ключевые слова, допустимые значения и описания записей в разделе ведения протокола

Ключевое слово	Значение	Описание
logFileSize	*число*	Размер файла протокола. *Число* задает максимальный размер файла протокола в килобайтах. Файл протокола будет перезаписан после достижения указанного размера. Если параметр `logFileSize` не задан, размер файла протокола не ограничен.
logFileName	"*имя файла*"	Имя файла протокола. *Имя файла* задает имя файла протокола. Имя файла по умолчанию - `/var/tmp/dhcpsdv6.log`.
numLogFiles	*число*	Указывает число файлов протокола для перезаписи. Значение по умолчанию - 0.
logItem	*тип*	Указывает тип требуемого ведения протокола. Допустимы следующие типы: **SYSERR** Системная ошибка в интерфейсе платформы. **OBJERR** Ошибка объекта, возникшая в процессе. **PROTERR** Ошибка протокола между клиентом и сервером. **WARNING** Предупреждение, призыв пользователя к вниманию. **EVENT** В процессе произошло некоторое событие. **ACTION** Процесс выполнил некоторое действие. **INFO** Информация, которая может оказаться полезной. **ACNTING** Кто и когда был обслужен. **TRACE** Коды для отладки.

Глобальные ключевые слова DHCPv6:

Описанные здесь значения ключевых слов предназначены для записей в разделе ключевых глобальных слов.

Глобальные ключевые слова вступают в силу только вне контейнера. Допустимы следующие значения:

Таблица 64. Ключевые слова, допустимые значения и описания записей в разделе глобальных ключевых слов

Ключевое слово	Значение	Описание
UsedIpAddressExpiredInterval	число [*единицы измерения*]	Указывает, с какой периодичностью проверяется правильность адресов, находящихся в состоянии НЕВЕРНЫЙ. Если единицы измерения не заданы, то по умолчанию время измеряется в секундах. Значение по умолчанию равно -1.
leaseExpiredInterval	число [*единицы измерения*]	Указывает, с какой периодичностью проверяется истечение срока действия адресов, находящихся в состоянии СВЯЗАН. Если срок действия адреса истек, то состояние адреса изменяется на УСТАРЕЛ. Если единицы измерения не заданы, то по умолчанию время измеряется в секундах. Значение по умолчанию - 900 секунд.
reservedTime	число [*единицы измерения*]	Указывает, как долго адрес должен находиться в состоянии ЗАРЕЗЕРВИРОВАН, прежде чем он будет переведен в состояние СВОБОДЕН. Если единицы измерения не заданы, то по умолчанию время измеряется в секундах. Значение по умолчанию равно -1.
reservedTimeInterval	число [*единицы измерения*]	Указывает, с какой периодичностью проверяется адрес, находящийся в состоянии ЗАРЕЗЕРВИРОВАН, на предмет изменения его состояния на СВОБОДЕН. Если единицы измерения не заданы, то по умолчанию время измеряется в секундах. Значение по умолчанию - 900 секунд.
saveInterval	число [*единицы измерения*]	Указывает, с какой периодичностью сервер **DHCP** должен проводить принудительное сохранение открытых баз данных. Для серверов, работающих с высокой нагрузкой, это значение должно составлять от 60 до 120 секунд. Если единицы измерения не заданы, то по умолчанию время измеряется в секундах. Значение по умолчанию - 3600 секунд.
clientpruneintv	число [*единицы измерения*]	Указывает, с какой периодичностью сервер **DHCP** удаляет из базы данных клиентов, не связанных с какими-либо адресами (т.е. клиентов с состоянием НЕИЗВЕСТНО). Это сокращает объем памяти, используемой сервером **DHCP**. Если единицы измерения не заданы, то по умолчанию время измеряется в секундах. Значение по умолчанию - 3600 секунд.
numprocessthreads	число	Указывает, сколько должно быть создано нитей процессоров для обработки пакетов. Минимальное значение равно единице. Каждая нить может обрабатывать запросы одного клиента. Значение по умолчанию - 30.
numpacketthreads	число	Указывает, сколько должно быть создано нитей пакетов. Минимальное значение - 1, значение по умолчанию - 5.
numloggingthreads	число	Указывает число нитей ведения протоколов. Значение по умолчанию - 1.
numduidbuckets	число	Этот параметр используется диспетчером таблиц и имеет прямое отношение к **numprocessthreads**. Значение по умолчанию - 53.
numclientbuckets	число	Указывает, сколько наборов будет использовано для хранения клиентских записей. Значение по умолчанию - 1021.
ignoreinterfacelist	*интерфейс* [*интерфейс*]	Список игнорируемых интерфейсов. Может быть указан один или несколько интерфейсов.
backupfile	"*имя файла*"	Файл для хранения резервных копий базы данных. Имя файла по умолчанию - /etc/dhcpv6/db_file6.crbk
checkpointfile	"*имя файла*"	Задает контрольные файлы базы данных. Имя первого контрольного файла - это "имя файла". Имя второго контрольного файла образуется путем замены последнего символа "имени файла" на цифру 2. В связи с этим, имя контрольного файла не должно оканчиваться цифрой 2.
clientrecorddb	"*имя файла*"	Задает файл для сохранения базы данных. В этом файле будут сохраняться записи всех клиентов, которых обслуживает сервер **DHCP**. Имя файла по умолчанию - **/etc/dhcpv6/db_file6.cr**

Таблица 64. Ключевые слова, допустимые значения и описания записей в разделе глобальных ключевых слов (продолжение)

Ключевое слово	Значение	Описание
duid	тип_ИД*значение* [*значение*]	Используется для идентификации сервера. Допустимы следующие значения: • duid 1 *интерфейс* • duid 2 *интерфейс* • duid 3 номер предприятия *идентификатор* • duid номер 0х*шестнадцатеричное значение*
preference-number	число	Позволяет клиенту указать приоритет сервера, с которого он предпочитает получать информацию. Чем больше это значение, тем выше вероятность того, что клиент будет использовать этот сервер. Значение по умолчанию - 255, оно же является максимальным.
unicast-enable	*стратегия*	Стратегия направленной рассылки, которая позволяет серверу обмениваться информацией с помощью направленной рассылки. По умолчанию эта опция включена.
tablemgr-policy	*стратегия*	Стратегия, которая позволяет серверу пользоваться диспетчером таблиц для более эффективного управления входящими клиентами. По умолчанию эта опция включена.
auth	*стратегия*	Включает поддержку отложенной идентификации. По умолчанию эта опция выключена.
auth-keyfile	"*имя файла*"	Файл, в котором содержатся ключи отложенной идентификации. Файл по умолчанию: **/etc/dhcpv6/dhcpsdv6.keys**.

Не вложенные операторы контейнеров DHCPv6:

Ключевое слово **subnet** сервера **DHCPv6** предназначено для записей операторов невложенных контейнеров.

Операторы невложенных контейнеров могут существовать только как часть глобальных ключевых слов.

Таблица 65. Ключевые слова, допустимые значения и описания записей операторов невложенных контейнеров

Элемент	Описание	
subnet	*ИД-подсети длина-префикса* [*диапазон*] {ОПЦИИ}	Указывает используемую подсеть. *ИД_подсети* должен быть адресом IPv6. *Длина_префикса* должна быть положительным целым числом меньше 128.

Вложенные операторы контейнеров DHCPv6:

Операторы вложенных контейнеров могут существовать только в виде опций внутри подсети.

У всех контейнеров могут быть вложенные контейнеры, если не указано обратное. Максимальная глубина вложенности - 7, включая подсеть и глобальный контейнер (в контейнере подсети могут содержаться только 5 вложенных друг в друга контейнеров).

У контейнеров Vendor и Inoption не может быть вложенных контейнеров.

Таблица 66. Ключевые слова, допустимые значения и описания записей операторов вложенных контейнеров

Ключевое слово	Значение	Описание
class	*имя* [*диапазон*] {ОПЦИИ|ОБЩИЕ ОПЦИИ }	Контейнер class. Значением поля *имя* может быть одна строка, строки через пробел, регулярное выражение, значение в формате hex 0x*шестнадцатеричное_число* или значение в формате 0x*шестнадцатеричное_число*
vendor	*имя* [*диапазон*] {ОПЦИИ|ОБЩИЕ ОПЦИИ }	Контейнер vendor. Значением поля *имя* может быть одна строка, строки через пробел, регулярное выражение, значение в формате hex 0x*шестнадцатеричное_число* или значение в формате 0x*шестнадцатеричное_число*
client	<*ИД* | 0 0xшестнадцатеричное число | регулярное выражение> <ip | *диапазон* | none | any> {ОПЦИИ|ОБЩИЕ ОПЦИИ }	Контейнер client. *ид* - одно-, двух- или трехзначное шестнадцатеричное число, <ip|диапазон|none|any> - IP-адрес для выделения клиентам с заданным ИД
inoption	*код ключ* [*диапазон*] { ОПЦИИ|ОБЩИЕ ОПЦИИ }	Контейнер inoption *код* - входящие код или число опции, указанные клиентом *ключ* - данные, относящиеся к опции, для сравнения.

Опции файла DHCPv6 cnf:

Описанные здесь опции файла `cnf` для **DHCPv6** могут существовать только внутри контейнера.

Таблица 67. Ключевые слова, допустимые значения и описания записей в разделе опций

Ключевое слово	Значение	Описание
exclude	*диапазон*	Диапазон IP-адресов, который нужно исключить из текущего диапазона. Часто используется, когда диапазон не указан как часть оператора контейнера.
exclude	IP	IP-адрес, который следует исключить из текущего диапазона
range	*диапазон*	Диапазон IP-адресов, который должен расширить текущий диапазон. Часто используется, когда диапазон не указан как часть оператора контейнера
диапазон	IP	IP-адрес, который следует добавить к текущему диапазону для его расширения
stealfromchildren	*стратегия*	Указывает, что следует взять адрес у вложенных контейнеров, если все адреса заняты. По умолчанию эта опция выключена.
stealfrompeer	*стратегия*	Указывает, что следует взять адрес у контейнеров того же уровня, если все адреса заняты. По умолчанию эта опция выключена.
stealfromparent	*стратегия*	Указывает, что следует взять адрес у контейнеров родительского уровня, если все адреса заняты. По умолчанию эта опция выключена.
balance-option	{ стратегия | <опция | опция опция ...> }	Контейнер опций балансировки. Указанные в нем опции будут переданы клиенту с учетом стратегии. Данное ключевое слово допустимо только на уровне контейнера subnet.
balance-policy	стратегия	В качестве значения "стратегия" может быть указано `fill` или `rotate`. Значение по умолчанию - `rotate`.
fill-count	число	Указывает, сколько раз опция будет пропущена, прежде чем будет обработана
interface-id	"*интерфейс*"	Допустимо только в подсети. Клиентским запросам, полученным по этому интерфейсу, будет разрешено получать адреса.

Общие опции DHCPv6:

Эти ключевые слова являются общими опциями **DHCPv6**.

Могут находиться внутри контейнеров или в глобальном разделе:

Таблица 68. Ключевые слова, допустимые значения и описания общих опций

Ключевое слово	Значение	Описание
reconfig-policy	*стратегия*	Разрешает серверу отправлять клиенту сообщение о повторной настройке. По умолчанию эта опция не задана и считается отключенной.
rapid-commit	*стратегия*	Позволяет серверу быстро фиксировать контейнер или глобальную группу. По умолчанию эта опция не задана и считается отключенной.
prefered-lifetime	`число [`*единицы измерения*`]`	Предпочитаемый срок действия IANA или IATA. Значение по умолчанию - 43200 секунд.
valid-lifetime	`число [`*единицы измерения*`]`	Действительный срок действия IANA или IATA. Значение по умолчанию - 86400 секунд.
rebind	`число`	Доля времени в процентах (0-100) на повторное выделение адреса. Значение по умолчанию - 80 процентов.
renew	`число`	Доля времени в процентах (0-100) на продление срока действия адреса. Значение по умолчанию - 50 процентов.
unicast-option	`стратегия`	Позволяет контейнерам предлагать обмен сообщениями с помощью направленной рассылки, что позволяет включать и выключать отдельные контейнеры и подсети даже при иной стратегии сервера. По умолчанию эта опция не задана и считается отключенной.
option	`число <строка|строка|hex>`	Список опций приведен в разделе “известные опции файла сервера DHCPv6”.
change-optiontable	`таблица опций`	Разрешено только в контейнере vendor.

известные опции файла сервера DHCPv6:

Здесь описаны известные опции файла сервера **DHCPv6**.

Ниже приведены поддерживаемые опции файла сервера **DHCPv6**. Опции, для которых в столбце **Можно указывать?** стоит "нет", нельзя указывать в файле конфигурации; если же они будут указаны, то они будут проигнорированы.

Номер опции	Тип данных по умолчанию	Можно указывать?	Описание
1	Нет	Нет	Требование
61 см	Нет	Нет	Уведомление
3	Нет	Нет	Запрос
4	Нет	Нет	Подтверждение
5	Нет	Нет	Адрес
6	Нет	Нет	Запрос опции
7	число	Нет	Предпочитаемый номер сервера
8	Нет	Нет	Затраченное время
9	Нет	Нет	Промежуточное сообщение
11	Нет	Нет	Идентификация
12	Строка ASCII - yes, no, true, false	Да	Направленная рассылка
13	Нет	Нет	Состояние
14	Строка ASCII - yes, no, true, false	Да	Быстрая фиксация
15	Нет	Нет	Класс пользователя
16	Нет	Нет	Класс вендора
17	Нет	Нет	Опция вендора

Номер опции	Тип данных по умолчанию	Можно указывать?	Описание
18	Нет	Нет	ИД интерфейса
19	Нет	Нет	Сообщение о повторной настройке
20	Строка ASCII - yes, no, true, false	Да	Принятие повторной настройки
23	Адреса IPv6 через пробел	Да	Серверы DNS
24	Строка ASCII	Да	Список доменов

Значения параметров DHCPv6:

Для параметров **DHCPv6** можно использовать следующие значения.

единицы измерения: second, seconds, minute, minutes, hour, hours, day, days, week, weeks, month, months, year, years

интерфейс: en0, en1, tr0

идентификатор: числа или символы

стратегия: yes, no, true, false

диапазон: ipv6_адрес-ipv6_адрес

регулярное выражение: "!выражение для сравнения$", "!выражение для сравнения^"

Пример файла /etc/dhcpv6/dhcpsdv6.cnf:

Показанный здесь пример файла `/etc/dhcpv6/dhcpsdv6.cnf` позволяет ознакомиться с содержимым файла.

```
logging_info{
        logFileSize 4000
        logItem     SYSERR
        logItem     PROTERR
        logItem     WARNING
        logItem     EVENT
        logItem     ACTION
        logItem     INFO
        logItem     ACNTING
        logItem     TRACE
        numLogFiles     3
        logFileName "/var/tmp/dhcpsdv6.log"
}
duid 1 en0
numprocessthreads 10
numpacketthreads  5
preference-number 255
reconfig-policy no
rapid-commit  no
unicast-option    yes
leaseExpiredInterval  3000 seconds
unicast-enable          yes
saveInterval            60 seconds
reservedTimeInterval    8000 seconds
reservedTime            10000 seconds
clientpruneintv         20  seconds

subnet bbbb:aaaa:: 40 bbbb:aaaa::0004-bbbb:aaaa::000f {
        balance-option {
               option 23 dead::beef
               option 23 beef::aaaa
```

```
                option 24 yahoo.com
        }

subnet dead:dead:aaaa:: 48 dead:dead:aaaa:aaaa::0006-dead:dead:aaaa:aaaa::000a {
        interface-id "en1"
        preferred-lifetime      100 seconds
        valid-lifetime          200 seconds
        rapid-commit  yes
        option 23 dead::beef beef:aaaa::bbbb:c aaaa:bbbb::cccc
        option 24 ibm.com austin.ibm.com
}
```

Настройка клиента DHCPv6

Для настройки клиентов **DHCPv6** служит файл `/etc/dhcpv6/dhcpc6.cnf`.

Ниже описаны директивы, которые можно включать в этот файл. Если **dhcpcd6** следует запускать при загрузке, добавьте запись в файл `/etc/rc.tcpip`.

Ключевые слова ведения протокола:

Здесь описаны допустимые ключевые слова ведения протоколов сервера **DHCPv6**.

Допустимы следующие ключевые слова:

Таблица 69. Ключевые слова ведения протокола и их описания

Ключевое слово	Описание
log-file-name	Путь и имя последнего файла протокола. К именам предыдущих файлов протокола добавляются номера от 1 до (n - 1); чем больше число, тем раньше был создан файл.
log-file-size	Максимальный размер файла протокола в килобайтах. Когда размер последнего файла протокола достигает этого значения, файл переименовывается и создается новый файл протокола.
log-file-num	Максимальное число файлов протокола, сохраняемых в случае, когда размер последнего файла достигает значения **log-file-size**, этот файл переименовывается и создается новый файл.
log-item	Элементы, которые следует заносить в протокол. **SYSERR** Системная ошибка **OBJERR** Ошибка объекта **PROTERR** Ошибка протокола **WARNING** Предупреждение **EVENT** Произошло событие **ACTION** Процесс выполнил некоторое действие **INFO** Дополнительная информация **ACNTING** Кто и когда был обслужен **TRACE** Коды для отладки

Ключевые слова DUID:

Ниже приведены значения ключевых слов для записей DUID.

Формат записей DUID следующий:

```
duid  <тип_duid> <значение>  <значение> ...
```

Тип DUID может быть ключевым словом или числом, что позволяет определять любые типы DUID, которые могут понадобиться в будущем. В настоящее время в RFC 3315 определены три типа DUID:

Таблица 70. Ключевые слова и значения для записей DUID

Ключевое слово	Описание
LLT	Тип DUID-LLT (значение 1)
LL	DUID-LL (значение 2)
EN	Тип DUID-EN (значение 3)

Особый формат записей DUID зависит от используемого ключевого слова.

```
duid  LLT    <имя интерфейса>
duid  LL     <имя интерфейса>
duid  EN     <номер предприятия>  <идентификатор предприятия>
duid  <число>  <шестнадцатеричные данные (с префиксом '0x')
```

Информационное ключевое слово:

Информационное ключевое слово имеет формат `info-only` *`имя-интерфейса`*.

Ниже описано информационное ключевое слово:

Таблица 71. Информационное ключевое слово и его описание

Ключевое слово	Описание
`info-only` *`имя интерфейса`*	Имя интерфейса, для которого клиент должен получить от сервера только информацию о конфигурации, но не адреса.

Ключевые слова продления срока действия и повторного выделения адреса:

Здесь приведено описание ключевых слов продления срока действия и повторного выделения адреса для сервера **DHCPv6**.

Таблица 72. Ключевые слова продления срока действия адреса и повторного выделения адреса

Ключевые слова	Описание
`rebind-time` *`значение`*	Если клиент не смог продлить срок действия выделенного ему адреса (из-за того, что сервер не ответил), то этот параметр задает время, за которое клиент должен соединиться с другими серверами для нового выделения адреса.
`renew-time` *`значение`*	Время, за которое клиент должен связаться с сервером, выделившим ему адрес, для продления срока действия адреса.

Ключевые слова повторной передачи запроса:

К ключевым словам повторной передачи запроса относятся **solicit-maxcount** и **solicit-timeout**.

Таблица 73. Ключевые слова повторной передачи запроса и их описания

Ключевые слова	Описания
solicit-maxcount	Ключевое слово solicit-maxcount задает число сообщений, которое клиент должен отправить серверу до получения ответа от сервера.
solicit-timeout	Время, в течение которого клиент должен пытаться отправить сообщение с запросом серверу, пока не получит ответ от сервера.

Ключевые слова опций:

Если ключевые слова опций появляются вне разделов 'интерфейс', то они расцениваются как глобальные. Эти опции применяются ко всем интерфейсам. Если ключевые слова опций появляются внутри разделов 'интерфейс', то они применяются только к этому интерфейсу.

Раздел опций имеет следующий формат:

```
option <ключевое слово | код опции>
option <ключевое слово | код опции> exec "выполняемая строка"
option <ключевое слово | код опции> { параметры опции }
option <ключевое слово | код опции> { параметры опции } exec "выполняемая строка"
```

Код опции можно указать с помощью зарегистрированного в IANA кода опции. Некоторые опции можно также указать с помощью приведенных ниже ключевых слов:

Ключевое слово	Код опции
ia-na	3
ia-ta	4
request-option	6
rapid-commit	14
user-class	15
vendor-class	16
vendor-opts	17
reconf-accept	20
dns-servers	23
domain-list	24

Ниже приведено объяснение каждого ключевого слова:

Ключевое слово	Назначение, формат и параметры
ia-na	**Назначение** Указывает опцию 3. Означает, что клиент запрашивает у сервера не временные адреса. **Формат** `option ia-na [ { параметры } ] [ exec "выполняемая строка" ]` **Параметры** Опция **ia-na** распознает следующие параметры: ia-id *значение* renew-time *значение* rebind-time *значение* Эти параметры указывают предпочитаемые пользователем значения и являются необязательными. *Значением* может быть десятичное число или шестнадцатеричное число с префиксом '0x'.

Ключевое слово	Назначение, формат и параметры
ia-ta	**Назначение** Указывает опцию 4. Означает, что клиент запрашивает временные адреса с сервера. **Формат** `option ia-ta [ { параметры } ] [ exec "выполняемая строка" ]` **Параметры** Опция **ia-ta** распознает следующие параметры: ia-id *значение* Этот параметр указывает предпочитаемые пользователем значения и является необязательным. *Значением* может быть десятичное число или шестнадцатеричное число с префиксом '0x'.
request-option	**Назначение** Указывает опцию 6. Означает, что клиент запрашивает список опций с сервера. **Формат** `option request-option { параметры } [ exec "выполняемая строка" ]` **Параметры** Аргументом опции **request-option** служит список кодов опций (в десятичном формате), разделенных пробелами
rapid-commit	**Назначение** Указывает опцию 14. Означает, что клиент сообщает о своей готовности к обмену сообщениями запрос-ответ. **Формат** `option rapid-commit [exec "выполняемая строка"]` **Параметры** Единственный возможный параметр - необязательный выполняемый оператор
user-class	**Назначение** Указывает опцию 15. Означает, что клиент указывает тип или категорию пользователя или приложений, которые он представляет. **Формат** `option user-class { параметры } [ exec "выполняемая строка" ]` **Параметры** В опции **user-class** указывается один или несколько экземпляров данных класса пользователя. Каждый экземпляр данных класса пользователя - это строка произвольной длины в кавычках или без кавычек. Если строка содержит символ пробела, ее следует заключить в кавычки. Параметры обязательны. Формат параметров следующий: class *значение* class *значение* где *значение* - это строка в кавычках или без кавычек.
vendor-class	**Назначение** Указывает опцию 16. Означает, что клиент предоставляет сведения о производителе используемого им аппаратного обеспечения. **Формат** option vendor-class { *параметры* } [exec "выполняемая строка"] **Параметры** В опции **vendor-class** указывается зарегистрированный номер предприятия вендора и один или несколько экземпляров данных класса вендора. Каждый экземпляр данных класса вендора - это строка произвольной длины в кавычках или без кавычек, описывающая некоторые параметры аппаратной конфигурации клиента. Эти параметры *обязательны*. Формат команды следующий: vendor-id *значение* class *значение* class *значение* где *значение* - это строка в кавычках или без кавычек.

Ключевое слово	Назначение, формат и параметры
vendor-opts	**Назначение** Указывает опцию 17. Означает, что клиент сообщает серверу информацию о вендоре. **Формат** `option vendor-opts <номер компании> { параметры } [ exec "выполняемая строка" ]]` **Параметры** В опции **vendor-opts** указывается зарегистрированный номер предприятия вендора и один или несколько экземпляров данных опции вендора. Каждый экземпляр данных опции вендора - это код опции вендора, за которым следуют данные опции в строковом или шестнадцатеричном формате. Эти параметры *обязательны*. Формат команды следующий: vendor-id *значение* option *код опции данные опции* option *код опции данные опции* где *данные опции* - это строка в кавычках или без кавычек или шестнадцатеричная строка с префиксом '0x'
reconf-accept	**Назначение** Указывает опцию 20. Означает, что клиент указывает серверу, желает ли он принять от сервера сообщение о повторной настройке. **Формат** `option reconf-accept [ { exec "выполняемая строка" } ]` **Параметры** Единственный возможный параметр **reconf-accept** - выполняемый оператор.
dns-servers	**Назначение** Указывает опцию 23. Означает, что клиент сообщает серверу предпочитаемый набор серверов DNS. **Формат** `option dns-servers [ { параметры } ] [ exec "выполняемая строка" ]` **Параметры** Аргументом опции **dns-servers** служит список адресов IPv6, разделенных пробелами или указанных в виде отдельных строк.
domain-list	**Назначение** Указывает опцию 24. Означает, что клиент указывает список предпочитаемых доменов. **Формат** `option domain-list [ { параметры } ] [ exec "выполняемая строка" ]` **Параметры** Аргументом опции **domain-list** служит список имен доменов, разделенных пробелами или указанных в виде отдельных строк.

Ключевые слова интерфейса:

Ключевое слово интерфейса имеет следующий формат: `interface <имя_интерфейса> [ { объявления опций } ]`.

Таблица 74. Ключевые слова интерфейса и их описания

Ключевые слова	Описания
`interface <имя_интерфейса> [ { объявления опций } ].`	Аргументами оператора interface служат объявления опций (одно или несколько). Если опции объявлены в разделе интерфейса, то они относятся к этому интерфейсу; если же опции объявлены вне раздела интерфейса, то они относятся ко всем интерфейсам.

```
interface  en1  {
             option ia-na {
            ia-id  01
            renew-time 0x40
            rebind-time 0x60
 }

 option  request-option  { 3 23 24 }
```

```
 option  user-class {
    class ibm
         class "userclassA and B"
         class "userclassB"
 }

option  vendor-class {
      vendor-id 1234
         class "vendorclassA"
         class "vendorclassB"
}

option  vendor-opts {
    vendor-id 2343
         option 89       vendoroption89
         option 90       vendoroption90
}

option reconf-accept
```

Промежуточный агент DHCP

Файл `/etc/dhcprd.cnf` содержит конфигурацию промежуточного агента **DHCP** и **BOOTP**. Ниже приведены сведения о формате файла и допустимых ключевых словах и директивах.

Директивы указываются в следующем формате:

<ключевое слово> <значение1> ... <значениеN>

Эти параметры и их значения используются для промежуточного агента при его запуске или повторном запуске.

Этот набор параметров указывает файлы протоколов, поддерживаемые сервером. Каждый параметр идентифицируется ключевым словом и следующим за ним значением.

Ключевое слово	Значение	Определение
numLogFiles	От 0 до *n*	Число файлов протокола. Если указано значение 0, файлы протокола не используются и сообщения протокола не выдаются. *n* - это максимальное число файлов протокола, сохраняемых в случае, когда размер последнего файла достигает максимального значения и создается новый файл.
logFileSize	В Кб	Максимальный размер файла протокола. Когда размер последнего файла протокола достигает этого значения, файл переименовывается и создается новый файл протокола.
logFileName	Путь к файлу	Имя последнего файла протокола. К именам предыдущих файлов протокола добавляются номера от 1 до (*n* - 1); чем больше число, тем раньше был создан файл.

Ключевое слово	Значение	Определение
logItem	Один элемент, который будет записан в протоколе.	**SYSERR** Системная ошибка в интерфейсе платформы. **OBJERR** Ошибка объекта, возникшая в процессе. **PROTERR** Ошибка протокола между клиентом и сервером. **WARNING** Предупреждение, призывающее пользователя обратить внимание. **EVENT** В процессе произошло некоторое событие. **ACTION** Процесс выполнил некоторое действие. **INFO** Информация, которая может оказаться полезной. **ACNTING** Кто и когда был обслужен. **TRACE** Коды для отладки.

Например, файл `/etc/dhcprd.cnf` может содержать следующие записи:

```
numLogFiles  4
logFileSize  1000
logFileName    /usr/tmp/dhcprd.log
logItem     SYSERR
logItem     OBJERR
logItem     PROTERR
logItem     WARNING
logItem     EVENT
logItem     ACTION
logItem     INFO
logItem     ACNTING
logItem     TRACE
```

Ключевое слово	Значение	Определение
relay	**IPv4**, **IPv6** или **ВСЕ**	Режим получателя пакетов. Если указано **IPv4**, то промежуточный агент работает как промежуточный агент **DHCPv4**. Этот режим применяется по умолчанию. Если указано **IPv6**, то промежуточный агент работает как промежуточный агент **DHCPv6**. Если указано **ALL**, то промежуточный агент работает как промежуточный агент **DHCPv4** и **DHCPv6**.
сервер	IP-адрес	IP-адрес сервера **BOOTP** или **DHCP**. Пакет будет переслан серверам, указанным в этом списке.
server6	Адрес IPv6	Адрес IPv6 сервера **DHCPv6**. Пакет будет переслан серверам, указанным в этом списке.

Ключевое слово	Значение	Определение
option6	*<код опции> <данные опции>*	Опции промежуточного агента **DHCPv6**. Ключевое слово допустимо, только если выбран режим IPv6. Значение *кода опции* - десятичное число. Значение *данных опции* - строка с кавычками или без кавычек или строка в шестнадцатеричном формате (с префиксом `0x`)
single-site		Означает, что устройство, на котором запущен промежуточный агент, принадлежит только одной системе.

Демон PXE Proxy DHCP

Сервер PXE Proxy **DHCP** работает примерно так же, как и сервер **DHCP**: он просматривает сообщения клиентов **DHCP** и отвечает на некоторые запросы. Однако, в отличие от сервера **DHCP**, сервер PXE Proxy **DHCP** не управляет сетевыми адресами, а всего лишь отвечает на запросы клиентов PXE.

В ответе сервера PXE Proxy **DHCP** содержится расположение загрузочного сервера, либо сетевой адрес и описание поддерживаемых и совместимых загрузочных серверов.

Применение сервера PXE Proxy **DHCP** вместе с сервером **DHCP** позволяет получить определенные преимущества. Во-первых, вы можете отделить администрирование сетевых адресов от администрирования загрузочных образов. Один процесс системы может управлять информацией о загрузке с помощью сервера PXE Proxy **DHCP**, а другой независимый процесс - конфигурацией сервера **DHCP**. Во-вторых, вы можете определить несколько загрузочных серверов, позволяя клиенту выбрать сервер непосредственно перед загрузкой. Различные загрузочные серверы могут содержать данные для различных операционных систем или конфигураций. Кроме того, сервер Proxy позволяет клиентам PXE находить совместимые серверы загрузки путем многоцелевой рассылки.

Сервер PXE Proxy **DHCP** может работать в той же системе, что и сервер **DHCP**, или в другой системе. Кроме того, его можно настроить в той же системе, в которой работает демон загрузочного сервера.

Компоненты proxy-сервера DHCP PXE

Сервер PXED состоит из трех основных компонентов.

Сервер PXED состоит из трех основных компонентов: базы данных, средств поддержки протокола и набора служебных нитей, для каждой из которых задается собственная информация о конфигурации.

База данных PXED:

Для создания списка ответов на пакет REQUEST, полученный от клиента, применяется база данных `db_file.dhcpo`.

Значения, возвращаемые базой данных, зависят от выбранного типа сервера. Они задаются с помощью ключевого слова **pxeservertype** в файле `pxed.cnf`.

База данных заполняется и проверяется в соответствии с информацией, заданной в файле конфигурации.

Служба протокола PXED:

Отправляемая клиентам информация о конфигурации определяется с помощью базы данных.

Модуль протокола PXED основан на спецификации Intel Preboot Execution Environment (PXE) версии 2.1 и все еще совместим со спецификацией Intel PXE версии 1.1.

Операции с нитями PXED:

Третий компонент сервера PXED - набор операций, которые, собственно, и обеспечивают его работу. Так как в сервере PXED реализована поддержка нескольких нитей, то эти операции в действительности организованы в виде набора нитей, которые в определенное время выполняют нужные действия и обеспечивают правильную работу.

Первая нить, *main*, обрабатывает запросы SRC (такие как **startsrc**, **stopsrc**, **lssrc**, **traceson** и **refresh**). Кроме того, эта нить согласовывает все операции, влияющие на остальные нити, и обрабатывает сигналы. Пример:

- SIGHUP (-1) обновляет все базы данных в файле конфигурации.
- SIGTERM (-15) завершает работу сервера в нормальном режиме.

Остальные нити выполняют обработку пакетов. В зависимости от типа сервера, может быть создана одна или две нити. Первая нить принимает запросы через порт 67, а вторая - через порт 4011. Обе нити могут обрабатывать запросы клиентов.

Конфигурация сервера PXED

По умолчанию сервер PXED считывает информацию из файла `/etc/pxed.cnf`, в котором хранится исходная база данных параметров и адресов сервера.

Сервер можно запустить с помощью SMIT или команд SRC.

Настройка сервера PXED - это самая трудная часть настройки PXED в сети. Во-первых, определите, в каких сетях будут размещаться клиенты PXE. Ниже приведен пример конфигурации демона **pxed**, работающего в одной системе с сервером DHCP:

```
pxeservertype       proxy_on_dhcp_server

subnet default
{
    vendor pxe
    {
        option    6      2       # Запретить поиск загрузочных серверов методом рассылки
        option    8      1   2     9.3.4.5    9.3.4.6   2    1    9.3.149.29
                                   # Выше приведен список загрузочных серверов
        option    9      0       "PXE bootstrap server" \
                                 1       "Microsoft Windows NT Boot Server"  \
                                 2       "DOS/UNDI Boot Server"
        option    10    20    "задержка (в секундах) перед автоматическим выбором первой опции меню"
     }
}
```

Опции контейнера vendor отправляются клиенту PXE только в том случае, если IP-адрес клиента принадлежит диапазону адресов подсети (9.3.149.0 - 9.3.149.255).

Ниже приведен пример конфигурации демона **pxed**, работающего в другой системе, чем сервер **DHCP**:

```
subnet default
{
    vendor pxe
    {
        option    6      10      # Имя файла загрузки указано в начальном пакете pxed,
                                 # отправляемом клиенту
        option    8      1   2     9.3.4.5    9.3.4.6   2    1    9.3.149.29
                                   # Выше приведен список загрузочных серверов
        option    9      0       "PXE bootstrap server" \
                                 1       "Microsoft Windows NT Boot Server"  \
                                 2       "DOS/UNDI Boot Server"
        option    10    20    "задержка (в секундах) перед автоматическим выбором первой опции меню"
        bootstrapserver          9.3.148.65
        pxebootfile    1    2    1   window.one
        pxebootfile    2    2    1   linux.one
```

```
        pxebootfile    1    2    1   hello.one
        client 6 10005a8ad14d any
        {
           pxebootfile    1    2    1   aix.one
           pxebootfile    2    2    1   window.one
        }
    }

Vendor pxeserver
{
    option     7     224.234.202.202
}
```

В файле конфигурации не указано ключевое слово **pxeservertype**, поэтому будет применяться значение по умолчанию **pdhcp_only**, означающее, что сервер PXED работает на другом компьютере, чем сервер **DHCP**. В этой конфигурации сервер PXED принимает пакеты BINLD REQUEST/INFORM через два порта (67 и 4011). Опция 7 отправляется серверу BINLD, когда сервер PXED получает от него пакет REQUEST/INFORM через порт 67, а опция 60 указывает на сервер PXED.

Директива `db_file` указывает, какой метод базы данных должен применяться для обработки этой части файла конфигурации. Комментарии начинаются с символа #. Текст, стоящий в строке после символа #, игнорируется сервером PXED. С помощью каждой строки `option` сервер задает для клиента какое-либо действие. В разделе “Опции контейнера вендора PXE” на стр. 306 описаны все поддерживаемые и известные опции. В разделе “Синтаксис файла сервера PXED для общих операций сервера” на стр. 307 описаны способы задания опций, неизвестных серверу.

Файл конфигурации PXED:

Файл конфигурации состоит из раздела адресов и раздела определения опций. Оба этих раздела реализуют концепцию контейнера, который может включать опции, модификаторы и, возможно, другие контейнеры.

Контейнер (по существу, это способ группирования параметров) позволяет объединять клиентов в группы на основании идентификатора. Типы контейнеров: subnet, class, vendor и client. Контейнеры, определяемые пользователем, в настоящее время не поддерживаются. Клиент однозначно определяется своим идентификатором, что позволяет всегда точно обнаруживать его, например, при переносе в другую подсеть. Для описания клиента может применяться несколько контейнеров.

Опции - это идентификаторы, возвращаемые клиенту. Это может быть, например, применяемые по умолчанию адреса шлюза и сервера DNS.

Контейнеры PXED:

При получении запроса сервер **DHCP** анализирует пакет и на основании ключей идентификации определяет, какие нужно выбрать контейнеры, параметры и адреса.

На примере конфигурации сервера PXED рассмотрен контейнер подсети. Его ключ идентификации - это расположение клиента в сети. Если клиент находится в данной сети, то он попадает в этот контейнер.

Для идентификации клиентов в разных типах контейнеров применяются различные опции:

- Для определения подсети, в которой находится клиент, контейнер subnet использует поле `giaddr` или адрес принимающего интерфейса.
- Контейнер class использует значение опции 77 (идентификатор класса пользователя).
- Контейнер vendor использует значение опции 60 (идентификатор класса вендора).
- Контейнер client использует значение опции 61 (идентификатор клиента) для клиентов PXE и поле `chaddr` из пакета **BOOTP** для клиентов **BOOTP**.

Во всех контейнерах, кроме subnet, можно применять шаблоны сравнения, включая регулярные выражения.

Существует также неявный контейнер *global*. Опции и модификаторы, указанные в контейнере global, действуют во всех контейнерах, если только они не отключены и не переопределены. Большинство контейнеров можно поместить в другие контейнеры в соответствии с областью видимости. С контейнерами могут быть связаны диапазоны адресов. С подсетями всегда связаны диапазоны адресов.

Основные правила для контейнеров и подконтейнеров:

- На глобальном уровне допустимы все контейнеры.
- Контейнеры subnet нельзя помещать в другие контейнеры.
- В контейнеры с ограничениями нельзя помещать обычные контейнеры того же типа. (Например, если контейнер содержит опцию, разрешающую применять только класс `Accounting`, то в него нельзя поместить контейнер, который содержит опцию, разрешающую применение всех классов, имена которых начинаются с буквы "a". Это недопустимо.)
- Контейнеры client с ограничениями не могут содержать вложенные контейнеры.

С помощью этих правил вы можете создавать иерархию контейнеров, в которой опции объединены в наборы, соответствующие конкретным клиентам или группам клиентов.

Каким образом организована обработка адресов и опций, если клиент входит в несколько контейнеров? Сервер **DHCP** получает сообщение, формирует запрос к базе данных (в данном случае - к файлу `db_file`) и получает список контейнеров. Контейнеры перечисляются в списке в порядке их вложенности и приоритета. Приоритет определяется как неявный иерархический уровень контейнера. Контейнеры с ограничениями имеют более высокий приоритет, чем обычные. Сортировка контейнеров выполняется в таком порядке: клиенты, классы, вендоры и подсети. В пределах одного типа контейнеры упорядочиваются по уровню вложенности. Созданный таким образом список упорядочивается от более конкретных объектов к менее конкретным. Например:

```
Subnet 1
--Class 1
--Client 1
Subnet 2
--Class 1
----Vendor 1
----Client 1
--Client 1
```

В данном примере есть две подсети `Subnet 1` и `Subnet 2`. Кроме того, определен один класс, `Class 1`, один вендор, `Vendor 1`, и один клиент, `Client 1`. `Class 1` и `Client 1` определены в нескольких местах. Поскольку эти определения находятся в разных контейнерах, имена объектов могут совпадать, однако указанные в них значения могут различаться. Если клиент `Client 1` отправит сообщение серверу **DHCP** из подсети `Subnet 1` с указанием класса `Class 1`, определенного в списке опций этого клиента, то сервер **DHCP** создаст следующий список контейнеров:

`Subnet 1, Class 1, Client 1`

Контейнер, определенный наиболее точно, заносится в список последним. Для получения адреса список просматривается в обратном порядке до обнаружения первого доступного адреса. Затем список просматривается в прямом порядке (в соответствии с иерархией) для получения опций. По мере просмотра списка, новые значения опций переопределяют прежние значения, если в контейнере не задана опция **deny**. Поскольку класс `Class 1` и клиент `Client 1` находятся в одной и той же подсети `Subnet 1`, они упорядочиваются в соответствии с приоритетом контейнеров. Если сообщение будет получено от клиента с тем же именем, находящегося в подсети `Subnet 2`, то будет создан следующий список контейнеров:

`Subnet 2, Class 1, Client 1` (на уровне `Subnet 2`), `Client 1` (на уровне `Class 1`)

Первой в списке указывается подсеть `Subnet 2`, затем класс `Class 1`, затем клиент `Client 1` на уровне `Subnet 2` (так как этот клиент находится в иерархии на один уровень ниже). Иерархия подразумевает, что клиент, имя которого указано в первом операторе, определен менее конкретно, чем клиент `Client 1`, определенный в классе `Class 1` подсети `Subnet 2`.

Приоритет, определяемый по уровню вложенности, выше, чем приоритет самих контейнеров. Например, если тот же клиент отправит такое же сообщение, указав идентификатор вендора, то список контейнеров будет следующим:

`Subnet 2, Class 1, Vendor 1, Client 1` (на уровне `Subnet 2`), `Client 1` (на уровне `Class 1`)

Организация поиска на основании приоритета контейнера повышает эффективность, поскольку контейнеры client обеспечивают наиболее точный способ определения одного или нескольких клиентов. В контейнере class адреса определены менее конкретно, чем в контейнере client; в контейнере vendor адреса определены еще менее конкретно, а контейнер subnet содержит самые общие определения адресов.

Адреса и диапазоны адресов PXED:

С контейнерами любого типа могут быть связаны диапазоны адресов. Контейнеры subnet обязательно имеют связанный диапазон адресов. Каждый диапазон адресов в пределах контейнера должен быть подмножеством диапазона адресов родительского контейнера и не должен пересекаться с диапазонами адресов других контейнеров.

Например, если внутри подсети определен класс с диапазоном адресов, то этот диапазон должен быть подмножеством диапазона адресов подсети. Диапазон внутри этого контейнера класса не должен пересекаться с любыми другими диапазонами этого уровня.

Диапазоны адресов могут задаваться в строке контейнера. Для задания несмежных диапазонов адресов можно воспользоваться операторами range и exclude. Таким образом, если в подсети есть два диапазона по десять адресов, то имеет смысл указать эти диапазоны в операторе subnet, чтобы уменьшить объем памяти и избежать конфликтов адресов с другими клиентами.

Когда адрес выбран, все последующие контейнеры, содержащие диапазоны адресов, удаляются из списка вместе со своими дочерними контейнерами. Необходимость такой процедуры обусловлена тем, что сетевые опции, заданные в удаленных контейнерах, недопустимы для адресов, не принадлежащих данному контейнеру.

Опции файла конфигурации PXED:

После первого просмотра списка и получения адресов для клиента создается набор опций.

В процессе выбора ранее определенные значения опций переопределяются новыми до тех пор, пока не встретится опция *deny* (запретить); при этом запрещенная опция удаляется из списка опций, отправляемых пользователю. Этот способ разрешает наследование опций родительских контейнеров и сокращает объем данных, которые нужно определять.

Ведение протоколов PXED:

Параметры протокола указываются в окне диалога, напоминающем окно настройки базы данных, но с ключевым словом **logging_info**.

На этапе обучения настройке PXED рекомендуется вести максимально подробный протокол. Кроме того, весьма полезно настроить протокол до начала работы с любыми другими файлами настройки, чтобы ошибки конфигурации были после инициализации системы занесены в протокол. Для включения опций протокола укажите ключевое слово **logitem**, а для их отключения удалите это ключевое слово. Другие ключевые слова позволяют задавать имя файла протокола, его размер, а также число взаимозаменяемых файлов протоколов.

Замечания о производительности PXED:

Обратите внимание, что набор ключевых слов и структура файла конфигурации влияют на объем используемой памяти и производительность сервера PXED.

Во-первых, понимание механизма наследования опций от контейнеров-предков к потомкам позволяет сократить объем используемой памяти. В среде, в которой не поддерживаются не указанные в списке клиенты, администратор должен явно перечислить всех клиентов в файле. При указании списка опций для каждого отдельного клиента на хранение дерева конфигурации расходуется больше ресурсов памяти сервера, чем при наследовании опций от контейнера-предка к потомку (в роли предка может выступать подсеть, сеть или глобальные контейнеры). Следовательно, администратор должен проверить, не повторяются ли какие-либо опции на уровне клиента в файле конфигурации, а если это так, то решить, можно ли указать эти опции в контейнере-предке и таким образом распространить их сразу на несколько клиентов.

Кроме того, если применяются записи **logItem** INFO и TRACE, то при обработке каждого сообщения клиента PXE в протокол заносится несколько сообщений. Добавление большого числа строк в файл протокола может привести к значительному увеличению нагрузки, поэтому ограничение объема заносимой в протокол информации позволяет повысить производительность сервера PXED. Вы можете динамически включать ведение протокола при возникновении ошибок в работе сервера PXED с помощью команды SRC **traceson**.

Опции контейнера вендора PXE

При поддержке клиента PXE, сервер **DHCP** передает на сервер BINLD следующую опцию, необходимую для самостоятельной конфигурации сервера BINLD:

Опция	Тип данных по умолчанию	Можно указывать?	Описание
6	Десятичное число	Да	PXE_DISCOVERY_CONTROL. Диапазон 0-16. Это битовое поле. Самый младший бит - бит 0. **бит 0** Если задан, отключает оповещение. **бит 1** Если задан, отключает многоцелевую рассылку. **бит 2** Если задан, то будут применяться только серверы из списка PXE_BOOT_SERVERS. **бит 3** Если этот бит задан, и в ответе PXED указано имя загрузочного файла, то система загружает этот файл (не показывая меню выбора загрузочного сервера). **биты 4-7** Должны быть равны 0. Если эта опция не указана, то клиент считает, что все биты равны 0.
7	Адрес	Да	IP-адрес для многоцелевой рассылки. Этот адрес применяется для поиска загрузочного сервера. Загрузочные серверы, поддерживающие поиск методом многоцелевой рассылки, должны принимать запросы, приходящие на этот адрес. Эта опция должна быть указана, если бит запрета многоцелевой рассылки (бит 1) в опции PXE_DISCOVERY_CONTROL не задан.

Опция	Тип данных по умолчанию	Можно указывать?	Описание
8	*Тип загрузочного сервера (0-65535)*	Да	PXE_BOOT_SERVERS *Число IP-адресов (0-256)* **Тип 0** Miscrosoft Windows *IP-адрес...IP-адрес* NT Boot Server *Тип загрузочного сервера IP-адрес* **Тип 1** Intel LCM Boot Server *число IP-адрес ...* **Тип 3** DOS/UNDI Boot Server *IP-адрес* **Тип 4** NEC ESMPRO Boot Server **Тип 5** IBM WSoD Boot Server **Тип 6** IBM LCCM Boot Server **Тип 7** CA Unicenter TNG Boot Server. **Тип 8** HP OpenView Boot Server. **Тип 9 - 32767** Зарезервировано. **Тип 32768 - 65534** Зарезервировано для вендора **Тип 65535** PXE API Test Server. Если для типа сервера указано нулевое *число IP-адресов*, то клиент может принимать ответы от любого загрузочного сервера этого типа. Загрузочные серверы не отвечают на запросы об обнаружении серверов тех типов, которые они не поддерживают.
9	*Тип загрузочного сервера (0-65535)*	Да	PXE_BOOT_MENU *"описание"* Порядок загрузки неявно задан в типе сервера. *"описание"...порядок в меню*.
10	*Тайм-аут в секундах (0-255)*	Да	PXE_MENU_PROMPT *"приглашение"* Тайм-аут - это время ожидания (в секундах), по истечении которого будет автоматически выбран первый пункт меню. На экране клиентской системы появится приглашение, после которого будет указано число секунд, оставшееся до автоматического выбора первого пункта меню. Если в клиентской системе будет нажата клавиша F8, то появится меню. Если указана эта опция, то меню появится автоматически, без вывода приглашения и тайм-аута. Если тайм-аут равен 0, то сразу будет выбран первый пункт меню. Если тайм-аут равен 255, то появится меню, и система будет ожидать выбора пользователя.

Синтаксис файла сервера PXED для общих операций сервера

Здесь описаны ключевые слова сервера PXED сервера **DHCPv6**, используемые для основных операций сервера. Описаны формы, субконтейнеры, значения по умолчанию и значения.

Примечание: Используемые в таблицы единицы времени (*единицы_времени*) указывать необязательно. Они применяются в качестве модификаторов значений времени. По умолчанию время указывается в минутах. Допустимо указывать время в секундах (1), минутах (60), часах (3600), сутках (86400), неделях (604800), месяцах (2392000) и годах (31536000). В скобках указан коэффициент, на который нужно умножить указанное значение времени, *n*, чтобы перевести его в секунды.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умол- чанию	Описание
database	database *тип базы данных*	Да	Нет	Основной контейнер, содержащий определения для пулов адресов, опций и операторов, задающих уровень доступа клиентов. *тип базы данных* - это имя модуля, который должен загружаться для обработки этой части файла. В текущей версии поддерживается только значение **db_file**.
logging_info	logging_info	Да	Нет	основной контейнер, определяющий параметры ведения протоколов.
logitem	logitem NONE	Нет	По умол- чанию все запре- щены.	Задает уровень ведения протокола. Можно указать несколько строк.
logitem	logitem SYSERR	Нет	По умол- чанию все запре- щены.	Задает уровень ведения протокола. Можно указать несколько строк.
logitem	logitem OBJERR	Нет	По умол- чанию все запре- щены.	Задает уровень ведения протокола. Можно указать несколько строк.
logitem	logitem PROTOCOL	Нет	По умол- чанию все запре- щены.	Задает уровень ведения протокола. Можно указать несколько строк.
logitem	logitem PROTERR	Нет	По умол- чанию все запре- щены.	Задает уровень ведения протокола. Можно указать несколько строк.
logitem	logitem WARN	Нет	По умол- чанию все запре- щены.	Задает уровень ведения протокола. Можно указать несколько строк.
logitem	logitem WARNING	Нет	По умол- чанию все запре- щены.	Задает уровень ведения протокола. Можно указать несколько строк.
logitem	logitem CONFIG	Нет	По умол- чанию все запре- щены.	Задает уровень ведения протокола. Можно указать несколько строк.
logitem	logitem EVENT	Нет	По умол- чанию все запре- щены.	Задает уровень ведения протокола. Можно указать несколько строк.
logitem	logitem PARSEERR	Нет	По умол- чанию все запре- щены.	Задает уровень ведения протокола. Можно указать несколько строк.
logitem	logitem ACTION	Нет	По умол- чанию все запре- щены.	Задает уровень ведения протокола. Можно указать несколько строк.
logitem	logitem ACNTING	Нет	По умол- чанию все запре- щены.	Задает уровень ведения протокола. Можно указать несколько строк.
logitem	logitem STAT	Нет	По умол- чанию все запре- щены.	Задает уровень ведения протокола. Можно указать несколько строк.
logitem	logitem TRACE	Нет	По умол- чанию все запре- щены.	Задает уровень ведения протокола. Можно указать несколько строк.
logitem	logitem RTRACE	Нет	По умол- чанию все запре- щены.	Задает уровень ведения протокола. Можно указать несколько строк.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умол- чанию	Описание
logitem	logitem START	Нет	По умол- чанию все запре- щены.	Задает уровень ведения протокола. Можно указать несколько строк.
numLogFiles	numLogFiles *n*	Нет	0	Указывает, сколько файлов протоколов нужно создать. Каждый последующий файл протокола создается после заполнения предыдущего. *n* - число создаваемых файлов.
logFileSize	logFileSize *n*	Нет	0	Задает размер каждого файла протокола в блоках по 1024 байта.
logFileName	logFileName *путь*	Нет	Нет	Задает путь к первому файлу протокола. Имя файла протокола имеет вид *имя_файла* или *имя_файла.расш*. *Имя_файла* должно состоять не более, чем из восьми символов. Имя следующего файла протокола создается на основе базового *имени_файла*, к которому добавляется номер, либо этот номер указывается вместо расширения. Например, если первому файлу присвоено имя `file`, то именем следующего файла будет `file01`. Если имя первого файла - `file.log`, то следующему файлу будет присвоено имя `file.01`.
pxeservertype	pxeservertype *тип_сервера*	Нет	dhcp_only	Указывает тип сервера **dhcpsd**. Если значение параметра *тип_сервера* равно **proxy_on_dhcp_server**, то сервер PXED работает на том же компьютере, что и сервер **DHCP**, и принимает запросы от клиентов PXE только через порт 4011; если значение этого параметра равно значению по умолчанию (**pdhcp_only**), то сервер PXED работает на другом компьютере и принимает запросы клиентов через порты 67 и 4011.

Синтаксис файла сервера PXED для базы данных db_file

Здесь описан синтаксис файла сервера PXED для базы данных `db_file`. Описаны формы, субконтейнеры, значения по умолчанию и значения.

Примечание:

1. Используемые в таблицы единицы времени (*единицы_времени*) указывать необязательно. Они применяются в качестве модификаторов значений времени. По умолчанию время указывается в минутах. Допустимо указывать время в секундах (1), минутах (60), часах (3600), сутках (86400), неделях (604800), месяцах (2392000) и годах (31536000). В скобках указан коэффициент, на который нужно умножить указанное значение времени, *n*, чтобы перевести его в секунды.
2. Элементы, описанные в одном контейнере, могут быть переопределены во вложенном контейнере. Например, можно определить клиентов **BOOTP** на глобальном уровне, но разрешить их работу только в конкретной подсети, указав ключевое слово supportBootp в обоих контейнерах.

3. В контейнерах client, class и vendor поддерживаются регулярные выражения. Если в контейнере class или vendor указана заключенная в кавычки строка, в которой после открывающей кавычки стоит символ !, то остаток строки будет обрабатываться как регулярное выражение. В контейнере client регулярные выражения можно указывать в полях **hwtype** и **hwaddr**. В обоих полях можно задавать строку следующего формата:

   ```
   десятичное_число-данные
   ```

   Если десятичное_число равно нулю, то данные представляют строку ASCII. Если указано любое другое число, то данные представляют собой набор шестнадцатеричных цифр.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умол- чанию	Описание
subnet	subnet default	Да	Нет	Определяет подсеть, с которой не связан диапазон адресов. Это подсеть применяется сервером только при ответе на запросы клиентов INFORM.
subnet	subnet *ид подсети маска*			Определяет подсеть и пул адресов. Если в строке не задан диапазон адресов и адреса не изменены в контейнере с помощью оператора range или exclude, то предполагается, что пул включает все адреса. Необязательный параметр диапазона представляет собой пару IP-адресов в десятичном формате с точками, разделенных дефисом. Можно (но не обязательно) указывать метку и приоритет. Они используются виртуальными подсетями для идентификации и упорядочения подсетей в виртуальной подсети. Метка и приоритет разделяются двоеточием. Эти контейнеры могут использоваться только на глобальном уровне или на уровне контейнера базы данных.
subnet	subnet *ид подсети маска диапазон*			Определяет подсеть и пул адресов. Если в строке не задан диапазон адресов и адреса не изменены в контейнере с помощью оператора range или exclude, то предполагается, что пул включает все адреса. Необязательный параметр диапазона представляет собой пару IP-адресов в десятичном формате с точками, разделенных дефисом. Можно (но не обязательно) указывать метку и приоритет. Они используются виртуальными подсетями для идентификации и упорядочения подсетей в виртуальной подсети. Метка и приоритет разделяются двоеточием. Эти контейнеры могут использоваться только на глобальном уровне или на уровне контейнера базы данных.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умол- чанию	Описание
subnet	subnet *ид подсети маска метка:приоритет*			Определяет подсеть и пул адресов. Если в строке не задан диапазон адресов и адреса не изменены в контейнере с помощью оператора range или exclude, то предполагается, что пул включает все адреса. Необязательный параметр диапазона представляет собой пару IP-адресов в десятичном формате с точками, разделенных дефисом. Можно (но не обязательно) указывать метку и приоритет. Они используются виртуальными подсетями для идентификации и упорядочения подсетей в виртуальной подсети. Метка и приоритет разделяются двоеточием. Эти контейнеры могут использоваться только на глобальном уровне или на уровне контейнера базы данных.
subnet	subnet *ид подсети маска диапазон метка:приоритет*			Определяет подсеть и пул адресов. Если в строке не задан диапазон адресов и адреса не изменены в контейнере с помощью оператора range или exclude, то предполагается, что пул включает все адреса. Необязательный параметр диапазона представляет собой пару IP-адресов в десятичном формате с точками, разделенных дефисом. Можно (но не обязательно) указывать метку и приоритет. Они используются виртуальными подсетями для идентификации и упорядочения подсетей в виртуальной подсети. Метка и приоритет разделяются двоеточием. Эти контейнеры могут использоваться только на глобальном уровне или на уровне контейнера базы данных.
subnet	subnet *ид подсети диапазон*	Да	Нет	Определяет подсеть, которая входит в контейнер сети. Задает диапазон адресов. Если не задан диапазон, то считается, что в подсеть входят все адреса. Маска подсети определяется родительским контейнером сети. **Примечание:** Задавать подсеть таким образом не рекомендуется.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умол- чанию	Описание
option	option *номер данные* ...	Нет	Нет	Задает опцию для отправки клиенту, или, в случае запрещения, опцию, отправка которой запрещена. Необязательная опция `* deny` означает, что все параметры, не определенные в данном контейнере, не должны возвращаться клиенту. Опция *номер*deny запрещает только указанную опцию. *Номер* - это 8-разрядное целое число без знака. *Данные* - это данные для опции (см. выше), заключенная в кавычки строка ASCII, либо шестнадцатеричные данные в одном из форматов: 0x*шестн._число* или hex "*шестн._число*" или hex"*шестн._число*". Если опция находится в контейнере vendor, то она будет включена вместе с другими опциями в опцию 43.
option	option *номер*deny	Нет	Нет	Задает опцию для отправки клиенту, или, в случае запрещения, опцию, отправка которой запрещена. Необязательная опция `* deny` означает, что все параметры, не определенные в данном контейнере, не должны возвращаться клиенту. Опция *номер*deny запрещает только указанную опцию. *Номер* - это 8-разрядное целое число без знака. *Данные* - это данные для опции (см. выше), заключенная в кавычки строка ASCII, либо шестнадцатеричные данные в одном из форматов: 0x*шестн._число* или hex "*шестн._число*" или hex"*шестн._число*". Если опция находится в контейнере vendor, то она будет включена вместе с другими опциями в опцию 43.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умол- чанию	Описание
option	option * deny	Нет	Нет	Задает опцию для отправки клиенту, или, в случае запрещения, опцию, отправка которой запрещена. Необязательная опция `* deny` означает, что все параметры, не определенные в данном контейнере, не должны возвращаться клиенту. Опция *номер*deny запрещает только указанную опцию. *Номер* - это 8-разрядное целое число без знака. *Данные* - это данные для опции (см. выше), заключенная в кавычки строка ASCII, либо шестнадцатеричные данные в одном из форматов: 0x*шестн._число* или hex "*шестн._число*" или hex"*шестн._число*". Если опция находится в контейнере vendor, то она будет включена вместе с другими опциями в опцию 43.
exclude	exclude *IP-адрес*	Нет	Нет	Изменяет диапазон адресов контейнера, в котором указан оператор exclude. На глобальном уровне и на уровне контейнера базы данных оператор exclude недопустим. Оператор exclude удаляет заданные адреса или диапазон адресов из диапазона, указанного для данного контейнера. Оператор exclude позволяет создавать для подсетей и других контейнеров не смежные диапазоны адресов.
exclude	exclude *адрес-адрес*	Нет	Нет	Изменяет диапазон адресов контейнера, в котором указан оператор exclude. На глобальном уровне и на уровне контейнера базы данных оператор exclude недопустим. Оператор exclude удаляет заданные адреса или диапазон адресов из диапазона, указанного для данного контейнера. Оператор exclude позволяет создавать для подсетей и других контейнеров не смежные диапазоны адресов.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умол- чанию	Описание
диапазон	range *IP-адрес*	Нет	Нет	Изменяет диапазон адресов контейнера, в котором указан оператор range. На глобальном уровне и на уровне контейнера базы данных оператор range недопустим. Если в строке описания контейнера не задан диапазон адресов и оператор range стоит в этом контейнере первым, то диапазон адресов контейнера будет определяться именно этим оператором range. Диапазоны, определяемые всеми остальными операторами range, стоящими после первого оператора range, или всеми операторами range, задающими диапазоны в строке описания контейнера, добавляются к текущему диапазону. С помощью оператора range к диапазону можно добавить отдельный адрес или набор адресов. Диапазон должен размещаться внутри группы адресов, определенной для контейнера подсети.
диапазон	range *адрес-адрес*	Нет	Нет	Изменяет диапазон адресов контейнера, в котором указан оператор range. На глобальном уровне и на уровне контейнера базы данных оператор range недопустим. Если в строке описания контейнера не задан диапазон адресов и оператор range стоит в этом контейнере первым, то диапазон адресов контейнера будет определяться именно этим оператором range. Диапазоны, определяемые всеми остальными операторами range, стоящими после первого оператора range, или всеми операторами range, задающими диапазоны в строке описания контейнера, добавляются к текущему диапазону. С помощью оператора range к диапазону можно добавить отдельный адрес или набор адресов. Диапазон должен размещаться внутри группы адресов, определенной для контейнера подсети.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умол- чанию	Описание
клиент	client *типаппаратный-адрес* NONE	Да	Нет	Определяет контейнер клиента, запрещающий выделение адреса клиенту с указанным *типом аппаратуры* и *аппаратным адресом*. Если параметр *тип аппаратуры* равен 0, то в параметре *аппаратный адрес* должна быть задана строка ASCII. В остальных случаях *тип аппаратуры* - это тип аппаратного обеспечения клиента, а *аппаратный адрес* - аппаратный адрес клиента. Если *аппаратный адрес* - это строка, то она может быть заключена в кавычки. Если *аппаратный адрес* - это шестнадцатеричная строка, то он может быть записан в виде 0x*шестн._число* или hex *шестн._число*. *Диапазон* разрешает выделять адрес из указанного *диапазона* клиенту с указанным *типом аппаратуры* и *аппаратным адресом*. Для сравнения нескольких клиентов должны использоваться регулярные выражения.
клиент	client *тип аппаратный-адрес* ANY	Да	Нет	Определяет контейнер клиента, запрещающий выделение адреса клиенту с указанным *типом аппаратуры* и *аппаратным адресом*. Если параметр *тип аппаратуры* равен 0, то в параметре *аппаратный адрес* должна быть задана строка ASCII. В остальных случаях *тип аппаратуры* - это тип аппаратного обеспечения клиента, а *аппаратный адрес* - аппаратный адрес клиента. Если *аппаратный адрес* - это строка, то она может быть заключена в кавычки. Если *аппаратный адрес* - это шестнадцатеричная строка, то он может быть записан в виде 0x*шестн._число* или hex *шестн._число*. *Диапазон* разрешает выделять адрес из указанного *диапазона* клиенту с указанным *типом аппаратуры* и *аппаратным адресом*. Для сравнения нескольких клиентов должны использоваться регулярные выражения.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умол- чанию	Описание
клиент	client *тип аппаратный адрес адрес*	Да	Нет	Определяет контейнер клиента, запрещающий выделение адреса клиенту с указанным *типом аппаратуры* и *аппаратным адресом*. Если параметр *тип аппаратуры* равен 0, то в параметре *аппаратный адрес* должна быть задана строка ASCII. В остальных случаях *тип аппаратуры* - это тип аппаратного обеспечения клиента, а *аппаратный адрес* - аппаратный адрес клиента. Если *аппаратный адрес* - это строка, то она может быть заключена в кавычки. Если *аппаратный адрес* - это шестнадцатеричная строка, то он может быть записан в виде 0х*шестн._число* или hex *шестн._число*. *Диапазон* разрешает выделять адрес из указанного *диапазона* клиенту с указанным *типом аппаратуры* и *аппаратным адресом*. Для сравнения нескольких клиентов должны использоваться регулярные выражения.
клиент	client *тип аппаратный адрес диапазон*	Да	Нет	Определяет контейнер клиента, запрещающий выделение адреса клиенту с указанным *типом аппаратуры* и *аппаратным адресом*. Если параметр *тип аппаратуры* равен 0, то в параметре *аппаратный адрес* должна быть задана строка ASCII. В остальных случаях *тип аппаратуры* - это тип аппаратного обеспечения клиента, а *аппаратный адрес* - аппаратный адрес клиента. Если *аппаратный адрес* - это строка, то она может быть заключена в кавычки. Если *аппаратный адрес* - это шестнадцатеричная строка, то он может быть записан в виде 0х*шестн._число* или hex *шестн._число*. *Диапазон* разрешает выделять адрес из указанного *диапазона* клиенту с указанным *типом аппаратуры* и *аппаратным адресом*. Для сравнения нескольких клиентов должны использоваться регулярные выражения.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умол- чанию	Описание
класс	class *строка*	Да	Нет	Определяет контейнер класса с именем *строка*. Строка может быть заключена в кавычки. Если строка заключена в кавычки, то перед сравнением они удаляются. Кавычки необходимы в том случае, если строка содержит пробелы или символы табуляции. Использование контейнера class допустимо на любом уровне. Для задания набора адресов, которые могут быть предложены клиенту с данным классом, рекомендуется пользоваться диапазонами. Диапазон может задаваться как одиночный IP-адрес в десятичном формате с точками или как два IP-адреса, разделенных дефисом.
класс	class *строка диапазон*	Да	Нет	Определяет контейнер класса с именем *строка*. Строка может быть заключена в кавычки. Если строка заключена в кавычки, то перед сравнением они удаляются. Кавычки необходимы в том случае, если строка содержит пробелы или символы табуляции. Использование контейнера class допустимо на любом уровне. Для задания набора адресов, которые могут быть предложены клиенту с данным классом, рекомендуется пользоваться диапазонами. Диапазон может задаваться как одиночный IP-адрес в десятичном формате с точками или как два IP-адреса, разделенных дефисом.
сеть	network *ид сети маска*	Да	Нет	Задает ИД сети на основании информации о классе (например, 9.3.149.0 с маской сети 255.255.255.0 - это сеть 9.0.0.0 255.255.255.0). В таком контейнере сети могут находиться подсети с одинаковым ИД сети и маской сети. Если задан диапазон адресов, то все адреса этого диапазона образуют пул. Диапазон адресов должен относиться к данной сети. Используется полная адресация класса. Это ключевое слово допустимо только на глобальном уровне или на уровне контейнера базы данных. **Примечание:** Вместо ключевого слова network лучше использовать контейнер subnet.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умол- чанию	Описание
сеть	network *ид_сети*	Да	Нет	Задает ИД сети на основании информации о классе (например, 9.3.149.0 с маской сети 255.255.255.0 - это сеть 9.0.0.0 255.255.255.0). В таком контейнере сети могут находиться подсети с одинаковым ИД сети и маской сети. Если задан диапазон адресов, то все адреса этого диапазона образуют пул. Диапазон адресов должен относиться к данной сети. Используется полная адресация класса. Это ключевое слово допустимо только на глобальном уровне или на уровне контейнера базы данных. **Примечание:** Вместо ключевого слова network лучше использовать контейнер subnet.
сеть	network *ид сети диапазон*	Да	Нет	Задает ИД сети на основании информации о классе (например, 9.3.149.0 с маской сети 255.255.255.0 - это сеть 9.0.0.0 255.255.255.0). В таком контейнере сети могут находиться подсети с одинаковым ИД сети и маской сети. Если задан диапазон адресов, то все адреса этого диапазона образуют пул. Диапазон адресов должен относиться к данной сети. Используется полная адресация класса. Это ключевое слово допустимо только на глобальном уровне или на уровне контейнера базы данных. **Примечание:** Вместо ключевого слова network лучше использовать контейнер subnet.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умол- чанию	Описание
вендор	vendor *ид_вендора*	Да	Нет	Определяет контейнер вендора. Контейнеры вендоров используются для возврата клиенту опции 43. ИД вендора может задаваться в виде строки, заключенной в кавычки, либо в виде двоичной строки в формате 0x*шестн._число* или hex "*шестн._число*". После ИД вендора можно указать диапазон. Диапазон задается двумя IP-адресами в десятичном формате с точками, разделенными дефисом. После диапазона может следовать первая часть опции 43 в виде шестнадцатеричной или ASCII-строки. При наличии в контейнере опций они добавляются к данным опции 43. После обработки всех опций к данным добавляется опция конца списка. Для возврата клиенту других опций (помимо опции 43) используйте регулярное выражение, позволяющее сравнивать всех клиентов и определения нормальных опций, возвращаемых на основании ИД вендора.
вендор	vendor *ид_вендора* hex ""			Определяет контейнер вендора. Контейнеры вендоров используются для возврата клиенту опции 43. ИД вендора может задаваться в виде строки, заключенной в кавычки, либо в виде двоичной строки в формате 0x*шестн._число* или hex "*шестн._число*". После ИД вендора можно указать диапазон. Диапазон задается двумя IP-адресами в десятичном формате с точками, разделенными дефисом. После диапазона может следовать первая часть опции 43 в виде шестнадцатеричной или ASCII-строки. При наличии в контейнере опций они добавляются к данным опции 43. После обработки всех опций к данным добавляется опция конца списка. Для возврата клиенту других опций (помимо опции 43) используйте регулярное выражение, позволяющее сравнивать всех клиентов и определения нормальных опций, возвращаемых на основании ИД вендора.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умол- чанию	Описание
вендор	vendor *ид_вендора* hex ""			Определяет контейнер вендора. Контейнеры вендоров используются для возврата клиенту опции 43. ИД вендора может задаваться в виде строки, заключенной в кавычки, либо в виде двоичной строки в формате 0x*шестн._число* или hex "*шестн._число*". После ИД вендора можно указать диапазон. Диапазон задается двумя IP-адресами в десятичном формате с точками, разделенными дефисом. После диапазона может следовать первая часть опции 43 в виде шестнадцатеричной или ASCII-строки. При наличии в контейнере опций они добавляются к данным опции 43. После обработки всех опций к данным добавляется опция конца списка. Для возврата клиенту других опций (помимо опции 43) используйте регулярное выражение, позволяющее сравнивать всех клиентов и определения нормальных опций, возвращаемых на основании ИД вендора.
вендор	vendor *ид_вендора* 0x*данные*			Определяет контейнер вендора. Контейнеры вендоров используются для возврата клиенту опции 43. ИД вендора может задаваться в виде строки, заключенной в кавычки, либо в виде двоичной строки в формате 0x*шестн._число* или hex "*шестн._число*". После ИД вендора можно указать диапазон. Диапазон задается двумя IP-адресами в десятичном формате с точками, разделенными дефисом. После диапазона может следовать первая часть опции 43 в виде шестнадцатеричной или ASCII-строки. При наличии в контейнере опций они добавляются к данным опции 43. После обработки всех опций к данным добавляется опция конца списка. Для возврата клиенту других опций (помимо опции 43) используйте регулярное выражение, позволяющее сравнивать всех клиентов и определения нормальных опций, возвращаемых на основании ИД вендора.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умол- чанию	Описание
вендор	vendor *ид_вендора* ""			Определяет контейнер вендора. Контейнеры вендоров используются для возврата клиенту опции 43. ИД вендора может задаваться в виде строки, заключенной в кавычки, либо в виде двоичной строки в формате 0x*шестн._число* или hex "*шестн._число*". После ИД вендора можно указать диапазон. Диапазон задается двумя IP-адресами в десятичном формате с точками, разделенными дефисом. После диапазона может следовать первая часть опции 43 в виде шестнадцатеричной или ASCII-строки. При наличии в контейнере опций они добавляются к данным опции 43. После обработки всех опций к данным добавляется опция конца списка. Для возврата клиенту других опций (помимо опции 43) используйте регулярное выражение, позволяющее сравнивать всех клиентов и определения нормальных опций, возвращаемых на основании ИД вендора.
вендор	vendor *ид_вендора диапазон*			Определяет контейнер вендора. Контейнеры вендоров используются для возврата клиенту опции 43. ИД вендора может задаваться в виде строки, заключенной в кавычки, либо в виде двоичной строки в формате 0x*шестн._число* или hex "*шестн._число*". После ИД вендора можно указать диапазон. Диапазон задается двумя IP-адресами в десятичном формате с точками, разделенными дефисом. После диапазона может следовать первая часть опции 43 в виде шестнадцатеричной или ASCII-строки. При наличии в контейнере опций они добавляются к данным опции 43. После обработки всех опций к данным добавляется опция конца списка. Для возврата клиенту других опций (помимо опции 43) используйте регулярное выражение, позволяющее сравнивать всех клиентов и определения нормальных опций, возвращаемых на основании ИД вендора.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умол- чанию	Описание
вендор	vendor *ид_вендорадиапазон* hex ""			Определяет контейнер вендора. Контейнеры вендоров используются для возврата клиенту опции 43. ИД вендора может задаваться в виде строки, заключенной в кавычки, либо в виде двоичной строки в формате 0x*шестн._число* или hex "*шестн._число*". После ИД вендора можно указать диапазон. Диапазон задается двумя IP-адресами в десятичном формате с точками, разделенными дефисом. После диапазона может следовать первая часть опции 43 в виде шестнадцатеричной или ASCII-строки. При наличии в контейнере опций они добавляются к данным опции 43. После обработки всех опций к данным добавляется опция конца списка. Для возврата клиенту других опций (помимо опции 43) используйте регулярное выражение, позволяющее сравнивать всех клиентов и определения нормальных опций, возвращаемых на основании ИД вендора.
вендор	vendor *ид_вендорадиапазон* hex ""			Определяет контейнер вендора. Контейнеры вендоров используются для возврата клиенту опции 43. ИД вендора может задаваться в виде строки, заключенной в кавычки, либо в виде двоичной строки в формате 0x*шестн._число* или hex "*шестн._число*". После ИД вендора можно указать диапазон. Диапазон задается двумя IP-адресами в десятичном формате с точками, разделенными дефисом. После диапазона может следовать первая часть опции 43 в виде шестнадцатеричной или ASCII-строки. При наличии в контейнере опций они добавляются к данным опции 43. После обработки всех опций к данным добавляется опция конца списка. Для возврата клиенту других опций (помимо опции 43) используйте регулярное выражение, позволяющее сравнивать всех клиентов и определения нормальных опций, возвращаемых на основании ИД вендора.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умол- чанию	Описание
вендор	vendor *ид_вендора диапазон* 0хзначение			Определяет контейнер вендора. Контейнеры вендоров используются для возврата клиенту опции 43. ИД вендора может задаваться в виде строки, заключенной в кавычки, либо в виде двоичной строки в формате 0х*шестн._число* или hex "*шестн._число*". После ИД вендора можно указать диапазон. Диапазон задается двумя IP-адресами в десятичном формате с точками, разделенными дефисом. После диапазона может следовать первая часть опции 43 в виде шестнадцатеричной или ASCII-строки. При наличии в контейнере опций они добавляются к данным опции 43. После обработки всех опций к данным добавляется опция конца списка. Для возврата клиенту других опций (помимо опции 43) используйте регулярное выражение, позволяющее сравнивать всех клиентов и определения нормальных опций, возвращаемых на основании ИД вендора.
вендор	vendor *ид_вендора диапазон* ""			Определяет контейнер вендора. Контейнеры вендоров используются для возврата клиенту опции 43. ИД вендора может задаваться в виде строки, заключенной в кавычки, либо в виде двоичной строки в формате 0х*шестн._число* или hex "*шестн._число*". После ИД вендора можно указать диапазон. Диапазон задается двумя IP-адресами в десятичном формате с точками, разделенными дефисом. После диапазона может следовать первая часть опции 43 в виде шестнадцатеричной или ASCII-строки. При наличии в контейнере опций они добавляются к данным опции 43. После обработки всех опций к данным добавляется опция конца списка. Для возврата клиенту других опций (помимо опции 43) используйте регулярное выражение, позволяющее сравнивать всех клиентов и определения нормальных опций, возвращаемых на основании ИД вендора.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умол- чанию	Описание
inoption	входящая опция *номер данные*	Да	Нет	Указывает контейнер, с которым следует сравнивать переданные клиентом опции с данным номером. *Номер* означает номер опции. *Данные* - это ключ для сравнения с этим контейнером в ходе выбора адреса и опции для данного клиента. Для стандартных опций *данные* указываются в явной форме: строка, заключенная в кавычки, IP-адрес или целое число; кроме того, их можно указать в форме шестнадцатеричной строки байтов, начинающейся с символов `0x`. Для опций, неизвестных серверу, допустим только второй вариант. Кроме того, *данные* могут представлять собой выражение, которое следует сравнить с содержимым строки данных опции, переданной клиентом. Такие выражения записываются в кавычках и начинаются с символов `"!` (кавычка и восклицательный знак). Опции, неизвестные серверу, должны указываться в виде шестнадцатеричной строки байтов БЕЗ начальных символов `0x`.
inoption	входящая опция *номер данные диапазон*	Да	Нет	Указывает контейнер, с которым следует сравнивать переданные клиентом опции с данным номером. *Номер* означает номер опции. *Данные* - это ключ для сравнения с этим контейнером в ходе выбора адреса и опции для данного клиента. Для стандартных опций *данные* указываются в явной форме: строка, заключенная в кавычки, IP-адрес или целое число; кроме того, их можно указать в форме шестнадцатеричной строки байтов, начинающейся с символов `0x`. Для опций, неизвестных серверу, допустим только второй вариант. Кроме того, *данные* могут представлять собой выражение, которое следует сравнить с содержимым строки данных опции, переданной клиентом. Такие выражения записываются в кавычках и начинаются с символов `"!` (кавычка и восклицательный знак). Опции, неизвестные серверу, должны указываться в виде шестнадцатеричной строки байтов БЕЗ начальных символов `0x`.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умол- чанию	Описание
virtual	virtual fill *ИД ИД* ...	Нет	Нет	Определяет виртуальную подсеть со стратегией. Слово `fill` означает, что перед переходом к следующему контейнеру необходимо использовать все адреса текущего контейнера. Слово `rotate` указывает, что для каждого запроса адрес выбирается из следующего указанного пула. `sfill` и `srotate` означают то же, что `fill` и `rotate`, однако в этом случае производится поиск, чтобы узнать, не соответствует ли клиент каким-либо контейнерам, вендорам или классам в подсети. Если обнаружено совпадение с контейнером, в котором выделяется адрес, то, адрес выбирается из этого контейнера, а не назначается в соответствии со стратегией. Число ИД в списке не ограничено. *ИД* - это либо ИД подсети, либо метка, заданная в описании подсети. Метка нужна при наличии нескольких подсетей с одинаковым идентификатором.
virtual	virtual sfill *ИД ИД* ...	Нет	Нет	Определяет виртуальную подсеть со стратегией. Слово `fill` означает, что перед переходом к следующему контейнеру необходимо использовать все адреса текущего контейнера. Слово `rotate` указывает, что для каждого запроса адрес выбирается из следующего указанного пула. `sfill` и `srotate` означают то же, что `fill` и `rotate`, однако в этом случае производится поиск, чтобы узнать, не соответствует ли клиент каким-либо контейнерам, вендорам или классам в подсети. Если обнаружено совпадение с контейнером, в котором выделяется адрес, то, адрес выбирается из этого контейнера, а не назначается в соответствии со стратегией. Число ИД в списке не ограничено. *ИД* - это либо ИД подсети, либо метка, заданная в описании подсети. Метка нужна при наличии нескольких подсетей с одинаковым идентификатором.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умол- чанию	Описание
virtual	virtual rotate *ИД ИД* ...	Нет	Нет	Определяет виртуальную подсеть со стратегией. Слово `fill` означает, что перед переходом к следующему контейнеру необходимо использовать все адреса текущего контейнера. Слово `rotate` указывает, что для каждого запроса адрес выбирается из следующего указанного пула. `sfill` и `srotate` означают то же, что `fill` и `rotate`, однако в этом случае производится поиск, чтобы узнать, не соответствует ли клиент каким-либо контейнерам, вендорам или классам в подсети. Если обнаружено совпадение с контейнером, в котором выделяется адрес, то, адрес выбирается из этого контейнера, а не назначается в соответствии со стратегией. Число ИД в списке не ограничено. *ИД* - это либо ИД подсети, либо метка, заданная в описании подсети. Метка нужна при наличии нескольких подсетей с одинаковым идентификатором.
virtual	virtual srotate *ИД ИД* ...	Нет	Нет	Определяет виртуальную подсеть со стратегией. Слово `fill` означает, что перед переходом к следующему контейнеру необходимо использовать все адреса текущего контейнера. Слово `rotate` указывает, что для каждого запроса адрес выбирается из следующего указанного пула. `sfill` и `srotate` означают то же, что `fill` и `rotate`, однако в этом случае производится поиск, чтобы узнать, не соответствует ли клиент каким-либо контейнерам, вендорам или классам в подсети. Если обнаружено совпадение с контейнером, в котором выделяется адрес, то, адрес выбирается из этого контейнера, а не назначается в соответствии со стратегией. Число ИД в списке не ограничено. *ИД* - это либо ИД подсети, либо метка, заданная в описании подсети. Метка нужна при наличии нескольких подсетей с одинаковым идентификатором.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умол- чанию	Описание
inorder:	inorder: *id id ...*	Нет	Нет	Определяет виртуальную подсеть со стратегией заполнения, т.е. перед переходом к следующему контейнеру должны быть использованы все адреса текущего контейнера. Число ИД в списке не ограничено. *ИД* - это либо ИД подсети, либо метка, заданная в описании подсети. Метка нужна при наличии нескольких подсетей с одинаковым идентификатором.
balance:	balance: *ИД ИД ...*	Нет	Нет	Определяет виртуальную подсеть со стратегией смены адресов, при которой каждый следующий адрес выбирается из следующего контейнера. Число ИД в списке не ограничено. *ИД* - это либо ИД подсети, либо метка, заданная в описании подсети. Метка нужна при наличии нескольких подсетей с одинаковым идентификатором.
bootstrapserver	bootstrapserver *IP-адрес*	Нет	Нет	Указывает, на каком сервере находятся файлы **TFTP**, которые должны использоваться клиентами после получения ими пакетов **BOOTP** или **DHCP**. Это значение задается в поле **siaddr** пакета. Допустимо на уровне любого контейнера.
giaddrfield	giaddrfield *IP-адрес*	Нет	Нет	Задает поле giaddrfield для ответных сообщений (пакетов). **Примечание:** Данная спецификация недопустима в протоколах **BOOTP** и **DHCP**, однако для работы некоторых клиентов необходимо, чтобы в поле **giaddr** был указан шлюз по умолчанию. Из-за возможных конфликтов ключевое слово **giaddrfield** должно использоваться только внутри контейнеров клиентов, хотя оно может работать на любом уровне.
bootfile	bootfile *полное_имя*	Нет	Нет	Определяет загрузочный файл, который должен быть указан в ответном пакете. Допустимо на уровне любого контейнера. Взаимодействие элементов, указанных в поступающем пакете с операторами загрузочного файла и домашнего каталога определяется стратегией загрузочного файла.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умол- чанию	Описание
pxebootfile	pxebootfile *System Arch MajorVer MinorVer Bootfilename*	Нет	Нет	Задает имя загрузочного файла, которое должно быть передано клиенту. Функция разбора файла конфигурации выдает сообщение об ошибке, если число параметров после ключевого слова меньше 4, и игнорирует все параметры после четвертого. Это ключевое слово может применяться только в контейнере.

Информация о других опциях приведена в разделах “Известные опции файла сервера DHCP” на стр. 219 и “Опции контейнера вендора среды исполнения перед загрузкой” на стр. 222.

Демон согласования загрузочных образов (BINLD)

Сервер демона согласования загрузочных образов (BINLD) применяется на третьем этапе загрузки клиентов PXE.

Сначала клиент обращается к серверу DHCP за IP-адресом, затем получает от proxy-сервера PXE DHCP информацию о расположении загрузочного сервера, а от загрузочного сервера - имя и каталог загрузочного файла. Если для загрузки клиента необходимо несколько файлов, то клиент PXE может обращаться к серверу несколько раз.

На последнем этапе клиент PXE получает загрузочный образ с загрузочного сервера. Загрузочный сервер сообщает клиенту PXE расположение сервера TFTP и имя файла, который необходимо загрузить.

Компоненты сервера BINLD

Здесь приведена информация о трех основных компонентах сервера BINLD.

Сервер BINLD состоит из трех основных компонентов: базы данных, средств поддержки протокола и набора служебных нитей, для каждой из которых задается собственная информация о конфигурации.

Базы данных BINLD:

Для создания списка ответов на пакет REQUEST, полученный от клиента, применяется база данных `db_file.dhcpo`.

Значения, возвращаемые базой данных, зависят от выбранного типа сервера. Они задаются с помощью ключевого слова **pxeservertype** в файле `binld.cnf`.

База данных заполняется и проверяется в соответствии с информацией, заданной в файле конфигурации.

Служба протокола BINLD:

Отправляемая клиентам информация о конфигурации определяется с помощью базы данных.

Модуль протокола PXED основан на спецификации Intel Preboot Execution Environment (PXE) версии 2.1, но еще совместим со спецификацией Intel PXE версии 1.1.

Операции с нитями BINLD:

Третья часть сервера BINLD - это набор операций, которые, собственно, и обеспечивают его работу.

Так как сервер BINLD представляет собой набор нитей, то все операции реализованы в виде нитей, которые в определенное время выполняют нужные действия и обеспечивают правильную работу.

Первая нить, *main*, обрабатывает запросы SRC (такие как **startsrc**, **stopsrc**, **lssrc**, **traceson** и **refresh**). Кроме того, эта нить согласовывает все операции, влияющие на остальные нити, и обрабатывает сигналы. Пример:

- SIGHUP (-1) обновляет все базы данных в файле конфигурации.
- SIGTERM (-15) завершает работу сервера в нормальном режиме.

Остальные нити выполняют обработку пакетов. В зависимости от типа сервера, может быть создана одна или две нити. Первая нить работает с портом 67, а вторая - с портом 4011. Обе нити могут обрабатывать запросы клиентов.

Конфигурация BINLD

По умолчанию сервер BINLD считывает информацию из файла `/etc/binld.cnf`, в котором хранится исходная база данных параметров и адресов сервера.

Сервер можно запустить с помощью SMIT или команд SRC.

Настройка сервера BINLD - это самая сложная часть настройки BINLD в сети. Во-первых, определите, в каких сетях будут размещаться клиенты PXE. Ниже приведен пример конфигурации сервера BINLD, работающего на одном компьютере с сервером DHCP:

```
pxeservertype       binld_on_dhcp_server

subnet default
{
    vendor pxe
    {
                bootstrapserver  9.3.149.6      #IP-адрес сервера TFTP
                pxebootfile    1    2    1   window.one    1    0
                pxebootfile    2    2    1   linux.one     2    3
                pxebootfile    1    2    1   hello.one     3    4
                client 6 10005a8ad14d any
                {
                   pxebootfile    1    2    1   aix.one     5   6
                   pxebootfile    2    2    1   window.one  6   7
       }
     }
}
```

В предыдущем примере сервер BINLD получает через порт 4011 обычные пакеты от клиентов, а также многоцелевые пакеты, если он получил адрес рассылки от dhcpsd/pxed. В ответ на пакет REQUEST/INFORM, полученный от клиента, сервер BINLD возвращает имя загрузочного файла и IP-адрес сервера TFTP. Если серверу BINLD не удается найти загрузочный файл того уровня, который был запрошен клиентом, он пытается найти загрузочный файл следующего уровня. Если серверу BINLD не удается найти загрузочный файл, отвечающий требованиям клиента (*Type*, *SystemArch*, *MajorVers*, *MinorVers*, and *Layer*), то он не отвечает на запрос.

Ниже приведен пример конфигурации сервера BINLD, работающего на отдельном компьютере от сервера DHCP/PXED.

```
subnet 9.3.149.0 255.255.255.0
{

     vendor pxe
     {

     bootstrapserver        9.3.149.6          # IP-адрес сервера TFTP
     pxebootfile    1    2    1   window.one    1    0
     pxebootfile    2    2    1   linux.one     2    3
     pxebootfile    1    2    1   hello.one     3    4
     client 6 10005a8ad14d any
```

```
        {
        pxebootfile    1    2    1   aix.one     5   6
        pxebootfile    2    2    1   window.one  6   7
        }
     }
}
```

В предыдущем примере ключевое слово *pxeservertype* не задано, поэтому применяется тип сервера по умолчанию, **binld_only**. Сервер BINLD принимает через порт 67 обычные и многоцелевые пакеты, а через порт 4011 - обычные пакеты, а также многоцелевые пакеты, если он получил адрес рассылки от dhcpsd/pxed. Имя загрузочного файла и IP-адрес сервера TFTP отправляются клиенту PXE только в том случае, если IP-адрес клиента относится к диапазону адресов подсети (9.3.149.0 - 9.3.149.255).

Ниже приведен пример конфигурации сервера BINLD, работающего на одном компьютере с сервером PXED:

```
pxeservertype       binld_on_proxy_server
subnet default
{
     vendor
     {
     bootstrapserver        9.3.149.6          # IP-адрес сервера TFTP
     pxebootfile    1    2    1   window.one    1    0
     pxebootfile    2    2    1   linux.one     2    3
     pxebootfile    1    2    1   hello.one     3    4
     client 6 10005a8ad14d any
       {
       pxebootfile    1    2    1   aix.one     5   6
       pxebootfile    2    2    1   window.one  6   7
       }
     }
}
```

В данном примере сервер BINLD принимает многоцелевые пакеты через порт 4011 при условии, что он получил адрес рассылки от dhcpsd/pxed. Если сервер не получает адрес рассылки, он завершает свою работу и заносит в файл протокола сообщение об ошибке.

Оператор базы данных `db_file` задает способ обработки этой части файла конфигурации. Комментарии начинаются с символа #. Текст, стоящий в строке после символа #, игнорируется сервером PXED. С помощью каждой строки `option` сервер задает для клиента какое-либо действие. В разделе “Опции контейнера вендора PXE” на стр. 306 описаны все поддерживаемые и известные опции. В разделе “Синтаксис файла сервера BINLD для общих операций сервера” на стр. 333 описаны способы задания опций, неизвестных серверу.

Файл конфигурации BINLD:

Файл конфигурации состоит из раздела адресов и раздела определения опций. Оба этих раздела реализуют концепцию контейнера, который может включать опции, модификаторы и, возможно, другие контейнеры.

Контейнер (по существу, это способ группирования параметров) позволяет объединять клиентов в группы на основании идентификатора. Типы контейнеров: subnet, class, vendor и client. Контейнеры, определяемые пользователем, в настоящее время не поддерживаются. Клиент однозначно определяется своим идентификатором, что позволяет всегда точно обнаруживать его, например, при переносе в другую подсеть. Для описания клиента может применяться несколько контейнеров.

Опции - это идентификаторы, возвращаемые клиенту. Это может быть, например, применяемые по умолчанию адреса шлюза и сервера DNS.

Контейнеры BINLD:

При получении запроса сервер DHCP анализирует пакет и на основании ключей идентификации определяет, какие нужно выбрать контейнеры, параметры и адреса.

Последний пример в разделе Настройка BINLD показан пример контейнера subnet. Его ключ идентификации - это расположение клиента в сети. Если клиент находится в данной сети, то он попадает в этот контейнер.

Для идентификации клиентов в разных типах контейнеров применяются различные опции:

- Для определения подсети, в которой находится клиент, контейнер subnet использует поле giaddr или адрес принимающего интерфейса.
- Контейнер class использует значение опции 77 (идентификатор класса пользователя).
- Контейнер vendor использует значение опции 60 (идентификатор класса вендора).
- В контейнере client применяется значение опции 61 (идентификатор клиента) для клиентов PXED и поле chaddr из пакета BOOTP для клиентов BOOTP.

Во всех контейнерах, кроме subnet, можно применять шаблоны сравнения, включая регулярные выражения.

Существует также неявный контейнер *global*. Опции и модификаторы, указанные в контейнере global, действуют во всех контейнерах, если только они не отключены и не переопределены. Большинство контейнеров можно поместить в другие контейнеры в соответствии с областью видимости. С контейнерами могут быть связаны диапазоны адресов. С подсетями всегда связаны диапазоны адресов.

Основные правила для контейнеров и подконтейнеров:

- На глобальном уровне допустимы все контейнеры.
- Контейнеры subnet нельзя помещать в другие контейнеры.
- В контейнеры с ограничениями нельзя помещать обычные контейнеры того же типа. (Например, если контейнер содержит опцию, разрешающую только класс `Accounting`, то в него нельзя помещать контейнер, который содержит опцию, разрешающую применение всех классов, имена которых начинаются с буквы "a". Это недопустимо.)
- Контейнеры client с ограничениями не могут содержать вложенные контейнеры.

С помощью этих правил вы можете создавать иерархию контейнеров, в которой опции объединены в наборы, соответствующие конкретным клиентам или группам клиентов.

Каким образом организована обработка адресов и опций, если клиент входит в несколько контейнеров? Сервер DHCP получает сообщение, формирует запрос к базе данных (в данном случае - к файлу `db_file`) и получает список контейнеров. Контейнеры перечисляются в списке в порядке их вложенности и приоритета. Приоритет определяется как неявный иерархический уровень контейнера. Контейнеры с ограничениями имеют более высокий приоритет, чем обычные. Сортировка контейнеров выполняется в таком порядке: клиенты, классы, вендоры и подсети. В пределах одного типа контейнеры упорядочиваются по уровню вложенности. Созданный таким образом список упорядочивается от более конкретных объектов к менее конкретным. Например:

```
Subnet 1
--Class 1
--Client 1
Subnet 2
--Class 1
----Vendor 1
----Client 1
--Client 1
```

В примере есть две подсети `Subnet 1` и `Subnet 2`. Кроме того, определен один класс, `Class 1`, один вендор, `Vendor 1`, и один клиент, `Client 1`. `Class 1` и `Client 1` определены в нескольких контейнерах. Их имена в разных контейнерах могут совпадать, однако указанные в них значения могут различаться. Если клиент

`Client 1` отправит сообщение серверу DHCP из подсети `Subnet 1` с указанием класса `Class 1`, определенного в списке опций этого клиента, то сервер DHCP создаст следующий список контейнеров:
`Subnet 1, Class 1, Client 1`

Контейнер, определенный наиболее точно, заносится в список последним. Для получения адреса список просматривается в обратном порядке до обнаружения первого доступного адреса. Затем список просматривается в прямом порядке (в соответствии с иерархией) для получения опций. По мере просмотра списка, новые значения опций переопределяют прежние значения, если в контейнере не задана опция *deny*. Поскольку класс `Class 1` и клиент `Client 1` находятся в одной и той же подсети `Subnet 1`, они упорядочиваются в соответствии с приоритетом контейнеров. Если сообщение будет получено от клиента с тем же именем, находящегося в подсети `Subnet 2`, то будет создан следующий список контейнеров:

`Subnet 2, Class 1, Client 1` (на уровне `Subnet 2`), `Client 1` (на уровне `Class 1`)

Первой в списке указывается подсеть `Subnet 2`, затем класс `Class 1`, затем клиент `Client 1` на уровне `Subnet 2` (так как этот клиент находится в иерархии на один уровень ниже). Иерархия подразумевает, что клиент, имя которого указано в первом операторе, определен менее конкретно, чем клиент `Client 1`, определенный в классе `Class 1` подсети `Subnet 2`.

Приоритет, определяемый по уровню вложенности, выше, чем приоритет самих контейнеров. Например, если тот же клиент отправит такое же сообщение, указав идентификатор вендора, то список контейнеров будет следующим:

`Subnet 2, Class 1, Vendor 1, Client 1` (на уровне `Subnet 2`), `Client 1` (на уровне `Class 1`)

Организация поиска на основании приоритета контейнера повышает эффективность, поскольку контейнеры client обеспечивают наиболее точный способ определения одного или нескольких клиентов. В контейнере class адреса определены менее конкретно, чем в контейнере client; в контейнере vendor адреса определены еще менее конкретно, а контейнер subnet содержит самые общие определения адресов.

Адреса и диапазоны адресов BINLD:

С контейнерами любого типа могут быть связаны диапазоны адресов. Контейнеры subnet обязательно имеют связанный диапазон адресов.

Каждый диапазон адресов в пределах контейнера должен быть подмножеством диапазона адресов родительского контейнера и не должен пересекаться с диапазонами адресов других контейнеров. Например, если внутри подсети определен класс с диапазоном адресов, то этот диапазон должен быть подмножеством диапазона адресов подсети. Диапазон внутри этого контейнера класса не должен пересекаться с любыми другими диапазонами этого уровня.

Диапазоны адресов могут задаваться в строке контейнера. Для задания несмежных диапазонов адресов можно воспользоваться операторами range и exclude. Таким образом, если в подсети есть два диапазона по десять адресов, то имеет смысл указать эти диапазоны в операторе subnet, чтобы уменьшить объем памяти и избежать конфликтов адресов с другими клиентами.

Когда адрес выбран, все последующие контейнеры, содержащие диапазоны адресов, удаляются из списка вместе со своими дочерними контейнерами. Необходимость такой процедуры обусловлена тем, что сетевые опции, заданные в удаленных контейнерах, недопустимы для адресов, не принадлежащих данному контейнеру.

Опции файла конфигурации BINLD:

После первого просмотра списка и получения адресов для клиента создается набор опций.

В процессе выбора ранее определенные значения опций переопределяются новыми до тех пор, пока не встретится опция *deny* (запретить); при этом запрещенная опция удаляется из списка опций, отправляемых пользователю. Этот способ разрешает наследование опций родительских контейнеров и сокращает объем данных, которые нужно определять.

Ведение протоколов BINLD:

Параметры протокола указываются в окне диалога, напоминающем окно настройки базы данных, но с ключевым словом **logging_info**.

На этапе обучения настройке PXED рекомендуется вести максимально подробный протокол. Кроме того, весьма полезно настроить протокол до начала работы с любыми другими файлами настройки, чтобы ошибки конфигурации были после инициализации системы занесены в протокол. Для включения опций протокола укажите ключевое слово **logitem**, а для их отключения удалите это ключевое слово. Другие ключевые слова позволяют задавать имя файла протокола, его размер, а также число взаимозаменяемых файлов протоколов.

Замечания о производительности BINLD:

Обратите внимание, что набор ключевых слов и структура файла конфигурации влияют на объем используемой памяти и производительность сервера PXED.

Во-первых, понимание механизма наследования опций от контейнеров-предков к потомкам позволяет сократить объем используемой памяти. В среде, в которой не поддерживаются не указанные в списке клиенты, администратор должен явно перечислить всех клиентов в файле. При указании списка опций для каждого отдельного клиента на хранение дерева конфигурации расходуется больше ресурсов памяти сервера, чем при наследовании опций от контейнера-предка к потомку (в роли предка может выступать подсеть, сеть или глобальные контейнеры). Следовательно, администратор должен проверить, не повторяются ли какие-либо опции на уровне клиента в файле конфигурации, а если это так, то решить, можно ли указать эти опции в контейнере-предке и таким образом распространить их сразу на несколько клиентов.

Кроме того, если применяются записи **logItem** INFO и TRACE, то при обработке каждого сообщения клиента PXE в протокол заносится несколько сообщений. Добавление большого числа строк в файл протокола может привести к значительному увеличению нагрузки, поэтому ограничение объема заносимой в протокол информации позволяет повысить производительность сервера PXED. Вы можете динамически включать ведение протокола при возникновении ошибок в работе сервера PXED с помощью команды SRC traceson.

Синтаксис файла сервера BINLD для общих операций сервера

Здесь приведены сведения о синтаксисе файла сервера BINLD для общих операций сервера. Описаны формы, субконтейнеры, значения по умолчанию и значения.

Примечание: Используемые в таблицы единицы времени (*единицы_времени*) указывать необязательно. Они применяются в качестве модификаторов значений времени. По умолчанию время указывается в минутах. Допустимо указывать время в секундах (1), минутах (60), часах (3600), сутках (86400), неделях (604800), месяцах (2392000) и годах (31536000). В скобках указан коэффициент, на который нужно умножить указанное значение времени, *n*, чтобы перевести его в секунды.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умол- чанию	Описание
database	database *тип базы данных*	Да	Нет	Основной контейнер, содержащий определения для пулов адресов, опций и операторов, задающих уровень доступа клиентов. *тип базы данных* - это имя модуля, который должен загружаться для обработки этой части файла. В текущей версии поддерживается только значение **db_file**.
logging_info	logging_info	Да	Нет	основной контейнер, определяющий параметры ведения протоколов.
logitem	logitem NONE	Нет	По умол- чанию все запре- щены.	Задает уровень ведения протокола. Можно указать несколько строк.
logitem	logitem SYSERR	Нет	По умол- чанию все запре- щены.	Задает уровень ведения протокола. Можно указать несколько строк.
logitem	logitem OBJERR	Нет	По умол- чанию все запре- щены.	Задает уровень ведения протокола. Можно указать несколько строк.
logitem	logitem PROTOCOL	Нет	По умол- чанию все запре- щены.	Задает уровень ведения протокола. Можно указать несколько строк.
logitem	logitem PROTERR	Нет	По умол- чанию все запре- щены.	Задает уровень ведения протокола. Можно указать несколько строк.
logitem	logitem WARN	Нет	По умол- чанию все запре- щены.	Задает уровень ведения протокола. Можно указать несколько строк.
logitem	logitem WARNING	Нет	По умол- чанию все запре- щены.	Задает уровень ведения протокола. Можно указать несколько строк.
logitem	logitem CONFIG	Нет	По умол- чанию все запре- щены.	Задает уровень ведения протокола. Можно указать несколько строк.
logitem	logitem EVENT	Нет	По умол- чанию все запре- щены.	Задает уровень ведения протокола. Можно указать несколько строк.
logitem	logitem PARSEERR	Нет	По умол- чанию все запре- щены.	Задает уровень ведения протокола. Можно указать несколько строк.
logitem	logitem ACTION	Нет	По умол- чанию все запре- щены.	Задает уровень ведения протокола. Можно указать несколько строк.
logitem	logitem ACNTING	Нет	По умол- чанию все запре- щены.	Задает уровень ведения протокола. Можно указать несколько строк.
logitem	logitem STAT	Нет	По умол- чанию все запре- щены.	Задает уровень ведения протокола. Можно указать несколько строк.
logitem	logitem TRACE	Нет	По умол- чанию все запре- щены.	Задает уровень ведения протокола. Можно указать несколько строк.
logitem	logitem RTRACE	Нет	По умол- чанию все запре- щены.	Задает уровень ведения протокола. Можно указать несколько строк.
logitem	logitem START	Нет	По умол- чанию все запре- щены.	Задает уровень ведения протокола. Можно указать несколько строк.
numLogFiles	numLogFiles *n*	Нет	0	Указывает, сколько файлов протоколов нужно создать. Каждый последующий файл протокола создается после заполнения предыдущего. *n* - число создаваемых файлов.
logFileSize	logFileSize *n*	Нет	0	Задает размер каждого файла протокола в блоках по 1024 байта.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
logFileName	logFileName *путь*	Нет	Нет	Задает путь к первому файлу протокола. Имя файла протокола имеет вид *имя_файла* или *имя_файла.расш*. Имя следующего файла протокола создается на основе базового *имени_файла*, к которому добавляется номер, либо этот номер указывается вместо расширения. Например, если первому файлу присвоено имя `file`, то именем следующего файла будет `file01`. Если имя первого файла - `file.log`, то следующему файлу будет присвоено имя `file.01`.
pxeservertype	pxeservertype *тип_сервера*	Нет	dhcp_only	Указывает тип сервера dhcpsd. Параметру *тип-сервера* можно присвоить одно из следующих значений: **binld_on_dhcp_server** (означает, что **BINLD** работает на том же компьютере, что и сервер DHCP, и принимает запросы клиентов PXE через порт 4011, в том числе многоцелевые запросы, если адрес рассылки был получен от сервера DHCP/PXED) или **binld_on_proxy_server** (означает, что **BINLD** работает на том же компьютере, что и сервер PXED, и принимает от клиентов PXE многоцелевые запросы, если адрес рассылки был получен от сервера DHCP/PXED). Значение по умолчанию - **binld_only**, то есть сервер BINLD работает на отдельном компьютере и принимает пакеты клиентов через порты 67 и 4011, в том числе многоцелевые пакеты, если адрес рассылки был получен от сервера DHCP/PXED.
dhcp_or_proxy _address	dhcp_or_proxy_address *IP-адрес*	Нет	Нет	Указывает IP-адрес сервера DHCP или PXED, которому сервер BINLD может отправить пакет типа REQUEST/INFORM для получения адреса рассылки. Это ключевое слово указывается только в том случае, если сервер DHCP или PXED находится в другой подсети, чем сервер BINLD.

Синтаксис файла сервера BINLD для базы данных db_file

Здесь описан синтаксис файла сервера BINLD для базы данных `db_file`. Описаны формы, субконтейнеры, значения по умолчанию и значения.

Примечание:

1. Используемые в таблицы единицы времени (*единицы_времени*) указывать необязательно. Они применяются в качестве модификаторов значений времени. По умолчанию время указывается в минутах. Допустимо указывать время в секундах (1), минутах (60), часах (3600), сутках (86400), неделях (604800), месяцах (2392000) и годах (31536000). В скобках указан коэффициент, на который нужно умножить указанное значение времени, *n*, чтобы перевести его в секунды.

2. Элементы, описанные в одном контейнере, могут быть переопределены во вложенном контейнере. Например, можно определить клиентов BOOTP на глобальном уровне, но разрешить их работу только в конкретной подсети, указав ключевое слово supportBootp в обоих контейнерах.
3. В контейнерах client, class и vendor поддерживаются регулярные выражения. Если в контейнере class или vendor указана заключенная в кавычки строка, в которой после открывающей кавычки стоит символ !, то остаток строки будет обрабатываться как регулярное выражение. В контейнере client регулярные выражения можно указывать в полях hwtype и hwaddr. В обоих полях можно задавать строку следующего формата:

   ```
   десятичное_число-данные
   ```

   Если десятичное_число равно нулю, то данные представляют строку ASCII. Если указано любое другое число, то данные представляют собой набор шестнадцатеричных цифр.

Ключевое слово	**Формат**	**Вложенные контейнеры**	**Значение по умол-чанию**	**Описание**
subnet	subnet default	Да	Нет	Определяет подсеть, с которой не связан диапазон адресов. Эта подсеть применяется сервером только для ответа на пакет INFORM, полученный от клиента, для которого не был найден ни один контейнер subnet.
subnet	subnet *ид подсети маска*	Да	Нет	Определяет подсеть и пул адресов. Если в строке не задан диапазон адресов и адреса не изменены в контейнере с помощью оператора range или exclude, то предполагается, что пул включает все адреса. Необязательный параметр диапазона представляет собой пару IP-адресов в десятичном формате с точками, разделенных дефисом. Можно (но не обязательно) указывать метку и приоритет. Они используются виртуальными подсетями для идентификации и упорядочения подсетей в виртуальной подсети. Метка и приоритет разделяются двоеточием. Эти контейнеры могут использоваться только на глобальном уровне или на уровне контейнера базы данных.
subnet	subnet *ид подсети маска диапазон*	Да	Нет	Определяет подсеть и пул адресов. Если в строке не задан диапазон адресов и адреса не изменены в контейнере с помощью оператора range или exclude, то предполагается, что пул включает все адреса. Необязательный параметр диапазона представляет собой пару IP-адресов в десятичном формате с точками, разделенных дефисом. Можно (но не обязательно) указывать метку и приоритет. Они используются виртуальными подсетями для идентификации и упорядочения подсетей в виртуальной подсети. Метка и приоритет разделяются двоеточием. Эти контейнеры могут использоваться только на глобальном уровне или на уровне контейнера базы данных.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умол-чанию	Описание
subnet	subnet *цд подсети маска метка:приоритет*	Да	Нет	Определяет подсеть и пул адресов. Если в строке не задан диапазон адресов и адреса не изменены в контейнере с помощью оператора range или exclude, то предполагается, что пул включает все адреса. Необязательный параметр диапазона представляет собой пару IP-адресов в десятичном формате с точками, разделенных дефисом. Можно (но не обязательно) указывать метку и приоритет. Они используются виртуальными подсетями для идентификации и упорядочения подсетей в виртуальной подсети. Метка и приоритет разделяются двоеточием. Эти контейнеры могут использоваться только на глобальном уровне или на уровне контейнера базы данных.
subnet	subnet *цд подсети маска диапазон метка:приоритет*			Определяет подсеть и пул адресов. Если в строке не задан диапазон адресов и адреса не изменены в контейнере с помощью оператора range или exclude, то предполагается, что пул включает все адреса. Необязательный параметр диапазона представляет собой пару IP-адресов в десятичном формате с точками, разделенных дефисом. Можно (но не обязательно) указывать метку и приоритет. Они используются виртуальными подсетями для идентификации и упорядочения подсетей в виртуальной подсети. Метка и приоритет разделяются двоеточием. Эти контейнеры могут использоваться только на глобальном уровне или на уровне контейнера базы данных.
subnet	subnet *цд подсети диапазон*	Да	Нет	Определяет подсеть, которая входит в контейнер сети. Задает диапазон адресов. Если не задан диапазон, то считается, что в подсеть входят все адреса. Маска подсети определяется родительским контейнером сети. **Примечание:** Задавать подсеть таким образом не рекомендуется.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
option	option *номер данные* ...	Нет	Нет	Задает опцию для отправки клиенту, или, в случае запрещения, опцию, отправка которой запрещена. Опция option *** deny** означает, что все параметры, не определенные в данном контейнере, не должны возвращаться клиенту. Опция *номер*deny запрещает только указанную опцию. *Номер* - это 8-разрядное целое число без знака. *Данные* - это данные для опции (см. выше), заключенная в кавычки строка ASCII, либо шестнадцатеричные данные в одном из форматов: 0x*шестн._число* или hex "*шестн._число*" или hex"*шестн._число*". Если опция находится в контейнере vendor, то она будет включена вместе с другими опциями в опцию 43.
option	option *номер*deny	Нет	Нет	Задает опцию для отправки клиенту, или, в случае запрещения, опцию, отправка которой запрещена. Опция option *** deny** означает, что все параметры, не определенные в данном контейнере, не должны возвращаться клиенту. Опция *номер*deny запрещает только указанную опцию. *Номер* - это 8-разрядное целое число без знака. *Данные* - это данные для опции (см. выше), заключенная в кавычки строка ASCII, либо шестнадцатеричные данные в одном из форматов: 0x*шестн._число* или hex "*шестн._число*" или hex"*шестн._число*". Если опция находится в контейнере vendor, то она будет включена вместе с другими опциями в опцию 43.
option	option * deny	Нет	Нет	Задает опцию для отправки клиенту, или, в случае запрещения, опцию, отправка которой запрещена. Опция option *** deny** означает, что все параметры, не определенные в данном контейнере, не должны возвращаться клиенту. Опция *номер*deny запрещает только указанную опцию. *Номер* - это 8-разрядное целое число без знака. *Данные* - это данные для опции (см. выше), заключенная в кавычки строка ASCII, либо шестнадцатеричные данные в одном из форматов: 0x*шестн._число* или hex "*шестн._число*" или hex"*шестн._число*". Если опция находится в контейнере vendor, то она будет включена вместе с другими опциями в опцию 43.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умол-чанию	Описание
exclude	exclude *IP-адрес*	Нет	Нет	Изменяет диапазон адресов контейнера, в котором указан оператор exclude. На глобальном уровне и на уровне контейнера базы данных оператор exclude недопустим. Оператор exclude удаляет заданные адреса или диапазон адресов из диапазона, указанного для данного контейнера. Оператор exclude позволяет создавать для подсетей и других контейнеров не смежные диапазоны адресов.
exclude	exclude *адрес-адрес*	Нет	Нет	Изменяет диапазон адресов контейнера, в котором указан оператор exclude. На глобальном уровне и на уровне контейнера базы данных оператор exclude недопустим. Оператор exclude удаляет заданные адреса или диапазон адресов из диапазона, указанного для данного контейнера. Оператор exclude позволяет создавать для подсетей и других контейнеров не смежные диапазоны адресов.
range	range *IP-адрес*	Нет	Нет	Изменяет диапазон адресов контейнера, в котором указан оператор range. На глобальном уровне и на уровне контейнера базы данных оператор range недопустим. Если в строке описания контейнера не задан диапазон адресов и оператор range стоит в этом контейнере первым, то диапазон адресов контейнера будет определяться именно этим оператором range. Диапазоны, определяемые всеми остальными операторами range, стоящими после первого оператора range, или всеми операторами range, задающими диапазоны в строке описания контейнера, добавляются к текущему диапазону. С помощью оператора range к диапазону можно добавить отдельный адрес или набор адресов. Диапазон должен размещаться внутри группы адресов, определенной для контейнера подсети.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
range	range *адрес-адрес*	Нет	Нет	Изменяет диапазон адресов контейнера, в котором указан оператор range. На глобальном уровне и на уровне контейнера базы данных оператор range недопустим. Если в строке описания контейнера не задан диапазон адресов и оператор range стоит в этом контейнере первым, то диапазон адресов контейнера будет определяться именно этим оператором range. Диапазоны, определяемые всеми остальными операторами range, стоящими после первого оператора range, или всеми операторами range, задающими диапазоны в строке описания контейнера, добавляются к текущему диапазону. С помощью оператора range к диапазону можно добавить отдельный адрес или набор адресов. Диапазон должен размещаться внутри группы адресов, определенной для контейнера подсети.
клиент	client *типаппаратный-адрес* NONE			Определяет контейнер клиента, запрещающий выделение адреса клиенту с указанным *типом аппаратуры* и *аппаратным адресом*. Если параметр *тип аппаратуры* равен 0, то в параметре *аппаратный адрес* должна быть задана строка ASCII. В остальных случаях *тип аппаратуры* - это тип аппаратного обеспечения клиента, а *аппаратный адрес* - аппаратный адрес клиента. Если *аппаратный адрес* - это строка, то она может быть заключена в кавычки. Если *аппаратный адрес* - это шестнадцатеричная строка, то он может быть записан в виде 0x*шестн._число* или hex *шестн._число*. *Диапазон* разрешает выделять адрес из указанного *диапазона* клиенту с указанным *типом аппаратуры* и *аппаратным адресом*. Для сравнения нескольких клиентов должны использоваться регулярные выражения.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умол-чанию	Описание
клиент	client *тип аппаратный-адрес* ANY			Определяет контейнер клиента, запрещающий выделение адреса клиенту с указанным *типом аппаратуры* и *аппаратным адресом*. Если параметр *тип аппаратуры* равен 0, то в параметре *аппаратный адрес* должна быть задана строка ASCII. В остальных случаях *тип аппаратуры* - это тип аппаратного обеспечения клиента, а *аппаратный адрес* - аппаратный адрес клиента. Если *аппаратный адрес* - это строка, то она может быть заключена в кавычки. Если *аппаратный адрес* - это шестнадцатеричная строка, то он может быть записан в виде 0х*шестн._число* или hex *шестн._число*. *Диапазон* разрешает выделять адрес из указанного *диапазона* клиенту с указанным *типом аппаратуры* и *аппаратным адресом*. Для сравнения нескольких клиентов должны использоваться регулярные выражения.
клиент	client *тип аппаратный адрес адрес*			Определяет контейнер клиента, запрещающий выделение адреса клиенту с указанным *типом аппаратуры* и *аппаратным адресом*. Если параметр *тип аппаратуры* равен 0, то в параметре *аппаратный адрес* должна быть задана строка ASCII. В остальных случаях *тип аппаратуры* - это тип аппаратного обеспечения клиента, а *аппаратный адрес* - аппаратный адрес клиента. Если *аппаратный адрес* - это строка, то она может быть заключена в кавычки. Если *аппаратный адрес* - это шестнадцатеричная строка, то он может быть записан в виде 0х*шестн._число* или hex *шестн._число*. *Диапазон* разрешает выделять адрес из указанного *диапазона* клиенту с указанным *типом аппаратуры* и *аппаратным адресом*. Для сравнения нескольких клиентов должны использоваться регулярные выражения.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
клиент	client *тип аппаратный адрес диапазон*			Определяет контейнер клиента, запрещающий выделение адреса клиенту с указанным *типом аппаратуры* и *аппаратным адресом*. Если параметр *тип аппаратуры* равен 0, то в параметре *аппаратный адрес* должна быть задана строка ASCII. В остальных случаях *тип аппаратуры* - это тип аппаратного обеспечения клиента, а *аппаратный адрес* - аппаратный адрес клиента. Если *аппаратный адрес* - это строка, то она может быть заключена в кавычки. Если *аппаратный адрес* - это шестнадцатеричная строка, то он может быть записан в виде 0x*шестн._число* или hex *шестн._число*. *Диапазон* разрешает выделять адрес из указанного *диапазона* клиенту с указанным *типом аппаратуры* и *аппаратным адресом*. Для сравнения нескольких клиентов должны использоваться регулярные выражения.
class	class *строка*	Да	Нет	Определяет контейнер класса с именем *строка*. Строка может быть заключена в кавычки. Если строка заключена в кавычки, то перед сравнением они удаляются. Кавычки необходимы в том случае, если строка содержит пробелы или символы табуляции. Использование контейнера class допустимо на любом уровне. Для задания набора адресов, которые могут быть предложены клиенту с данным классом, рекомендуется пользоваться диапазонами. Диапазон может задаваться как одиночный IP-адрес в десятичном формате с точками или как два IP-адреса, разделенных дефисом.
class	class *строка диапазон*	Да	Нет	Определяет контейнер класса с именем *строка*. Строка может быть заключена в кавычки. Если строка заключена в кавычки, то перед сравнением они удаляются. Кавычки необходимы в том случае, если строка содержит пробелы или символы табуляции. Использование контейнера class допустимо на любом уровне. Для задания набора адресов, которые могут быть предложены клиенту с данным классом, рекомендуется пользоваться диапазонами. Диапазон может задаваться как одиночный IP-адрес в десятичном формате с точками или как два IP-адреса, разделенных дефисом.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умол-чанию	Описание
сеть	network *ид сети маска*	Да	Нет	Задает ИД сети на основании информации о классе (например, 9.3.149.0 с маской сети 255.255.255.0 - это сеть 9.0.0.0 255.255.255.0). В таком контейнере сети могут находиться подсети с одинаковым ИД сети и маской сети. Если задан диапазон адресов, то все адреса этого диапазона образуют пул. Диапазон адресов должен относиться к данной сети. Используется полная адресация класса. Это ключевое слово допустимо только на глобальном уровне или на уровне контейнера базы данных. **Примечание:** Вместо ключевого слова network лучше использовать контейнер subnet.
сеть	network *ид_сети*	Да	Нет	Задает ИД сети на основании информации о классе (например, 9.3.149.0 с маской сети 255.255.255.0 - это сеть 9.0.0.0 255.255.255.0). В таком контейнере сети могут находиться подсети с одинаковым ИД сети и маской сети. Если задан диапазон адресов, то все адреса этого диапазона образуют пул. Диапазон адресов должен относиться к данной сети. Используется полная адресация класса. Это ключевое слово допустимо только на глобальном уровне или на уровне контейнера базы данных. **Примечание:** Вместо ключевого слова network лучше использовать контейнер subnet.
сеть	network *ид сети диапазон*			Задает ИД сети на основании информации о классе (например, 9.3.149.0 с маской сети 255.255.255.0 - это сеть 9.0.0.0 255.255.255.0). В таком контейнере сети могут находиться подсети с одинаковым ИД сети и маской сети. Если задан диапазон адресов, то все адреса этого диапазона образуют пул. Диапазон адресов должен относиться к данной сети. Используется полная адресация класса. Это ключевое слово допустимо только на глобальном уровне или на уровне контейнера базы данных. **Примечание:** Вместо ключевого слова network лучше использовать контейнер subnet.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
vendor	vendor *ид_вендора*	Да	Нет	Определяет контейнер вендора. Контейнеры вендоров используются для возврата клиенту опции 43. ИД вендора может задаваться в виде строки, заключенной в кавычки, либо в виде двоичной строки в формате 0x*шестн._число* или hex "*шестн._число*". После ИД вендора можно указать диапазон. Диапазон задается двумя IP-адресами в десятичном формате с точками, разделенными дефисом. После диапазона может следовать первая часть опции 43 в виде шестнадцатеричной или ASCII-строки. При наличии в контейнере опций они добавляются к данным опции 43. После обработки всех опций к данным добавляется опция конца списка. Для возврата клиенту других опций (помимо опции 43) используйте регулярное выражение, позволяющее сравнивать всех клиентов и определения нормальных опций, возвращаемых на основании ИД вендора. Если после ключевого слова **vendor** указано значение **pxe**, то будет создан контейнер vendor для клиента PXE. Если после ключевого слова **vendor** указано значение **pxeserver**, то будет создан контейнер vendor для сервера PXE.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умол-чанию	Описание
vendor	vendor *ид_вендора* hex ""	Да	Нет	Определяет контейнер вендора. Контейнеры вендоров используются для возврата клиенту опции 43. ИД вендора может задаваться в виде строки, заключенной в кавычки, либо в виде двоичной строки в формате 0x*шестн._число* или hex "*шестн._число*". После ИД вендора можно указать диапазон. Диапазон задается двумя IP-адресами в десятичном формате с точками, разделенными дефисом. После диапазона может следовать первая часть опции 43 в виде шестнадцатеричной или ASCII-строки. При наличии в контейнере опций они добавляются к данным опции 43. После обработки всех опций к данным добавляется опция конца списка. Для возврата клиенту других опций (помимо опции 43) используйте регулярное выражение, позволяющее сравнивать всех клиентов и определения нормальных опций, возвращаемых на основании ИД вендора. Если после ключевого слова **vendor** указано значение **pxe**, то будет создан контейнер vendor для клиента PXE. Если после ключевого слова **vendor** указано значение **pxeserver**, то будет создан контейнер vendor для сервера PXE.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умол-чанию	Описание
vendor	vendor *ид_вендора* hex ""	Да	Нет	Определяет контейнер вендора. Контейнеры вендоров используются для возврата клиенту опции 43. ИД вендора может задаваться в виде строки, заключенной в кавычки, либо в виде двоичной строки в формате 0x*шестн._число* или hex "*шестн._число*". После ИД вендора можно указать диапазон. Диапазон задается двумя IP-адресами в десятичном формате с точками, разделенными дефисом. После диапазона может следовать первая часть опции 43 в виде шестнадцатеричной или ASCII-строки. При наличии в контейнере опций они добавляются к данным опции 43. После обработки всех опций к данным добавляется опция конца списка. Для возврата клиенту других опций (помимо опции 43) используйте регулярное выражение, позволяющее сравнивать всех клиентов и определения нормальных опций, возвращаемых на основании ИД вендора. Если после ключевого слова **vendor** указано значение **pxe**, то будет создан контейнер vendor для клиента PXE. Если после ключевого слова **vendor** указано значение **pxeserver**, то будет создан контейнер vendor для сервера PXE.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умол-чанию	Описание
vendor	vendor *ид_вендора* 0x*данные*	Да	Нет	Определяет контейнер вендора. Контейнеры вендоров используются для возврата клиенту опции 43. ИД вендора может задаваться в виде строки, заключенной в кавычки, либо в виде двоичной строки в формате 0x*шестн._число* или hex "*шестн._число*". После ИД вендора можно указать диапазон. Диапазон задается двумя IP-адресами в десятичном формате с точками, разделенными дефисом. После диапазона может следовать первая часть опции 43 в виде шестнадцатеричной или ASCII-строки. При наличии в контейнере опций они добавляются к данным опции 43. После обработки всех опций к данным добавляется опция конца списка. Для возврата клиенту других опций (помимо опции 43) используйте регулярное выражение, позволяющее сравнивать всех клиентов и определения нормальных опций, возвращаемых на основании ИД вендора. Если после ключевого слова **vendor** указано значение **pxe**, то будет создан контейнер vendor для клиента PXE. Если после ключевого слова **vendor** указано значение **pxeserver**, то будет создан контейнер vendor для сервера PXE.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умол-чанию	Описание
vendor	vendor *ид_вендора* ""	Да	Нет	Определяет контейнер вендора. Контейнеры вендоров используются для возврата клиенту опции 43. ИД вендора может задаваться в виде строки, заключенной в кавычки, либо в виде двоичной строки в формате 0x*шестн._число* или hex "*шестн._число*". После ИД вендора можно указать диапазон. Диапазон задается двумя IP-адресами в десятичном формате с точками, разделенными дефисом. После диапазона может следовать первая часть опции 43 в виде шестнадцатеричной или ASCII-строки. При наличии в контейнере опций они добавляются к данным опции 43. После обработки всех опций к данным добавляется опция конца списка. Для возврата клиенту других опций (помимо опции 43) используйте регулярное выражение, позволяющее сравнивать всех клиентов и определения нормальных опций, возвращаемых на основании ИД вендора. Если после ключевого слова **vendor** указано значение **pxe**, то будет создан контейнер vendor для клиента PXE. Если после ключевого слова **vendor** указано значение **pxeserver**, то будет создан контейнер vendor для сервера PXE.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умол-чанию	Описание
vendor	vendor *ид_вендора диапазон*	Да	Нет	Определяет контейнер вендора. Контейнеры вендоров используются для возврата клиенту опции 43. ИД вендора может задаваться в виде строки, заключенной в кавычки, либо в виде двоичной строки в формате 0x*шестн._число* или hex "*шестн._число*". После ИД вендора можно указать диапазон. Диапазон задается двумя IP-адресами в десятичном формате с точками, разделенными дефисом. После диапазона может следовать первая часть опции 43 в виде шестнадцатеричной или ASCII-строки. При наличии в контейнере опций они добавляются к данным опции 43. После обработки всех опций к данным добавляется опция конца списка. Для возврата клиенту других опций (помимо опции 43) используйте регулярное выражение, позволяющее сравнивать всех клиентов и определения нормальных опций, возвращаемых на основании ИД вендора. Если после ключевого слова **vendor** указано значение **pxe**, то будет создан контейнер vendor для клиента PXE. Если после ключевого слова **vendor** указано значение **pxeserver**, то будет создан контейнер vendor для сервера PXE.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умол-чанию	Описание
vendor	vendor *ид_вендорадиапазон* hex ""	Да	Нет	Определяет контейнер вендора. Контейнеры вендоров используются для возврата клиенту опции 43. ИД вендора может задаваться в виде строки, заключенной в кавычки, либо в виде двоичной строки в формате 0x*шестн._число* или hex "*шестн._число*". После ИД вендора можно указать диапазон. Диапазон задается двумя IP-адресами в десятичном формате с точками, разделенными дефисом. После диапазона может следовать первая часть опции 43 в виде шестнадцатеричной или ASCII-строки. При наличии в контейнере опций они добавляются к данным опции 43. После обработки всех опций к данным добавляется опция конца списка. Для возврата клиенту других опций (помимо опции 43) используйте регулярное выражение, позволяющее сравнивать всех клиентов и определения нормальных опций, возвращаемых на основании ИД вендора. Если после ключевого слова **vendor** указано значение **pxe**, то будет создан контейнер vendor для клиента PXE. Если после ключевого слова **vendor** указано значение **pxeserver**, то будет создан контейнер vendor для сервера PXE.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умол-чанию	Описание
vendor	vendor *ид_вендорадиапазон* hex ""	Да	Нет	Определяет контейнер вендора. Контейнеры вендоров используются для возврата клиенту опции 43. ИД вендора может задаваться в виде строки, заключенной в кавычки, либо в виде двоичной строки в формате 0x*шестн._число* или hex "*шестн._число*". После ИД вендора можно указать диапазон. Диапазон задается двумя IP-адресами в десятичном формате с точками, разделенными дефисом. После диапазона может следовать первая часть опции 43 в виде шестнадцатеричной или ASCII-строки. При наличии в контейнере опций они добавляются к данным опции 43. После обработки всех опций к данным добавляется опция конца списка. Для возврата клиенту других опций (помимо опции 43) используйте регулярное выражение, позволяющее сравнивать всех клиентов и определения нормальных опций, возвращаемых на основании ИД вендора. Если после ключевого слова **vendor** указано значение **pxe**, то будет создан контейнер vendor для клиента PXE. Если после ключевого слова **vendor** указано значение **pxeserver**, то будет создан контейнер vendor для сервера PXE.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умол-чанию	Описание
vendor	vendor *ид_вендора диапазон* 0xзначение	Да	Нет	Определяет контейнер вендора. Контейнеры вендоров используются для возврата клиенту опции 43. ИД вендора может задаваться в виде строки, заключенной в кавычки, либо в виде двоичной строки в формате 0x*шестн._число* или hex "*шестн._число*". После ИД вендора можно указать диапазон. Диапазон задается двумя IP-адресами в десятичном формате с точками, разделенными дефисом. После диапазона может следовать первая часть опции 43 в виде шестнадцатеричной или ASCII-строки. При наличии в контейнере опций они добавляются к данным опции 43. После обработки всех опций к данным добавляется опция конца списка. Для возврата клиенту других опций (помимо опции 43) используйте регулярное выражение, позволяющее сравнивать всех клиентов и определения нормальных опций, возвращаемых на основании ИД вендора. Если после ключевого слова **vendor** указано значение **pxe**, то будет создан контейнер vendor для клиента PXE. Если после ключевого слова **vendor** указано значение **pxeserver**, то будет создан контейнер vendor для сервера PXE.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умол-чанию	Описание
vendor	vendor *ид_вендора диапазон* ""	Да	Нет	Определяет контейнер вендора. Контейнеры вендоров используются для возврата клиенту опции 43. ИД вендора может задаваться в виде строки, заключенной в кавычки, либо в виде двоичной строки в формате 0x*шестн._число* или hex "*шестн._число*". После ИД вендора можно указать диапазон. Диапазон задается двумя IP-адресами в десятичном формате с точками, разделенными дефисом. После диапазона может следовать первая часть опции 43 в виде шестнадцатеричной или ASCII-строки. При наличии в контейнере опций они добавляются к данным опции 43. После обработки всех опций к данным добавляется опция конца списка. Для возврата клиенту других опций (помимо опции 43) используйте регулярное выражение, позволяющее сравнивать всех клиентов и определения нормальных опций, возвращаемых на основании ИД вендора. Если после ключевого слова **vendor** указано значение **pxe**, то будет создан контейнер vendor для клиента PXE. Если после ключевого слова **vendor** указано значение **pxeserver**, то будет создан контейнер vendor для сервера PXE.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умол-чанию	Описание
vendor	vendor pxe	Да	Нет	Определяет контейнер вендора. Контейнеры вендоров используются для возврата клиенту опции 43. ИД вендора может задаваться в виде строки, заключенной в кавычки, либо в виде двоичной строки в формате 0x*шестн._число* или hex "*шестн._число*". После ИД вендора можно указать диапазон. Диапазон задается двумя IP-адресами в десятичном формате с точками, разделенными дефисом. После диапазона может следовать первая часть опции 43 в виде шестнадцатеричной или ASCII-строки. При наличии в контейнере опций они добавляются к данным опции 43. После обработки всех опций к данным добавляется опция конца списка. Для возврата клиенту других опций (помимо опции 43) используйте регулярное выражение, позволяющее сравнивать всех клиентов и определения нормальных опций, возвращаемых на основании ИД вендора. Если после ключевого слова **vendor** указано значение **pxe**, то будет создан контейнер vendor для клиента PXE. Если после ключевого слова **vendor** указано значение **pxeserver**, то будет создан контейнер vendor для сервера PXE.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умол-чанию	Описание
vendor	vendor pxeserver	Да	Нет	Определяет контейнер вендора. Контейнеры вендоров используются для возврата клиенту опции 43. ИД вендора может задаваться в виде строки, заключенной в кавычки, либо в виде двоичной строки в формате 0x*шестн._число* или hex "*шестн._число*". После ИД вендора можно указать диапазон. Диапазон задается двумя IP-адресами в десятичном формате с точками, разделенными дефисом. После диапазона может следовать первая часть опции 43 в виде шестнадцатеричной или ASCII-строки. При наличии в контейнере опций они добавляются к данным опции 43. После обработки всех опций к данным добавляется опция конца списка. Для возврата клиенту других опций (помимо опции 43) используйте регулярное выражение, позволяющее сравнивать всех клиентов и определения нормальных опций, возвращаемых на основании ИД вендора. Если после ключевого слова **vendor** указано значение **pxe**, то будет создан контейнер vendor для клиента PXE. Если после ключевого слова **vendor** указано значение **pxeserver**, то будет создан контейнер vendor для сервера PXE.
входящая опция	входящая опция *номер данные*	Да	Нет	Указывает контейнер, с которым следует сравнивать переданные клиентом опции с данным номером. *Номер* означает номер опции. *Данные* - это ключ для сравнения с этим контейнером в ходе выбора адреса и опции для данного клиента. Для стандартных опций *данные* указываются в явной форме: строка, заключенная в кавычки, IP-адрес или целое число; кроме того, их можно указать в форме шестнадцатеричной строки байтов, начинающейся с символов `0x`. Для опций, неизвестных серверу, допустим только второй вариант. Кроме того, *данные* могут представлять собой выражение, которое следует сравнить с содержимым строки данных опции, переданной клиентом. Такие выражения записываются в кавычках и начинаются с символов `"!` (кавычка и восклицательный знак). Опции, неизвестные серверу, должны указываться в виде шестнадцатеричной строки байтов БЕЗ начальных символов `0x`.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
входящая опция	входящая опция *номер данные диапазон*	Да	Нет	Указывает контейнер, с которым следует сравнивать переданные клиентом опции с данным номером. *Номер* означает номер опции. *Данные* - это ключ для сравнения с этим контейнером в ходе выбора адреса и опции для данного клиента. Для стандартных опций *данные* указываются в явной форме: строка, заключенная в кавычки, IP-адрес или целое число; кроме того, их можно указать в форме шестнадцатеричной строки байтов, начинающейся с символов `0x`. Для опций, неизвестных серверу, допустим только второй вариант. Кроме того, *данные* могут представлять собой выражение, которое следует сравнить с содержимым строки данных опции, переданной клиентом. Такие выражения записываются в кавычках и начинаются с символов `"!` (кавычка и восклицательный знак). Опции, неизвестные серверу, должны указываться в виде шестнадцатеричной строки байтов БЕЗ начальных символов `0x`.
virtual	virtual fill *ИД ИД* ...	Нет	Нет	Определяет виртуальную подсеть со стратегией. Слово `fill` означает, что перед переходом к следующему контейнеру необходимо использовать все адреса текущего контейнера. Слово `rotate` указывает, что для каждого запроса адрес выбирается из следующего указанного пула. `sfill` и `srotate` означают то же, что `fill` и `rotate`, однако в этом случае производится поиск, чтобы узнать, не соответствует ли клиент каким-либо контейнерам, вендорам или классам в подсети. Если обнаружено совпадение с контейнером, в котором выделяется адрес, то, адрес выбирается из этого контейнера, а не назначается в соответствии со стратегией. Число ИД в списке не ограничено. *ИД* - это либо ИД подсети, либо метка, заданная в описании подсети. Метка нужна при наличии нескольких подсетей с одинаковым идентификатором.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умол-чанию	Описание
virtual	virtual sfill *ИД ИД* ...	Нет	Нет	Определяет виртуальную подсеть со стратегией. Слово `fill` означает, что перед переходом к следующему контейнеру необходимо использовать все адреса текущего контейнера. Слово `rotate` указывает, что для каждого запроса адрес выбирается из следующего указанного пула. `sfill` и `srotate` означают то же, что `fill` и `rotate`, однако в этом случае производится поиск, чтобы узнать, не соответствует ли клиент каким-либо контейнерам, вендорам или классам в подсети. Если обнаружено совпадение с контейнером, в котором выделяется адрес, то, адрес выбирается из этого контейнера, а не назначается в соответствии со стратегией. Число ИД в списке не ограничено. *ИД* - это либо ИД подсети, либо метка, заданная в описании подсети. Метка нужна при наличии нескольких подсетей с одинаковым идентификатором.
virtual	virtual rotate *ИД ИД* ...	Нет	Нет	Определяет виртуальную подсеть со стратегией. Слово `fill` означает, что перед переходом к следующему контейнеру необходимо использовать все адреса текущего контейнера. Слово `rotate` указывает, что для каждого запроса адрес выбирается из следующего указанного пула. `sfill` и `srotate` означают то же, что `fill` и `rotate`, однако в этом случае производится поиск, чтобы узнать, не соответствует ли клиент каким-либо контейнерам, вендорам или классам в подсети. Если обнаружено совпадение с контейнером, в котором выделяется адрес, то, адрес выбирается из этого контейнера, а не назначается в соответствии со стратегией. Число ИД в списке не ограничено. *ИД* - это либо ИД подсети, либо метка, заданная в описании подсети. Метка нужна при наличии нескольких подсетей с одинаковым идентификатором.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
virtual	virtual srotate *ИД ИД* ...	Нет	Нет	Определяет виртуальную подсеть со стратегией. Слово `fill` означает, что перед переходом к следующему контейнеру необходимо использовать все адреса текущего контейнера. Слово `rotate` указывает, что для каждого запроса адрес выбирается из следующего указанного пула. `sfill` и `srotate` означают то же, что `fill` и `rotate`, однако в этом случае производится поиск, чтобы узнать, не соответствует ли клиент каким-либо контейнерам, вендорам или классам в подсети. Если обнаружено совпадение с контейнером, в котором выделяется адрес, то, адрес выбирается из этого контейнера, а не назначается в соответствии со стратегией. Число ИД в списке не ограничено. *ИД* - это либо ИД подсети, либо метка, заданная в описании подсети. Метка нужна при наличии нескольких подсетей с одинаковым идентификатором.
virtual		Нет	Нет	Определяет виртуальную подсеть со стратегией. Слово `fill` означает, что перед переходом к следующему контейнеру необходимо использовать все адреса текущего контейнера. Слово `rotate` указывает, что для каждого запроса адрес выбирается из следующего указанного пула. `sfill` и `srotate` означают то же, что `fill` и `rotate`, однако в этом случае производится поиск, чтобы узнать, не соответствует ли клиент каким-либо контейнерам, вендорам или классам в подсети. Если обнаружено совпадение с контейнером, в котором выделяется адрес, то, адрес выбирается из этого контейнера, а не назначается в соответствии со стратегией. Число ИД в списке не ограничено. *ИД* - это либо ИД подсети, либо метка, заданная в описании подсети. Метка нужна при наличии нескольких подсетей с одинаковым идентификатором.
inorder:	inorder: *id id* ...	Нет	Нет	Определяет виртуальную подсеть со стратегией заполнения, т.е. перед переходом к следующему контейнеру должны быть использованы все адреса текущего контейнера. Число ИД в списке не ограничено. *ИД* - это либо ИД подсети, либо метка, заданная в описании подсети. Метка нужна при наличии нескольких подсетей с одинаковым идентификатором.

Ключевое слово	Формат	Вложенные контейнеры	Значение по умолчанию	Описание
balance:	balance: *ИД ИД* ...	Нет	Нет	Определяет виртуальную подсеть со стратегией смены адресов, при которой каждый следующий адрес выбирается из следующего контейнера. Число ИД в списке не ограничено. *ИД* - это либо ИД подсети, либо метка, заданная в описании подсети. Метка нужна при наличии нескольких подсетей с одинаковым идентификатором.
bootstrapserver	bootstrapserver *IP-адрес*	Нет	Нет	Указывает, на каком сервере находятся файлы TFTP, которые должны использоваться клиентами после получения ими пакетов BOOTP или DHCP. Это значение задается в поле **siaddr** пакета. Допустимо на уровне любого контейнера.
giaddrfield	giaddrfield *IP-адрес*	Нет	Нет	Задает поле giaddrfield для ответных сообщений (пакетов). **Примечание:** Данная спецификация недопустима в протоколах BOOTP и DHCP, однако для работы некоторых клиентов необходимо, чтобы в поле **giaddr** был указан шлюз по умолчанию. Из-за возможных конфликтов ключевое слово giaddrfield должно использоваться только внутри контейнеров клиентов, хотя оно может работать на любом уровне.
bootfile	bootfile *полное_имя*	Нет	Нет	Определяет загрузочный файл, который должен быть указан в ответном пакете. Допустимо на уровне любого контейнера. Взаимодействие элементов, указанных в поступающем пакете с операторами загрузочного файла и домашнего каталога определяется стратегией загрузочного файла.
pxebootfile	pxebootfile *архитектура-системы старшая-версия младшая-версия загрузочный-файл тип уровень*	Нет	Нет	Указывает загрузочный файл, имя которого будет передано клиенту PXE. Функция разбора файла конфигурации выдает сообщение об ошибке, если число параметров после ключевого слова меньше 4, игнорирует все параметры после седьмого, и полагает, что тип и уровень равны нулю, если указано ровно 4 параметра. Это ключевое слово допустимо только в контейнере.

Информация о других опциях приведена в разделах “Известные опции файла сервера DHCP” на стр. 219 и “Опции контейнера вендора PXE” на стр. 306.

Демоны TCP/IP

Демоны (или *серверы*) - это процессы, которые работают в фоновом режиме и выполняют запросы других процессов. **Протокол управления передачей/Протокол Internet** применяет программы-демоны для выполнения определенных функций в операционной системе.

Эти программы выполняются в фоновом режиме, то есть не прерывают работу других процессов (если это не относится к функциям демона).

Демоны могут запускаться из командной строки, из сценариев и другими демонами. Кроме того, для управления демонами могут применяться демон **inetd**, сценарий **rc.tcpip** и Контроллер системных ресурсов (SRC).

Подсистемы и субсерверы

Подсистемами называются демоны, работой которых управляет SRC. *Субсервер* - это демон, который управляется подсистемой. (Команды и имена программ-демонов обычно обозначаются буквой **d** в конце имени.)

Понятия подсистемы и субсервера являются взаимоисключающими. Это значит, что демон не может быть одновременно и подсистемой, и субсервером. Единственной подсистемой **TCP/IP**, которая управляет другими программами-демонами, является демон **inetd**. Все субсерверы **TCP/IP** являются также субсерверами **inetd**.

Список демонов **TCP/IP** приведен в “Демоны TCP/IP” на стр. 437.

Управление ресурсами системы

Кроме других функций, SRC позволяет запускать и завершать программы-демоны, а также выполнять трассировку их деятельности. Кроме того, SRC предоставляет возможность группировать демоны в подсистемы и субсерверы.

Контроллер системных ресурсов специально разработан для помощи в управлении работой демонов. SRC позволяет устанавливать флаги и параметры для каждой из команд демонов.

Дополнительные сведения об управлени системными ресурсами приведены в разделе System Resource Controller в книге *Управление операционной системой и устройствами*.

Список команд SRC приведен в “Команды SRC” на стр. 435.

Настройка демона inetd

Выполните эти шаги для настройки демона **TCP/IP inetd**.

Для настройки программы-демона **inetd**:

1. Укажите, какие субсерверы будут запускаться при запуске **inetd**.
2. Укажите параметры перезапуска субсерверов путем изменения параметров демона **inetd**.

Таблица 75. Настройка задач демона inetd

Процедура	Команды быстрого доступа SMIT	Команда или файл
Запуск демона **inetd**	`smit mkinetd`	**startsrc -s inetd**
Изменение параметров демона **inetd**	`smit chinetd` или `smit lsinetd`	
Завершение демона **inetd**	`smit rminetd`	**stopsrc -s inetd**
Получение списка всех субсерверов **inetd**	`smit inetdconf`	
Добавить субсервер **inetd** [1]	`smit mkinetdconf`	отредактируйте файл `/etc/inetd.conf` и вызовите команду **refresh -s inetd** или **kill -1** *inetdPID*[2]
Изменить/показать параметры субсервера **inetd**	`smit inetdconf`	отредактируйте файл `/etc/inetd.conf` и вызовите команду **refresh -s inetd** или **kill -1** *inetdPID*[2]
Удаление субсервера **inetd**	`smit rminetd`	отредактируйте файл `/etc/inetd.conf` и вызовите команду **refresh -s inetd** или **kill -1** *inetdPID*[2]

Примечание:

1. При добавлении субсервера **inetd** конфигурация демона **inetd** изменяется таким образом, что он вызывает субсервер при необходимости.
2. Команды **refresh** и **kill** информируют демон **inetd** об изменении файла конфигурации.

Сетевые службы клиента

Сетевые службы клиента - это набор протоколов **TCP/IP**, доступных в операционной системе. Для работы с ними применяется инструмент SMIT (команда `smit clientnet`).

Каждому протоколу (или службе) соответствует номер порта, который он использует в сети. Такой номер называется *стандартным портом*. По соглашению между программистами для ссылки на порт можно использовать как имена, так и номера. Например, почтовый протокол **TCP/IP** использует порт 25 и известен под именем **smtp**. Если протокол указан в файле `/etc/services` (строка с именем протокола не помечена как комментарий), то хост может применять этот протокол.

По умолчанию в файле `/etc/services` определены все протоколы **TCP/IP**. Этот файл не требуется изменять вручную. Если вы написали приложение клиент-сервер и хотите зарезервировать порт и имя новой службы, добавьте запись об этой службе в файл `/etc/services`. При назначении портов новым службам в файле `/etc/services` не забывайте о том, что номера портов с 0 по 1024 зарезервированы для использования системой.

Таблица 76. Задачи сетевых служб клиента

Процедура	Команды быстрого доступа SMIT	Команда или файл
Получение списка всех служб	`smit lsservices`	view `/etc/services`
Добавление службы	`smit mkservices`	edit `/etc/services`
Изменить/показать параметры службы	`smit chservices`	edit `/etc/services`
Удаление службы	`smit rmservices`	edit `/etc/services`

Сетевые службы сервера

К Сетевым службам сервера относится управление удаленным доступом, запуск и завершение работы **TCP/IP** и управление драйвером устройства **pty**. Более подробная информация об этих функциях приведена в следующей таблице.

Драйвер устройства **pty** автоматически устанавливается вместе с системой. По умолчанию он настраивается для поддержки 16 символических ссылок типа BSD и может использоваться системой во время загрузки.

Таблица 77. Задачи сетевых служб сервера

Процедура	Команды быстрого доступа SMIT	Команда или файл
Управление удаленным доступом		См. разделы "Remote Command Execution Access" и "Restricted File Transfer Program Users" в книге *Защита*.
Запуск, перезапуск и завершение подсистем TCP/IP	`smit otherserv`	См. раздел “Управление ресурсами системы” на стр. 360.
Изменить/показать параметры драйвера pty	`smit chgpty`	**chdev -l pty0 -P -a num=***X* где *X*может принимать значения от 0 до 64
Сделать драйвер устройства pty недоступным	`smit pty` затем выберите **Удалить PTY; Сохранить определение**	Связанные команды или файлы отсутствуют.
Сделать драйвер устройства pty доступным	`smit pty` затем выберите **Настроить определенный PTY**	Связанные команды или файлы отсутствуют.
Создать отчет об ошибках	`smit errpt`	Связанные команды или файлы отсутствуют.
Трассировать pty	`smit trace`	Связанные команды или файлы отсутствуют.

Маршрутизация TCP/IP

Маршрутом называется путь, по которому пакеты пересылаются от отправителя к получателю.

Маршрут определяет не полный путь, а только сегмент пути от хоста до шлюза (или от шлюза до шлюза), который может переслать пакеты целевому хосту. Существует пять типов маршрутов:

Элемент	Описание
маршрут до хоста	Определяет шлюз, который может переслать пакеты указанному хосту в другой сети.
маршрут к сети	Определяет шлюз, который может переслать пакеты другому хосту указанной сети.
маршрут по умолчанию	Определяет шлюз, которому будут отправлены пакеты, если не был задан маршрут до целевого хоста или маршрут к сети целевого хоста.
циклический маршрут	Маршрут по умолчанию для всех пакетов, отправляемых по адресам локальной сети. IP-адрес циклического маршрута всегда 127.0.0.1.
маршрут оповещения	Маршрут по умолчанию для всех пакетов оповещения. Каждой подсети, в которой у сети есть IP-адрес, автоматически присваиваются два маршрута оповещения (один - адресу подсети, а другой - адресу оповещения подсети).

Список маршрутов хранится в *таблице маршрутизации* ядра. Описание маршрута содержит такую информацию, как список сетей, достижимых локальным хостом, и список шлюзов для отправки пакетов в удаленные сети. При получении дейтаграммы шлюз ищет в таблицах маршрутизации следующий узел ее маршрута до целевого хоста и отправляет дейтаграмму этому узлу.

В таблицу маршрутизации ядра можно добавлять несколько маршрутов к одному и тому же хосту. Процедура выбора маршрута сначала находит все маршруты, соответствующие запросу, а потом выбирает маршрут с минимальной метрикой расстояния. При наличии нескольких маршрутов одинаковой длины выбирается тот маршрут, который задан наиболее точно. Если несколько маршрутов совпадают по обоим критериям, то эти маршруты применяются по-очереди.

Статическая и динамическая маршрутизация

В **TCP/IP** предусмотрено два типа маршрутизации: *статическая* и *динамическая*.

Статическая маршрутизация означает, что таблицы маршрутизации обслуживаются вручную с помощью команды **route**. Этот тип маршрутизации рекомендуется применять тогда, когда ваша сеть взаимодействует с одной или двумя другими сетями. Однако если сеть соединена с большим числом сетей, то число шлюзов резко возрастает, и для обслуживания таблиц маршрутизации вручную требуется значительное время.

При динамической маршрутизации таблицы маршрутизации автоматически обновляются демонами. Демоны маршрутизации непрерывно получают информацию, рассылаемую путем оповещения другими демонами маршрутизации, поэтому они постоянно обновляют таблицы маршрутизации.

В **TCP/IP** предусмотрено два демона, поддерживающих динамическую маршрутизацию: **routed** и **gated**. Демон **gated** поддерживает одновременно **Протокол информации о маршрутизации (RIP)**, **Протокол информации о маршрутизации следующего поколения (RIPng)**, **Протокол внешних шлюзов (EGP)**, **Протокол граничных шлюзов (BGP)** и **BGP4+**, **протокол (HELLO)**, **Протокол кратчайшего пути (OSPF)**, **протоколы IS-IS** и **ICMP и ICMPv6/Router Discovery**. Кроме того, демон **gated** поддерживает **Простой протокол управления сетью (SNMP)**. Демон **routed** поддерживает только **Протокол информации о маршрутизации**.

В зависимости от опций, указанных при запуске демона маршрутизации, он может работать в одном из двух режимов - *пассивном* или *активном*. В активном режиме демон маршрутизации периодически отправляет шлюзам и хостам оповещающие сообщения, содержащие информацию о маршрутизации для их локальных сетей, а также получает информацию о маршрутизации от других хостов и шлюзов. В пассивном режиме демон маршрутизации только получает информацию о маршрутизации и не пытается обновить информацию о маршрутизации удаленных шлюзов (то есть он не распространяет собственную информацию о маршрутизации).

Два описанных типа маршрутизации применяются не только шлюзами, но и хостами сети. Статическая маршрутизация применяется для шлюзов точно так же, как и для других хостов. Однако демоны динамической маршрутизации, которые выполняются не на шлюзах, могут работать только в пассивном (тихом) режиме.

Шлюзы маршрутизации TCP/IP

Шлюз - это один из типов маршрутизаторов. *Маршрутизаторы* соединяют несколько сетей и выполняют функции маршрутизации пакетов. Например, некоторые маршрутизаторы передают данные по маршруту на уровне сетевого интерфейса или на физическом уровне. *Шлюзы* осуществляют маршрутизацию на сетевом уровне.

Шлюзы пересылают IP-дейтаграммы, полученные от других шлюзов или хостов, хостам локальной сети, а также передают IP-дейтаграммы из одной сети в другую. Например, если шлюз соединяет две сети Token-Ring, то у него есть две карты адаптера Token-Ring, с каждой из которых связан собственный сетевой интерфейс Token-Ring. Шлюз получает дейтаграммы по одному сетевому интерфейсу и отправляет их с помощью другого интерфейса. Периодически шлюзы проверяют состояние своих сетевых соединений с помощью сообщений о состоянии интерфейса.

При пересылке пакетов шлюзы ориентируются на адрес целевой сети, а не на адрес конкретного хоста. То есть шлюз не должен хранить список маршрутов до всех возможных целевых хостов пакета. Шлюз направляет пакет в сеть, к которой подключен целевой хост. За пересылку пакета целевому хосту будут отвечать маршрутизаторы этой сети. Таким образом, обычно для работы шлюза требуется только ограниченный объем оперативной памяти и, возможно, ограниченный объем дисковой памяти.

Расстояние, проходимое сообщением от отправителя к получателю, измеряется числом *транзитных участков между шлюзами*. Путь до шлюза, расположенного в сети, к которой подключен источник - это нулевой транзитный участок, путь к сети, которая достижима с этого шлюза - первый транзитный участок, и т.д. Расстояние до получателя сообщения обычно измеряется в *числе транзитных участков* (иногда оно называется *метрикой*).

Внутренние и внешние шлюзы маршрутизации:

Внутренними шлюзами называются шлюзы, которые относятся к одной и той же автономной системе. Эти шлюзы обмениваются сообщениями с помощью **Протокола информации о маршрутизации (RIP)**, **Протокола информации о маршрутизации следующего поколения (RIPng)**, **межсистемного протокола ISIS** protocol, **протокола кратчайшего пути (OSPF)** или **протокола HELLO**. Внешние шлюзы относятся к различным автономным системам. Они работают на основе **Протокола внешних шлюзов (EGP)**, **Протокола граничных шлюзов (BGP)** или **BGP4+**.

Для примера рассмотрим две автономные системы. Первая из них состоит из сетей, которые управляются компанией Widget. Вторая система состоит из сетей, которые управляются компанией Gadget. В компании Widget есть один компьютер с именем apple, выполняющий роль шлюза для соединения с Internet. В компании Gadget есть один компьютер с именем orange, который также выполняет роль шлюза для соединений с Internet. Автономные системы обеих компаний состоят из нескольких внутренних сетей. Шлюзы, соединяющие внутренние сети, представляют собой внутренние шлюзы. Однако apple и orange - внешние шлюзы.

Каждый внешний шлюз взаимодействует не со всеми внешними шлюзами. Вместо этого, внешний шлюз регистрирует набор соседей (других внешних шлюзов), с которыми он обменивается информацией. Соседи определяются не по географическому принципу, а в зависимости от соединений, установленных между шлюзами. Соседний шлюз, в свою очередь, обменивается информацией со своими соседними внешними шлюзами. В результате информация о маршрутизации распространяется среди внешних шлюзов, и их таблицы маршрутизации обновляются.

Информация о маршрутизации представлена в виде пар (N,D), где N - это сеть, а D - расстояние до этой сети, указанное в соответствии с применяемой метрикой расстояния. Каждый шлюз рассылает информацию о

достижимых сетях и о расстояниях до этих сетей. Шлюз, получивший такое сообщение, подсчитывает наикратчайший путь до других сетей и передает эту информацию своим соседям. Таким образом, каждый внешний шлюз непрерывно получает информацию о маршрутизации, обновляет свои таблицы маршрутизации и передает эту информацию дальше своим соседям.

Протоколы шлюза:

Все шлюзы, внутренние и внешние, обмениваются информацией согласно определенным протоколам. Ниже приведено краткое описание наиболее распространенных протоколов шлюзов **TCP/IP**:

Протокол HELLO (HELLO)
HELLO - это один из протоколов для обмена информацией между внутренними шлюзами. **HELLO** подсчитывает наикратчайший путь к другим сетям на основе минимальной временной задержки.

Протокол информации о маршрутизации (RIP)
Протокол информации о маршрутизации - это один из протоколов для обмена информацией между внутренними шлюзами. Как и **протокол HELLO**, **RIP** подсчитывает наикратчайший путь к другим сетям. В отличие от **HELLO**, **RIP** оценивает расстояние не на основе временной задержки, а на основе числа транзитных участков. Демон **gated** сохраняет значения всех метрик в виде временных задержек, поэтому он преобразует число транзитных участков **RIP** во временную задержку.

Протокол информации о маршрутизации следующего поколения
RIPng - это расширение протокола **RIP** для поддержки **IPv6**.

Протокол кратчайшего пути (OSPF)
OPSF - это протокол для обмена информацией между внутренними шлюзами. Этот протокол основан на обмене сообщениями о состоянии канала связи между маршрутизаторами и сетями, и он лучше приспособлен для сложных сетей со многими маршрутизаторами, чем **RIP**. Кроме того, он поддерживает работу с несколькими равноправными маршрутами.

Протокол внешних шлюзов (EGP)
Протокол внешних шлюзов предназначен для обмена информацией между внешними шлюзами. **EGP** не подсчитывает наикратчайший путь к другим сетям. Он применяется внешними шлюзами для того, чтобы получить информацию о достижимости сетей.

Протокол граничных шлюзов (BGP)
Этот протокол предназначен для обмена информацией между внешними шлюзами. Он позволяет шлюзам различных автономных систем обмениваться информацией о достижимости сетей и предоставляет больше возможностей, чем **EGP**. Дополнительная информация о каждом маршруте хранится в атрибутах **EGP**, которые позволяют выбрать наилучший маршрут.

Протокол граничных шлюзов 4+
BGP4+ - это расширение протокола **BGP** версии 4 с поддержкой **IPv6** и другими улучшениями.

Межшлюзовый протокол (IS-IS)
Протокол **IS-IS** применяется внутренними шлюзами при общении друг с другом. Это протокол, хранящий информацию о состоянии канала. Он может передавать пакеты ISO/CLNP и, как и протокол **OSPF**, применяет для определения маршрута алгоритм кратчайшего пути.

Заметки о шлюзах

Перед настройкой шлюза выполните следующие действия.

Перед настройкой шлюзов для своей сети нужно выполнить следующие действия:

1. Определить число необходимых шлюзов. Число необходимых шлюзов зависит от следующих факторов:
   - Число сетей, которые вы хотите соединить.
   - Способ соединения этих сетей.
   - Уровень загруженности соединенных сетей.

   Предположим, что нужно установить соединение между сетями 1, 2 и 3.

На рисунке показаны три сети с номерами 1, 2 и 3. Сети 1 и 2 соединены через шлюз A. Сети 2 и 3

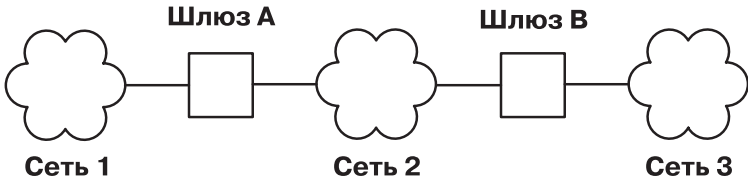


Рисунок 24. Простая конфигурация шлюза

соединены через шлюз B.

Для того чтобы непосредственно соединить сеть 1 и сеть 2, вам потребуется один шлюз (шлюз A). Для того чтобы сеть 2 непосредственно соединить с сетью 3, вам потребуется еще один шлюз (шлюз B). Если теперь предположить, что все необходимые маршруты определены, пользователи всех трех сетей смогут передавать друг другу информацию.

Однако если нагрузка в сети 2 очень высока, то данные будут передаваться из сети 1 в сеть 3 со значительной задержкой. Если большая часть данных передается между сетями 1 и 3, то разумно устанавливать соединение напрямую, а не через промежуточную сеть. Для этого необходимо установить соединение между дополнительной парой шлюзов: шлюзом C (в сети 1) и шлюзом D (в сети 3). Предложенный вариант неэффективен, так как один шлюз может соединять более двух сетей.

Более эффективный способ - соединить шлюз A со шлюзом B и с сетью 2. Для этого в каждом из шлюзов A и B необходимо установить второй сетевой адаптер. В общем случае число сетей, которые могут быть соединены через один шлюз, ограничивается числом карт адаптера, которые можно установить в шлюз.

2. Выбрать тип маршрутизации.

   Статическую маршрутизацию применяют в небольших сетях, конфигурация которых изменяется редко. Если же вы работаете с большой сетью, конфигурация которой часто меняется, то рекомендуется применять динамическую маршрутизацию. В сети может применяться комбинация статической и динамической маршрутизации. Это значит, что для некоторых маршрутов будет задано статическое определение, а другие маршруты будут обновляться демонами. Информация о статических маршрутах не передается другим шлюзам и не обновляется демонами маршрутизации.

3. При использовании динамической маршрутизации выберите демон маршрутизации в зависимости от типа шлюза и протокола, поддерживаемого этим шлюзом. Для внутреннего шлюза, поддерживающего только протокол **RIP**, выберите демон **routed**. Если шлюз должен поддерживать несколько протоколов или представляет собой внешний шлюз, то выберите демон **gated**.

   Примечание: Одновременное применение демонов **gated** и **routed** на одном хосте может привести к непредсказуемым результатам.

Настройка шлюза

Для настройки шлюза выполните описанную ниже процедуру.

Для простоты в этой процедуре предполагается, что шлюз должен соединять две сети, а соответствующая система уже настроена для работы с одной из этих сетей.

1. Установите и настройте второй сетевой адаптер, если вы еще этого не сделали. (Обратитесь к разделам “Установка сетевого адаптера” на стр. 159 и “Управление и настройка адаптера” на стр. 160.)
2. Выберите IP-адрес для второго сетевого интерфейса и настройте его в соответствии с инструкциями из раздела “Управление сетевым интерфейсом” на стр. 165.
3. Добавьте маршрут ко второй сети.
4. Для того чтобы настроить компьютер в качестве маршрутизатора, соединяющего сети **TCP/IP**, введите:

   ```
   no -o ipforwarding=1
   ```

Теперь компьютер шлюза доступен в обеих сетях, к которым он непосредственно подключен.

1. Если для соединения с хостами за пределами этих двух сетей вы хотите применять статическую маршрутизацию, то добавьте все необходимые маршруты.

2. Для настройки динамической маршрутизации выполните инструкции из раздела “Настройка демона routed” на стр. 368 или “Настройка демона gated” на стр. 369. Если ваша сеть подключена к Internet, выполните инструкции из раздела “Номера автономных систем” на стр. 371.

Таблица 78. Задачи настройки шлюза

Процедура	Команды быстрого доступа SMIT	Файл или команда
Просмотр таблицы маршрутизации	`smit lsroute`	**netstat -rn**[1]
Добавление статического маршрута	`smit mkroute`	**route add** *целевой-хост шлюз*[2]
Удаление статического маршрута	`smit rmroute`	**route delete** *целевой-хост шлюз*[2]
Очистка таблицы маршрутизации	`smit fshrttbl`	**route flush**

Примечание:

1. В таблице предусмотрены столбцы для адреса назначения, адреса шлюза, флагов, числа транзитных участков и сетевого интерфейса. (Более подробная информация об этих полях приведена в описании команды **netstat** в книге *Справочник по командам, том 4*). Если кадры не доставлены целевому хосту, а в таблице маршрутизации задан правильный маршрут, то, возможно, произошло одно из следующих событий:
   - Сбой в работе сети.
   - Сбой в работе удаленного хоста или шлюза.
   - Удаленный хост или шлюз выключен или не готов к приему кадров.
   - На удаленном хосте не определен обратный маршрут к исходной сети.
2. Значение *конечный_пункт_маршрута* - это IP-адрес или имя целевого хоста или сети, а значение *шлюз* - это IP-адрес или имя шлюза. (В маршруте по умолчанию в качестве целевого хоста указывается 0).

Ограничения применения маршрутизации

Использование маршрута можно ограничить таким образом, чтобы он применялся только некоторыми пользователями. Ограничения основываются на принадлежности пользователя основной группе с указанным ИД.

С помощью команды **route** вы можете указать до 32 групп, которым разрешено или не разрешено применение маршрута. Если вы задаете список разрешенных групп, то любой пользователь из любой группы может использовать маршрут. Если вы задаете список запрещенных групп, то маршрут могут применять только пользователи, не указанные ни в одной группе этого списка. Пользователь root может применять любой маршрут.

С помощью команды **ifconfig** можно задать список групп, связанных с определенным интерфейсом. В этом случае пакет разрешается переслать по любому маршруту, разрешенному для групп, связанных с интерфейсом, который получил пакет.

Если существует несколько маршрутов к целевому хосту, то все сообщения об изменении маршрута ICMP игнорируются, а информация об MTU для данного пути не рассылается.

Функция обнаружения сбоев в работе шлюза

Хост может проверять, работает ли применяемый им шлюз, и изменять таблицу маршрутизации соответствующим образом.

Если параметр сети **-passive_dgd** равен 1, то во всей системе применяется пассивная проверка работоспособности шлюза. Если шлюзу было последовательно отправлено **dgd_packets_lost** запросов **ARP**, а ответ так и не был получен, то считается, что этот шлюз не работает, и для всех маршрутов, проходящих через этот шлюз, устанавливается максимальная метрика расстояния (также называемая *числом транзитных участков* или *ценой маршрута*). По истечении **dgd_retry_time** минут восстанавливается исходная цена маршрута. Кроме того, хост предпринимает некоторые дополнительные действия при обнаружении потери пакетов **TCP**. Если было потеряно **dgd_packets_lost** последовательно отправленных пакетов **TCP**, то запись

ARP для применявшегося шлюза удаляется, а соединение **TCP** устанавливается с использованием следующего по списку оптимального маршрута. Если при следующем обращении к шлюзу окажется, что он не работает, то будут выполнены описанные выше действия. Параметры **passive_dgd**, **dgd_packets_lost** и **dgd_retry_time** можно задать с помощью команды **no**.

Для того чтобы включить активную проверку работоспособности шлюза, выполняемую для каждого маршрута, укажите флаг **-active_dgd** в команде **route**. Система будет отправлять пакеты PING всем шлюзам, указанным в таблице маршрутизации, каждые **dgd_ping_time** секунд. Если ответ от шлюза не был получен, ему отправляются дополнительные пакеты PING до тех пор, пока не будет достигнуто значение **dgd_packets_lost**. Если ни на один из этих пакетов ответ не был получен, то цена всех маршрутов, проходящих через этот шлюз, увеличивается. Пакеты PING будут отправляться до тех пор, пока не будет получен ответ, после чего цена всех таких маршрутов восстанавливается. Параметр **dgd_ping_time** можно настроить с помощью команды **no**.

Определение работоспособности шлюза обычно применяется в случае статической, а не динамической маршрутизации. Работа этой функции в пассивном режиме не повлияет на производительность системы, поэтому ее рекомендуется применять во всех сетях с несколькими шлюзами. Однако пассивное определение работоспособности шлюза выполняется только на основе оптимальных значений маршрута. Некоторые протоколы, например, **UDP**, не отправляют уведомления о потере пакетов данных, поэтому для них пассивный режим неприменим.

Режим активного определения работоспособности шлюза применяется в тех случаях, когда хост должен узнавать о сбое шлюза немедленно. В этом случае хост опрашивает все шлюзы с периодичностью в несколько секунд, поэтому повышается нагрузка на сеть. Этот режим рекомендуется применять только на хостах, выполняющих наиболее важные задачи, а также в сетях с небольшим числом хостов.

Примечание: Режим определения работоспособности шлюза и протоколы маршрутизации, применяемые демонами **gated** и **routed**, выполняют схожие функции. Они отслеживают изменения, вносимые в конфигурацию сети, и соответствующим образом обновляют таблицу маршрутизации. Однако для реализации этих функций они используют разные механизмы, которые нельзя применять одновременно. В связи с этим режим определения работоспособности шлюза не следует включать для систем, в которых запущен демон **gated** или **routed**.

Если функция обнаружения сбоев в работе шлюза обнаружит, что основной маршрут снова доступен и параметр **dgd_flush_cached_route** включен, то текущие маршруты всех активных соединений удаляются из кэша. Маршруты всех текущих активных соединений проверяются снова с целью поиска оптимального маршрута для отправки данных. Параметр **dgd_flush_cached_route** можно настроить с помощью команды **no**. По умолчанию параметр **dgd_flush_cached_route** выключен.

Примечание: Параметр **dgd_flush_cached_route** следует включать только в стабильной сетевой среде. В противном случае возможные более серьезные неполадки производительности, связанные с неправильной или нестабильной работой аппаратных маршрутизаторов, поскольку функция обнаружения сбоев в работе шлюза будет слишком часто обновлять таблицу маршрутизации. Частая очистка кэшированных маршрутов также может использовать слишком много ресурсов.

Дублирование маршрутов

Дублирование маршрутов позволяет создавать маршрут к хосту для каждого хоста, с которым система осуществляет обмен данными.

Перед отправкой потока данных по сети выполняется поиск маршрута к целевому хосту в таблице маршрутизации. Если найден конкретные маршрут, то он применяется для отправки данных. Если конкретный маршрут к хосту не найден, то может быть найден сетевой маршрут или маршрут по умолчанию. Если у найденного маршрута задан флаг дублирования 'c', то маршрут к целевому хосту будет создан с применением шлюза дублируемого маршрута. Последующие операции поиска маршрута для этого хоста в таблицах маршрутизации возвращают дублированный маршрут. У дублированных маршрутов установлен флаг 'W'. Срок действия этих маршрутов ограничен и они удаляются из таблицы

маршрутизации, если не используются в течение времени, заданного в минутах в параметре *route_expire*. Изменить значение *route_expire* можно с помощью команды **no**.

Функция дублирования маршрутов применяется, главным образом протоколом вычисления MTU маршрута AIX, который позволяет этой системе вести учет данных MTU маршрута для каждой целевой системы. Если параметру сети **tcp_pmtu_discover** или **udp_pmtu_discover** (значение которых задается с помощью команды **no**) присвоено значение 1, то для всех сетевых маршрутов в системе устанавливается флаг дублирования. Протокол вычисления MTU маршрута по умолчанию включен.

Примечание: Для того чтобы добавить запись дублированного маршрута вручную, внесите изменения в таблицу маршрутизации с помощью команды **route**.

Информация, связанная с данной:

Команда route

Динамическое удаление маршрутов

Если настройка маршрутов выполняется с помощью демона **routed**, то маршрут, удаленный вручную, *не заменяется* на маршрут, указанный в поступившей информации RIP (так как используется ioctl).

Если настройка маршрутов выполняется с помощью демона **gated**, а флаг **-n** не указан, то маршрут, удаленный вручную, *замещается* маршрутом, который указан в поступающей информации RIP.

Настройка демона routed

С помощью данных действий можно настроить демон **routed**.

Для настройки демона **routed** выполните следующие действия:

1. В сценарии оболочки `/etc/rc.tcpip` удалите символ комментария # из строки запуска routed. В результате демон **routed** будет автоматически запускаться при каждом запуске системы.
   - Укажите режим, в котором должен работать шлюз: активный (флаг **-s**) или пассивный (флаг **-q**).
   - Укажите, должна ли выполняться трассировка пакетов (флаг **-t**). Трассировку пакетов можно включить после запуска демона **routed**, отправив демону с помощью команды **kill** сигнал **SIGUSR1**. Этот же сигнал применяется для увеличения уровня трассировки (всего предусмотрено четыре уровня). Кроме того, трассировку пакетов можно отключить после запуска демона **routed**, отправив демону с помощью команды **kill** сигнал **SIGUSR2**. Более подробная информация приведена в описании демона **routed** и команды **kill**.
   - Укажите, должна ли выполняться отладка (флаг **-d**). Вместе с этим флагом нужно указать файл протокола, в который будет заносится информация об отладке, или выбрать вывод информации на консоль.
   - Укажите, выполняет ли система, в которой запускается демон **routed**, функции шлюза (флаг **-g**).

     Примечание: Если демон **routed** запущен не в системе шлюза, то он может работать только в пассивном режиме.
2. Перечислите все известные сети в файле `/etc/networks`. Более подробные сведения приведены в разделе Networks File Format for TCP/IP книги *Справочник по файлам* . Пример файла `networks` расположен в каталоге `/usr/samples/tcpip`.
3. Укажите в файле `/etc/gateways` маршруты ко всем известным шлюзам, не подключенным непосредственно к вашей сети. Примеры записей файла `/etc/gateways` можно найти в разделе Gateways File Format for TCP/IP книги *Справочник по файлам* . Кроме того, пример файла `gateways` расположен в каталоге `/usr/samples/tcpip`.

Внимание: Не запускайте демоны **routed** и **gated** одновременно. Это может привести к непредсказуемым результатам.

Настройка демона gated

При настройке демона **gated** следует выбрать протокол маршрутизации, который лучше всего подходит для вашей системы.

Для настройки демона **gated** выполните следующие действия:

1. Выберите протокол маршрутизации, который лучше всего подходит для вашей системы. Возможны следующие протоколы маршрутизации: **EGP**, **BGP**, **RIP**, **RIPng**, **HELLO**, **OSPF**, **ICMP/Router Discovery** и **IS-IS**. Кроме того, с помощью протокола **SNMP** вы можете просмотреть или изменить управляющую информацию об объекте сети удаленного хоста.

   Примечание: Для передачи информации об адресах сетей из автономной системы шлюзам других автономных систем воспользуйтесь протоколом **EGP**, **BGP** или **BGP4+**. Если вы подключены к Internet, то **EGP**, **BGP** или **BGP4+** применяются для передачи информации о достижимости сетей базовой системы шлюза. Протоколы внутренней маршрутизации предназначены для распространения информации о достижимости сетей внутри автономных систем.
2. Перечислите все известные сети в файле `/etc/networks`. Более подробные сведения приведены в разделе Networks File Format for TCP/IP книги *Справочник по файлам* . Пример файла `networks` расположен в каталоге `/usr/samples/tcpip`.
3. Измените файл `/etc/gated.conf`, задав в нем необходимые параметры демона **gated**.

   Примечание: В AIX 4.3.2 и более поздних версиях применяется демон **gated** версии 3.5.9. Формат файла `/etc/gated.conf` изменился. Приведенные ниже примеры относятся к версии 3.5.9 демона **gated**. При настройке файла `/etc/gated.conf` в версиях AIX младше AIX 4.3.2 применяется формат, указанный в самом файле `/etc/gated.conf`.
   a. Укажите уровень подробности вывода трассировки. Если трассировку нужно включить до анализа файла `gated.conf`, укажите флаг **-t** при запуске демона. Дополнительная информация приведена в разделе gated Daemon книги *Справочник по командам, том 2*.
   b. Укажите протоколы маршрутизации, которые должны применяться в данной системе. Для каждого протокола предусмотрен собственный оператор. Удалите символ комментария (#) и измените операторы в соответствии с набором протоколов.
      - Если будет применяться **EGP**:
        – Укажите оператор **EGP** `autonomoussystem`. Узнайте номер автономной системы у лица, ответственного за подключение к Internet, а если вы не подключены к Internet, то присвойте автономной системе номер в соответствии с номерами других автономных систем вашей сети.
        – Присвойте оператору **EGP** значение `yes`.
        – Задайте предложения `group` для всех автономных систем.
        – Укажите предложения `neighbor` для всех соседей данной автономной системы. Например:

```
autonomoussystem 283 ;

egp yes {
        group maxup 1 {
             neighbor nogendefault 192.9.201.1 ;
             neighbor nogendefault 192.9.201.2 ;
       } ;
       group {
             neighbor 192.10.201.1 ;
             neighbor 192.10.201.2 ;
       } ;
} ;
```

      - Если будет применяться протокол **RIP** или **HELLO**:
        – Укажите в операторе **RIP** или **HELLO** значение `yes`.

– В операторах **RIP** или **HELLO** укажите опцию `nobroadcast`, если шлюз должен только принимать информацию о маршрутизации, но не рассылать ее. Если шлюз должен обмениваться информацией о маршрутизации с другими шлюзами, укажите в этих операторах значение `broadcast`.

– Если вы хотите, чтобы шлюз отправлял информацию непосредственно исходным шлюзам, укажите опцию `sourcegateways`. Укажите в опции `sourcegateways` имя шлюза или его IP-адрес в десятичном формате с точками. Например:

```
# Оповещение конкретных шлюзов

rip/hello yes {
          sourcegateways
                101.25.32.1
                101.25.32.2 ;
} ;
```

В следующем примере приведен раздел **RIP**/**HELLO** файла `gated.conf` для хоста, который не отправляет и не принимает пакеты **RIP** через интерфейс tr0.

```
rip/hello nobroadcast {
         interface tr0 noripin ;
} ;
```

- Если будет применяться **BGP**:
  – Добавьте раздел **BGP** `autonomoussystem`. Узнайте номер автономной системы у лица, ответственного за подключение к Internet, а если вы не подключены к Internet, то присвойте автономной системе номер в соответствии с номерами других автономных систем вашей сети.
  – Присвойте оператору **BGP** значение `yes`.
  – Задайте предложения `peer` для всех соседей данной автономной системы. Например:

```
# Выполнить все операции BGP

bgp yes {
        peer 192.9.201.1 ;
} ;
```

- Если будет применяться **SNMP**:
  – Присвойте оператору **SNMP** значение `yes`.

```
snmp yes ;
```

Настройка демона gated для выполнения IPv6:

С помощью этой процедуры можно настроить демон **gated** для выполнения **протокола Internet версии 6 (IPv6)**.

Перед настройкой демона **gated** для работы с **протоколом Internet версии 6 (IPv6)**, убедитесь, что ваша система может выполнять маршрутизацию пакетов **IPv6** и **IPv6**:

1. Запустите **autoconf6** для автоматической настройки интерфейсов протокола **IPv6**.
2. Настройте локальные адреса для каждого интерфейса **IPv6**, который будет применяться при маршрутизации **IPv6**, с помощью следующей команды:

   `ifconfig` *`интерфейс`* `inet6 fec0:`*`n`*`::`*`адрес`*`/64 alias`

   где

   интерфейс
   Имя интерфейса, например `tr0` или `en0`.

   n Десятичное число. Например, `11`.

   агента Часть интерфейса **IPv6**, стоящая после двойного двоеточия; например, для адреса **IPv6** `fe80::204:acff:fe86:298d` в поле *адрес* следует указать `204:acff:fe86:298d`.

Примечание: Адрес **IPv6**, связанный с интерфейсом, можно узнать с помощью команды **netstat -i**.

Если адаптеру Token-Ring `tr0` присвоен адрес **IPv6** `fe80::204:acff:fe86:298d`, вызовите следующую команду:

```
ifconfig tr0 inet6 fec0:13::204:acff:fe86:298d/64 alias
```

3. Включите маршрутизацию **IPv6** следующей командой:

   ```
   no -o ip6forwarding=1
   ```

4. Запустите **ndpd-router** следующей командой:

   ```
   ndpd-router -g
   ```

   Для того, чтобы узнать, какие флаги следует использовать для вашей сетевой конфигурации, см. **ndpd-router** в *Справочник по командам, том 4*.

   Запуск демона **ndpd-router** позволяет системе играть роль маршрутизатора для протокола **NDP**. Этот протокол применяется для рассылки хостам информации о маршрутизации, необходимой для передачи пакетов **IPv6**.

   Демон **ndpd-host** должен быть запущен на всех хостах в сети **IPv6**. Хосты, на которых запущен демон **ndpd-host**, образуют сеть **IPv6**, в рамках которой они могут применять протокол **NDP** для определения адресов уровня канала связи соседних компьютеров и рассылки пакетов.

   Для получения дополнительной информации см. **ndpd-router** и **ndpd-host** в *Справочник по командам, том 4* или RFC 1970, *Обнаружение соседних узлов*.

5. Затем настройте демон **gated**:

   a. Выберите протокол шлюзов **IPv6** для своей системы. Два возможных варианта - **Улучшенный протокол граничных шлюзов (BGP4+)** и **Протокол информации о маршрутизации следующего поколения (RIPng)**.

   b. Измените файл `/etc/gated.conf`, задав в нем необходимые параметры демона **gated**.

      Примечание: В AIX 4.3.2 и более поздних версиях применяется демон **gated** версии 3.5.9. Формат файла `gated.conf` несколько изменился. Для получения информации о допустимом формате обратитесь к документации `gated.conf` в в книге *Справочник по файлам*, или воспользуйтесь примером файла, который находится в каталоге `/usr/sample/tcpip`.

      При настройке **BGP4+** или **RIPng** используйте IP-адреса в формате **IPv6**.

      Примечание: По умолчанию **RIPng** применяет для рассылки пакетов групповой адрес.

После изменения файла `/etc/gated.conf` запустите демон **gated**.

Номера автономных систем

Для применения протокола **EGP** или **BGP** необходимо получить официальный *номер автономной системы* для шлюза.

Для этого отправьте соответствующий запрос в организацию NIC по адресу:

`INFO@INTERNIC.NET`

Mobile IPv6

Протокол Mobile **IPv6** обеспечивает поддержку переадресации для **IPv6**. С его помощью пользователь может применять один и тот же IP-адрес в любой точке земного шара, а приложения, работающие с этим адресом, сохраняют связь и соединения верхнего уровня, независимо от местонахождения пользователя. Поддержка переадресации осуществляется в однородных и разнородных средах.

Например, протокол Mobile **IPv6** позволяет переместить узел из сегмента Ethernet в кластер беспроводной LAN, не изменяя IP-адрес этого мобильного узла.

В протоколе Mobile **IPv6** каждый мобильный узел обозначен двумя IP-адресами: домашним адресом и текущим адресом. Домашний адрес - постоянный IP-адрес, указывающий на данный узел, независимо от его местонахождения. Текущий адрес изменяется вместе с точкой подключения и содержит информацию о текущем местонахождении мобильного узла. Подключившись к локальной сети по месту нахождения, мобильный узел должен получить текущий адрес, который применяется в течение пребывания узла в этой сети. Для получения текущего адреса могут применяться функции обнаружения соседних узлов **IPv6** (см. “Автоматическая настройка Neighbor discovery/адрес без сохранения состояния” на стр. 126). Автоматическая настройка может выполняться как с сохранением, так и без сохранения состояния. Текущий адрес может быть также задан вручную. Для протокола Mobile **IPv6** неважно, как был получен текущий адрес.

В домашней сети должен быть настроен по крайней мере один домашний агент, а мобильный узел должен знать IP-адрес своего домашнего агента. Мобильный узел отправляет домашнему агенту пакет с опцией изменения домашнего агента. Домашний агент получает этот пакет и связывает домашний адрес с полученным текущим адресом мобильного узла. В ответ домашний агент отправляет пакет с подтверждением.

Домашний агент хранит в кэше связи между домашними адресами и текущими адресами обслуживаемых им мобильных узлов. Домашний агент перехватывает пакеты, отправленные по домашнему адресу, и перенаправляет их мобильным узлам. После этого мобильный узел отправляет пакет переадресации узлу-корреспонденту со своим текущим адресом, и узел-корреспондент создает связывающую запись в кэше, чтобы направлять потоки данных непосредственно на мобильный узел по его текущему адресу.

Поддержка переадресации позволяет системе AIX выступать в следующих ролях:

В роли **Домашнего агента**:
- Хранение записей в кэше связывания для всех обслуживаемых мобильных узлов.
- Перехват пакетов, направленных мобильному узлу, для которого система является домашним агентом, по его домашнему адресу, если мобильный узел отсутствует в локальной сети.
- Перенаправление перехваченных пакетов по основному текущему адресу мобильного узла, указанному в записи в кэше связывания агента.
- Отправка подтверждения связывания в ответ на пакет изменения связывания с установленным битом подтверждения.
- Обработка случаев выявления совпадающих адресов на текущем адресе мобильного узла для обеспечения уникальности адресов **IPv6**.
- Поддержка динамического определения адреса домашних агентов мобильными узлами.
- Поддержка приема запросов на выделение мобильного префикса и отправка сведений о мобильном префиксе.

В роли **Стационарного корреспондента**:
- Обработка домашнего адреса, полученного в любом пакете **IPv6**.
- Обработка опции изменения связывания, полученной в пакете, и возврат подтверждения связывания, если в полученном пакете установлен бит подтверждения (A).
- Хранение в кэше связывания полученных в опциях изменения связывания текущих адресов.
- Отправка пакетов с применением заголовка маршрутизации при наличии записи с текущим адресом мобильного узла в кэше связывания.

В роли **Маршрутизатора** в сети, в которой временно находится мобильный узел.
- Отправка опции интервала оповещения в оповещениях маршрутизации для более простого обнаружения передвижения мобильных узлов. Эта функция настраивается параметром **-m** демона **ndpd-router**
- Поддержка отправки незапрошенного многоцелевого оповещения маршрутизатора с более высокой скоростью в соответствии с RFC 2461. Эта функция настраивается параметрами **-m** и **-D** демона **ndpd-router**

- Отправка информации о домашнем агенте (параметры и время жизни домашнего агента) в пакетах оповещения о маршрутизаторах для упрощения выбора домашнего агента мобильными узлами. Эта функция настраивается параметром **-H** демона **ndpd-router**.

Защита Mobile IPv6

Сообщения об обновлении связывания и о подтверждении связывания, передаваемые между мобильным узлом и домашним агентом, должны быть защищены средствами защиты IP с помощью ESP с непустым алгоритмом идентификации для передаваемых данных.

Дополнительная информация о защите IP-сетей приведена в разделе *Защита*.

Защита связывания мобильного узла и узла корреспондента обеспечивается с помощью процедуры обратной маршрутизации. Эта процедура шифрует с помощью ESP сообщений, передаваемых между узлом домашнего агента и мобильными узлами. Поскольку сообщения об обновлении и о подтверждении связывания, передаваемые между узлом и мобильным узлом, защищены процедурой обратной маршрутизации, то требования по защите IP для взаимодействующих корреспондентов не оговариваются. Однако если для ограничения доступа узел корреспондента применяет защиту IP, то должны быть разрешены сообщения протокола MH (135).

Туннели можно определять вручную или с помощью IKE в режиме ответа (поддерживается только ускоренный режим). Как минимум для домашнего узла с заголовком ESP будут определены следующие туннели IP:

- Туннель в режиме транспорта с протоколом MH (135) между IP-адресом домашнего агента и домашним адресом каждого мобильного узла, регистрируемого этим домашним агентом.
- Туннель в режиме туннеля с протоколом MH (135) между каждым IP-адресом и домашним адресом каждого мобильного узла, регистрируемого этим домашним агентом.

Соответствующие туннели должны быть определены на мобильных узлах.

Примечание: Сообщения об обновлении и о подтверждении связывания передаются с помощью заголовка Mobility Header (MH) переадресации и должны быть защищены с помощью ESP.

В предыдущих реализациях Mobile IPv6 в AIX поддерживалась отправка мобильными узлами сообщений об обновлении связывания с помощью пакетов опции пункта назначения. Такие сообщения можно было защитить с помощью заголовка Authentication Header (AH).

Для того чтобы домашний агент или узел корреспондента могли принимать такие сообщения, необходимо перед запуском Mobile **IPv6** отредактировать файл `/etc/rc.mobip6` и включить в нем переменную **Enable_Draft13_Mobile**. После этого в случае применения средств защиты IP для защиты сообщений об обновлении связывания необходимо будет определить вручную или с помощью IKE туннель в режиме транспорта с протоколом 60, что позволит защитить сообщения об обновлении и о подтверждении связывания.

Для того чтобы домашний агент или узел корреспондента могли принимать сообщения об обновлении связывания, не защищенные средствами защиты IP, отредактируйте файл `/etc/rc.mobip6` и отключите в нем переменную **Check_IPsec**. Однако делать это не рекомендуется, поскольку такой подход представляет собой серьезную угрозу безопасности, поскольку появляется возможность влиять на маршрутизацию пакетов, адресованных мобильному узлу.

Настройка Mobile IPv6

Здесь приведена информация о настройке Mobile **IPv6**. Для применения Mobile **IPv6** необходимо установить набор файлов `bos.net.mobip6.rte`.

Информация об установке наборов файлов приведена в разделе Установка обновлений дополнительного программного обеспечения и служб в книге *Установка и миграция*

Запуск Mobile IPv6 в качестве домашнего агента:

С помощью этой процедуры можно запустить Mobile **IPv6** в качестве домашнего агента.

1. Определите туннели IKE (этапов 1 и 2) в режиме ответа с протоколом **ESP** или вручную задайте определения связей защиты IP ESP между IP-адресом домашнего агента и домашним адресом каждого мобильного узла, с которым корреспондент может взаимодействовать.
2. Разрешите применение системы в качестве домашнего агента Mobile **IPv6** и узла. Введите в командной строке `smit enable_mobip6_home_agent`.
3. Укажите, когда необходимо включить поддержку.

Запуск Mobile IPv6 в качестве корреспондента:

С помощью этой процедуры можно запустить Mobile IPv6 в качестве корреспондента.

1. Определите туннели IKE (этапов 1 и 2) в режиме ответа с протоколом **ESP** или вручную задайте определения связей защиты IP ESP между IP-адресом домашнего агента и домашним адресом каждого мобильного узла, с которым корреспондент может взаимодействовать.
2. Разрешите применение системы в качестве узла корреспондента Mobile **IPv6**. Введите в командной строке `smit enable_mobip6_correspondent`.
3. Укажите, когда необходимо включить поддержку.

Запуск Mobile IPv6 в качестве маршрутизатора:

С помощью этой процедуры можно запустить Mobile **IPv6** в качестве маршрутизатора.

Введите следующую команду для обеспечения обнаружения передвижений:

```
ndpd-router -m
```

Завершение работы Mobile IPv6:

Используйте эту процедуру для завершения работы Mobile **IPv6**.

1. Введите в командной строке `smit disable_mobip6`.
2. Укажите, когда следует остановить Mobile **IPv6**.
3. Укажите, следует ли завершить работу демона **ndpd-router**.
4. Укажите, следует ли отключить пересылку **IPv6**.

Устранение неполадок в Mobile IPv6

Для устранения неполадок в Mobile **IPv6** используйте команду **mobip6ctrl -b**.

1. Введите следующую команду, чтобы получить сведения о состоянии связей:

   ```
   mobip6ctrl -b
   ```

2. Информация о работе с утилитами устранения неполадок **TCP/IP** приведена в разделе “Устранение неполадок TCP/IP” на стр. 424.

Виртуальный IP-адрес

Виртуальный IP-адрес устраняет зависимость хоста от отдельных сетевых интерфейсов.

Входящие пакеты отправляются по адресу VIPA системы, но передаются через физические сетевые интерфейсы.

Раньше в случае выхода интерфейса из строя все соединения с этим интерфейсом разрывались. Применение VIPA в системе и в протоколах маршрутизации позволяет автоматически изменять маршрут и выполнять восстановление после сбоя без нарушения работы существующих пользовательских соединений, в которых применяется виртуальный интерфейс, если пакеты можно передать через другой физический интерфейс. В

системах с VIPA обеспечивается более высокий коэффициент готовности, так как сбои адаптеров не влияют на активные соединения. Так как поток данных IP системы передается через несколько физических адаптеров, то общая нагрузка не концентрируется на одном адаптере и в связанной с ним подсети.

Функция VIPA AIX прозрачная для сетевого оборудования. Для ее реализации не нужно дополнительное сетевое оборудование или другое аппаратного обеспечение. Для реализации функции VIPA необходимы следующие ресурсы:

- Два или более установленных интерфейсов IP любого физического типа в разных подсистемах, соединенных с корпоративной сетью.
- Протоколы маршрутизации IP, работающие в корпоративной сети.

Настройка VIPA

Настройка VIPA, как и любого другого сетевого интерфейса IP, выполняется с помощью SMIT. Кроме того, настраивая VIPA, можно указать группу интерфейсов.

В такой конфигурации во всех исходящих соединениях, инициированных хостом VIPA через эти интерфейсы, предназначенные для применения VIPA, в качестве исходного адреса в заголовки исходящих пакетов **TCP/IP** помещается виртуальный адрес.

1. Для настройки VIPA IPv4 введите к командной строке `smit mkinetvi`. Для настройки VIPA IPv6 введите к командной строке `smit mkinetvi6`.
2. Заполните обязательные поля. Дополнительная информация приведена в разделе “Пример среды VIPA”. Нажмите Enter.

Добавление адаптера в VIPA

С помощью этой процедуры можно добавить адаптер в виртуальный IP-адрес.

Для добавления адаптера в интерфейс VIPA выполните следующие действия:

1. Введите в командной строке команду `smit chvi`.
2. Выберите интерфейс VIPA, в который необходимо добавить адаптер, и нажмите клавишу Enter.
3. Укажите добавляемый адаптер в поле **Имена интерфейсов**.
4. Укажите `добавить` в поле **добавить/удалить интерфейсы** и нажмите клавишу Enter.

Удаление адаптера из VIPA

С помощью этой процедуры можно удалить адаптер из виртуального IP-адреса.

Для удаления адаптера из интерфейса VIPA выполните следующие действия:

1. Введите в командной строке команду `smit chvi`.
2. Выберите интерфейс VIPA, из которого необходимо удалить адаптер, и нажмите клавишу Enter.
3. Укажите удаляемый адаптер в поле **Имена интерфейсов**.
4. Укажите `удалить` в поле **добавить/удалить интерфейсы** и нажмите клавишу Enter.

Пример среды VIPA

Следующий пример среды VIPA с соединениями Ethernet включает в себя систему с виртуальным IP-адресом и дв а физических соединения.

В системе настроен виртуальный IP-адрес `vi0` `10.68.6.1` и два физических соединения: `en1` с IP-адресом `10.68.1.1` и `en5` - с IP-адресом `10.68.5.1`. В данном примере оба физических соединения применяют интерфейс Ethernet, но будет поддерживаться любое сочетание интерфейсов IP, например token-ring или FDDI, пока подсети подключены к более крупной корпоративной сети и определены на маршрутизаторе этой сети.

Команда **lsattr -El vi0** возвращает следующий вывод:

```
netaddr          10.68.6.1      н/д                                        True
state            работает       Стандартный сетевой интерфейс Ethernet     True
netmask          255.255.255.0  Макс. размер пакета IP для устройства      True
netaddr6                        Макс. размер пакета IP для удал. сетей     True
alias6                          IP-адрес                                   True
prefixlen                       Текущее состояние интерфейса               True
alias4                          Инкапсуляция на уровне концевика связи     True
interface_names  en1,en5        Интерфейсы, применяющие виртуальный адрес True
```

Команда **ifconfig vi0** возвращает следующий вывод:

```
vi0: flags=84000041<UP,RUNNING,64BIT>
       inet 10.68.6.1 netmask 0xffffff00
       iflist : en1 en5
```

Команда **netstat -rn** возвращает следующий вывод:

```
Таблицы маршрутизации
Пункт назн.       Шлюз          Флаги     Ссылк.  Исп.   Инт.   PMTU Exp Groups

Дерево маршрутов для группы протоколов 2 (Internet):
default           10.68.1.2      UG        3      1055   en1     -   -
10.68.1/24        10.68.1.1      U         0      665    en1     -   -
10.68.5/24        10.68.5.1      U         0      1216   en5     -   -
127/8             127.0.0.1      U         4      236    lo0     -   -
10.68.6.1         127.0.0.1      UH        0      0      lo0     -   -
```

В исходящих пакетах без исходного адреса, передаваемых через интерфейсы `en1` и `en5`, в качестве исходного адреса будет задан виртуальный адрес (`10.68.6.1`). Входящие пакеты направляются по адресу VIPA (`10.68.6.1`), распространяемому в сети. Так как интерфейс `vi0` является виртуальным и не связан ни с одним устройством, то в таблице маршрутизации всей системы, показанной командой **netstat -rn**, соответствующая ему запись будет отсутствовать. Это означает, что при настройке этого маршрута в SMIT маршрут интерфейса не добавляется.

В случае выхода из строя или сбоя одного из физических интерфейсов, соединений или маршрутов сети, сетевые протоколы передают пакеты через другой физический интерфейс той же системы. Если удаленная система устанавливает соединение Telnet с адресом `vi0`, то пакеты, направленные по этому адресу, могут передаваться через интерфейс `en1` или `en5`. Если, например, интерфейс `en1` выйдет из строя, то пакеты могут передаваться через интерфейс `en5`. Учтите, что на оповещение маршрутов у протоколов маршрутизации может уйти некоторое время.

При использовании VIPA конечные системы и промежуточные маршрутизаторы должны иметь возможность доставить пакеты, направленные по виртуальному адресу (`vi0`), в один из физических интерфейсов (`en1` или `en5`).

VIPA и псевдонимы

VIPA принципиально схожи с псевдонимами IP, только адреса не связываются с физическими интерфейсами.

VIPA обладают несколькими преимуществами, по сравнению с псевдонимами IP:

- VIPA создает виртуальное устройство, которое можно включать и выключать независимо от физических интерфейсов.
- Адреса VIPA можно изменять, а псевдонимы - только добавлять и удалять.

Доступ по IP-адресу физических адаптеров

При реализации VIPA доступ из удаленных систем к отдельным интерфейсам сохраняется. Однако при обращении к IP-адресам физических интерфейсов не используется такое преимущество VIPA, как обмен данными, независимый от физических адаптеров. VIPA скрывает сбои физических адаптеров от внешних клиентов. Применение физических адресов восстанавливает зависимость связи от физических адаптеров.

Если удаленная система связывается с системой VIPA по адресу VIPA, или приложение в системе VIPA устанавливает соединение с другой системой, то вместо исходного адреса в заголовках IP-пакетов будет указан адрес VIPA. Но если удаленная система создает сеанс связи с помощью IP-адреса физического интерфейса, то в качестве исходного адреса в заголовках ответных пакетов будет указан физический адрес этого интерфейса. Существует одно исключение из этого правила. В исходящих пакетах приложений, связанных с определенным интерфейсом IP, указан исходный адрес интерфейса, с которым связаны эти приложения.

VIPA и протоколы маршрутизации

Для поддержки VIPA в демон gated были внесены некоторые изменения, и теперь этот демон не добавляет маршрут интерфейса и не отправляет оповещения через виртуальные интерфейсы.

Протокол **OSPF**, поддерживаемый демоном gated, оповещает о виртуальном интерфейсе соседние маршрутизаторы. Другие хосты сети могут обмениваться данными с хостом VIPA через маршрутизатор первого транзитного участка.

Несколько адресов VIPA

В системе можно настроить несколько виртуальных интерфейсов. Несколько интерфейсов VIPA может применяться, например, в том случае, если маршрутизаторы могут обеспечивать приоритет при обработке пакетов с определенным целевыми или исходными адресами VIPA.

Кроме того, несколько интерфейсов VIPA может применяться для связывания различных приложений с отдельными интерфейсами VIPA. Например, для обеспечения работы нескольких веб-серверов для нескольких компаний в одной системе можно было бы настроить следующие адреса:

- vi0 200.1.1.1 www.companyA.com
- vi1 200.1.1.2 www.companyB.com
- vi2 200.1.1.3 www.companyC.com

Канал EtherChannel и объединение линий IEEE 802.3ad

Канал EtherChannel и объединение линий IEEE 802.3ad - это технологии объединения сетевых портов, позволяющие объединить несколько адаптеров Ethernet в одно псевдоустройство Ethernet.

Например, адаптеры `ent0` и `ent1` можно объединить в устройство EtherChannel `en3`. Затем интерфейсу `en3` можно назначить IP-адрес. С точки зрения системы объединенные адаптеры представляют собой один адаптер. Протокол IP настраивается для этих адаптеров как и для любого другого адаптера Ethernet. Кроме того, всем адаптерам, включенным в канал EtherChannel или объединение линий, присваивается одинаковый аппаратный адрес (MAC), поэтому удаленные системы также рассматривают их как один адаптер. Объединение линий EtherChannel и IEEE 802.3ad требует поддержки коммутатором, чтобы было понятно, какие порты следует считать одним.

Примечание: Адаптер EtherChannel присваивает недопустимый адрес MAC-адрес `02:00:00:00:00:00` порту адаптера Ethernet хоста (HEA) неактивного канала в конфигурации EtherChannel. Недопустимый MAC-адрес присваивается при создании EtherChannel или при добавлении портов HEA в неактивный канал во время выполнения. В процессе переключения или восстановления EtherChannel недопустимый MAC-адрес заменяется допустимым MAC-адресом, а во время выполнения допустимый MAC-адрес заменяется недопустимым MAC-адресом.

Основное достоинство канала EtherChannel и объединения линий IEEE 802.3ad заключается в том, что они позволяют получить логическое сетевое устройство, обладающее суммарной пропускной способностью всех его адаптеров. В случае сбоя какого-либо адаптера данные автоматически перенаправляются на следующий доступный адаптер без прерывания уже установленных сеансов связи. После исправления неполадки адаптера он автоматически возобновляет работу в составе канала EtherChannel или объединения линий.

Между каналом EtherChannel и объединением линий IEEE 802.3ad существует ряд различий. С учетом различий, перечисленных в Табл. 79 на стр. 378, выберите оптимальную технологию.

Таблица 79. Отличия канала EtherChannel от объединения линий IEEE 802.3ad

EtherChannel	IEEE 802.3ad, объединение линий
Требуется настроить коммутатор.	Для обмена данными Link Aggregation Control Protocol Data Unit (LACPDU) требуется настроить коммутатор.
Контрольные сигналы не передаются между портом коммутатора и смежным системным портом.	Контрольные сигналы (LACPDU) передаются с частотой, указанной в стандарте IEEE 802.3ad. Контрольные сигналы обеспечивают дополнительную защиту в случае сбоя.

В операционной системе AIX поддерживается функция динамической настройки адаптеров. Эта функция позволяет добавлять адаптеры в EtherChannel и удалять их, не прерывая установленные пользователями соединения.

Понятия, связанные с данным:

“Динамическая настройка адаптеров” на стр. 388
В версиях, предшествовавших AIX 5L версии 5.2 с рекомендуемым пакетом обслуживания 5200-03, для добавления адаптера в EtherChannel или для его удаления сначала нужно было отключить интерфейс, прервав тем самым соединения всех пользователей. Для устранения этого ограничения в AIX 5L версии 5.2 с рекомендуемым пакетом обслуживания 5200-03 появилась функция динамической настройки адаптеров (DAM).

“EtherChannel”
Адаптеры, относящиеся к каналу EtherChannel, должны быть подключены к одному коммутатору, поддерживающему EtherChannel. Если адаптеры подключены к другим коммутаторам, то эти коммутаторы необходимо добавить в стек, объединив тем самым в один коммутатор.

“Настройка объединения линий IEEE 802.3ad” на стр. 391
IEEE 802.3ad - это стандартный способ объединения нескольких линий связи в одну. Принципиально он ничем не отличается от технологии EtherChannel, то есть несколько адаптеров Ethernet объединяются в один виртуальный адаптер, обеспечивающий более высокую пропускную способность и более надежную защиту от неполадок.

“Сценарии стыкуемости” на стр. 395
Ознакомьтесь с этими сценариями перед настройкой канала EtherChannel или объединения линий IEEE 802.3ad.

EtherChannel

Адаптеры, относящиеся к каналу EtherChannel, должны быть подключены к одному коммутатору, поддерживающему EtherChannel. Если адаптеры подключены к другим коммутаторам, то эти коммутаторы необходимо добавить в стек, объединив тем самым в один коммутатор.

Этот коммутатор необходимо вручную настроить таким образом, чтобы он обслуживал порты, относящиеся к EtherChannel, как объединенный канал. В документации по коммутатору эта функция может называться *link aggregation* (объединение линий) или *trunking*.

Для правильной работы канала EtherChannel, необходимо, чтобы механизм опроса, периодически проверяющий состояние соединения, был включен на всех адаптерах перед созданием канала EtherChannel. Поток данных распределяется между адаптерами стандартным способом (адаптер для передачи пакетов выбирается в соответствии с заданным алгоритмом) или карусельным методом (пакеты равномерно распределяются между адаптерами). Входящий поток данных распределяется в соответствии со стратегией, заданной в конфигурации коммутатора. Его распределение не зависит от режима работы EtherChannel.

В отдельной системе можно настроить несколько каналов EtherChannel. Если все соединения в одном канале EtherChanel подключены к одному коммутатору, то в случае сбоя или отключения этого коммутатора теряется весь канал EtherChannel. Для предотвращения такой неполадки предусмотрен резервный адаптер, позволяющий продолжить обслуживание в случае сбоя основного канала EtherChannel. Резервный адаптер и адаптеры EtherChannel должны быть подключены к разным сетевым коммутаторам, которые должны быть связаны между собой для правильной работы данной конфигурации. В случае выхода из строя всех

адаптеров канала EtherChannel для отправки и приема данных будет применяться резервный адаптер. При восстановлении, по крайней мере, одного из адаптеров EtherChannel управление возвращается каналу EtherChannel.

Например, для создания канала EtherChannel `en3` адаптеры `ent0` и `ent1` можно настроить в качестве основных адаптеров EtherChannel, а `ent2` - в качестве резервного адаптера. В идеальном случае адаптеры `ent0` и `ent1` должны быть подключены к одному коммутатору с поддержкой EtherChannel, а `ent2` должен быть подключен к другому коммутатору. В данном случае все данные, передаваемые через `en3` (интерфейс EtherChannel), будут по умолчанию отправляться через `ent0` или `ent1` (в зависимости от схемы распределения пакетов EtherChannel), в то время как `ent2` будет простаивать. Если в какой-либо момент времени адаптеры `ent0` и `ent1` выйдут из строя, то все данные будут передаваться через резервный адаптер `ent2`. При восстановлении `ent0` или `ent1` они вновь будут использоваться для передачи данных.

Канал EtherChannel может работать в режиме резервного сетевого интерфейса, который обеспечивает защиту от единичных неполадок сети Ethernet. Для настройки режима резервного сетевого интерфейса дополнительное аппаратное обеспечение не требуется, однако для обеспечения максимальной надежности резервный адаптер должен быть подключен к отдельному коммутатору. В этом режиме данные передаются только через один адаптер. EtherChannel проверяет активный адаптер и, при необходимости, маршрут к указанному пользователем узлу. При обнаружении сбоя для передачи данных начинает использоваться следующий адаптер. Режим резервного сетевого интерфейса обеспечивает обнаружение сбоя и возобновление работы без отключения пользовательских соединений. Резервный сетевой интерфейс был вначале реализован как один из режимов в меню SMIT EtherChannel. Аналогичную функцию выполняет резервный адаптер, поэтому данный режим был удален из меню SMIT. Информация о настройке резервного сетевого интерфейса приведена в разделе “Настройка резервного сетевого интерфейса” на стр. 384.

Замечания по настройке канал EtherChannel

Обратитесь к этому списку перед настройкой EtherChannel.

- Канал EtherChannel допускает настройку до восьми основных адаптеров Ethernet и до восьми резервных адаптеров.
- В системе можно настроить несколько каналов EtherChannel, но каждый из них представляет отдельный сетевой интерфейс Ethernet. Может понадобиться увеличение значения опции **ifsize** команды **no**, чтобы помимо интерфейсов Ethernet адаптеров оно учитывало и настроенные каналы EtherChannel. В системах AIX 5.2 и более ранних версий значение параметра **ifsize** по умолчанию равно восьми. Размер по умолчанию - 256.
- В состав EtherChannel могут входить любые поддерживаемые адаптеры Ethernet (см. “Поддерживаемые адаптеры” на стр. 396). Однако адаптеры Ethernet должны быть подключены к коммутатору, поддерживающему EtherChannel. Информация о поддержке EtherChannel приведена в документации по коммутатору (в документации по эта функция может называться объединением каналов ("link aggregation" или "trunking").
- Для всех адаптеров канала EtherChannel должна быть установлена одинаковая скорость передачи данных (например, 100 Мбит/с) и дуплексный режим передачи.
- Адаптеры, выделенные каналу EtherChannel, станут недоступны системе после настройки EtherChannel. Изменения любых атрибутов, таких как скорость передачи данных, размеры очередей приема и передачи и т.д., необходимо вносить до включения адаптеров в состав канала EtherChannel.
- Перед выполнения этой процедуры нужно убедиться, что с адаптерами, которые планируется объединить в канал EtherChannel, не связаны IP-адреса. При настройке канала EtherChannel с адаптерами, для которых ранее уже были заданы IP-адреса, убедитесь, что соответствующие им интерфейсы находятся в `отключенном` состоянии. В базе данных ODM не должно быть интерфейсов, находящихся в состоянии `up` и связанных с адаптерами, которые планируется объединить в канал EtherChannel (это возможно, если адаптерам были назначены IP-адреса с помощью SMIT). Это может привести к возникновению неполадок при включении EtherChannel во время перезагрузки компьютера, так как такие интерфейсы будут настроены до того, как канал EtherChannel и информация о нем будут найдены в базе данных ODM. В результате при настройке EtherChannel будет обнаружено, что некоторые из необходимых адаптеров уже заняты. Для предотвращения подобной неполадки перед созданием EtherChannel введите `smitty chinet`,

выберите интерфейсы тех адаптеров, которые планируется объединить в EtherChannel, и измените их **состояние** на значение `отключен`. Это гарантирует отсутствие ошибок при настройке канала EtherChannel во время перезагрузки компьютера.

Дополнительные сведения о ODM приведены в разделе Администратор объектных данных (ODM) книги *Программирование: Разработка и отладка программ*.

- Если для создания EtherChannel планируется применять адаптеры 10/100 Ethernet в версиях AIX до AIX 5L версии 5.2 с рекомендуемым пакетом обслуживания 5200-03, то перед добавлением этих адаптеров в конфигурацию EtherChannel для них может потребоваться включить функцию опроса линии связи. Для этого введите команду `smitty chgenet`. Укажите в поле **Включить опрос линии связи** значение `да` и нажмите Enter.

  Примечание: В AIX 5L версии 5.2 с рекомендуемым пакетом обслуживания 5200-03 и в более поздних версиях явно включать опрос линии не нужно. Он включается автоматически.

- Если вы планируете применять большие кадры, то может потребоваться включить соответствующую функцию для всех адаптеров перед созданием EtherChannel, а также в конфигурации самого EtherChannel. Для этого введите команду `smitty chgenet`. Укажите в поле **Включить поддержку больших кадров** значение `да` и нажмите Enter. Повторите эту операцию для каждого адаптера, для которого нужно включить поддержку больших кадров. Позднее вы сможете включить поддержку больших кадров и для канала EtherChannel.

  Примечание: Включать поддержку больших кадров на всех адаптерах не нужно - достаточно включить ее только для самого канала EtherChannel. Если вы присвоите атрибуту **Включить поддержку больших кадров** значение `да`, то все необходимые операции будут выполнены автоматически.

- AIX 5.3 и AIX 6.1 поддерживают следующие конфигурации Host Ethernet Adapter (HEA).
  - Поддерживается объединение каналов между выделенным портом HEA и адаптером PCI/PCI-E для ручного объединения и объединения LACP.
  - Конфигурация EtherChannel с поддержкой невыделенного порта HEA, EtherChannel с резервным адаптером, настроенным в качестве PCI/PCI-E, или виртуальный Ethernet.

    Примечание: Для невыделенного порта HEA в конфигурации EtherChannel действуют ограничения объединения каналов.

- AIX версии 6.1 с технологическим пакетом обслуживания 6100-06 и выше поддерживают EtherChannel на стековых коммутаторах.
- Сетевая загрузка или сетевая установка с помощью EtherChannel в клиенте Управление сетевой установкой (NIM) не поддерживается.

Настройка EtherChannel

Выполните эту процедуру для настройки EtherChannel.

1. Введите в командной строке `smitty etherchannel`.
2. Выберите в списке пункт **Добавить канал EtherChannel/объединение линий** и нажмите Enter.
3. Выберите основные адаптеры Ethernet для канала EtherChannel и нажмите клавишу Enter. Если будет применяться резервный адаптер EtherChannel, то не выбирайте этот адаптер на данном этапе.

   Примечание: Окно **Доступные сетевые адаптеры** содержит список всех адаптеров Ethernet. Если выбрать уже используемый адаптер Ethernet (для которого определен интерфейс), то будет выведено сообщение об ошибке. Для применения такого адаптера вначале нужно отключить его интерфейс.
4. Заполните поля формы, следуя приведенным ниже указаниям:
   - **Родительский адаптер:** Это поле содержит сведения о родительском устройстве EtherChannel (например, когда канал EtherChannel принадлежит общему адаптеру Ethernet). Это поле содержит значение `HET` если канал EtherChannel не содержится в другом адаптере (по умолчанию). Если канал EtherChannel содержится в другом адаптере, в поле отображается имя родительского адаптера (например, `ent6`). Это поле носит информационный характер и не может быть изменено. Родительский адаптер можно задать только в AIX 5.3и более поздних версиях.

- **Адаптеры канала EtherChannel/объединения линий:** В списке должны быть указаны все основные адаптеры канала EtherChannel. Эти адаптеры были выбраны на предыдущем шаге.
- **Задать альтернативный адрес:** Это необязательное поле. Если задать в этом поле значение `да`, то можно будет указать адрес MAC для данного канала EtherChannel. Если задать в этом поле значение `нет`, то канал EtherChannel будет применять адрес MAC первого адаптера.
- **Альтернативный адрес:** Если в поле **Задать альтернативный адрес** было указано значение `да`, введите в этом поле необходимый адрес MAC. Указанный 12-значный шестнадцатеричный адрес должен начинаться с символов `0x` (например, 0x001122334455).
- **Включить большие кадры Gigabit Ethernet:** Это поле заполнять необязательно. Для применения этой функции коммутатор должен поддерживать большие кадры. Эта функция поддерживается только интерфейсами Standard Ethernet (en), но не интерфейсами IEEE 802.3 (et). Для включения функции укажите значение `да`.
- **Режим:** Возможны следующие значения:
  - **standart**: В этом режиме канал EtherChannel выбирает адаптер для передачи пакетов в соответствии с заданным алгоритмом. Алгоритм включает в себя взятие значения данных, его деление на число адаптеров в канале EtherChannel, и выбор исходящей линии по полученному остатку. Значение режима хэширования указывает, какое именно значение данных является исходным для этого алгоритма (см. описание атрибута Режим хэширования). Например, если выбран режим хэширования `standard`, то применяется целевой IP-адрес пакета. Если адрес равен 10.10.10.11 и канал EtherChannel состоит из 2 адаптеров, (1 / 2) = 0 с остатком 1, т.е. будет применяться второй адаптер (нумерация адаптеров начинается с 0). Адаптеры нумеруются в том порядке, в котором они указаны в меню SMIT. Этот режим применяется по умолчанию.
  - **round_robin**: В этом режиме пакеты, передаваемые по EtherChannel, циклически распределяются между адаптерами. Последовательность отправки пакетов может отличаться от той, в которой они были переданы в канал EtherChannel, но этот способ обеспечивает эффективное использование всей пропускной способности. Если режим хэширования отличен от `default`, то выбирать данный режим нельзя. Если вы выбрали карусельный режим, то не изменяйте режим хэширования, равный `default`.
  - **netif_backup**: Для применения режима резервного сетевого интерфейса можно настроить несколько адаптеров в основном и резервном EtherChannel. Дополнительная информация приведена в разделе “Настройка резервного сетевого интерфейса” на стр. 384.
  - **8023ad**: Эта опция позволяет использовать для автоматического объединения каналов протокол управления объединением каналов IEEE 802.3ad (LACP). Дополнительная информация приведена в разделе “Настройка объединения линий IEEE 802.3ad” на стр. 391.
- **Интервал IEEE 802.3ad**: Можно выбрать одно из следующих значений:
  - **long**: Интервал по умолчанию. EtherChannel запрашивает пакеты LACP с длинным интервалом, указанным протоколом.
  - **short**: EtherChannel запрашивает пакеты LACP с коротким интервалом, указанным протоколом.

    Примечание: Значение интервала применяется только в том случае, если EtherChannel работает в режиме IEEE 802.3ad. В противном случае это значение игнорируется.

    Примечание: AIX поддерживает как длинные, так и короткие интервалы запросов.
- **Режим хэширования**: Вы можете выбрать один из следующих режимов, определяющих исходные данные, применяемые алгоритмом для выбора адаптера:
  - **default**: В этом режиме для определения применяемого адаптера используется целевой IP-адрес пакета. При обработке пакетов других протоколов, например, ARP, для вычислений применяется последний байт целевого адреса MAC. В этом режиме при передаче пакетов по каналу EtherChannel сохраняется очередность их отправки, но не всегда обеспечивается эффективное использование пропускной способности канала.

– **src_port**: В этом режиме для определения применяемого адаптера используется номер исходного порта UDP или TCP пакета. Если пакет не относится к UDP или TCP, то применяется последний байт целевого IP-адреса. Если пакет не относится к IP, то применяется последний байт целевого MAC-адреса.
– **dst_port**: В этом режиме для определения применяемого адаптера используется номер целевого порта UDP или TCP пакета. Если пакет не относится к UDP или TCP, то применяется последний байт целевого IP-адреса. Если пакет не относится к IP, то применяется последний байт целевого MAC-адреса.
– **src_dst_port**: В этом режиме для определения применяемого адаптера используются номера исходного и целевого порта UDP или TCP пакета (номера исходного и целевого портов складываются и делятся пополам, а полученное значение передается алгоритму выбора). Если пакет не относится к UDP или TCP, то применяется последний байт целевого IP-адреса. Если пакет не относится к IP, то применяется последний байт целевого MAC-адреса. Этот режим обеспечивает хорошее распределение пакетов в большинстве случаев, как для клиентов, так и для серверов.

Примечание: Сочетание режима хэширования, отличного от `default`, с режимом `round_robin` недопустимо.

Дополнительная информация о распределении пакетов и о выравнивании нагрузки приведена в разделе “Опции выравнивания нагрузки EtherChannel” на стр. 386.

- **Резервный адаптер:** Это поле заполнять необязательно. Укажите резервные адаптеры для канала EtherChannel.
- **Опрашиваемый IP-адрес:** Это необязательное поле. Оно действует только в том случае, если применяется режим **Резервный сетевой интерфейс** или канал EtherChannel содержит один или несколько адаптеров, а также один или несколько резервных адаптеров. Канал EtherChannel будет проверять связь с указанным IP-адресом или именем хоста. Если канал EtherChannel не получит от этого IP-адреса ответ на запросы, число которых указано в поле **Количество повторов**, с интервалом, заданным в поле **Тайм-аут повтора**, то EtherChannel переключается на другие резервные адаптеры.
- **Количество повторов:** Укажите количество неполученных ответов при проверки связи, при достижении которого канал EtherChannel сменит активный адаптер. Значение по умолчанию - 3. Это поле необязательно для заполнения и применяется только в том случае, если задан **Опрашиваемый IP-адрес.**
- **Тайм-аут повтора:** Укажите (в секундах) интервалы отправки пакетов по **Опрашиваемому IP-адресу**. Значение по умолчанию - одна секунда. Это поле необязательно для заполнения и применяется только в том случае, если задан **Опрашиваемый IP-адрес.**

5. Заполнив необходимые поля, нажмите клавишу Enter, чтобы создать канал EtherChannel.
6. Введите команду `smitty chinet`, чтобы настроить для созданного устройства EtherChannel протокол IP.
7. Выберите из списка созданный интерфейс EtherChannel.
8. Укажите в полях формы всю необходимую информацию и нажмите клавишу Enter.

Дополнительные задачи, которые можно выполнить после настройки EtherChannel, описаны в разделе “Просмотр списка каналов EtherChannel или объединений линий” на стр. 388.

Восстановление после сбоя и связанные с ним опции

Для EtherChannel или объединения линий IEEE 802.3ad доступны функции восстановления и передачи управления.

С их помощью можно внести следующие улучшения:

- предотвратить потерю пакетов при восстановлении
- настроить мгновенный запуск передачи управления
- отключить автоматическое восстановление чтобы резервный адаптер оставался функционирующим
- принудить объединение линий передавать управление от основного канала резервному или наоборот

Восстановление без потерь:

Функция восстановления без потерь обеспечивает восстановление основного канала из резервного адаптера с минимальной потерей пакетов.

Перед восстановлением без потерь EtherChannel или IEEE 802.3ad выполнит восстановление основного канала при обнаружении восстановления одного из основных адаптеров. В некоторых случаях коммутатор адаптера будет находиться в состоянии, отличном от необходимого для приема и передачи данных и некоторые пакеты будут потеряны сразу после выполнения восстановления.

При использовании восстановления без потерь адаптер EtherChannel или IEEE 802.3ad будет восстановлен до основного канала только когда он будет способен фактически принимать данные. Это гарантирует полную инициализацию порта коммутатора и отсутствие потерянных пакетов.

Передача управления без потерь:

Функция передачи управления без потерь изменяет поведение функции восстановления без потерь.

Когда сбой проверки связи вызывает передачу управления, восстановление без потерь используется по умолчанию. Оно подразумевает определенный период ожидания, позволяющий коммутатору неактивного адаптера завершить прием данных прежде чем завершиться передача управления. Если атрибут **noloss_failover** имеет значение `no`, передача управления из-за сбоя проверки связи выполняется немедленно.

Автоматическое восстановление:

После передачи управления от основного канала резервному адаптеру объединение линий EtherChannel и IEEE 802.3ad автоматически запускает восстановление основного канала при восстановлении хотя бы одного из его адаптеров.

Этот вариант восстановления не поддерживается в режиме IEEE 802.3ad и переключение на резервный адаптер выполняется в результате сбоя протокола Link Aggregation Control Protocol (LACP). Сбой LACP возникает, если все адаптеры в основном канале не получают блоки данных LACP (LACPDU) в течение тайм-аута. Тайм-аут определяется стандартом IEEE, который основан на интервале, настроенном для узла IEEE 802.3ad.

Это поведение можно изменить задав для атрибута **auto_recovery** значение `no`. При этом объединение линий EtherChannel или IEEE 802.3ad по-прежнему будет выполняться в резервном адаптере после передачи управления. Выполнение операций в резервном адаптере продолжиться до тех пор, пока не произойдет одно из следующих событий.

- Выполняется принудительная передача управления.
- Возникает ошибка резервного адаптера.
- Обнаруживается сбой проверки резервного адаптера.

Принудительная передача управления:

Для EtherChannel или объединения линий IEEE 802.3ad можно задать принудительную передачу управления от основного канала резервному адаптеру, или от резервного адаптера основному каналу.

Принудительная передача управления работает только если задан резервный адаптер и если неактивный канал настроен и функционирует. Например, для принудительной передачи управления от основного канала резервному адаптеру резервный адаптер должен функционировать.

Для применения этой функции введите `smitty etherchannel` и выберите опцию **Принудительная передача управления в EtherChannel / объединении линий**. Затем выберите канал EtherChannel объединение линий IEEE 802.3ad, для которого следует использовать принудительную передачу управления.

Настройка резервного сетевого интерфейса

Резервный сетевой интерфейс обеспечивает защиту от одиночного сбоя сети с помощью функций обнаружения неполадки и возобновления работы без вмешательства в работу пользовательских соединений. В этом режиме одновременно применяется только один адаптер канала.

В случае выхода этого адаптера из строя для передачи всего потока данных применяется следующий адаптер канала EtherChannel. При работе в режиме резервного сетевого интерфейса не обязательно подключать адаптеры к коммутатору, поддерживающему EtherChannel.

Конфигурация резервного сетевого интерфейса действует наиболее эффективно, если адаптеры канала подключены к разным сетевым коммутаторам, в результате чего обеспечивается более надежная защита от сбоев, чем при подключении всех адаптеров к одному коммутатору. При подключении адаптеров к разным коммутаторам необходимо, чтобы эти коммутаторы были соединены. Это позволяет переносить поток данных с одного адаптера на другой, так как всегда существует маршрут к применяемому в данный момент адаптеру.

Адаптер, настроенный в основном канале EtherChannel, обладает большим приоритетом по сравнению с резервным адаптером. Основной адаптер применяется вплоть до выхода из строя. В этом в предыдущих выпусках заключается отличие резервного адаптера от режима резервного сетевого интерфейса, в котором резервный адаптер также применялся до выхода из строя, независимо от того, была ли устранена неполадка на основном адаптере.

Например, для создания канала EtherChannel `ent3` можно настроить основной адаптер `ent0` и резервный `ent2`. В идеальном случае адаптеры `ent0` и `ent2` должны быть подключены к разным коммутаторам. В данном случае все данные, передаваемые через `ent3` (интерфейс EtherChannel), будут по умолчанию отправляться через `ent0`, в то время как `ent2` будет простаивать. Если в какой-либо момент времени адаптер `ent0` выйдет из строя, то все данные будут передаваться через резервный адаптер `ent2`. После восстановления `ent0` он вновь будет использоваться для передачи данных.

Также возможно настроить EtherChannel для обнаружения сбоев линии связи и отсутствия доступа к сети для нескольких каналов EtherChannel с резервным адаптером. Для этого необходимо указать с помощью атрибута **netaddr** IP-адрес или имя хоста удаленной системы, с которой должна непрерывно поддерживаться связь. Канал EtherChannel периодически опрашивает эту систему, чтобы определить наличие сетевого маршрута к ней. В случае отсутствия ответов на указанное число пакетов канал EtherChannel переводит поток данных на другой резервный, пытаясь найти маршрут к удаленной системе через этот адаптер. В этой конфигурации не только все адаптеры должны быть подключены к разным коммутаторам, но и все коммутаторы должны быть соединены с целевой системой разными маршрутами.

Для одного или нескольких каналов EtherChannels с резервным адаптером доступна функция проверки связи. Однако, если переключение было вызвано неудачной проверкой ping с помощью основного адаптера, то резервный адаптер будет оставаться активным на протяжении неограниченного времени. Во время работы резервного адаптера невозможно определить, можно ли достичь требуемого хоста с помощью основного адаптера. Для того чтобы избежать непрерывного переключения между основным и резервным адаптером система просто продолжает использовать для работы резервный адаптер (до тех пор, пока на отправляемые с него запросы ping не перестанут приходить ответы или пока резервный адаптер сам не выйдет из стройка; в этом случае система переключится на основной адаптер). Однако, если поток данных был переключен на другой адаптер в связи с выходом из строя основного адаптера (а не из-за отсутствия ответов на запросы ping), то канал EtherChannel снова начнет использовать основной адаптер сразу после его восстановления.

Инструкции по настройке резервного сетевого интерфейса в более поздних версиях приведены в разделе “Настройка резервного сетевого интерфейса”.

Настройка резервного сетевого интерфейса:

Ниже описана процедура настройки резервного сетевого интерфейса в более новых версиях.

1. Войдя в систему как пользователь root, введите команду `smitty etherchannel`.

2. Выберите в списке пункт **Добавить канал EtherChannel/объединение линий** и нажмите Enter.
3. Выберите основной адаптер Ethernet и нажмите клавишу Enter. Этот адаптер будет применяться, пока не выйдет из строя.

   Примечание: В поле **Доступные сетевые адаптеры** отображаются все адаптеры Ethernet. Если выбрать уже используемый адаптер Ethernet, то будет выведено сообщение об ошибке. В этом случае необходимо отключить интерфейс адаптера. Сведения об отключении интерфейса приведены в “Внесение изменений в EtherChannel в 5200-01 и более ранних версиях” на стр. 390.
4. Заполните поля формы, следуя приведенным ниже указаниям:
   - **Родительский адаптер:** Это поле содержит сведения о родительском устройстве EtherChannel (например, когда канал EtherChannel принадлежит общему адаптеру Ethernet). Это поле содержит значение `НЕТ` если канал EtherChannel не содержится в другом адаптере (по умолчанию). Если канал EtherChannel содержится в другом адаптере, в поле отображается имя родительского адаптера (например, `ent6`). Это поле носит информационный характер и не может быть изменено. Резервный адаптер можно задать в операционной системе AIX.
   - **Адаптеры канала EtherChannel/объединения линий**: В этом поле должен быть указан основной адаптер, выбранный на предыдущем шаге.
   - **Задать альтернативный адрес**: Это необязательное поле. Если задать в этом поле значение `да`, то можно будет указать адрес MAC для данного канала EtherChannel. Если задать в этом поле значение `нет`, то канал EtherChannel будет применять адрес MAC основного адаптера.
   - **Альтернативный адрес**: Если в поле **Задать альтернативный адрес** было указано значение `да`, то введите в этом поле необходимый адрес MAC. Указанный 12-значный шестнадцатеричный адрес должен начинаться с символов `0x` (например, 0x001122334455).
   - **Включить большие кадры Gigabit Ethernet**: Это поле заполнять необязательно. Для применения этой функции коммутатор должен поддерживать большие кадры. Эта функция поддерживается только интерфейсами Standard Ethernet (en), но не интерфейсами IEEE 802.3 (et). Для включения функции укажите значение `да`.
   - **Режим**: Неважно, какой режим работы будет выбран, так как в канале EtherChannel есть только один основной адаптер. Все пакеты будут отправляться через этот адаптер, пока он не выйдет из строя. Режим `netif_backup` отсутствует, так как его функции можно заменить резервным адаптером.
   - **Режим хэширования**: Неважно, какой режим работы будет выбран, поскольку в канале EtherChannel есть только один основной адаптер. Все пакеты будут отправляться через этот адаптер, пока он не выйдет из строя.
   - **Резервный адаптер**: укажите один или несколько адаптеров, которые требуется включить в резервную группу EtherChannel. После переключения вследствие потери основной группы EtherChannel резервные адаптеры используются до восстановления основной группы EtherChannel.
   - **Опрашиваемый IP-адрес**: Это поле необязательно для заполнения. Канал EtherChannel проверяет связь с указанным в этом поле IP-адресом или именем хоста. Если канал EtherChannel не получит от этого IP-адреса ответ на запросы, число которых указано в поле **Количество повторов**, с интервалом, заданным в поле **Тайм-аут повтора**, то активный адаптер заменяется резервным.
   - **Количество повторов**: Укажите количество неполученных ответов при проверке связи, по достижении которого канал EtherChannel сменит активный адаптер. Значение по умолчанию - 3. Это поле необязательно для заполнения и применяется только в том случае, если задан **Опрашиваемый IP-адрес.**
   - **Тайм-аут повтора**: Укажите интервалы отправки пакетов по **Опрашиваемому IP-адресу** (в секундах). Значение по умолчанию - одна секунда. Это поле необязательно для заполнения и применяется только в том случае, если задан **Опрашиваемый IP-адрес.**
5. Заполнив необходимые поля, нажмите клавишу Enter, чтобы создать канал EtherChannel.
6. Введите команду `smitty chinet`, чтобы настроить для созданного интерфейса протокол IP.
7. Выберите из списка созданный интерфейс EtherChannel.
8. Укажите в полях формы всю необходимую информацию и нажмите клавишу Enter.

Теперь резервный сетевой интерфейс настроен.

Опции выравнивания нагрузки EtherChannel

В EtherChannel существует два способа выравнивания нагрузки, связанной с передачей исходящих данных: карусельный способ, равномерно распределяющий исходящие данные между всеми адаптерами EtherChannel, а также стандартный способ, выбирающий адаптер в соответствии с заданным алгоритмом.

Исходное числовое значение, передаваемое алгоритму выбора адаптера, определяется режимом хэширования.

В следующей таблице перечислены допустимые сочетания опций выравнивания нагрузки.

Таблица 80. Распределение нагрузки для различных сочетаний режима и режима хэширования.

Режим	Режим хэширования	Распределение исходящего потока данных
standard или 8023ad	default	Обычная работа AIX. Алгоритм выбора адаптера использует последний байт целевого IP-адреса (для данных **TCP/IP**) или MAC-адреса (для ARP и других данных, не относящихся к IP). Этот режим обычно является хорошей начальной точкой для серверов с большим числом клиентов.
standard или 8023ad	src_dst_port	Адаптер для передачи данных выбирается алгоритмом в зависимости от исходного и целевого номера порта TCP или UDP. Поскольку каждое соединение использует уникальный порт TCP или UDP, то три режима распределения пакетов на основании номеров портов обеспечивают более гибкий подход в ситуации, когда есть несколько разных соединений TCP или UDP с одним и тем же IP-адресом.
standard или 8023ad	src_port	Алгоритм выбора адаптера использует номер исходного порта TCP или UDP. В выводе команды **netstat -an** номер порта указывается после адреса TCP/IP в столбце `Локальный`.
standard или 8023ad	dst_port	Алгоритм выбора адаптера использует номер целевого порта. В выводе команды **netstat -an** номер порта TCP или UDP указывается после адреса TCP/IP в столбце `Удаленный`.
round-robin	default	Исходящие данные равномерно распределяются между всеми адаптерами EtherChannel. Этот режим обычно используется при непосредственном соединении двух хостов (без промежуточного коммутатора).

Карусельный метод распределения:

Исходящие данные равномерно распределяются между всеми адаптерами EtherChannel. Такой подход обеспечивает наивысший уровень оптимизации пропускной способности для серверов AIX. Несмотря на то, что карусельное распределение идеально для равномерного использования всех каналов связи, следует помнить, что его применение может привести к получению целевой системой пакетов в неправильном порядке.

В целом карусельный режим идеален для непосредственного соединения больших серверов с применением больших кадров. В этом случае отсутствуют промежуточные коммутаторы, а значит выполняемая коммутатором обработка не может привести к изменению времени доставки пакетов, их порядка или к изменению адаптера. При непосредственном соединении систем кабелем пакеты принимаются в том порядке, в котором они были отправлены. Поддержка больших кадров (MTU 9000 байт) всегда обеспечивает

более высокую скорость передачи файлов, чем стандартное значение MTU, равное 1500 байт. Однако в этом случае существует и еще одно преимущество. Отправка больших пакетов занимает больше времени, что снижает вероятность прерывания получающего хоста большим числом внеочередных пакетов.

Карусельный режим может быть реализован и в других конфигурациях, однако при этом возрастает риск получения внеочередных пакетов принимающей системой. Этот риск особенно высок при наличии нескольких продолжительных соединений TCP, работающих в потоковом режиме. При наличии большого числа таких соединений между двумя хостами пакеты различных соединений могут приходить вперемешку. Проверьте статистические данные по внеочередным пакетам, приведенные в разделе `tcp` вывода команды **netstat -s**. Возрастающее значение указывает на наличие возможных неполадок в потоке данных, передаваемых EtherChannel.

Если в системе, использующей стандартное значение MTU Ethernet и подключенной через коммутатор возникает большое число внеочередных пакетов, то попробуйте различные режимы хэширования, поддерживаемые в стандартном режиме работы. Каждый режим имеет свои преимущества, но чаше всего применяются режимы default и src_dst_port, поэтому начать рекомендуется именно с них.

Стандартный алгоритм (802.3ad):

В использовании стандартного алгоритма EtherChannel есть свои преимущества.

Стандартный алгоритм применяется как для обычного объединения каналов связи, так и для объединения IEEE 802.3ad. AIX делит последний байт исходного числового значения на число адаптеров канала EtherChannel и выбирает линию для передачи исходящих данных в соответствии с полученным остатком. Если остаток равен нулю, то используется первый адаптер EtherChannel; остаток, равный единице, означает выбор второго адаптера и т. д. (адаптеры выбираются в том порядке, в котором они перечислены в атрибуте **adapter_names**).

Режим хэширования задает исходное число, которое применяется при вычислении. По умолчанию используется последний байт целевого IP-адреса или MAC-адреса, однако могут также применяться номера исходных и целевых портов TCP и UDP. Перечисленные варианты позволяют оптимальным образом настроить распределение данных между физическими адаптерами EtherChannel.

В режиме хэширования по умолчанию алгоритм выбора адаптера использует в качестве исходных данных целевой IP адрес (для данных IP). Для данных ARP и для других данных, не относящихся к IP, та же самая формула применяется к последнему байту целевого MAC-адреса. При отсутствии сбоев адаптеров все данные, передаваемые между двумя хостами в стандартном режиме по умолчанию передаются через один и тот же адаптер. Режим хэширования по умолчанию может быть очень эффективным в том случае, когда локальный хост устанавливает соединение с множеством различных IP-адресов.

Если локальный хост устанавливает продолжительные соединения с несколькими IP-адресами, то вы заметите что нагрузка на некоторые адаптеры будет выше, чем на другие, поскольку все данные, передаваемые по определенному получателю, отправляются через один и тот же адаптер. Несмотря на то, что это позволяет избежать появления внеочередных пакетов, в этом случае пропускная способность сети используется не самым эффективным образом. Режимы хэширования по номеру порта также обеспечивают правильный порядок пакетов, однако они допускают передачу пакетов, относящихся к разным соединениям UDP и TCP, даже если эти соединения установлены с одним и тем же адресом, по разным адаптерам, используя при этом возможности всех адаптеров.

В режиме хэширования **src_dst_port** номера исходного и целевого порта TCP или UDP складываются, а затем делятся пополам. Полученное целое число (без десятичных знаков) передается стандартному алгоритму. Данные TCP и UDP передаются с помощью адаптера, выбранного стандартным алгоритмом на основании значения, выбранного в соответствии с данным режимом хэширования. Данные, не относящиеся к TCP и UDP, по-прежнему используют режим хэширования по умолчанию, т. е. вычисления выполняются по последнему байту целевого IP-адреса или MAC-адреса. Режим хэширования **src_dst_port** использует номера и исходного и целевого порта TCP или UDP. В этом режиме все пакеты, относящиеся к одному соединению

TCP или UDP, передаются с помощью одного и того же адаптера, а значит, они принимаются в правильном порядке, однако при этом достигается равномерное распределение нагрузки, поскольку соединения (даже с одним и тем же хостом) могут устанавливаться с помощью разных адаптеров. При этом результаты применения данного режима хэширования не связаны с направлением установленного соединения, так как используются номера и исходного и целевого порта TCP или UDP.

В режиме хэширования **src_port** используется номер исходного порта TCP или UDP. В режиме хэширования **dst_port** используется номер целевого порта TCP или UDP отправляемого пакета. Режимы хэширования **src_port** и **dst_port** следует применять в тех случаях, когда номера портов меняются от соединения к соединению и опция **src_dst_port** не обеспечивает требуемого распределения нагрузки.

Просмотр списка каналов EtherChannel или объединений линий

Ниже описана процедура просмотра списка каналов EtherChannel или объединений линий.

1. Введите в командной строке `smitty etherchannel`.
2. Выберите пункт **Показать список каналов EtherChannel/объединений линий** и нажмите Enter.

Изменение альтернативного адреса

Выполните следующие шаги чтобы указать адрес MAC для канала EtherChannel или объединения линий.

1. В зависимости от того, какая версия AIX используется, вам может понадобиться отсоединить интерфейс:
   - В AIX 5.2 с 5200-01 и более ранних версиях введите `smitty chinet` и выберите интерфейс, принадлежащий вашему EtherChannel. Измените атрибут **Текущее СОСТОЯНИЕ**, указав значение **detach**, и нажмите Enter.
   - В AIX 5L версии 5.2 с рекомендуемым пакетом обслуживания 5200-03 и более поздних версий изменить альтернативный адрес EtherChannel можно без отключения интерфейса.
2. Введите в командной строке `smitty etherchannel`.
3. Выберите пункт **Изменить параметры канала EtherChannel** и нажмите клавишу Enter.
4. При наличии нескольких каналов EtherChannel выберите канал, для которого нужно задать альтернативный адрес.
5. Укажите в поле **Задать альтернативный адрес EtherChannel** значение `да`.
6. Укажите альтернативный адрес в поле **Альтернативный адрес EtherChannel**. Указанный 12-значный шестнадцатеричный адрес должен начинаться с символов `0x` (например, 0x001122334455).
7. Нажмите клавишу Enter, чтобы завершить операцию.

   Примечание: Изменение MAC-адреса EtherChannel во время работы может привести к кратковременному разрыву соединений. Это связано с необходимостью сброса адаптеров и передачей им нового аппаратного адреса, а инициализация некоторых адаптеров может занимать несколько секунд.

Динамическая настройка адаптеров

В версиях, предшествовавших AIX 5L версии 5.2 с рекомендуемым пакетом обслуживания 5200-03, для добавления адаптера в EtherChannel или для его удаления сначала нужно было отключить интерфейс, прервав тем самым соединения всех пользователей. Для устранения этого ограничения в AIX 5L версии 5.2 с рекомендуемым пакетом обслуживания 5200-03 появилась функция динамической настройки адаптеров (DAM).

Эта функция позволяет добавлять адаптеры в EtherChannel и удалять их, не прерывая установленные пользователями соединения. С ее помощью можно также удалить или добавить резервный адаптер; канал EtherChannel можно первоначально создать без резервного адаптера, а затем добавить адаптер в случае необходимости.

Теперь вы можете не только добавлять и удалять адаптеры, не мешая работе пользователей, но и изменять большинство атрибутов EtherChannel. Например, вы можете включить проверку "ping" для резервного сетевого интерфейса во время работы EtherChannel или в любой момент изменить адрес проверяемого удаленного хоста.

Вы также можете включить обычный канал EtherChannel в объединение линий IEEE 802.3ad (и наоборот), предоставив пользователям возможность поэкспериментировать с этими двумя функциями, не удаляя канал EtherChannel и не создавая его заново.

Более того, DAM позволяет создать канал EtherChannel с одним адаптером. Канал EtherChannel с одним адаптером работает точно так же, как обычный адаптер, однако в случае выхода этого адаптера из строя его можно заменить, не прерывая соединение. Для выполнения этой задачи необходимо добавить в EtherChannel временный адаптер, удалить неисправный адаптер, с помощью функции оперативной замены установить вместо него новый, добавить новый адаптер в канал EtherChannel, а затем удалить временный адаптер. На протяжении всей этой процедуры установленные соединения будут сохранены. Если же адаптер работает в автономном режиме, то перед его удалением с помощью функции оперативной замены его нужно будет отключить, прервав тем самым передачу всех данных.

Добавление, удаление или изменений адаптера в канале EtherChannel или в объединении линий

Существует два способа добавления, удаления и изменения адаптеров в канале EtherChannel или в объединении линий.

Первый способ требует отключения интерфейса EtherChannel или объединения линий. Во втором способе отключать интерфейс необязательно (в этом случае применяется возможность динамической настройки адаптеров, поддерживаемая в AIX 5L версии 5.2 с рекомендуемым пакетом обслуживания 5200-03 и в более поздних версиях).

Внесение изменений в канал EtherChannel с помощью динамической настройки адаптеров:

Внесение изменений с помощью динамической настройки адаптеров не требует прекращения передачи данных через EtherChannel путем отключения интерфейса.

Перед выполнением операции ознакомьтесь со следующей информацией:

1. При добавлении адаптера во время работы обратите внимание, что разные адаптеры Ethernet поддерживают разные возможности (например, возможность аппаратного вычисления контрольных сумм, использования частных сегментов, отправки больших блоков данных и т.д.) Если в одном и том же канале EtherChannel применяются адаптеры разных типов, то на уровень интерфейса передается информация о возможностях, поддерживаемых всеми адаптерами (например, если все адаптеры, кроме одного, поддерживают частные сегменты, то EtherChannel не будет поддерживать частные сегменты; если все адаптеры поддерживают отправку больших блоков данных, то канал также будет поддерживать такую операцию). При добавлении адаптера в EtherChannel во время работы необходимо следить, чтобы добавляемый адаптер поддерживал как минимум те же возможности, что и адаптеры, уже входящие в состав EtherChannel. Если вы попытаетесь добавить адаптер, который поддерживает не все возможности, поддерживаемые EtherChannel, то произойдет ошибка. Однако помните, что если интерфейс EtherChannel отключен, то можно добавлять любые адаптеры (независимо от поддерживаемых ими возможностей), поскольку при активации интерфейса EtherChannel заново проверить все возможности, поддерживаемые новым набором адаптеров.
2. Если вы не применяете альтернативный адрес и планируете удалить адаптер, MAC-адрес которого использовался в EtherChannel (MAC-адрес, используемый EtherChannel, "принадлежит" одному из адаптеров), то EtherChannel будет применять MAC-адрес следующего доступного адаптера (т.е. того адаптера, который станет первым после удаления или резервного адаптера, если все основные адаптеры удалены). Например, если в EtherChannel есть основные адаптеры `ent0` и `ent1` и резервный адаптер `ent2`, то по умолчанию будет применяться MAC-адрес адаптера `ent0` (в этом случае говорят, что MAC-адрес "принадлежит" `ent0`). При удалении `ent0` EtherChannel будет использовать MAC-адрес адаптера `ent1`. Если затем удалить адаптер `ent1`, то EtherChannel будет использовать MAC-адрес адаптера `ent2`. Если после этого снова добавить `ent0` в EtherChannel, то будет по-прежнему применяться MAC-адрес `ent2`, поскольку MAC-адрес теперь принадлежит `ent2`. Если затем удалить `ent2` из EtherChannel, то будет снова применяться MAC-адрес `ent0`.

Удаление адаптера, MAC-адрес которого применялся в EtherChannel, может привести к временному прерыванию соединений, поскольку в этом случае необходимо сбросить все адаптеры EtherChannel и сообщить им новый аппаратный адрес. Инициализация некоторых адаптеров может занимать несколько секунд.

Если в EtherChannel применяется альтернативный адрес (т.е. указанный вами MAC-адрес), то этот адрес будет применяться независимо от добавления и удаления адаптеров. Более того, в этом случае при добавлении или удалении адаптеров связь не будет прерываться даже на короткое время, поскольку в EtherChannel нет адаптеров, которым "принадлежит" MAC-адрес.

3. Теперь почти все атрибуты EtherChannel можно изменять во время работы. Единственное исключение составляет атрибут **Включить поддержку больших кадров Gigabit Ethernet**. Для изменения атрибута **Включить поддержку больших кадров Gigabit Ethernet** необходимо сначала отключить интерфейс EtherChannel, и лишь после этого изменить значение.
4. Для всех атрибутов, которые нельзя изменить во время работы (в настоящее время это относится только к атрибуту **Включить поддержку больших кадров Gigabit Ethernet**) предусмотрено поле **Применить изменения только к базе данных**. Если в этом поле указано значение `да`, то вы можете изменять во время работы даже те значения, которые обычно изменять во время работы нельзя. Если в поле **Применить изменения только к базе данных** указано значение `да`, то атрибут будет изменен только в ODM и его изменение не будет отражено в работающем канале EtherChannel до тех пор, пока он не будет перезагружен (например, при отключении интерфейса, вводе команд `rmdev -l EtherChannel_device` и `mkdev -l EtherChannel_device`) или до тех пор, пока не будет перезагружена вся система. Это очень удобный способ внесения изменений при следующей загрузке системы без нарушения работы EtherChannel.
5. В логическом разделе при удалении адаптера из EtherChannel дополнительно необходимо удалить связанный порт коммутатора из EtherChannel. В противном случае связь может быть потеряна, поскольку коммутатор может использовать тот же порт для обмена данными.

Для внесения изменений в EtherChannel или в объединение каналов с помощью динамической настройки адаптеров выполните следующие действия:

1. Введите в командной строке `smitty etherchannel`.
2. Выберите пункт **Изменить параметры канала EtherChannel/объединения линий**.
3. Выберите канал EtherChannel или объединение линий, которое необходимо изменить.
4. Заполните все обязательные поля в соответствии со следующими инструкциями:
   - В поле **Добавить адаптер** или **Удалить адаптер** выберите адаптер Ethernet для добавления или удаления.
   - В поле **Добавить резервный адаптер** или **Удалить резервный адаптер** выберите адаптер Ethernet, который необходимо начать или прекратить использовать в качестве резервного.
   - Почти все атрибуты EtherChannel, кроме атрибута **Включить поддержку больших кадров Gigabit Ethernet**, можно изменять во время работы.
   - Для включения обычного канала EtherChannel в объединение линий IEEE 802.3ad измените атрибут **Режим** на `8023ad`. Для включения объединения линий IEEE 802.3ad в EtherChannel измените атрибут **Режим** на `standard` или `round_robin`.
5. Укажите необходимые значения и нажмите Enter.

Внесение изменений в EtherChannel в 5200-01 и более ранних версиях:

Данная процедура предназначена для отсоединения интерфейса и внесения изменений в 5200-01 и более ранних версиях.

1. Введите `smitty chinet` и выберите интерфейс, принадлежащий вашему EtherChannel. Измените атрибут **Текущее СОСТОЯНИЕ**, указав значение **detach**, и нажмите Enter.
2. Введите в командной строке `smitty etherchannel`.
3. Выберите пункт **Изменить параметры канала EtherChannel/объединения линий** и нажмите Enter.
4. Выберите канал EtherChannel или объединение линий, которое необходимо изменить.

5. Измените атрибуты EtherChannel или объединения линий и нажмите Enter.
6. Укажите в полях формы всю необходимую информацию и нажмите клавишу Enter.

Удаление канала EtherChannel или объединения линий:

Ниже приведена процедура удаления канала EtherChannel или объединения линий.

1. Введите `smitty chinet` и выберите интерфейс, принадлежащий вашему EtherChannel. Измените атрибут **Текущее СОСТОЯНИЕ**, указав значение **detach**, и нажмите Enter.
2. Введите в командной строке `smitty etherchannel`.
3. Выберите пункт **Удалить канал EtherChannel** и нажмите Enter.
4. Выберите удаляемый канал EtherChannel и нажмите клавишу Enter.

Настройка или удаление резервного адаптера в существующем EtherChannel или объединении линий:

Ниже описана процедура добавления и удаления резервного адаптера канала EtherChannel или объединения линий.

1. Введите `smitty chinet` и выберите интерфейс, принадлежащий вашему EtherChannel. Измените атрибут **Текущее СОСТОЯНИЕ**, указав значение **detach**, и нажмите Enter.
2. Введите в командной строке `smitty etherchannel`.
3. Выберите пункт **Изменить параметры канала EtherChannel/объединения линий**.
4. Выберите канал EtherChannel или объединение линий, для которого нужно добавить или удалить резервный адаптер.
5. Укажите резервный адаптер в поле **Резервный адаптер** или выберите значение **Нет**, чтобы удалить резервный адаптер из канала.

Настройка объединения линий IEEE 802.3ad

IEEE 802.3ad - это стандартный способ объединения нескольких линий связи в одну. Принципиально он ничем не отличается от технологии EtherChannel, то есть несколько адаптеров Ethernet объединяются в один виртуальный адаптер, обеспечивающий более высокую пропускную способность и более надежную защиту от неполадок.

Например, адаптеры `ent0` и `ent1` можно объединить в объединение линий IEEE 802.3ad `ent3`. Затем интерфейсу `ent3` можно назначить IP-адрес. С точки зрения системы объединенные адаптеры представляют собой один адаптер. Протокол IP настраивается для этих адаптеров как и для любого другого адаптера Ethernet.

Для применения IEEE 802.3ad необходим коммутатор, поддерживающий эту технологию.

Преимущества применения объединения каналов IEEE 802.3ad перед EtherChannel заключается в том, что можно использовать коммутаторы, которые поддерживают IEEE 802.3ad, но не поддерживают EtherChannel. Кроме того, IEEE 802.3ad обеспечивает защиту от сбоев адаптеров.

После настройки объединения линий IEEE 802.3ad сервер (система хоста) начинает обмениваться со смежным коммутатором блоками данных управляющего протокола объединения линий (LACPDU). Только активный канал, который может быть основным каналом или резервным адаптером, обменивается LACPDU со смежным коммутатором.

Коммутатор позволяет объединить только те адаптеры, для которых установлена одинаковая скорость передачи данных (например, 100 Мбит/с или 1 Гбит/с) и дуплексный режим передачи. AIX допускает объединение адаптеров с разными скоростями или режимами передачи данных, однако при объединении таких адаптеров на коммутаторе может возникнуть ошибка. Если коммутатору не удастся объединить адаптеры, производительность сети может значительно снизиться. Информация о том, как узнать, были ли адаптеры объединены на коммутаторе, приведена в разделе “Устранение неполадок объединения линий IEEE 802.3ad” на стр. 394.

В соответствии со спецификацией IEEE 802.3ad все пакеты с одинаковым целевым IP-адресом отправляются через один и тот же адаптер. Следовательно, при работе в режиме `802.3ad` пакеты всегда распределяются стандартным, а не карусельным методом.

Объединение линий IEEE 802.3ad поддерживает функцию резервного адаптера, как и EtherChannel. Кроме того, резервный адаптер поддерживает LACP IEEE 802.3ad. Порт коммутатора, подключенный к резервному адаптеру, также должен поддерживать IEEE 802.3ad.

Примечание: Действия, которые необходимо выполнить для включения поддержки IEEE 802.3ad, зависят от особенностей конкретного коммутатора. Ознакомьтесь с документацией по коммутатору и выполните приведенные в ней инструкции по включению LACP на коммутаторе.

Информация о настройке объединения линий IEEE 802.3ad приведена в разделе “Настройка объединения линий IEEE 802.3ad”.

Перед настройкой объединения линий IEEE 802.3ad обратите внимание на следующее:

- Хотя официально это не подтверждено, реализация технологии IEEE 802.3ad в AIX позволяет Объединению линий связи включать в себя адаптеры с различным быстродействием линий. Однако необходимо объединить только адаптеры с одинаковой скоростью передачи, работающие в дуплексном режиме. В противном случае могут возникнуть неполадки при настройке объединения линий на коммутаторе. Дополнительная информация о типах объединения, поддерживаемых коммутатором, приведена в документации по коммутатору.
- Если в объединение линий планируется добавить адаптеры 10/100 Ethernet, то перед добавлением этих адаптеров для них необходимо включить функцию опроса линии. Для этого введите команду `smitty chgenet`. Укажите в поле **Включить опрос линии связи** значение `да` и нажмите Enter. Повторите это действие операцию каждого адаптера 10/100 Ethernet, который планируется добавить в объединение линий.

  Примечание: В AIX 5L версии 5.2 с рекомендуемым пакетом обслуживания 5200-03 и в более поздних версиях явно включать опрос линии не нужно. Он включается автоматически.

Настройка объединения линий IEEE 802.3ad:

Для настройки объединения линий IEEE 802.3ad выполните следующие действия.

1. Введите в командной строке `smitty etherchannel`.
2. Выберите в списке пункт **Добавить канал EtherChannel/объединение линий** и нажмите Enter.
3. Выберите основные адаптеры Ethernet, которые нужно включить в объединение линий, и нажмите Enter. Если будет применяться резервный адаптер, не выбирайте этот адаптер на данном этапе.

   Примечание: Окно **Доступные сетевые адаптеры** содержит список всех адаптеров Ethernet. Если выбрать уже используемый адаптер Ethernet (для которого определен интерфейс), то будет выведено сообщение об ошибке. Для применения этих адаптеров сначала необходимо отключить их интерфейсы.
4. Заполните поля формы, следуя приведенным ниже указаниям:
   - **Родительский адаптер:** Это поле содержит сведения о родительском устройстве EtherChannel (например, когда канал EtherChannel принадлежит общему адаптеру Ethernet). Это поле содержит значение `HET` если канал EtherChannel не содержится в другом адаптере (по умолчанию). Если канал EtherChannel содержится в другом адаптере, в поле отображается имя родительского адаптера (например, `ent6`). Это поле носит информационный характер и не может быть изменено. Родительский адаптер можно задать только в AIX 5.3и более поздних версиях.
   - **Адаптеры канала EtherChannel/объединения линий:** В списке должны быть указаны все основные адаптеры объединения линий. Эти адаптеры были выбраны на предыдущем шаге.
   - **Задать альтернативный адрес:** Это необязательное поле. Если вы укажете в этом поле значение `yes`, то можно будет задать адрес MAC для данного объединения линий. Если вы укажете в этом поле значение `no`, то объединение линий будет применять адрес MAC первого адаптера.

- **Альтернативный адрес:** Если в поле **Задать альтернативный адрес** было указано значение `да`, введите в этом поле необходимый адрес MAC. Указанный 12-значный шестнадцатеричный адрес должен начинаться с символов `0x` (например, 0x001122334455).
- **Включить большие кадры Gigabit Ethernet:** Это поле заполнять необязательно. Для применения этой функции коммутатор должен поддерживать большие кадры. Эта функция поддерживается только интерфейсами Standard Ethernet (en), но не интерфейсами IEEE 802.3 (et). Для включения функции укажите значение `да`.
- **Режим:** Введите значение `8023ad`.
- **Режим хэширования**: Вы можете выбрать один из следующих режимов, определяющих исходные данные, применяемые алгоритмом для выбора адаптера:
  - **default**: В этом режиме для определения применяемого адаптера используется целевой IP-адрес пакета. При обработке пакетов других протоколов, например, ARP, для вычислений применяется последний байт целевого адреса MAC. В этом режиме при передаче пакетов по каналу EtherChannel сохраняется очередность их отправки, но не всегда обеспечивается эффективное использование пропускной способности канала.
  - **src_port**: В этом режиме для определения применяемого адаптера используется номер исходного порта UDP или TCP пакета. Если пакет не относится к UDP или TCP, то применяется последний байт целевого IP-адреса. Если пакет не относится к IP, то применяется последний байт целевого MAC-адреса.
  - **dst_port**: В этом режиме для определения применяемого адаптера используется номер целевого порта UDP или TCP пакета. Если пакет не относится к UDP или TCP, то применяется последний байт целевого IP-адреса. Если пакет не относится к IP, то применяется последний байт целевого MAC-адреса.
  - **src_dst_port**: В этом режиме для определения применяемого адаптера используются номера исходного и целевого порта UDP или TCP пакета (номера исходного и целевого портов складываются и делятся пополам, а полученное значение передается алгоритму выбора). Если пакет не относится к UDP или TCP, то применяется последний байт целевого IP-адреса. Если пакет не относится к IP, то применяется последний байт целевого MAC-адреса. Этот режим обеспечивает хорошее распределение пакетов в большинстве случаев, как для клиентов, так и для серверов.

  Дополнительная информация о распределении пакетов и о выравнивании нагрузки приведена в разделе “Опции выравнивания нагрузки EtherChannel” на стр. 386.
- **Резервный адаптер:** Это поле заполнять необязательно. В нем можно указать резервный адаптер.
- **Опрашиваемый IP-адрес:** Это необязательное поле. Оно доступно если объединение линий состоит из одного или нескольких основных и одного резервного адаптера. Объединение линий будет проверять связь с IP-адресом или хостом, указанным в этом поле. Если объединение линий не получит от этого IP-адреса ответ на запросы, число которых указано в поле **Количество повторов** с интервалом, заданным в поле **Тайм-аут повтора**, то активный адаптер заменяется резервным.
- **Количество повторов:** Укажите количество неудачных попыток проверить связь, при достижении которого в объединении линий должно происходить переключение на другой адаптер. Значение по умолчанию - 3. Это поле необязательно для заполнения и применяется только в том случае, если задан **Опрашиваемый IP-адрес.**
- **Тайм-аут повтора:** Укажите интервал отправки пробных пакетов **Опрашиваемому IP-адресу** в секундах. Значение по умолчанию - одна секунда. Это поле необязательно для заполнения и применяется только в том случае, если задан **Опрашиваемый IP-адрес.**

5. Изменив необходимые поля, нажмите клавишу Enter, чтобы создать объединение линий.
6. Введите команду `smitty chinet`, чтобы настроить для созданного устройства объединения линий протокол IP.
7. Выберите в списке интерфейс объединения линий.
8. Укажите в полях формы всю необходимую информацию и нажмите клавишу Enter.

Устранение неполадок объединения линий IEEE 802.3ad:

С помощью команды **entstat** можно устранить неполадки объединения линий IEEE 802.3ad.

Если в работе объединения линий IEEE 802.3ad возникают ошибки, проверьте режим работы объединения линий с помощью следующей команды:

```
entstat -d устройство
```

где *устройство* - это устройство объединения линий.

Эта команда предоставляет самый простой способ определить состояние LACP на основе LACPDU, полученных от коммутатора. Допустимы следующие состояния:

- Не активен: LACP не запущен. Это состояние означает, что объединение линий еще не настроено, так как ему не назначен IP-адрес, либо его интерфейс отключен.
- Согласование: LACP запущен, но коммутатор еще не объединил адаптеры. Если объединение линий будет находиться в этом состоянии более минуты, проверьте правильность настройки коммутатора. Например, убедитесь, что протокол LACP включен для портов.
- Объединение создано: Процедура LACP успешно выполнена, и коммутатор объединил адаптеры.
- Сбой: Произошел сбой LACP. Возможно, объединяемые адаптеры применяют разные скорости или режимы передачи данных, либо они подключены к разным коммутаторам. Проверьте правильность конфигурации адаптеров.

  Обратите внимание, что некоторые коммутаторы позволяют объединять только те порты, номера которых образуют непрерывный диапазон, и накладывают ограничение на максимальное число объединяемых адаптеров. Ознакомьтесь с документацией по коммутатору, узнайте, какие ограничения накладываются коммутатором, и проверьте правильность конфигурации коммутатора.

Примечание: Состояние объединения линий указывается только для целей диагностики и не относится к конфигурации, заданной в операционной системе AIX. Информация о состоянии была получена оптимальным способом. Для отладки неполадок объединения линий рекомендуется проверить конфигурацию коммутатора.

Следующая статистика объединения линий IEEE 802.3ad представляет состояние LACP для каждого порта объединения.

Сведения приведены как для актора (объединение линий IEEE 802.3ad) так и для партнера (порт коммутатора).

```
Приоритет системы: значение приоритета для данной системы
Система: уникальное значение, идентифицирующее систему
Ключ выполнения: значение, указывающее какие порты могут быть
        объединены
Приоритет порта: значение приоритета для данного порта
Порт: уникальное значение, идентифицирующее порт при объединении
Состояние:
        Активность LACP: активен или пассивен, указывает, следует ли
                всегда отправлять LACPDU или только в ответ на другой
                LACPDU: объединение линий IEEE 802.3ad всегда находится
                в режиме Активен
        Тайм-аут LACP: короткий или длинный - время ожидания
                перед отправкой LACPDU:объединение линий IEEE 802.3ad
                всегда использует длинный тайм-аут
        Объединение: отдельное или агрегированное - можно ли
                объединять этот порт с другими портами или только с самим
                собой: порт, находящийся в одноадаптерном объединении
                линий IEEE 802.3ad будет помечен как отдельный,
```

либо как объединяемый при наличии нескольких портов

Синхронизация: IN_SYNC или OUT_OF_SYNC - указывает что объединение обнаружило синхронизацию с партнером или нет

Сбор пакетов: включено или выключено - указывает, принимает ли объединение линий IEEE 802.3ad пакеты или нет

Распространение: включено или выключено - указывает, распространяет ли объединение линий IEEE 802.3ad пакеты или нет

По умолчанию: истина или ложь - указывает, следует ли объединению линий IEEE 802.3ad использовать значения по умолчанию для сведений о партнере

С истекшим сроком: истина или ложь, указывает, является ли объединение линий IEEE 802.3ad рабочим в режиме истекшего срока

Следующие статистические сведения приведены как для каждого порта, так и для их объединений:

Полученные LACPDU: Полученные пакет LACPDU
Переданные LACPDU: Отправленные пакеты LACPDU
Принятые PDU маркеров: полученные PDU маркеров
Переданные PDU маркеров: отправленные PDU маркеров:
в данной версии протокола протокол маркеров не реализован поэтому эти сведения всегда будут иметь нулевое значение
Принятые PDU ответов маркеров: полученные PDU маркеров
Переданные PDU ответов маркеров: отправленные PDU маркеров:
в данной версии протокола протокол маркеров не реализован поэтому эти сведения всегда будут иметь нулевое значение
Полученные неизвестные PDU: принятые PDU неизвестного типа
Принятые недопустимые PDU: принятые PDU известного типа, но неправильно скомпонованные, недопустимого размера или неизвестного подтипа

Сценарии стыкуемости

Ознакомьтесь с этими сценариями перед настройкой канала EtherChannel или объединения линий IEEE 802.3ad.

Описание сценариев приведено после таблицы.

Таблица 81. Возможные сочетания конфигураций AIX и коммутатора

Режим EtherChannel	Конфигурация коммутатора	Результат
8023ad	IEEE 802.3ad LACP	OK - AIX отправит LACPDU, чтобы активировать объединение линий IEEE 802.3ad на коммутаторе.
standard или round_robin	EtherChannel	OK - Канал EtherChannel будет работать в обычном режиме.
8023ad	EtherChannel	Нежелательное сочетание - AIX и коммутатор не смогут объединиться. AIX инициирует LACPDU, однако коммутатор их проигнорирует и не отправит LACPDU в AIX. Поскольку отсутствует LACPDU, AIX не будет передавать пакеты в канал или порт. В результате связь по сети будет нарушена.
standard или round_robin	IEEE 802.3ad LACP	Нежелательное сочетание - Коммутатор не сможет объединить адаптеры. Результатом может стать снижение производительности, связанное с тем, что коммутатор будет передавать адрес MAC между портами коммутатора.

Ниже приведено краткое описание каждой комбинации конфигурации:

- 8023ad с EtherChannel:

  В этом случае коммутатор не будет отвечать на блоки данных LACPDU, отправляемые AIX, так как он будет настроен в качестве канала EtherChannel. Поскольку отсутствует LACPDU, AIX не будет использовать канал или порт для передачи пакетов. В результате связь по сети будет нарушена.

  Примечание: В этом случае в выводе команды `entstat -d` объединение линий будет всегда находиться в состоянии Согласование. Кроме того, в выводе entstat в разделе Статистика порта IEEE 802.3ad состояние **Передача** для **Агента** будет показано как выключенное.

- standard или round_robin с EtherChannel:

  Это наиболее распространенная конфигурация EtherChannel.

- standard или round_robin с IEEE 802.3ad LACP:

  Такая конфигурация недопустима. Коммутатор попытается создать объединение с помощью LACP, однако ему не удастся это сделать, так как AIX не будет отвечать на отправленные LACPDU. Для применения объединения линий в AIX нужно установить режим `8023ad`.

Поддерживаемые адаптеры

Канал EtherChannel и объединение линий IEEE 802.3ad поддерживается адаптерами Ethernet IBM Power Systems Peripheral Component Interconnect-X (PCI-X) и PCI Express (PCIe).

Ниже приведена дополнительная информация:

- Виртуальный адаптер ввода-вывода Ethernet

  Виртуальные адаптеры ввода-вывода Ethernet поддерживаются только в двух возможных конфигурациях EtherChannel:

  – Один основной виртуальный адаптер ввода-вывода Ethernet и один резервный виртуальный адаптер ввода-вывода Ethernet. В такой конфигурации атрибут **IP-адрес для проверки** должен быть включен, чтобы EtherChannel мог обнаружить сбой удаленного соединения. Для VIOS версии 2.2.3.0 или выше и AIX версии 7.1 с технологическим уровнем 3 можно использовать функцию состояния виртуальной линии Ethernet для определения сбоев обслуживающего VIOS или общего адаптера Ethernet (SEA) путем установки значения `yes` в атрибуте **`poll_uplink`** устройства виртуального Ethernet.

  – Один поддерживаемый физический адаптер Ethernet является основным, один виртуальный адаптер ввода-вывода Ethernet является резервным. В такой конфигурации атрибут **IP-адрес для проверки** должен быть включен, чтобы EtherChannel мог обнаружить сбой удаленного соединения.

- Адаптер хоста Ethernet(HEA)

  Логические порты HEA поддерживаются в EtherChannel только в том случае, если все адаптеры в EtherChannel являются логическими портами HEA. В случае применения выделенного порта HEA поддерживается объединение каналов с адаптером PCI/PCI-E. Кроме того, в качестве резервного адаптера можно использовать адаптер PCI/PCI-E или виртуальный адаптер Ethernet (если основной адаптер содержит HEA).

  При использовании нескольких логических портов HEA в качестве основных адаптеров в EtherChannel, физические порты, связанные с логическими портами HEA, также должны находиться в EtherChannel в коммутаторе Ethernet. Следовательно, все разделы, использующие логические порты HEA, относящиеся к одним и тем же физическим портам HEA, также должны находиться в EtherChannel.

  Например, допустим, что Раздел 1 имеет следующую конфигурацию:

  – Логический порт HEA из физического порта HEA 0

  – Логический порт HEA из физического порта HEA 1

  – Канал EtherChannel, созданный с использованием логических портов HEA, перечисленных выше

  Если другой раздел в той же системе должен использовать логический порт HEA из физического порта HEA 0 или из физического порта HEA 1, необходимо создать канал EtherChannel для этого раздела по обоим логическим портам HEA, аналогично конфигурации Раздела 1. Попытка использования любого из

этих двух логических портов HEA в качестве автономных портов в других разделах может привести к неполадкам в соединениях, поскольку доставка пакетов может осуществляться на другой логический порт HEA, вместо нужного порта.

Ограничение не действует при использовании логических портов HEA в конфигурации Резервного сетевого интерфейса (1 основной и 1 резервный), поскольку физические порты HEA не требуют специальной конфигурации на коммутаторе Ethernet.

Примечание: Если логические порты, связанные с физическими портами HEA, настроены в ходе объединения LACP (802.3ad), то эти физические порты должны использоваться только соответствующим логическим разделом. HMC не запрещает назначение портов другим разделам, однако не поддерживает такую конфигурацию.

- Объединенные сетевые адаптеры Fibre Channel over Ethernet

  Объединение линий между общим портом (порт, применяемый для передачи данных Ethernet и Fiber Channel) и другими поддерживаемыми адаптерами поддерживается только в том случае, если коммутатор, подключенный к общему порту, поддерживает объединение линий без снижения производительности Fiber Channel.
- Адаптеры SR-IOV

  Объединение линий с логическими портами SR-IOV можно осуществить одним из следующих способов:
  - Объединением линий IEEE 802.3ad (LACP)
  - Посредством резервного сетевого интерфейса (NIB)
  - LACP и NIB

  Для сетевых приложений, где требуется пропускная способность более одного порта, можно использовать объединение линий IEEE 802.3ad для объединения нескольких логических портов SR-IOV. Логический порт SR-IOV, участвующий в объединении линий IEEE 802.3ad, должен быть единственным логическим портом, настроенным для физического порта. Несколькими логическими портами SR-IOV, настроенными для одного физического порта, где один из логических портов SR-IOV настроен как часть конфигурации объединения линий IEEE 802.3ad, коммутатор может управлять неправильно, поскольку через физический порт могут передаваться данные более одного партнера LACP. Для того чтобы предотвратить настройку второго логического порта SR-IOV на том же физическом порте в качестве логического порта SR-IOV в конфигурации объединения линий IEEE 802.3ad, должно быть указано значение 100 (100 %) для емкости логического порта во время настройки логического порта.

  Для сетевых приложений, где требуется пропускная способность меньше одного порта вместе с защитой от одного сетевого сбоя, логические порты SR-IOV могут быть частью конфигурации NIB. Когда логический порт SR-IOV настроен как часть основного или резервного адаптера в конфигурации NIB, физический порт может использоваться другими логическими адаптерами SR-IOV. В такой конфигурации можно включить атрибут **Опрашиваемый IP-адрес** для обнаружения сбоев удаленных соединений. Логический порт SR-IOV может быть основным или резервным адаптером для другого логического порта SR-IOV, виртуальным адаптером Ethernet или портом физического адаптера.

С дополнительной информацией о новых адаптерах можно ознакомиться в Информации о выпуске AIX, соответствующей вашей версии AIX.

Важно: Применение в одном канале EtherChannel адаптеров с разной скоростью передачи данных, даже если один из этих адаптеров является резервным, не поддерживается. Это *не* значит, что такая конфигурация не будет работать. Драйвер EtherChannel предпримет все возможные меры для обеспечения работоспособности даже в конфигурации с разной скоростью адаптеров.

Информация, связанная с данной:

SR-IOV

Устранение неполадок EtherChannel

Если при работе с EtherChannel возникли неполадки, следует ознакомиться с различными сценариями их устранения.

Для определения неполадки, возможно связанной с восстановлением после сбоя или применением больших кадров, можно воспользоваться трассировкой и статистикой.

Трассировка EtherChannel:

Для устранения неполадок канала EtherChannel можно воспользоваться командами **tcpdump** и **iptrace**.

Идентификатор точки трассировки для пакетов передачи - 2FA, а для остальных событий - 2FB. Выполнить трассировку входящих пакетов можно только для точек трассировки каждого из адаптеров канала, но не для всего канала EtherChannel.

Статистика EtherChannel:

Для получения статистики по всем адаптерам, входящим в состав EtherChannel, вызовите команду **entstat**.

Например, команда `entstat ent3` покажет совокупную статистику для ent3. Указав флаг **-d**, можно также просмотреть отдельные статистические данные для каждого адаптера. Например, команда `entstat -d ent3` показывает совокупные статистические данные по всему каналу EtherChannel, а также по каждому отдельному адаптеру этого канала.

Примечание: В разделе `Общая статистика` значение `Число сбросов адаптера` обозначает количество операций восстановления после сбоя. Для резервного адаптера EtherChannel возвращение к основному адаптеру EtherChannel не рассматривается как восстановление после сбоя. Подсчитывается только количество переключений с основного адаптера на резервный.

Резервный адаптер учитывается в значении `Количество адаптеров`.

Медленное восстановление после сбоя:

Если в режиме резервного сетевого интерфейса или при использовании резервного адаптера EtherChannel восстановление после сбоя выполняется слишком долго, убедитесь, что на коммутаторе не используется протокол STP.

При изменении правил соответствия портов коммутатора адресам MAC коммутатор запускает алгоритм связующего дерева, чтобы проверить сеть на отсутствие циклов. В случае применения резервного сетевого интерфейса или резервного адаптера EtherChannel правила соответствия портов адресам MAC могут меняться.

Для каждого порта коммутатора задан счетчик задержки пересылки, который определяет, как скоро после инициализации порта должна начаться пересылка или отправка пакетов. Следовательно, в случае восстановления основного канала после сбоя соединение восстанавливается с определенной задержкой, тогда как переключение на резервный адаптер происходит безо всякой задержки. Установите для счетчика задержки пересылки коммутатора минимальное значение, для того чтобы ускорить обратное переключение на основной канал.

Для правильной работы функции резервного адаптера EtherChannel необходимо, чтобы задержка пересылки составляла не более 10 секунд. В противном случае при обратном переключении на основной канал EtherChannel может возникнуть ошибка. Рекомендуется задать на коммутаторе минимальное значение задержки.

НЕ запускается восстановление после сбоя для адаптеров:

Если вы применяете AIX 5.2 с 5200-01или более раннюю версию и в случае сбоя адаптера не запускается процедура восстановления после сбоя, то необходимо проверить, нужно ли включить функцию опроса линии связи для обнаружения неполадок.

Некоторые адаптеры не могут автоматически проверить состояние линии связи. Для таких адаптеров нужно включить функцию опроса линии связи, которая периодически проверяет состояние линии с помощью таймера. По умолчанию функция опроса линии связи выключена. Для правильной работы канала EtherChannel с такими адаптерами необходимо, чтобы механизм опроса был включен на всех адаптерах перед созданием канала EtherChannel. Если вы применяете AIX 5L версии 5.2 с рекомендуемым пакетом обслуживания 5200-03 или более позднюю версию, то опрос линии включается автоматически.

У адаптеров, поддерживающих механизм опроса линии, есть атрибут ODM **poll_link**, для которого следует задать значение `yes`, чтобы включить функцию опроса линии. Перед тем как создать канал EtherChannel, введите следующую команду для каждого адаптера, который будет включен в канал:

```
smitty chgenet
```

Укажите в поле **Включить опрос линии связи** значение `да`.

Большие кадры:

Требуется не только включить атрибут **use_jumbo_frame** канала EtherChannel, но и включить поддержку больших кадров для каждого адаптера перед созданием EtherChannel.

Для этого выполните следующую команду:

```
smitty chgenet
```

Поддержка больших кадров автоматически включается для всех адаптеров, если атрибут **use_jumbo_frame** EtherChannel содержит значение yes.

Удаленный дамп:

Создание удаленного дампа по каналу EtherChannel не поддерживается.

Протокол IP для InfiniBand (IPoIB)

Пакеты IP-протокола могут быть отправлены через интерфейс InfiniBand (IB). При этом IP-пакеты заключаются в пакеты IB с помощью сетевого интерфейса.

Для использования IP по IB следует установить и настроить в системе драйвер InfiniBand connection manager (ICM) и по крайней мере одно устройство IB. Для того чтобы определить, имеется ли уже установленное устройство IB, запустите команду **lsdev -C | grep iba**. Имя набора файлов, содержащего интерфейс IB: `devices.common.IBM.ib`. Набор файлов `devices.chrp.IBM.lhca` - пример набора файлов поддерживаемого в настоящее время адаптера.

Для настройки драйвера ICM обратитесь к разделу “Настройка драйвера InfiniBand Communication Manager” на стр. 402.

Для создания интерфейса InfiniBand (IB IF) должна существовать возможность присоединения к этому интерфейсу существующей группы многоцелевой рассылки с предоставляемым пользователем ключом PKEY (либо ключом PKEY = 0xFFFF, применяемым по умолчанию в том случае, если ключ не предоставлен пользователем) и предоставляемым пользователем ключом Q_Key (либо ключом Q_Key = 0x1E, применяемым по умолчанию в том случае, если ключ не предоставлен пользователем). Группа многоцелевой рассылки представляет собой группу, которую следует присоединить к интерфейсу для отправки пакетов оповещения и пакетов **ARP**. Если такая группа не существует или не может быть создана, создание интерфейса IB IF невозможно.

Можно создать или изменить интерфейс IB IF с помощью командной строки или пользовательского интерфейса SMIT. Для создания интерфейса IB IF необходимы следующие параметры:

- *имя интерфейса*
- *имя адаптера*

- *номер порта*
- *IP-адрес интерфейса*

Следующие параметры позволяют изменить IB IF:

- *IP-адрес*
- *маска сети*
- *Размер MTU* (равный желаемому MTU, меньше 4 байтов для заголовка IB)
- *состояние*
- *Размер очереди приема и отправки (значение по умолчанию - 4000)*
- *Ключ очереди многоцелевой рассылки*
- *Поддержка расширенных пакетов*

Ниже приведен пример команды для создания интерфейса IB IF из командной строки:

```
$ /usr/sbin/mkiba -i ib0 -p 1 -A iba0 -a 1.2.3.8 [-P -1 -S "up" -m "255.255.254.0" -M 2044]
```

где:

Элемент	Описание
`-M 2044`	Максимальный блок для передачи.
`-m "255.255.254.0"`	Маска сети.
`-p 1`	Номер порта (значение по умолчанию *1*, если параметр не задан).
`-A iba0`	Имя устройства IB.
`-a 1.2.3.8`	IP-адрес IF.
`-i ib0`	Имя интерфейса.
`-P -1`	Ключ разделения (Значение по умолчанию *PKEY*, если параметр не задан. После создания интерфейса ключ PKEY нельзя изменить; пользователь должен получить ключ PKEY, отличный от заданного по умолчанию, у администратора сети).
`-S "up"`	Состояние интерфейса.
`-q 8000`	Размеры очередей приема и отправки.
`-Q 0x1E`	Ключ очереди многоцелевой рассылки присваивается группе многоцелевой рассылки (значение по умолчанию - Q_KEY = 0x1E).
`-k "on"`	Расширенные пакеты позволяют увеличить размер MTU TCP/IP до 64 КБ. Этот параметр также должен быть задан для удаленного хоста.

Ниже приведен пример команды для создания интерфейса IB IF из пользовательского интерфейса SMIT:

```
$ smitty inet
```

После отображения меню Выбор сетевого протокола выполните следующую процедуру:

1. Выберите **Добавить сетевой интерфейс** или **Изменить/Показать параметры сетевого интерфейса.** Появится меню Добавить сетевой интерфейс.
2. В этом меню выберите **Добавить сетевой интерфейс IB**. Появится меню Добавить сетевой интерфейс IB.
3. В этом меню внесите необходимые изменения и нажмите Enter.

Создание, просмотр, добавление и удаление записей ARP и изменение таймеров ARP

Запись **Протокола преобразования адресов (ARP)** позволяет интерфейсу взаимодействовать с другим интерфейсом даже в том случае, если они входят в разные группы многоцелевой рассылки.

Запись **ARP** можно создать вручную с помощью команды **arp -t ib**.

Для просмотра всех записей **ARP** запустите команду **$ arp -t ib -a**. Если необходимо просмотреть определенное число записей **ARP**, следует указать число. Например, команда `$ arp -t ib -a 5` отображает 5 записей **ARP**.

Следующая команда добавляет запись **ARP**:

```
$ arp -t ib -s имя интерфейса IB dlid <16-разрядный DLID> dqp
16-разрядный шестнадцатеричный номер пары целевой очереди
ipaddr <Целевой IP-адрес>
```

где:

Элемент	Описание
DLID	- локальный целевой ИД.
DGID	- глобальный целевой ИД.

Следующая команда удаляет запись **ARP**:

```
$ arp -t ib -d IP-адрес
```

Следующая команда позволяет изменить значение таймера для неполных и полных записей ARP. Эти значения применяются для удаления записей ARP по истечении заданного времени:

```
arp -t ib -i <время в минутах для удаления
неполных записей ARP>
-c <время в минутах для удаления
полных записей ARP>
```

Текущее значение времени удаления неполных записей ARP - 3 минуты. Время по умолчанию для полных записей ARP - 24 часа. Если требуется изменить значения, то в ходе выполнения команды будут изменены только значения всех настроенных интерфейсов. В случае настройки новых интерфейсов команда будет выполнена снова. Кроме того, значения изменяются в ODM.

Значения можно изменить в динамическом режиме для отдельных интерфейсов с помощью команды **ifconfig**:

```
Таймер
неполных записей ARP можно изменить с помощью следующей команды:
ifconfig ib0 inc_timer 4
ifconfig ib0 com_timer 60
```

Изменение параметров интерфейса InfiniBand

Параметры интерфейса IB IF можно изменить с помощью команд пользовательского интерфейса SMIT либо командной строки.

Для того чтобы изменить параметры интерфейса IB IF с помощью SMIT, выполните следующее:

1. Запустите команду **$ smitty inet**. Появится меню Выбрать сетевой интерфейс.
2. В этом меню выберите **Изменить / Показать параметры сетевого интерфейса**. Появится меню Доступные сетевые интерфейсы.
3. В меню Доступные сетевые интерфейсы выберите пункт **Интерфейс InfiniBand**. Появится меню Изменить / Показать интерфейс IB.
4. Измените необходимые параметры.

Для того чтобы изменить параметры интерфейса IB IF с помощью командной строки, запустите команду **$ ifconfig**. Следующая команда изменяет параметры интерфейса IB IF из командной строки:

```
$ ifconfig ib0 [ib_port номер порта mtu минимальная единица передачи p_key
16-разрядный шестнадцатеричный ключ раздела  ib_adapter имя адаптера InfiniBand маска сети
десятичное число с точками]
```

```
$ ifconfig ib0 inc_timer  3  com_timer 60
```

- *inc_timer* - это срок действия неполной записи ARP в минутах. Значение по умолчанию - 2 минуты.
- *com_timer* - это срок действия полной записи ARP в минутах. Значение по умолчанию - 24 часа.

Настройка драйвера InfiniBand Communication Manager

Используйте эту процедуру для настройки InfiniBand Communication Manager.

1. Запустите команду **$ smitty icm**. Появится меню InfiniBand Communication Manager.
2. В этом меню выберите **Добавить InfiniBand Communication Manager**.
3. В меню Добавить InfiniBand Communication Manager выберите **Добавить InfiniBand Communication Manager**. Появится меню Имя нового IB Communication Manager.
4. В меню Имя IB Communication Manager для добавления выберите пункт **Управление icm InfiniBand**.
5. Используйте значения по умолчанию либо измените необходимые параметры и нажмите Enter.

Инициатор ПО iSCSI и целевой объект ПО

Программный инициатор iSCSI позволяет AIX получать доступ к запоминающим устройствам по сети TCP/IP с использованием адаптеров Ethernet. Целевой объект ПО iSCSI обеспечивает AIX доступ других инициаторов iSCSI к экспортированной локальной памяти с использованием протокола iSCSI, определенного в RFC 3720.

Применение технологии iSCSI, называемой также технологией SAN по сети IP, позволяет развертывать сеть хранения данных с использованием соединений IP. iSCSI представляет собой открытый стандартизированный подход, в рамках которого информация SCSI инкапсулируется в пакеты **TCP/IP**, что позволяет передавать ее по сетям Ethernet и Gigabit Ethernet. Технология iSCSI делает возможной передачу команд и данных SCSI по существующей сети Ethernet, независимо от местоположения. В состав iSCSI входят следующие компоненты, образующие единую систему:

- Инициаторы

  Это драйверы устройств, которые расположены на клиенте. Они инкапсулируют команды SCSI и передают их по сети IP на целевое устройство.

- Целевые программы

  Программы, принимающие инкапсулированные команды SCSI по сети IP. Эти же программы могут обеспечивать настройку и управление памятью.

- Целевые устройства

  Это может быть запоминающее устройство со встроенной памятью, либо шлюз или мост без собственной внутренней памяти.

Настройка инициатора ПО iSCSI

Для того чтобы настроить программный инициатор с помощью SMIT, выполните следующие действия:

1. Выберите **Устройства**.
2. Выберите **iSCSI**.
3. Выберите **Настроить устройство протокола iSCSI**.
4. Выберите **Изменить параметры устройства протокола iSCSI**.
5. Проверьте, правильно ли указано **Имя инициатора**. **Имя инициатора** используется целевым устройством iSCSI при входе в систему.

   Примечание: Имя по умолчанию присваивается инициатору при установке программного обеспечения. Пользователь может изменить его в соответствии с правилами присвоения имен в локальной сети.
6. Значение поля **Максимальное число целевых устройств** соответствует максимальному числу целевых устройств iSCSI, которые можно настроить. Если вы уменьшаете это значение, то вы также уменьшаете объем памяти, первоначально выделяемой для драйвера протокола iSCSI во время настройки.
7. Настройте метод обнаружения iSCSI в поле **Стратегия обнаружения**, чтобы найти целевые устройства iSCSI. ПО инициатора iSCSI поддерживает следующие 4 метода обнаружения:

   file Информация о целевых устройствах сохраняется в файле конфигурации.

   odm Информация о целевых устройствах сохраняется в объектах Администратора объектных данных (ODM). Если диск iSCSI используется как загрузочный диск или в процедуре загрузки **rootvg**, то

необходимо использовать метод обнаружения **odm**. Дополнительная информация приведена в разделе Добавление статически обнаруживаемых целевых устройств iSCSI в ODM.

isns Информация о целевых устройствах сохраняется на сервере Internet Storage Name Service (iSNS) и автоматически получается с сервера в ходе настройки инициатора iSCSI.

slp Информация о целевых устройствах сохраняется в служебном агенте протокола SLP или агенте каталогов и автоматически получается из него в ходе настройки инициатора iSCSI.

После настройки программного инициатора выполните следующие действия:

1. Если стратегия поиска - это **file**, отредактируйте файл `/etc/iscsi/targets`, включив в него целевые устройства iSCSI, которые потребуются при настройке устройств.

   Одному целевому устройству iSCSI соответствует одна строка файла (если она не является комментарием). Дополнительная информация приведена в разделе targets File книги *Справочник по файлам* .

   Если стратегия обнаружения - это **odm**, используйте команду **mkiscsi** или панели smit для создания определений целевых устройств в ODM. Дополнительная информация приведена в разделе Добавление статически обнаруживаемых целевых устройств iSCSI в ODM.

   Если стратегия обнаружения - это **isns** или **slp**, то необходимо правильно настроить сервер iSNS или SLP и сделать его доступным ля инициатора iSCSI.

   Для настройки устройства iSCSI необходимо, чтобы к нему можно было получить доступ через правильно настроенный сетевой интерфейс. Хотя программный инициатор iSCSI может работать с локальной сетью 10/100 Ethernet, он разрабатывался для работы с сетью Gigabit Ethernet, отдельно от остального сетевого потока данных.
2. После определения целевых устройств введите следующую команду:

   ```
   cfgmgr -l iscsi0
   ```

   Эта команда изменит конфигурацию драйвера программного инициатора.

   При вводе этой команды драйвер попытается соединиться с целевыми устройствами, перечисленными в файле `/etc/iscsi/targets`, и определить новый жесткий диск (hdisk) для всех LUN на найденных целевых устройствах. Дополнительная информация приведена в описании команды **cfgmgr** в книге *Справочник по командам, том 1*.

   Примечание: Если соответствующие диски не определены, проверьте, правильно ли настроена конфигурация инициатора, целевого устройства и всех шлюзов iSCSI, а затем повторно введите команду **cfgmgr**.

Если вы хотите выполнить дальнейшую настройку параметров устройств инициатора iSCSI, воспользуйтесь программой SMIT:

1. Выберите **Устройства**.
2. Выберите **Жесткий диск**.

Типичное устройство программного инициатора обычно выглядит следующим образом:

```
hdisk2   Доступен              Другой диск iSCSI
```

Если диск iSCSI поддерживает очереди тегов команд и NACA=1 в управляющем байте, рекомендуется увеличить длину очереди для диска. В ряде случаев это позволяет повысить производительность. Оптимальная длина очереди не должна превышать фактический размер очереди на диске. В противном случае может производительность может снизиться. Для определения размера очереди на диске обратитесь к документации по диску.

Настройка целевого объекта ПО iSCSI

Драйвер целевого объекта ПО iSCSI позволяет AIX действовать в качестве единого целевого устройства iSCSI или в качестве нескольких целевых устройств iSCSI. Драйвер целевого объекта ПО iSCSI выполняет

экспорт локальных дисков, локальных файлов или логических томов инициаторам iSCSI, которые подключаются к AIX согласно протоколам iSCSI и TCP/IP.

У каждого целевого устройства есть полное имя iSCSI (IQN) и набор номеров логических накопителей (LUN), доступных для инициаторов, подключенных к виртуальному целевому объекту iSCSI. Для каждого целевого устройства можно задать сетевой интерфейс и номера портов TCP/IP, которые будут использоваться драйвером целевого объекта для приема входящих соединений.

Примечание: Необходимо установить набор файлов iSCSI devices.tmiscsw.rte, входящий в состав пакета расширения AIX.

Для настройки драйвера целевого объекта iSCSI выполните следующие действия:

1. Создайте один экземпляр драйвера целевого объекта iSCSI согласно указанному ниже пути SMIT. Этот экземпляр выступает в качестве контейнера для других объектов iSCSI.

   Устройства > iSCSI > Целевое устройство iSCSI > Протокол целевого объекта iSCSI > Добавить протокол целевого объекта iSCSI
2. Создайте одно целевое устройство iSCSI для каждого виртуального целевого объекта iSCSI, выделенного драйвером целевого объекта iSCSI. С помощью следующего пути SMIT создайте каждое целевое устройство iSCSI:

   Устройства > iSCSI > Целевое устройство iSCSI > Целевые объекты iSCSI > Добавить целевые объекты iSCSI
3. Определите один или несколько номеров логических накопителей (LUN) для каждого целевого устройства, согласно следующему пути SMIT:

   Примечание: LUN доступны для инициаторов, подключенных к виртуальному целевому объекту. На целевом объекте iSCSI каждый LUN может быть связан либо с ранее определенным логическим томом, физическим томом или файлов, созданным в локальной файловой системе. Ни один физический том, связанный с целевым логическим накопителем iSCSI, не может быть никаким иным образом использован системой AIX, на которой запущен драйвер целевого объекта iSCSI.

   Устройства > iSCSI > Целевое устройство iSCSI > iSCSI LUN целевого объекта

   Этим действием обычно завершается процедура настройки. Тем не менее, если вы пользуетесь Протоколом идентификации с квитированием связи по вызову (CHAP) или Списками управления доступом (ACL) для определения взаимного соответствия между инициаторами и доступными для них номерами LUN, то для завершения настройки необходимо выполнить еще одно действие.

   - Если вы применяете к инициаторам идентификацию CHAP, отредактируйте файл `/etc/tmiscsi/autosecrets`, добавив в него шифры, используемые инициаторами для входа в систему. В файле `/etc/tmiscsi/autosecrets` содержится по одной записи для каждого целевого объекта. Формат каждой записи следующий:

     имя_целевого_объекта имя_chap шифр_chap
   - При использовании ACL для указания того, какие LUN доступны для определенных инициаторов, отредактируйте файл `/etc/tmiscsi/access_lists`, добавив по одной записи для каждого целевого объекта. Формат каждой записи следующий:

     имя_целевого_объекта|*имя_lun имя_iSCSI*, *имя_iSCSI,...*

Информация, связанная с данной:

/etc/tmiscsi/autosecrets

/etc/tmiscsi/access_lists

/etc/tmiscsi/isns_servers

Дополнительная информация об инициаторе ПО iSCSI

Работая с инициаторами ПО iSCSI, примите во внимание следующую информацию.

- Поиск целевого устройства

  ПО инициатора iSCSI поддерживает следующие 4 метода обнаружения целевых устройств:

file Для настройки каждого целевого устройства применяется текстовый файл.

odm Для настройки каждого целевого устройства применяются объекты ODM. Если диск iSCSI используется как загрузочный диск или в процедуре загрузки rootvg, то необходимо использовать метод обнаружения **odm**.

isns Целевые устройства регистрируются на одном или нескольких серверах Internet Storage Name Service (iSNS).

slp Целевые устройства регистрируются в одном или нескольких служебных агентах протокола SLP или агентах каталогов.

- Идентификация iSCSI

  Для настройки идентификации инициатора может применяться только CHAP(MD5). Идентификация целевых устройств не поддерживается.

- Число настроенных LUN

  Максимальное число настроенных LUN для одного целевого устройства iSCSI, тестируемых с помощью программного инициатора iSCSI, равно 128. Инициатор использует одно соединение TCP для каждого целевого устройства iSCSI (одно соединение на сеанс iSCSI). Это соединение TCP является общим для всех LUN, настроенных для целевого устройства. Размер буфера отправки и буфера приема для сокета TCP инициатора устанавливается равным максимальному размеру буфера сокета в системе. Для задания максимально допустимого размера буфера применяется сетевая опция **sb_max**. Значение по умолчанию - 1 MB.

- Группы томов

  Для того чтобы избежать проблем и ошибок при создании групп томов с помощью устройств iSCSI, следуйте приведенным ниже рекомендациям:

  – Настройка групп томов, создаваемых с помощью устройств iSCSI, должна выполняться после перезагрузки, когда они находятся в неактивном состоянии. После настройки устройств iSCSI вручную активируйте группы томов, созданные на основе iSCSI. Затем смонтируйте соответствующие файловые системы.

    Группы томов и программный драйвер iSCSI активируются на разных этапах загрузки. По этой причине активировать группы томов iSCSI в процессе загрузки нельзя.

  – Не размещайте группы томов на устройствах, отличных от iSCSI.

- Ошибки ввода-вывода

  Если связь с целевыми устройствами iSCSI потеряна, происходит ошибка ввода-вывода. Для того чтобы предотвратить такие ошибки ввода-вывода и избежать повреждения файловых систем, перед тем как выполнять действия, которые могут привести к долговременной потере связи с активными целевыми устройствами iSCSI, остановите все операции ввода-вывода и размонтируйте файловые системы на основе iSCSI.

  Если приложения пытаются выполнять операции ввода-вывода с устройствами iSCSI, но при этом связь с целевыми устройствами iSCSI потеряна, то это приведет к возникновению ошибок ввода-вывода. В этом случае размонтировать файловые системы на основе iSCSI будет невозможно, так как используемые ими устройства iSCSI останутся занятыми.

  Если из-за потери связи с активными целевыми устройствами iSCSI возникают ошибки ввода-вывода, необходимо выполнить обслуживание файловых систем. Для этого следует ввести команду **fsck**.

- Не используйте инициатор ПО iSCSI AIX или целевой объект ПО iSCSI AIX с циклическим интерфейсом (`lo0`). Обработка прерываний циклического интерфейса отличается от обработки прерываний сетевого интерфейса физического или виртуального адаптера Ethernet. Работа операционной системы AIX может быть прервана, если циклический интерфейс применяется вместе с драйверами программного обеспечения iSCSI.

Информация, связанная с данной:

Добавление статически обнаруживаемых адресатов iSCSI в ODM

Замечания о защите iSCSI:

Защита каталогов `/etc/iscsi`, `/etc/tmiscsi` и файлов в них осуществляется посредством настройки прав доступа к файлам и принадлежности.

Информация о паролях CHAP хранится в файлах `/etc/iscsi/targets` и `/etc/tmiscsi/autosecrets` в текстовом виде.

Примечание: Не изменяйте исходные права доступа к файлам и принадлежность этих файлов.

Замечания о производительности iSCSI:

Обратите внимание на следующие конфигурации для получения максимальной производительности iSCSI.

Для достижения наилучшей производительности выполните следующие действия:

- Включите следующие опции адаптера AIX Gigabit Ethernet Adapter и интерфейса целевого устройства iSCSI: Большой буфер отправки TCP, Управление потоком исходящих и поступающих данных TCP, Поддержка больших кадров.
- Установите такие значения сетевых опций и параметров интерфейса, чтобы в системе AIX достигалась максимальная производительность ввода-вывода iSCSI.
  - Включите сетевую опцию RFC 1323.
  - Задайте подходящие значения для сетевых опций **tcp_sendspace**, **tcp_recvspace** , **sb_max** и **mtu_size** и опций сетевого интерфейса.

    Максимальный объем передаваемых данных программного инициатора iSCSI равен 256 Кб. Исходя из предположения, что максимальное системное значение для параметров **tcp_sendspace** и **tcp_recvspace** равно 262 144 байтам, команда **ifconfig**, предназначенная для настройки интерфейса Gigabit Ethernet, может выглядеть следующим образом:

    ```
    ifconfig en2 10.1.2.216 mtu 9000 tcp_sendspace 262144 tcp_recvspace 262144
    ```
  - Значение сетевой опции **sb_max** не должно быть меньше 524 288; оптимальное значение равно 1 048 576.
  - Опции **mtu_size** присвойте значение 9000.
  - Для некоторых адресатов iSCSI для повышения производительности следует отключить алгоритм Нэгла TCP. Воспользуйтесь командой **no** чтобы задать для параметра **tcp_nagle_limit** значение `0`, что приведет к отключению алгоритма Нэгла.

Примечание: Информация о настройке сетевых опций приведена в описании команды **no** в книге *Справочник по командам, том 4*.

Дополнительная информация и рекомендации по дополнительной настройке параметров приведены в главе Настройка производительности TCP и UDP.

Дополнительная информация о целевом объекте ПО iSCSI

При определении целевого объекта ПО iSCSI и экспорте номеров логических накопителей (LUN) следует принять во внимание следующее:

- Полное имя iSCSI (IQN) каждого из виртуальных целевых объектов должно быть указано в программе SMIT при определении целевого объекта ПО. Панель SMIT не предусматривает ограничений относительно формата этого имени. Тем не менее, для некоторых инициаторов iSCSI необходимо, чтобы IQN было указано в формате, определенном протоколом iSCSI. Употребление имени в некорректном формате может помешать инициатору войти в систему целевого объекта и получить доступ к дискам, экспортированным целевым объектом.

  Для просмотра текущего имени целевого устройства iSCSI выполните следующие действия:

  1. Запустите команду, аналогичную описанной ниже. В данном примере, допустим, что целевым устройством iSCSI является устройство `target0`.

```
lsattr -E -l target0
```

2. Проверьте атрибут `iscsi_name`.

- В справочных данных, возвращенных для экспортированного номера логического накопителя (LUN), содержатся следующие значения:
  - `ИД вендора: AIX`
  - `ИД продукта: iSCSI_VDASD`
  - `Номер версии ANSI: 3`
- Не используйте инициатор ПО iSCSI AIX или целевой объект ПО iSCSI AIX с циклическим интерфейсом (`lo0`). Обработка прерываний циклического интерфейса отличается от обработки прерываний сетевого интерфейса физического или виртуального адаптера Ethernet. Работа операционной системы AIX может быть прервана, если циклический интерфейс применяется вместе с драйверами программного обеспечения iSCSI.

Протокол управления потоком передачи

Протокол управления потоком передачи (SCTP) - это протокол с установлением соединения, как TCP, но передающий данные сообщениями, как **UDP**. Операционная система AIX поддерживает RFC 4960.

В следующей таблице приведены различия между SCTP и протоколами TCP и UDP.

Таблица 82. Различия между TCP, UDP и SCTP

Атрибут	TCP	UDP	SCTP
Надежность	Надежный	Ненадежный	Надежный
Управление соединением	С установлением соединения	Без установления соединения	С установлением соединения
Передача данных	В виде байтов	в виде сообщений	в виде сообщений
Управление потоком	Да	Нет	Да
Управление нагрузкой	Да	Нет	Да
Устойчивость к сбоям	Нет	Нет	Да
Доставка данных	Строго упорядоченная	Неупорядоченная	Частично упорядоченная
Защита	Да	Да	Улучшенная

В целом **SCTP** предлагает более гибкие возможности для некоторых приложений, таких как **Voice over IP (VoIP)**, которые требуют, чтобы данные передавались надежно, но вместе с тем в виде сообщений. Для этого типа приложений **SCTP** подходит лучше, чем **TCP** или **UDP**.

- **TCP** обеспечивает надежную и строго упорядоченную доставку данных. Использование **TCP** приложениями, требовательными к надежности, но терпимыми к неупорядоченности или частичной упорядоченности данных, может вызывать ненужные задержки из-за объединения данных в блоки. **SCTP** использует принцип "несколько потоков в одном соединении", что обеспечивает строго упорядоченную доставку с логическим разделением потоков.
- **SCTP** ориентирован на сообщения, тогда как **TCP** - на байты. Из-за ориентированной на байты структуры **TCP** приложениям приходится добавлять свои собственные маркировки записей для сохранения границ сообщений.
- **SCTP** обеспечивает некоторую устойчивость к сбоям за счет использования функции множества сетевых адресов. Сервер считается хостом с несколькими сетевыми адресами, когда к нему прикреплено несколько сетевых интерфейсов, находящихся в одной или разных сетях. Между двумя хостами с несколькими сетевыми адресами можно установить связь **SCTP**. В этом случае при запуске соединения происходит обмен всеми IP-адресами обоих конечных точек соединения; это позволяет каждой конечной точке соединения в случае сбоя одного из интерфейсов использовать любой из этих адресов в течение срока действия соединения, при условии, что нужный узел доступен через другие интерфейсы.

- **SCTP** также предлагает дополнительные функции защиты, отсутствующие в **TCP** и **UDP**. В **SCTP** выделение ресурсов во время установки соединения откладывается до тех пор, пока идентификация клиента не будет подтверждена с помощью cookie, что снижает вероятность атак типа Отказ в обслуживании.

Запуск и завершение связи SCTP

Здесь приведена информация о запуске и завершении связи **SCTP**.

Связь **SCTP** состоит из четырехэтапного квитирования, которое происходит в следующем порядке:

1. Клиент отправляет серверу сигнал **INIT**, чтобы инициировать связь.
2. После получения сигнала **INIT** сервер отправляет клиенту ответ **INIT-ACK**. Этот сигнал **INIT-ACK** содержит cookie состояния. Cookie состояния содержит Идентификационный код сообщения (MAC), а также системное время создания cookie, срок действия cookie и информацию, необходимую для установки связи. MAC подсчитывается сервером на основе секретного ключа, известного только ему.
3. После получения этого сигнала **INIT-ACK** клиент отправляет ответ **COOKIE-ECHO**, который просто повторяет cookie состояния.
4. После проверки идентичности cookie состояния с помощью секретного ключа сервер выделяет ресурсы для связи, отправляет ответ **COOKIE-ACK**, подтверждая получение сигнала **COOKIE-ECHO**, и переводит связь в состояние **ESTABLISHED** (установлено).

SCTP также поддерживает контролируемое завершение активной связи по запросу пользователя **SCTP**. Это происходит следующим образом:

1. Клиент отправляет сигнал **SHUTDOWN** серверу, сообщая о своей готовности закрыть соединение.
2. Сервер отвечает, отправляя подтверждение **SHUTDOWN-ACK**.
3. Затем клиент отправляет сигнал **SHUTDOWN-COMPLETE** обратно серверу.

SCTP также поддерживает быстрое закрытие активной связи (сигнал **ABORT**) по требованию клиента **SCTP** или из-за ошибки в стеке **SCTP**. Тем не менее, **SCTP** не поддерживает полуоткрытые соединения. Более подробные сведения об этом протоколе содержатся в RFC 4960.

Кроме указанных выше отличий **SCTP** от существующих транспортных протоколов, в**SCTP** предусмотрены следующие функции:

- **Последовательная доставка в потоках**: потоком в контексте **SCTP** называется последовательность пользовательских сообщений, передаваемых между конечными точками. Связь **SCTP** может поддерживать многопоточность. В ходе установки связи пользователь может указать число потоков. Фактическое значение числа потоков фиксируется после переговоров с другим участником соединения. В каждом потоке строго сохраняется порядок доставки данных. Однако доставка данных в параллельных потоках происходит независимо. Таким образом, потеря данных в одном потоке не мешает передаче данных в других потоках. Это позволяет пользовательским приложениям применять разные потоки для передачи логически независимых данных. Данные можно передавать и неупорядоченно - для этого служит специальная опция. Это может оказаться полезным, если требуется срочная передача данных.
- **Фрагментация пользовательских данных**: **SCTP** может фрагментировать пользовательские сообщения, чтобы размер пакетов, передаваемых на нижний уровень, не превышал MTU пути. После получения фрагменты объединяются в сообщение и передаются пользователю. Несмотря на то, что фрагментация может происходить и на сетевом уровне, фрагментация на транспортном уровне имеет некоторые преимущества перед фрагментацией на IP-уровне. К этим преимуществам относятся отсутствие необходимости повторно отправлять все сообщение целиком при потере отдельных фрагментов в сети и снижение нагрузки на маршрутизаторы, которым в ином случае пришлось бы выполнять IP-фрагментацию.
- **Подтверждение и управление нагрузкой**: для надежной доставки данных необходимо подтверждение получения пакетов. Когда **SCTP** не получает подтверждения доставки отправленного пакета за отведенное время, он посылает этот пакет еще раз. **SCTP** использует практически те же алгоритмы управления

нагрузкой, что и **TCP**. Помимо применения совокупных подтверждений, используемых в **TCP**, **SCTP** использует и выборочные подтверждения (SACK), позволяющие выборочно подтверждать получение пакетов.

- **Комплекты порций**: порция может содержать пользовательские данные или управляющую информацию **SCTP**. Несколько порций могут быть объединены под одним заголовком **SCTP**. Комплекты порций формируются, образуя пакет **SCTP**, отправителем и расформировываются получателем.
- **Проверка пакетов**: каждый пакет **SCTP** содержит поле с тегом проверки, который задается каждой конечной точкой в момент установления связи. Все пакеты отправляются с одним и тем же тегом проверки на протяжении всего срока действия связи. Если в течение срока действия связи поступает пакет с иным тегом проверки, то такой пакет отбрасывается. Кроме того, отправитель должен задать контрольную сумму CRC-32 в каждом пакете **SCTP**, чтобы снизить вероятность повреждения данных при передаче по сети. Все пакеты с неправильной контрольной суммой CRC-32 отбрасываются.
- **Управление путями**: во время установления связи каждая конечная точка может предложить список имеющихся у нее транспортных адресов. Однако для обычной передачи данных в связи **SCTP** указывается только один главный путь. Если использовать главный путь становится невозможно, используются другие транспортные адреса. В течение срока действия связи по всем путям отправляются периодические сигналы, так называемый "пульс", для отслеживания состояния путей.

API сокета SCTP

API сокета **SCTP** обеспечивает согласованность, доступность и совместимость.

API сокетов SCTP предназначены для того, чтобы:

- Поддерживать согласованность с существующими API сокетов
- Предоставлять базу для доступа к новым функциям **SCTP**
- Обеспечивать совместимость, чтобы большинство существующих приложений **TCP** и **UDP** можно было использовать с **SCTP** с незначительными изменениями

Для облегчения переноса имеющихся приложений **TCP** и **UDP** были созданы два различных стиля **SCTP**:

- API типа **UDP** - семантика похожа на используемую в протоколах без установления соединения, таких как **UDP**
- API типа **TCP** - семантика похожа на используемую в протоколах без установления соединения, таких как **TCP**

Несмотря на то, что **SCTP** позволяет определять и использовать и API типа **TCP**, и API типа **UDP**, в AIX 5.3 поддерживается только синтаксис типа **UDP**, поскольку API типа **UDP** предоставляют большую гибкость при доступе к новым функциям **SCTP**. При использовании API типа **UDP** обычный сервер применяет следующую последовательность вызовов в течение срока действия связи:

1. **socket**()
2. **bind**()
3. **listen**()
4. **recvmsg**()
5. **sendmsg**()
6. **close**()

Обычный клиент пользуется следующей последовательностью вызовов API сокетов:

1. **socket**()
2. **sendmsg**()
3. **recvmsg**()
4. **close**()

Связь, созданная с помощью указанной последовательности вызовов, называется созданной явно. Связь может быть создана неявно после создания сокета, простым вызовом **sendmsg**(), **recvmsg**() или **sendto**() и

recvto(). В случае неявной связи вызовы **bind**() и **listen**() необязательны. Синтаксис всех этих системных вызовов схож с тем, который применяется с сокетами **UDP**. Для функций сокета поле **Type** должно быть равно SOCK_SEQPACKET, а поле **Протокол** - IPPROTO_SCTP. Помимо этих стандартных API сокетов, **SCTP** предоставляет два новых API: **sctp_peeloff**() и **sctp_opt_info**(). Подробные сведения об использовании API сокетов для **SCTP** приведены в документе SCTP Socket API Draft. В AIX 5.3 **SCTP** реализован в виде расширения ядра. Для загрузки и выгрузки расширения ядра SCTP служит команда **sctpctrl**.

С помощью этой команды можно также просматривать и изменять различные показатели и параметры расширения ядра **SCTP**, используя другие опции, например get и set. Дополнительная информация о команде **sctpctrl** приведена в описании команды **sctpctrl** в книге *Справочник по командам, том 5*

Функция sctp_bindx:

Добавляет или удаляет адрес связывания сокета.

Библиотека

/usr/lib/libsctp.a

Синтаксис

```
#include <sys/types.h>
#include  <sys/socket.h>
#include <netinet/sctp.h>

int sctp_bindx(int sd, struct sockaddr * addrs, int addrcnt, int flags);
```

Описание

Функция **sctp_bindx** добавляет или удаляет набор адресов связывания, передаваемый в массиве **addrs** сокету **sd** или получаемый от сокета. Параметр **addrcnt** задает число адресов в массиве, а параметр **flags** указывает, требуется ли добавить или удалить адреса.

Если сокет **sd** - это сокет IPv4, то необходимо передавать адреса IPv4. Если сокет **sd** - это сокет IPv6, то можно передавать адреса IPv4 или IPv6.

Параметр **addrs** - это указатель на массив с одним или несколькими адресами сокета. Каждый адрес содержится в соответствующей структуре, **struct sockaddr_in** или **struct sockaddr_in6**. Для определения длины адреса необходимо использовать семейство типов адресов. Вызывающий агент задает число адресов в массиве в параметре **addrcnt**.

Параметр **flags** может принимать значения **SCTP_BINDX_ADD_ADDR** или **SCTP_BINDX_REM_ADDR**. Приложение может использовать SCTP_BINDX_ADD_ADDR для связывания дополнительных адресов с конечной точкой после вызова команды **bind**. Параметр **SCTP_BINDX_REM_ADDR** указывает, что SCTP должен удалить данные адреса из связывания. Вызывающий агент не может удалить все связанные адреса. Такая команда завершается с кодом ошибки **EINVAL**.

Коды возврата

В случае успеха команда **sctp_bindx()** возвращает 0. В случае сбоя команда **sctp_bindx()** возвращает -1, и в параметре **errno** регистрируется соответствующий код ошибки.

Коды ошибок

Ошибка	Описание
EINVAL	Код ошибки **EINVAL** указывает, что порт или адрес недопустим или что команда пытается удалить все связанные адреса.
EOPNOTSUPP	Код **EOPNOTSUPP** указывает, что команда пытается добавить или удалить связанные адреса при наличии установленного соединения.

Функции sctp_getladdrs и sctp_freeladdrs:

Возвращает все локально связанные адреса сокета.

Библиотека

/usr/lib/libsctp.a

Синтаксис

```
#include <sys/types.h>
#include  <sys/socket.h>
#include <netinet/sctp.h>

int sctp_getladdrs(int sd, sctp_assoc_t assoc_id, struct sockaddr **addrs);
void sctp_freeladdrs(struct sockaddr *addrs);
```

Описание

Функция **sctp_getladdrs** возвращает все локально связанные адреса сокета. Возвращаемый параметр **addrs** указывает на динамически размещенный массив структур **sockaddr** соответствующего типа для каждого локального адреса. Для освобождения памяти необходимо использовать параметр **sctp_freeladdrs**.

Примечание: Входной или выходной параметр **addrs** не может быть равен NULL.

Если **sd** - это сокет IPv4, то все возвращаемые адреса имеют тип IPv4. Если **sd** - это сокет IPv6, то возвращаемые адреса могут быть адресами IPv4 или IPv6.

Для сокетов типа one-to-many поле **id** указывает связь с запросом. Для сокетов типа one-to-one поле **id** игнорируется. Если поле **id** равно 0, то локально связанные адреса возвращаются безотносительно к конкретному связыванию.

Функция **sctp_freeladdrs** освобождает все ресурсы, размещенные функцией **sctp_getladdrs**.

Возвращаемое значение

В случае успеха функция **sctp_getladdrs** возвращает число локальных адресов, связанных с сокетом. Если сокет не связан, то возвращается значение 0, при этом значение поля ***addrs** не определено. В случае ошибки функция **sctp_getladdrs** возвращает значение -1, и значение поля ***addrs** не определено.

Функции sctp_getpaddrs и sctp_freepaddrs:

Возвращает все адреса партнеров в ассоциации.

Библиотека

/usr/lib/libsctp.a

Синтаксис

```
#include <sys/types.h>
#include  <sys/socket.h>
#include <netinet/sctp.h>

int sctp_getpaddrs(int sd, sctp_assoc_t assoc_id, struct sockaddr **addrs);
void sctp_freepaddrs(struct sockaddr *addrs);
```

Описание

Функция **sctp_getpaddrs** возвращает все адреса партнеров в ассоциации. Возвращаемый параметр **addrs** указывает на динамически размещенный массив структур **sockaddr** соответствующего типа для каждого адреса. Для освобождения памяти необходимо использовать функцию **sctp_freepaddrs**.

Примечание: Входной или выходной параметр **addrs** не может быть равен NULL.

Если **sd** - это сокет IPv4, то все возвращаемые адреса имеют тип IPv4. Если **sd** - это сокет IPv6, то возвращаемые адреса могут быть адресами IPv4 или IPv6. Для сокетов типа one-to-many поле **id** указывает связь с запросом. Для сокетов типа one-to-one поле **id** игнорируется.

Функция **sctp_freepaddrs** освобождает все ресурсы, размещенные функцией **sctp_getpaddrs**.

Возвращаемое значение

В случае успеха функция **sctp_getpaddrs** возвращает число адресов партнеров, связанных с сокетом. Если сокет не связан, то возвращается значение 0, при этом значение поля ***addrs** не определено. В случае ошибки функция **sctp_getpaddrs** возвращает значение -1, и значение поля ***addrs** не определено.

Вычисление MTU маршрута

Если маршрут, связывающий два хоста, проходит через несколько сетей, и размер передаваемого пакета превышает наименьшее значение MTU этих сетей, то пакет разбивается на фрагменты. Так как фрагментация пакетов может снизить производительности сети, ее желательно избегать путем передачи пакетов, размер которых не превышает значений MTU на протяжении всего маршрута. Этот размер называется MTU маршрута.

Операционная система поддерживает алгоритм вычисления MTU маршрута, описанный в RFC 1191. Включить вычисление маршрута MTU в приложениях **TCP** и **UDP** можно изменив опции **tcp_pmtu_discover** и **udp_pmtu_discover** команды **no**. Если применение этого алгоритма включено, то размер всех пакетов, передаваемых приложениями **TCP**, не будет превышать MTU маршрута. Поскольку приложения **UDP** сами определяют размер передаваемых пакетов, они должны получать информацию об MTU маршрута с помощью опции **IP_FINDPMTU** сокета, даже если включена опция **udp_pmtu_discover** команды **no**. По умолчанию параметры **tcp_pmtu_discover** и **udp_pmtu_discover** включены.

При попытке вычисления MTU маршрута в таблице MTU маршрут (PMTU) создается запись pmtu. Эту таблицу можно посмотреть с помощью команды **pmtu**. Можно избежать создания большого числа записей pmtu, ограничив время существования неиспользуемых записей и разрешив их удаление. Время существования записи PMTU можно задать с помощью опции pmtu_expire команды **no**. По умолчанию значение pmtu_expire равно 10 минутам.

Значение MTU маршрута может динамически изменяться вместе с самим маршрутом. Уменьшение значения MTU маршрута приводит к фрагментации пакетов, поэтому периодически должна проводиться проверка этого значения. По умолчанию значение MTU маршрута проверяется каждые 10 минут. Периодичность проверки можно изменить, указав необходимое значение в параметре **pmtu_default_age** команды **no**.

Приложения **UDP** всегда требуют включения опции сокета **IP_DONTFRAG** для обнаружения уменьшения PMTU. Это включает немедленное обнаружение снижения MTU маршрута, а не проверку его снижения каждые pmtu_default_age минут.

При увеличении MTU маршрута появляется возможность повышения производительности сети, поэтому периодически выполняется проверка возможности увеличения MTU маршрута. По умолчанию значение MTU маршрута проверяется каждые 30 минут. Периодичность проверки можно изменить, указав необходимое значение в параметре **pmtu_rediscover_interval** команды **no**.

Поскольку не все маршрутизаторы в сети поддерживают RFC 1191, иногда невозможно определить точное значение MTU маршрута. В таких случаях для добавления или удаления значений MTU маршрута можно использовать команду **mmtu**.

Примечание:

1. Алгоритм вычисления MTU маршрута неприменим для одинаковых маршрутов, в том числе тех, которые созданы для групповой маршрутизации (обратитесь к разделу “Ограничения применения маршрутизации” на стр. 366). Вычисление MTU маршрута можно использовать на дублирующих маршрутах.
2. При включении функции вычисления MTU маршрута параметру **arpqsize** команды **no** присваивается значение не меньше 5. Это значение не уменьшается после выключения функции вычисления MTU маршрута.

Quality of Service TCP/IP

Quality of Service (QoS) - это семейство развивающихся стандартов Internet по специальной обработке передаваемых по Internet данных определенного типа.

Наличие поддержки QoS вдоль всего маршрута позволяет снизить потери производительности, вызванные различными задержками в очередях и перегрузками. В данной операционной системе предусмотрена поддержка QoS для разделения всех исходящих данных на классы обслуживания и резервирования ресурсов по запросу клиентского приложения.

QoS можно применять в организациях для внедрения и развития стратегий управления пропускной способностью сети. Средства QoS позволяют хосту:

- Регулировать объем данных определенного типа, поступающих в сеть;
- Помечать выбранные пакеты в соответствии с установленной стратегией, чтобы последующие маршрутизаторы могли предоставлять нужный уровень обслуживания;
- Поддерживать такие службы, как служба виртуальной выделенной линии, и обеспечивать должный уровень поддержки QoS вдоль маршрута;
- Участвовать в обработке запросов на резервирование ресурсов, поступающих от получателей, и объявлять о наличии сеансов отправителя, которые могут выполнить такое резервирование.

QoS предоставляет следующие функции:

- Дифференцированные службы, согласно определению в RFC 2474
- Стратегии управления потоком данных
- Подразделение пакетов на удовлетворяющие и не удовлетворяющие заданным критериям обслуживания
- Изменение параметров передаваемых данных
- Измерения
- Интегрированные службы для приложений клиента и сервера, согласно определению в RFC 1633
- Сигнализация RSVP (RFC 2205)
- Гарантийная служба (RFC 2212)
- Служба управляемой загрузки (RFC 2211)

- Управление сетью на основе стратегий
- Общая библиотека RAPI для приложения

Подсистема QoS состоит из четырех компонентов:

Расширение ядра QoS (/usr/lib/drivers/qos)
Расширение ядра QoS находится в каталоге /usr/lib/drivers/qos. Загрузка и выгрузка его из памяти осуществляется с помощью методов настройки **cfgqos** и **ucfgqos**. Это расширение ядра обеспечивает поддержку QoS.

Агент стратегии (/usr/sbin/policyd)
Агент стратегии - это демон пользовательского уровня, находящийся в каталоге /usr/sbin/policyd. Этот демон обеспечивает поддержку управления стратегиями и интерфейсов с расширением ядра QoS, необходимую для задания, изменения и удаления правил стратегии. Правила стратегии можно определить в локальном файле конфигурации (/etc/policyd.conf) или загрузить с центрального сервера стратегии управления сетью с помощью LDAP; возможно также сочетание этих способов.

Агент RSVP (/usr/sbin/rsvpd)
Агент RSVP - это демон пользовательского уровня, находящийся в каталоге /usr/sbin/rsvpd. Этот демон реализует семантику протокола сигнализации RSVP.

Общая библиотека RAPI (/usr/lib/librapi.a)
С помощью API RSVP (RAPI) приложения могут запрашивать более высокий уровень обслуживания, определенный в модели QoS для интегрированных служб сети Internet. Эта библиотека совместно с локальным агентом RSVP отправляет запросы QoS по маршруту передачи данных с помощью протокола RSVP. Этот API является открытым стандартом.

Примечание: Данная реализация QoS основана на наборе стандартов и проектов стандартов Internet, созданных Рабочей группой Internet (IETF) и ее различными подгруппами. По мере развития стандартов эта технология будет становиться более полной и более тщательно определенной. Важно отметить также, что QoS - это сравнительно молодая технология, которая только-только начинает внедряться в Internet. QoS предоставляет разработчику множество преимуществ на всех стадиях разработки. Однако в полной мере эти преимущества можно реализовать только в том случае, если QoS поддерживается на всем отдельно взятом маршруте.

Модели QoS

Модели QoS для Internet являются открытыми стандартами, разработанными IETF.

В настоящее время IETF приняла два стандарта моделей QoS для Internet: *интегрированные службы* и *дифференцированные службы*. Эти две Internet-модели QoS дополняют и улучшают традиционную модель службы, описанную в RFC 1812.

Интегрированные службы:

Интегрированные службы (IS) представляют собой динамическую модель резервирования ресурсов для Internet, описанную в RFC 1633.

С помощью протокола сигнализации, называемого Протоколом резервирования ресурсов (RSVP), хосты динамически запрашивают конкретный уровень качества обслуживания в сети. Параметры QoS передаются в сообщениях RSVP, так что каждый узел сети вдоль маршрута устанавливает параметры, необходимые для достижения требуемого качества обслуживания. Эти параметры QoS описывают одну или две определенных в настоящее время службы, а именно гарантийную службу и службу управляемой загрузки. Важная особенность интегрированных служб (IS) заключается в том, что сигнализация выполняется для каждого потока данных, а резервы создаются на каждом транзитном участке вдоль маршрута. Хотя эта модель хорошо подходит для обслуживания тех приложений, требования которых динамически изменяются, по ряду соображений ее нельзя реализовать в сети, в которой отдельные маршрутизаторы обрабатывают несколько потоков данных.

Дифференцированные службы:

В Дифференцированных службах (DS) упразднены инструменты масштабируемости для потока данных и для транзитных участков, а взамен реализован упрощенный механизм классификации пакетов.

Для распределения пакетов по классам в DS применяется не динамическая сигнализация, а биты в байте типа обслуживания IP (TOS). Конкретный битовый шаблон байта IP TOS называется кодовым набором DS; с его помощью маршрутизаторы определяют качество обслуживания, предоставляемое на данном конкретном транзитном участке (практически тем же способом маршрутизаторы выполняют пересылку IP с помощью поиска в таблицах маршрутизации). Способ обработки пакета в соответствии с данным кодовым набором DS называется способом обработки на транзитном участке (PHB); этот способ задается отдельно на каждом узле сети. Совокупность всех этих независимых друг от друга PHB обеспечивает обслуживание на всем маршруте.

Рабочая группа IETF, занимающаяся разработкой стандартов для дифференцированных служб, определила три вида PHB: PHB с ускоренной пересылкой (EF), группа PHB с надежной пересылкой (AF) и PHB по умолчанию (DE). PHB EF предназначен для передачи с низким временем ожидания, низкой вероятностью потери данных и высокой защищенностью. Примером может служить виртуальная выделенная линия (VLL). AF представляет собой семейство PHB, называемое группой PHB. PHB этой группы применяются для распределения пакетов по уровням приоритета. Уровень приоритета, присвоенный пакету, определяет относительную важность этого пакета в классе AF. PHB этой группы можно использовать для организации обслуживания типа *Olympic*, состоящего из трех классов: bronze, silver и gold (т.е. бронзового, серебряного и золотого). PHB DE - это традиционная модель оптимального обслуживания, стандарт которого описан в RFC 1812.

Поддерживаемые стандарты и проекты стандартов

Ниже перечислены RFC и проекты стандартов Internet, в которых описаны спецификации, на которых основана данная реализация QoS:

Элемент	Описание
RFC 2474	Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers
RFC 2475	An Architecture for Differentiated Services
RFC 1633	Integrated Services in the Internet Architecture: an Overview
RFC 2205	Resource ReSerVation Protocol (RSVP)
RFC 2210	The Use of RSVP with IETF Integrated Services
RFC 2211	Specification of the Controlled-Load Network Element Service
RFC 2212	Specification of Guaranteed Quality of Service
RFC 2215	General Characterization Parameters for Integrated Service Network Elements

Элемент	Описание
draft-ietf-diffserv-framework-01.txt, октябрь 1998 года	A Framework for Differentiated Services
draft-ietf-diffserv-rsvp-01.txt, ноябрь 1998 года	A Framework for Use of RSVP with DIFF-serv Networks
draft-ietf-diffserv-phb-ef-01.txt	An Expedited Forwarding PHB
draft-ietf-diffserv-af-04.txt	Assured Forwarding PHB Group
draft-rajan-policy-qosschema-00.txt, октябрь 1998 года	Schema for Differentiated Services and Integrated Services in Networks
draft-ietf-rap-framework-01.txt, ноябрь 1998 года	A Framework for Policy-based Admission Control[25]
draft-ietf-rap-rsvp-ext-01.txt, ноябрь 1998 года	RSVP Extensions for Policy Control

Примечание: QoS - это новая технология Internet. QoS предоставляет разработчику множество преимуществ на всех стадиях разработки. Однако в полной мере эти преимущества можно реализовать только в том случае, если QoS поддерживается на всем отдельно взятом маршруте.

Установка QoS

QoS поставляется в наборе файлов `bos.net.tcp.server`. Для работы с QoS необходимо установить этот набор файлов.

Для работы с общей библиотекой `RAPI` нужно дополнительно установить набор файлов `bos.adt.include`.

Запуск и остановка подсистемы QoS

Для запуска и завершения работы подсистемы QoS можно воспользоваться командой SMIT `smit qos` или командами **mkqos** и **rmqos**.

1. Для завершения работы подсистемы QoS сейчас и после следующего запуска системы введите команду:
   ```
   /usr/sbin/rmqos -B
   ```
2. Для запуска подсистемы QoS только сейчас введите команду:
   ```
   /usr/sbin/mkqos -N
   ```

Флаги запуска и завершения работы подсистемы перечислены в описаниях команд **mkqos** и **rmqos**.

Для настройки демонов **policyd** и **rsvpd** применяются файлы конфигурации `/etc/policyd.conf` и `/etc/rsvpd.conf`, соответственно. Для настройки подсистемы QoS в конкретной локальной среде *необходимо* отредактировать эти файлы. Если вы не измените конфигурацию по умолчанию, подсистема QoS не будет работать правильно.

Настройка агента RSVP

Агент RSVP необходим, если хост должен поддерживать протокол RSVP.

Для настройки агента RSVP служит файл конфигурации `/etc/rsvpd.conf`, синтаксис которого описан в примере файла конфигурации `/etc/rsvpd.conf`.

Ниже приведен пример конфигурации RSVP для хоста, в котором создано 4 виртуальных или физических интерфейса, соответствующих четырем IP-адресам: `1.2.3.1`, `1.2.3.2`, `1.2.3.3`, и `1.2.3.4`.

```
interface 1.2.3.1
interface 1.2.3.2 disabled
interface 1.2.3.3 disabled
interface 1.2.3.4
{
  trafficControl
}

rsvp 1.2.3.1
{
  maxFlows 64
}

rsvp 1.2.3.4
{
  maxFlows 100
}
```

Интерфейс `1.2.3.1` подключен для протокола RSVP. Однако поскольку опция управления потоком не задана, при получении сообщений RSVP RESV в подсистеме TCP не будут резервироваться ресурсы. Этот интерфейс может поддерживать до 64 одновременных сеансов RSVP.

Интерфейсы `1.2.3.2` и `1.2.3.3` отключены. Агент RSVP не может отправлять или принимать сообщения RSVP через эти интерфейсы.

Интерфейс `1.2.3.4` подключен для протокола RSVP. Кроме того, через этот интерфейс можно резервировать ресурсы в подсистеме **TCP** с помощью сообщений RSVP RESV. Данный интерфейс поддерживает до 100 сеансов RSVP.

Все остальные интерфейсы хоста, явно не указанные в файле /etc/rsvpd.conf, выключены.

Настройка агента стратегии

Агент стратегии - это обязательный компонент подсистемы QoS.

Для его настройки служит файл конфигурации /etc/policyd.conf. Формат этого файла описан в примере файла конфигурации, /etc/policyd.conf.

Агент стратегии можно настроить путем редактирования файла /etc/policyd.conf. Кроме того, для настройки стратегий предусмотрены следующие команды:

- **qosadd**
- **qosmod**
- **qoslist**
- **qosremove**

В приведенном ниже примере создается категория службы premium, которая применяется в правиле стратегии tcptraffic. Характеристики этой категории службы следующие: максимальная скорость передачи данных - 110000 Кбит/с, размер стека маркеров - 10000 бит, значение TOS для исходящих пакетов IP - 11100000 в двоичной системе. По правилу стратегии tcptraffic обслуживание класса premium предоставляется всем данным с IP-адресом отправителя 1.2.3.6, IP-адресом получателя 1.2.3.3 и номером порта получателя в диапазоне от 0 до 1024.

```
ServiceCategories  premium
{
   PolicyScope    DataTraffic
   MaxRate         110000
   MaxTokenBucket  10000
   OutgoingTOS     11100000
}

ServicePolicyRules   tcptraffic
{
   PolicyScope    DataTraffic
   ProtocolNumber 6  # tcp
   SourceAddressRange      1.2.3.6-1.2.3.6
   DestinationAddressRange 1.2.3.3-1.2.3.3
   DestinationPortRange    0-1024
   ServiceReference        premium
}
```

В следующем примере показана настройка категории обслуживания по умолчанию (default). Эта категория будет применяться для ограничения потока данных UDP, поступающего через интерфейсы 1.2.3.1-1.2.3.4 на IP-адреса 1.2.3.6-1.2.3.10, порт 8000.

```
ServiceCategories  default
{
   MaxRate         110000
   MaxTokenBucket  10000
   OutgoingTOS     00000000
}

ServicePolicyRules   udptraffic
{
   ProtocolNumber 17  # udp
   SourceAddressRange      1.2.3.1-1.2.3.4
   DestinationAddressRange 1.2.3.6-1.2.3.10
   DestinationPortRange    8000-8000
   ServiceReference        default
}
```

Приведенный ниже пример конфигурации можно применять для загрузки правил с сервера LDAP по отличительному имени поддерева, а также для поиска стратегий на хосте сервера LDAP.

```
    ReadFromDirectory
    {
      LDAP_Server      1.2.3.27
      Base             ou=NetworkPolicies,o=myhost.mydomain.com,c=us
    }
```

Устранение неполадок QoS

С помощью команды **qosstat** можно просмотреть информацию о состоянии установленных и активных стратегий в подсистеме QoS. Эта информация поможет локализовать неполадку при анализе конфигурации QoS.

Команда **qosstat** выдает отчет примерно следующего вида:

```
Действие:
  Быстродействие стека маркеров (бит/с): 10240
  Размер стека маркеров (бит): 1024
  Пиковое быстродействие (бит/с): 10240
  Мин. размер блока для стратегии (бит): 20
  Макс. размер пакета (бит): 1452
  Тип: IS-CL
  Флаги: 0x00001001 (POLICE,SHAPE)

  Статистика:
    Согласованных пакетов: 1423 (440538 байт)

  Параметры:
    Адрес отправителя     Адрес получателя      Протокол
    192.168.127.39:8000   192.168.256.29:35049  tcp       (1 соединение)

Действие:
  Быстродействие стека маркеров (бит/с): 10240
  Размер стека маркеров (бит): 1024
  Пиковое быстродействие (бит/с): 10240
  Исходящий TOS (согласованный): 0xc0
  Исходящий TOS (несогласованный): 0x00
  Флаги: 0x00001011 (POLICE,MARK)
  Тип: DS

  Статистика:
    Согласованных пакетов: 335172 (20721355 байт)
    Несогласованных пакетов: 5629 (187719 байт)

  Параметры:
    Адрес отправителя     Адрес получателя      Протокол
    192.168.127.39:80     *:*                   tcp       (1 соединение)
    192.168.127.40:80     *:*                   tcp       (5 соединений)
```

Спецификация стратегии QoS

В этом разделе описаны классы объектов и атрибуты, применяемые агентом стратегий для описания стратегий QoS, управляющих отправкой данных.

После описания классов и атрибутов приведены указания по их применению и настройке.

В данном разделе применяются следующие обозначения.

```
     p  : один из допустимых параметров
     B  : целое значение размером в один байт (т.е. 0 =< B =< 255)
     b  : строка битов, начинающаяся с самого левого бита  (т.е. 101 - это
          10100000 в поле размером в байт)
     i  : целое значение
     s  : символьная строка
     a  : IP-адрес в формате B.B.B.B
    (R) : обязательный параметр
    (O) : необязательный параметр
```

Определение ReadFromDirectory:

Это определение задает параметры для установления сеанса LDAP.

Определение ReadFromDirectory применяется в файле /etc/policyd.conf для создания сеанса LDAP.

```
ReadFromDirectory
{
  LDAP_Server    a   # IP-адрес сервера каталогов LDAP
  LDAP_Port     i    # Номер порта на сервере LDAP
  Base     s         # Отличительное имя
  LDAP_SelectedTag s # Тег для сравнения с SelectorTag в классе объектов
}
```

где

```
LDAP_Server (R): IP-адрес сервера LDAP
LDAP_Port   (0): Уникальный номер порта; значение по умолчанию - 389
Base        (R): Пример: o=ibm, c=us, где o - организация, a c - страна
LDAP_SelectedTag (R): Уникальная строка для сравнения с атрибутом SelectorTag из класса объектов
```

Определение ServiceCategories:

Это определение задает тип обслуживания, который должен обеспечиваться при передаче потока IP-пакетов (например, соединение **TCP** или данные **UDP**) по сети.

Можно задать несколько определений ServiceCategories с различными именами, для того чтобы ссылаться на них в будущем. Для того чтобы определение стратегии было задано полностью, помимо объекта ServiceCategories необходимо задать объект ServicePolicyRules.

```
ServiceCategories   s
{
   SelectorTag     s     # Обязательный тег для поиска LDAP
   MaxRate        i     # Целевая скорость передачи данных для этого класса обслуживания
   MaxTokenBucket i     # Размер набора маркеров
   OutgoingTOS    b     # Значение TOS для отправляемых данных в этом классе обслуживания
   FlowServiceType   p  # Тип потока данных

}
```

где

```
s           (R)     : имя категории обслуживания
SelectorTag (R)     : Тег, необходимый LDAP для выбора классов объектов
MaxRate     (O)     : скорость в Кбит/с, значение по умолчанию - 0
MaxTokenBucket(O)   : размер в Кб; значение по умолчанию совпадает с системным максимумом
OutgoingTOS (O)     : значение по умолчанию - 0
FlowServiceType  (O): ControlledLoad | Guaranteed, по умолчанию - ControlledLoad
```

Определение ServicePolicyRules:

Это определение задает свойства IP-пакетов, относящихся к определенной категории обслуживания.

Другими словами, оно описывает набор IP-дейтаграмм, при передаче которых должен обеспечиваться определенный уровень обслуживания. Объект ServicePolicyRules ссылается на объект ServiceCategories при помощи атрибута ServiceReference. Если два правила относятся к одному классу ServiceCategory, для каждого из них создается свой экземпляр объекта ServiceCategory.

```
ServicePolicyRules   s
{
   SelectorTag     s     # Обязательный тег для поиска LDAP
   ProtocolNumber    i   # ИД транспортного протокола для правила стратегии
   SourceAddressRange      a1-a2
   DestinationAddressRange a1-a2
   SourcePortRange         i1-i2
```

```
    DestinationPortRange    i1-i2
    PolicyRulePriority  i       # Первым обрабатывается элемент с самым высоким приоритетом
    ServiceReference        s  # Имя категории обслуживания для данного правила
}
```

где

```
s           (R): имя правила стратегии
SelectorTag (R): требуется LDAP для поиска класса объектов
ProtocolNumber     (R): значение по умолчанию - 0, оно не соответствует никакому протоколу
SourceAddressRange (O): от a1 до a2, где a2 >= a1, по умолч. - 0, доп. любой исх. адрес
SourcePortRange    (O): от i1 до i2, где i2 >= i1, по умолч. - 0, доп. любой исх. порт
DestinationAddressRange (O): то же, что и SourceAddressRange
DestinationPortRange    (O): то же, что и SourcePortRange
PolicyRulePriority (O): это значение необходимо указать при наличии аналогичной стратегии
ServiceReference   (R): категория обслуживания для данного правила
```

Руководство по среде DiffServ

Ниже приведены указания по определению стратегий в среде DiffServ.

1. **Только маркировка**

   ```
   OutgoingTOS  : Требуемый тип обслуживания
   FlowServiceType : ControlledLoad
   MaxRate         : Значение по умолчанию - 0
   ```

2. **Только формирование**

   ```
   OutgoingTOS  : Значение по умолчанию - 0
   FlowServiceType : Guaranteed
   MaxRate         : Скорость передачи потока данных (положительное целое число)
   ```

3. **Маркировка и ограничение (см. примечание)**

   ```
   OutgoingTOS     : Требуемый тип обслуживания
   FlowServiceType : ControlledLoad
   MaxRate         : Скорость передачи потока данных (положительное целое число)
   ```

4. **Маркировка и формирование**

   ```
   OutgoingTOS     : Требуемый тип обслуживания
   FlowServiceType : Guaranteed
   MaxRate         : Скорость передачи потока данных (положительное целое число)
   ```

Примечание: В данной стратегии тип обслуживания, заданный для внепрофильных пакетов, равен нулю.

Пример файла конфигурации policyd

Ниже приведен пример файла конфигурации `/etc/policyd.conf`.

```
#loglevel  511    # Подробные сообщения

######################################################################
#
# Пометить данные rsh, отправляемые через исходные порты TCP 513 и 514.
ServiceCategories       tcp_513_514_svc
{
        MaxRate                 0               # Только маркировка
        OutgoingTOS             00011100  # двоичное
        FlowServiceType         ControlledLoad
}

ServicePolicyRules      tcp_513_514_flt
{
        ProtocolNumber          6  # TCP
        SourceAddressRange      0.0.0.0-0.0.0.0 # Любой исходный IP-адрес
        DestinationAddressRange 0.0.0.0-0.0.0.0 # Любой целевой IP-адрес
        SourcePortRange         513-514
        DestinationPortRange    0-0             # Любой целевой порт
        ServiceReference        tcp_513_514_svc
}
#
```

```
######################################################################
#
# Формировать данные UDP, отправляемые через порт 9000.
ServiceCategories       udp_9000_svc
{
        MaxRate                 8192     # килобит
        MaxTokenBucket          64       # килобит
        FlowServiceType         Guaranteed
}

ServicePolicyRules      udp_9000_flt
{
        ProtocolNumber          17  # UDP
        SourceAddressRange      0.0.0.0-0.0.0.0 # Любой исходный IP-адрес
        DestinationAddressRange 0.0.0.0-0.0.0.0 # Любой целевой IP-адрес
        SourcePortRange         9000-9000
        DestinationPortRange    0-0             # Любой целевой порт
        ServiceReference        udp_9000_svc
}
#
######################################################################
#
# Маркировать и ограничивать поток данных FUNGER, отправляемых через исходный порт TCP 79.
ServiceCategories       tcp_79_svc
{
        MaxRate                 8         # килобит
        MaxTokenBucket          32        # килобит
        OutgoingTOS             00011100  # двоичное
        FlowServiceType         ControlledLoad
}

ServicePolicyRules      tcp_79_flt
{
        ProtocolNumber          6  # TCP
        SourceAddressRange      0.0.0.0-0.0.0.0 # Любой исходный IP-адрес
        DestinationAddressRange 0.0.0.0-0.0.0.0 # Любой целевой IP-адрес
        SourcePortRange         79-79
        DestinationPortRange    0-0             # Любой целевой порт
        ServiceReference        tcp_79_svc
}
#
######################################################################
#
# Маркировать и формировать данные FTP, отправляемые через исходный порт TCP 20.
ServiceCategories       tcp_20_svc
{
        MaxRate                 81920        # килобит
        MaxTokenBucket          128          # килобит
        OutgoingTOS             00011101     # двоичное
        FlowServiceType         Guaranteed
}

ServicePolicyRules      tcp_20_flt
{
        ProtocolNumber          6  # TCP
        SourceAddressRange      0.0.0.0-0.0.0.0 # Любой исходный IP-адрес
        DestinationAddressRange 0.0.0.0-0.0.0.0 # Любой целевой IP-адрес
        SourcePortRange         20-20
        DestinationPortRange    0-0             # Любой целевой порт
        ServiceReference        tcp_20_svc
}
#
######################################################################
#
# Запись сервера LDAP.
#ReadFromDirectory
#{
```

```
#   LDAP_Server             9.3.33.138  # IP-адрес сервера LDAP
#   Base                    o=ibm,c=us  # Основное отличительное имя
#   LDAP_SelectedTag        myhost      # Имя хоста клиента
#}
#
####################################################################
```

Загрузка стратегий Сервера каталогов IBM SecureWay

Если демон стратегий применяется вместе с сервером каталогов LDAP IBM SecureWay, то перед запуском сервера LDAP обновите файл `/etc/ldapschema/V3.modifiedschema`, руководствуясь приведенной ниже схемой.

Более подробная информация приведена в разделе “Планирование и настройка преобразования имен LDAP (Схема IBM SecureWay Directory)” на стр. 202.

```
objectClasses {
( ServiceCategories-OID NAME 'ServiceCategories' SUP top MUST
( objectClass $ SelectorTag $ serviceName ) MAY
( description $ FlowServiceType $ MaxRate $ MaxTokenBucket $ OutgoingTos ) )
( ServicePolicyRules-OID NAME 'ServicePolicyRules' SUP top MUST
( objectClass $ PolicyName $ SelectorTag ) MAY
( description $ DestinationAddressRange $ DestinationPortRange $
ProtocolNumber $ ServiceReference $ SourceAddressRange $ SourcePortRange ) )
}
attributeTypes {
( DestinationAddressRange-OID NAME 'DestinationAddressRange' SYNTAX
1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )
( DestinationPortRange-OID NAME 'DestinationPortRange' SYNTAX
 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )
( FlowServiceType-OID NAME 'FlowServiceType'
 SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )
( MaxRate-OID NAME 'MaxRate' SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )
( MaxTokenBucket-OID NAME 'MaxTokenBucket' SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )
( OutgoingTos-OID NAME 'OutgoingTos' SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )
( PolicyName-OID NAME 'PolicyName' SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )
( ProtocolNumber-OID NAME 'ProtocolNumber' SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )
( SelectorTag-OID NAME 'SelectorTag' SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )
( ServiceReference-OID NAME 'ServiceReference' SYNTAX
 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )
( SourceAddressRange-OID NAME 'SourceAddressRange' SYNTAX
 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )
( SourcePortRange-OID NAME 'SourcePortRange' SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE )
}

IBMattributeTypes {
( DestinationAddressRange-OID DBNAME ( 'DestinationAddressRange' 'DestinationAddressRange' ) )
( DestinationPortRange-OID DBNAME ( 'DestinationPortRange' 'DestinationPortRange' ) )
( FlowServiceType-OID DBNAME ( 'FlowServiceType' 'FlowServiceType' ) )
( MaxRate-OID DBNAME ( 'MaxRate' 'MaxRate' ) )
( MaxTokenBucket-OID DBNAME ( 'MaxTokenBucket' 'MaxTokenBucket' ) )
( OutgoingTos-OID DBNAME ( 'OutgoingTos' 'OutgoingTos' ) )
( PolicyName-OID DBNAME ( 'PolicyName' 'PolicyName' ) )
( ProtocolNumber-OID DBNAME ( 'ProtocolNumber' 'ProtocolNumber' ) )
( SelectorTag-OID DBNAME ( 'SelectorTag' 'SelectorTag' ) )
( ServiceReference-OID DBNAME ( 'ServiceReference' 'ServiceReference' ) )
( SourceAddressRange-OID DBNAME ( 'SourceAddressRange' 'SourceAddressRange' ) )
( SourcePortRange-OID DBNAME ( 'SourcePortRange' 'SourcePortRange' ) )
}

ldapSyntaxes {
}

matchingRules {
}
```

Настройка системы QoS

Перекрывающиеся стратегии устанавливаются Администратором QoS в произвольном порядке. При наличии перекрывающихся стратегий нужно задать порядок применения стратегий с помощью атрибутов `PolicyRulePriority` объектов `ServicePolicyRules` . Атрибуту `PolicyRulePriority` присваивается целое значение. В случае перекрывающихся стратегий выбирается правило с наибольшим значением атрибута.

QoS поддерживает только сокеты **UDP**, между которыми установлено соединение.

Стратегии и агенты RSVP применяются независимо друг от друга. Поэтому не следует указывать стратегии, конфликтующие с резервированием RSVP или покрываемые им. При наличии таких конфликтов система применяет первую стратегию или резервирование, а остальные помечает как недопустимые.

Значение атрибута `MaxTokenBucket` должно быть не меньше максимального значения MTU для всех интерфейсов, настроенных в системе.

Стратегии автоматически изменяются агентом путем удаления старых стратегий и установки новых. При выполнении этой процедуры поток данных, связанный с изменяемой стратегией, может какое-то время обслуживаться на уровне, установленном по умолчанию (обычно это оптимальный уровень обслуживания).

Соответствие стандартам IETF для моделей IntServ и DiffServ

Этот выпуск совместим с текущими стандартами Internet Engineering Task Force (IETF) для дифференцированных (DiffServ) и интегрированных (IntServ) служб Internet.

Компоненты модели IntServ описаны в следующих документах RFC:

- The Use of RSVP with IETF Integrated Services (RFC 2210)
- Specification of the Controlled-Load Network Element Service (RFC 2211)
- Specification of Guaranteed Quality of Service (RFC 2212)

Компоненты модели DiffServ описаны в следующих документах RFC:

- Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers (RFC 2474)
- An Architecture for Differentiated Services (RFC 2475)

Текущий формат октета IP TOS описан в следующем RFC:

- Type of Service in the Internet Protocol Suite (RFC 1349)

Планируемый в будущем формат октета IP TOS описан в следующих RFC:

- Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers (RFC 2474)
- Assured Forwarding PHB Group (RFC 2597)
- An Expedited Forwarding PHB (RFC 2598)

Поддержка IPv6

QoS поддерживает только протокол IPv4. IPv6 не поддерживается.

Управление демоном стратегии

Демоном стратегий можно управлять с помощью Контроллера системных ресурсов (SRC).

Например, команда:

```
startsrc -s policyd -a "-i 60"
```

запускает агент стратегий с интервалом обновления в 60 секунд.

Команда

```
stopsrc -s policyd
```

останавливает демон стратегий.

Примечание: При остановке демона установленные в ядре стратегии не удаляются. При повторном запуске демона старые стратегии (установленные до этого в ядре) удаляются, и вместо них устанавливаются стратегии, определенные в файле `/etc/policyd.conf`.

Команда SRC **refresh** в данный момент не поддерживается.

Команды и методы QoS

ЗДесь прведены сведения о командах и методах QoS **TCP/IP**.

Самая свежая информация по данному вопросу приведена в файле README, расположенном в каталоге `/usr/samples/tcpip/qos`.

Поддерживаются следующие команды QoS:

- **qosadd**
- **qoslist**
- **qosmod**
- **qosremove**
- **qosstat**
- **mkqos**
- **rmqos**

Поедрживаются следующие методы QoS:

- **cfgqos**
- **ucfgqos**

Устранение неполадок TCP/IP

Многие неполадки в работе **Протокола управления передачей/Протокола Internet (TCP/IP)** можно локализовать с помощью команды **netstat**.

Многие неполадки в работе сети можно локализовать с помощью команды **netstat**. После локализации неполадки с помощью этой команды вы можете перейти к более сложным диагностическим средствам. Например, помощью команд **netstat -i** и **netstat -v** вы можете определить, что неполадка связана с аппаратным интерфейсом, а затем запустить программу диагностики. Если же команда **netstat -s** свидетельствует о наличии ошибок протокола, вы можете воспользоваться командой **trpt** или **iptrace**.

Неполадки связи

В распространенным неполадкам связи **TCP/IP** относится невозможность связаться с хостом в вашей сети и неполадки маршрутизации. Для устранения этих неполадок предлагаются определенные решения.

Если вы не можете установить соединение с хостом сети:

- Попробуйте подключиться к хосту с помощью команды **ping**. Для проверки работы локального сетевого интерфейса выполните команду **ping** для локального хоста.
- Попытайтесь получить имя хоста с помощью команды **host**. Если преобразование не выполнено, значит вы имеете дело с неполадкой службы преобразования имен. Дополнительная информация приведена в разделе “Неполадки преобразования имен” на стр. 425.

Если адрес хоста будет определен, но вам не удается установить соединение с этим хостом, то неполадка вызвана неправильной маршрутизацией. Дополнительная информация приведена в разделе “Неполадки маршрутизации TCP/IP” на стр. 426.

- Если вы работаете с сетью Token-Ring, проверьте, не подключен ли целевой хост к другой сети. Если это так, то, вероятнее всего, неправильно установлено значение поля **allcast**. Выполните команду SMIT `smit`

chinet, для того чтобы перейти в меню Сетевые интерфейсы. В окне диалога Token-Ring укажите значение **нет** в поле Ограничить оповещение локальной сетью Token-Ring.

- Если по сети передается большое число пакетов **протокола преобразования адресов (ARP)**, проверьте, правильно ли задана маска подсети. Слишком большое число пакетов оповещения может привести к снижению производительности сети.

Неполадки преобразования имен

Для определения IP-адресов хостов в сети используются следующие средства (в том порядке, в котором они перечислены).

1. Сервер имен доменов (**named**)
2. Служба информации о сети (NIS)
3. Локальный файл /etc/hosts

Определение неполадок хоста клиента:

Если вам не удается определить IP-адрес по имени хоста и в системе используется простая схема определения IP-адресов (файл /etc/hosts), то проверьте, правильно ли заданы IP-адреса и имена хостов в файле /etc/hosts.

Если вам не удается определить IP-адрес хоста, и при этом применяется сервер имен, то выполните следующие действия:

1. Убедитесь в том, что файл resolv.conf существует, и в нем указаны имя локального домена и IP-адрес сервера имен.
2. Проверьте связь с локальным сервером имен с помощью команды **ping**, указав в ней IP-адрес сервера имен из файла resolv.conf.
3. Если локальный сервер имен работает, то проверьте, запущен ли на нем демон **named**. Для этого вызовите на сервере имен команду **lssrc -s named**.
4. Если в системе запущена программа **syslogd**, просмотрите сообщения из протокола. Имя файла вывода, в который помещаются такие сообщения, указано в файле /etc/syslog.conf.

Если после выполнения всех указанных действий вам не удалось локализовать неполадку, то, скорее всего, неполадка вызвана неправильной работой сервера имен.

Определение неполадок хоста сервера имен:

С помощью этой процедуры можно определить неполадки сервера имен.

Если вам не удается определить IP-адрес хоста:

1. Проверьте, запущен ли демон **named**:

   ```
   lssrc -s named
   ```

2. Убедитесь, что в базе данных сервера имен указан правильный адрес нужного хоста. Отправьте демону **named** сигнал **SIGINT**, чтобы он скопировал дамп базы данных и кэш-памяти в файл /var/tmp/named_dump.db. Убедитесь, что адрес, который вы пытаетесь преобразовать, действительно указан в базе данных и в нем нет ошибок.

   При необходимости добавьте или исправьте имена и IP-адреса хостов в файле данных **named** на главном сервере имен доменов. После этого для повторного считывания информации из файла данных введите следующую команду **SRC**:

   ```
   refresh -s named
   ```

3. Убедитесь, что запросы на преобразование имен обрабатываются правильно. Для этого запустите программу **named** в командной строке и задайте уровень отладки. Допустимы уровни отладки от 1 до 9. Чем выше уровень, тем больше информации заносится в протокол при отладке.

   ```
   startsrc -s named -a
   "-d уровень_отладки"
   ```

4. Убедитесь в отсутствии ошибок в файлах данных **named**. Дополнительная информация приведена в разделе “Преобразование серверов имен” на стр. 182. Более подробные сведения приведены в разделах "DOMAIN Data File Format,", "DOMAIN Reverse Data File Format," "DOMAIN Cache File Format," и "DOMAIN Local Data File Format" книги *Справочник по файлам* .

   Примечание: Очень часто встречаются ошибки, связанные с неправильным использованием символов **.** (точка) и **@** (коммерческое "а") в файлах данных DOMAIN.

Если внешние пользователи не могут обратиться к вашим доменам, убедитесь, что на всех серверах, отличных от главного сервера имен (т.е. на подчиненных серверах и на серверах подсказок), в файлах данных о доменах задано одинаковое время хранения данных в кэш-памяти (TTL).

Если внешние распознаватели постоянно отправляют запросы к вашим серверам, убедитесь, что ваши серверы распространяют файлы данных о домене с приемлемыми значениями TTL. Если значение параметра TTL очень мало или равно нулю, то переданные данные быстро становятся недействительными. Для устранения этой неполадки задайте в записях начала области ответственности (SOA) значение, равное по крайней мере одной неделе.

Неполадки маршрутизации TCP/IP

Если вам не удается установить соединение с каким-либо удаленным хостом, рассмотрите следующие решения ситуаций.

- При получении сообщения `Сеть недоступна` проверьте правильность определения маршрута к шлюзу. Для этого нужно просмотреть таблицы маршрутизации ядра с помощью команды **netstat -r**.
- При получении сообщения `Не задан маршрут к хосту` введите команду **ifconfig** имя_интерфейса и проверьте, работает ли локальный сетевой интерфейс. В выводе этой команды должно быть указано, работает ли интерфейс. Попытайтесь обратиться к другому хосту вашей сети с помощью команды **ping**.
- При получении сообщения `Тайм-аут соединения` выполните следующие действия:
  - Убедитесь, что локальный шлюз работает. Для этого вызовите команду **ping**, указав в ней имя или IP-адрес шлюза.
  - Убедитесь, что в вашей системе правильно задан маршрут к шлюзу. Для этого нужно просмотреть таблицы маршрутизации ядра с помощью команды **netstat -r**.
  - Убедитесь, что на хосте, с которым вы пытаетесь установить соединение, задан маршрут к вашему хосту.
- Если в вашей сети применяется статическая маршрутизация, то проверьте, определены ли маршруты к целевому хосту и шлюзу. Для этого нужно просмотреть таблицы маршрутизации ядра с помощью команды **netstat -r**.

  Примечание: Убедитесь, что на хосте, с которым вы пытаетесь установить соединение, задан маршрут к вашему хосту.
- Если в вашей сети применяется динамическая маршрутизация, то введите команду **netstat -r** и убедитесь, что в таблицах маршрутизации ядра правильно задан маршрут к шлюзу.
- Если на шлюзе применяется **протокол информации о маршрутизации** (RIP) и демон **routed**, то п роверьте, задан ли в файле `/etc/gateways` статический маршрут к целевому хосту.

  Примечание: Это необходимо сделать только в том случае, если демон маршрутизации не может определить маршрут к целевому хосту с помощью других шлюзов.
- Если хост шлюза применяет протокол **RIP** и демон **gated**, то проверьте, задан ли статический маршрут к целевому хосту в файле `gated.conf`.
- В случае применения динамической маршрутизации с демоном **routed**:
  - Если демону **routed** не удается определить маршрут с помощью запросов (например, если на целевом хосте не запущен протокол **RIP**), о проверьте, задан ли маршрут к целевому хосту в файле `/etc/gateways`.

  - Убедитесь, что шлюзы, отвечающие за пересылку пакетов на хост, работают, и на них запущен протокол **RIP**. В противном случае вам нужно будет задать статический маршрут.
  - Запустите демон **routed** с опцией отладки, чтобы получить протокол с информацией о получении пакетов с ошибками. Запустите демон с помощью следующей команды:

    ```
    startsrc -s routed -a "-d"
    ```
  - Запустите демон **routed** с флагом **-t**. В этом случае все отправляемые и принимаемые пакеты будут дублироваться в поток stdout. В таком режиме программа **routed** управляется с того терминала, с которого она была запущена. В связи с этим получение программой прерывания с управляющего терминала приводит к ее уничтожению.
- В случае применения динамической маршрутизации с демоном **gated**:
  - Убедитесь, что в файле `/etc/gated.conf` указана правильная информация, а в системе запущены все необходимые протоколы.
  - Убедитесь, что шлюз в сети отправителя применяет тот же протокол, что и шлюз в сети получателя.
  - Убедитесь, что в системе, с которой вы пытаетесь установить соединение, задан маршрут к вашему хосту.
  - Убедитесь, что имена шлюзов, указанные в файле `gated.conf`, совпадают с именами шлюзов из файла `/etc/networks`.
- Если применяется протокол **RIP** или **HELLO**, и маршрут к пункту назначения не удается определить с помощью запросов маршрутизации, убедитесь, что в файле `gated.conf` задан маршрут к целевому хосту. Статические маршруты следует задавать в следующих случаях:
  - На целевом и исходном хостах применяются разные протоколы, поэтому они не могут обмениваться информацией о маршрутизации.
  - Хост должен быть достижим с удаленного шлюза (шлюза, находящегося в другой автономной системе по отношению к исходному хосту). Протокол **RIP** может применяться только хостами из одной автономной системы.

Если вам не удалось устранить неполадку, выполнив описанные выше действия, рекомендуется включить трассировку демона маршрутизации (**routed** или **gated**). Для этого введите команду SRC **traceson** или отправьте программе-демону сигнал, задающий необходимый уровень трассировки. Сигналы демонов **gated** и **routed** приведены в описании этих программ.

Определение неполадок поддержки SRC

С помощью данных инструкций можно определить распространенные неполадки в работе Контроллера системных ресурсов.

- Если изменения, внесенные в файл `/etc/inetd.conf`, не вступили в силу:

  Обновите демон **inetd** с помощью команды **refresh -s inetd** или команды **kill -1** *InetdPID*.
- Если команда **startsrc -s** [*подсистема*] выдает сообщение:

  ```
  0513-00  SRC не активен.
  ```

  Не запущен контроллер системных ресурсов для подсистемы. Запустите SRC командой **srcmstr &**, а затем снова введите команду**startsrc**.

  Демон можно запустить из командной строки без помощи SRC.
- Если команда **refresh -s** [*подсистема*] или **lssrc -ls** [*подсистема*] выдает сообщение:

  ```
  Подсистема не поддерживает эту опцию.
  ```

  Подсистема не поддерживает запрошенную опцию SRC. Обратитесь к описанию подсистемы и определите поддерживаемые опции.
- Если было выдано следующее сообщение:

  ```
  SRC не найден, выполнение будет продолжено без поддержки SRC.
  ```

Демон был запущен непосредственно из командной строки, а не командой **startsrc**. Это не неполадка. Однако для работы с подсистемой, запущенной из командной строки, не могут применяться команды SRC, в том числе **stopsrc** и **refresh**.

Если демон **inetd** работает правильно, запущены все необходимые службы, но соединение по-прежнему установить не удается, то попробуйте запустить процессы демона **inetd** в режиме отладки.

1. Временно остановите демон **inetd**.

   ```
   stopsrc -s inetd
   ```

   Команда **stopsrc** останавливает подсистемы; в данном случае - это подсистема демона **inetd**.
2. Добавьте в конец файла `syslog.conf` строку отладки. Например:

   ```
   vi /etc/syslog.conf
   ```

   a. Добавьте строку `*.debug /tmp/myfile` в конец файла и закройте редактор.
   b. Указанный файл должен существовать (в этом примере - `/tmp/myfile`). Для создания файла можно воспользоваться командой **touch**.
3. Обновите файл:
   - Если вы работаете с SRC, введите:

     ```
     refresh -s syslogd
     ```
   - В противном случае убейте процесс демона **syslogd**:

     ```
     kill -1 `ps -e | grep /etc/syslogd | cut -c1-7`
     ```
4. Запустите демон **inetd** в режиме отладки:

   ```
   startsrc -s inetd -a "-d"
   ```

   Флаг **-d** включает отладку.
5. Попытайтесь установить соединение, чтобы сообщения об ошибке были записаны в файл `/tmp/myfile`. Например:

   ```
   tn bastet
   Выполнение запроса...
   Установлено соединение с bastet
   login:>
   Соединение закрыто
   ```
6. Попробуйте найти в файле с информацией об отладке сообщения об ошибке. Например:

   ```
   tail -f /tmp/myfile
   ```

Локализация неполадок telnet или rlogin

Ниже приведена полезная информация об определении причин неполадок, возникающих при выполнении команд **telnet** и **rlogin**.

Если при работе с полноэкранными приложениями наблюдаются искажения изображения:

1. Проверьте значение переменной среды **TERM** с помощью следующей команды:

   ```
   env
   ```

   ```
   echo $TERM
   ```
2. Убедитесь, что значение переменной **TERM** соответствует типу вашего терминала.

Для отладки **telnet** можно использовать следующие команды:

Элемент	Описание
display	Показывает значения параметров и переключателей.
toggle	Переключает режим отображения сетевых данных (шестнадцатеричный или обычный).
toggle опции	Показывает внутренние опции процесса **telnet**.

Если демон **inetd** может запустить службу **telnet**, но вам по-прежнему не удается установить соединение с помощью команды **telnet**, то, возможно, ошибка связана с неправильной работой интерфейса **telnet**.

1. Убедитесь, что **telnet** применяет правильный тип терминала.
   a. Проверьте значение переменной **$TERM** на вашем компьютере:
      ```
      echo $TERM
      ```
   b. Проверьте значение переменной **$TERM** на компьютере, с которым вы пытаетесь установить соединение:
      ```
      echo $TERM
      ```
2. Воспользуйтесь функциями отладки **telnet**, введя команду **telnet** без параметров.
   ```
   telnet
   tn>
   ```
   a. Введите `open` *`хост`*, где *хост* - это имя компьютера.
   b. Нажмите Ctrl-T. Появится приглашение `tn%gt;`.
   c. В приглашении `tn>` введите `debug` для перехода в режим отладки.
3. Попробуйте подключиться к другому компьютеру с помощью интерфейса **telnet**:
   ```
   telnet bastet
   Выполнение запроса...
   Установлено соединение с bastet
   Escape character is '^T'.
   ```

   На экране появится последовательность команд. Например:
   ```
   SENT do ECHO
   SENT do SUPPRESS GO AHEAD
   SENT will TERMINAL TYPE (reply)
   SENT do SUPPORT SAK
   SENT will SUPPORT SAK (reply)
   RCVD do TERMINAL TYPE (don't reply)
   RCVD will ECHO (don't reply)
   RCVD will SUPPRESS GO AHEAD (don't reply)
   RCVD wont SUPPORT SAK (reply)
   SENT dont SUPPORT SAK (reply)
   RCVD do SUPPORT SAK (don't reply)
   SENT suboption TELOPT_NAWS Width 80, Height 25
   RCVD suboption TELOPT_TTYPE SEND
   RCVD suboption TELOPT_TTYPE aixterm
   ...
   ```
4. Убедитесь, что в файле `/etc/termcap` или `/usr/lib/terminfo` задано определение `aixterm`. Например:
   ```
   ls -a /usr/lib/terminfo
   ```
5. Если определение `aixterm` отсутствует, то добавьте его путем создания файла `ibm.ti`. Например:
   ```
   tic ibm.ti
   ```

   Команда **tic** служит компилятором информации о терминале.

При применении команд **rlogin** и **telnet** в программах, использующих Extended Curses, могут возникать неполадки, связанные с нажатием функциональных клавишей и клавишей перемещения курсора. Функциональные клавиши и клавиши управления курсором генерируют Escape-последовательности, которые расщепляются в том случае, если на каждую последовательность нажатий клавиш выделяется слишком мало времени. Функция Extended Curses ждет в течение определенного промежутка времени, чтобы

определить, на что указывает символ Esc: только на нажатие клавиши Esc, или на начало многобайтовой Escape-последовательности, генерируемой другими клавишами, например, клавиши управления курсором, клавишей действия и функциональными клавишами.

Если после символа Esc в течение заданного времени не поступает никаких данных, или эти данные недопустимы, то функция Curses решает, что этот символ указывает на нажатие клавиши Esc, и последовательность расщепляется. Задержка, возникающая при выполнении команд **rlogin** и **telnet**, зависит от сети. Иногда клавиши управления курсором и функциональные клавиши работают, иногда - нет, в зависимости от быстродействия сети. Для устранения этой неполадки увеличьте значение переменной среды **ESCDELAY** (от 1000 до 1500).

Неполадки при настройке TCP/IP

Сетевые интерфейсы настраиваются автоматически во время первого запуска системы после установки карты адаптера. Тем не менее, вам все равно придется задать некоторые начальные значения для **TCP/IP**, включая имя хоста, IP-адрес и маску подсети.

Это можно сделать с помощью интерфейса SMIT:

- Для задания имени хоста, IP-адреса и маски подсети введите команду быстрого доступа `smit mktcpip`.
- Для указания имени сервера, который будет обеспечивать преобразование имен, введите команду быстрого доступа `smit mktcpip`. (Учтите, что `smit mktcpip` настраивает только один сетевой интерфейс.)
- Для настройки других сетевых атрибутов введите команду быстрого доступа `smit chinet`.

Кроме того, вы можете задать статические маршруты, необходимые для отправки хостом информации о передаче - например, маршрут к локальному шлюзу. Эту информацию можно внести в базу данных конфигурации с помощью команды SMIT `smit mkroute`.

Для устранения других неполадок конфигурации обратитесь к разделу “Настройка сети TCP/IP” на стр. 105.

Распространенные неполадки TCP/IP с сетевыми интерфейсами

Сетевые интерфейсы настраиваются автоматически при первом запуске системы после установки карты адаптера. Однако перед запуском **TCP/IP** вы должны будете задать некоторые параметры. К ним относятся имя хоста и IP-адрес, которые можно задать с помощью команды SMIT `smit mktcpip`.

Если вы выбрали SMIT, то занесите эти значения в базу данных конфигурации с помощью команды быстрого доступа `smit mktcpip`. Для изменения этих значений в работающей системе воспользуйтесь командами `smit chinet` и `smit hostname`. Команда быстрого доступа `smit mktcpip` позволяет выполнить минимальную настройку **TCP/IP**. Для добавления адаптеров перейдите в меню дополнительной конфигурации; это можно сделать с помощью команды быстрого доступа `smit tcpip`.

Если вы убедились в правильности всех настроек, но по-прежнему сталкиваетесь с неполадками при отправке и приеме информации, то проверьте следующие условия:

- С помощью команды **netstat -i** убедитесь, что для сетевого адаптера создан сетевой интерфейс. Интерфейс (например, **tr0**) должен быть указан в выводе в столбце `Имя`. Если имя не показано, создайте сетевой интерфейс с помощью команды быстрого доступа SMIT `smit mkinet`.
- С помощью команды **netstat -i** проверьте, правильно ли задан IP-адрес сетевого интерфейса. IP-адрес должен быть указан в списке адресов в колонке Сеть. Если IP-адрес задан неправильно, измените его с помощью команды SMIT `smit chinet`.
- С помощью команды **arp** убедитесь, что IP-адрес целевого компьютера указан полностью. Например:

  ```
  arp -a
  ```

  Команда **arp** пытается найти физический адрес адаптера. Эта команда может выдать неполный адрес. Например:

  ```
  ? (192.100.61.210) на (неполный_адрес)
  ```

Такая ситуация может возникнуть в том случае, если компьютер выключен, по указанному адресу нет компьютера, или из-за неполадок аппаратного обеспечения (например, если компьютер может принимать пакеты, но не может отвечать на них).

- Проверьте правильность работы карты адаптера. Например:

  ```
  netstat -v
  ```

  Команда **netstat -v** показывает статистику драйверов устройств Ethernet, Token Ring, X.25 и 802.3. Кроме того, эта команда выводит данные об ошибках и работе сети для всех драйверов устройств, связанных с интерфейсом, например: `Нет ошибок буферов памяти`, `Нет ошибок расширения буферов памяти` и `Число переданных пакетов и число ошибок адаптера`.
- Чтобы убедиться, что неполадки не связаны с адаптером, просмотрите протокол ошибок с помощью команды **errpt**.
- Запустите программу диагностики карты адаптера и убедитесь, что карта работает правильно. Для этого воспользуйтесь быстрого доступа `smit diag` или командой **diag**.

Неполадки TCP/IP при использовании сетевого интерфейса SLIP:

В общем случае наиболее эффективный способ устранения неполадок интерфейса **Протокола подключения к Internet по последовательной линии (SLIP)** заключается в пошаговой проверке конфигурации.

Кроме того, вы можете выполнить следующие действия:

- С помощью команды **ps -ef** убедитесь, что процесс **slattach** запущен и подключен к нужному порту tty. Если это не так, введите команду **slattach**. (Точный формат команды приведен в разделе “Настройка SLIP через модем” на стр. 625 или “Настройка SLIP через нуль-модемный кабель” на стр. 627.)
- С помощью команды `smit chinet` проверьте, правильно ли заданы IP-адреса локального и удаленного хостов.

  Выберите интерфейс **SLIP**. Проверьте правильность значений, указанных в полях **IP-адрес** и **Адрес получателя**.

Если модем не работает или работает неправильно:

- Проверьте, правильно ли установлен модем. Обратитесь к руководству по установке модема.
- Убедитесь, что на модеме выключена функция управления потоком.

Если терминал работает неправильно, то с помощью команды `smit tty` проверьте, правильно ли указаны в базе данных конфигурации скорость передачи данных и другие параметры модема.

Неполадки TCP/IP при использовании сетевого интерфейса Ethernet:

Обратитесь к следующей справочной таблице при наличии неполадок **TCP/IP** при работе с сетевым интерфейсом Ethernet.

Если сетевой интерфейс инициализируется, адреса заданы правильно, и вы проверили исправность карты адаптера, то:

- Убедитесь, что к приемопередатчику подключен T-образный разъем.
- Убедитесь, что используется стандартный кабель Ethernet. (Кабель Ethernet на 50 Ом.)
- Убедитесь, что используются терминаторы Ethernet. (Терминаторы Ethernet на 50 Ом.)
- Адаптеры Ethernet могут работать как с приемопередатчиком, установленным на карте, так и с внешним приемопередатчиком. Тип применяемого приемопередатчика задается переключателем на карте адаптера. Проверьте положение этого переключателя (соответствующие инструкции приведены в описании адаптера).
- Проверьте тип разъема Ethernet (тонкий - BNC; толстый - DIX). Если вы изменили тип разъема, то установите флаг Применить изменение только к базе данных с помощью команды быстрого доступа

SMIT **smit chgenet**. (Укажите значение Да в SMIT.) Перезагрузите систему для применения внесенных изменений. (Обратитесь к разделу “Управление и настройка адаптера” на стр. 160.)

Неполадки TCP/IP при использовании сетевого интерфейса Token-Ring:

Используйте данную информацию при устранении неполадок связи в сетевом интерфейсе.

Если вам не удается установить соединение с другими системами сети, но сетевой интерфейс инициализируется, адреса заданы правильно, и вы проверили исправность карты адаптера, то:

- Проверьте, не относятся ли хосты, с которыми не удается установить соединение, к другому кольцу Token-Ring. Если это так, то с помощью команды быстрого доступа SMIT `smit chinet` проверьте значение, указанное в поле Ограничить оповещение только локальной сетью Token-Ring. *Не* указывайте значение Нет в SMIT.
- Убедитесь, что для адаптера Token-Ring задано правильное быстродействие сети. Если быстродействие задано неправильно, измените соответствующий атрибут адаптера с помощью SMIT (см. раздел “Управление и настройка адаптера” на стр. 160). После перезапуска **TCP/IP** быстродействие адаптера Token-Ring будет равно быстродействию сети.

Неполадки TCP/IP при использовании моста Token-Ring/Ethernet:

Если вам не удается установить соединение между сетями Token-Ring и Ethernet с помощью моста, хотя мост работает правильно, то пакеты могут отбрасываться адаптером Ethernet.

Это происходит в том случае, если размер поступающего пакета (с учетом заголовка) превышает размер максимального блока передачи (MTU) для адаптера. Например, к пакету размером 1500 байт, отправленному через мост адаптером Token-Ring, будет добавлен 8-байтовый заголовок управления логическим каналом связи (LLC), что приводит к увеличению размера пакета до 1508 байт. Если размер MTU принимающего адаптера Ethernet равен 1500, то пакет будет уничтожен.

Проверьте значения MTU для обоих сетевых адаптеров. Для того чтобы учесть 8-байтовый заголовок LLC, который адаптер Token-Ring добавляет к отправляемым сообщениям, необходимо, чтобы значение MTU для адаптера Token-Ring было по крайней мере на 8 байт меньше значения MTU для адаптера Ethernet. Например, чтобы установить соединение между адаптером Token-Ring и адаптером Ethernet с MTU, равным 1500, значение MTU для этого адаптера Token-Ring должно быть равно 1492.

Неполадки TCP/IP при использовании моста Token-Ring/Token-Ring:

При подключении через мост максимальный размер блока передачи (MTU), равный по умолчанию 1500, должен быть заменен на значение, которое на 8 байт меньше, чем максимальный размер информационного поля (I-кадра), указываемый мостом в поле управления маршрутизацией.

Для того чтобы узнать значение поля управления маршрутизацией, просмотрите получаемые пакеты с помощью демона **iptrace**. Биты 1, 2 и 3 первого байта задают максимальный размер кадра, т.е. максимальный размер информационного поля, которое может быть передано по определенному маршруту между двумя станциями. Формат поля управления маршрутизацией показан на следующем рисунке:

На рисунке показаны байты 0 и 1 поля управления маршрутизацией. Восемь битов байта 0 равны B, B, B, B,

Рисунок 25. Поле управления маршрутизацией

L, L, L, L. Восемь битов байта 1 равны D, F, F, F, r, r, r, r.

Возможны следующие сочетания значений битов:

Элемент	Описание
000	Задает максимальный размер информационного поля, равный 516 байтам.
001	Задает максимальный размер информационного поля, равный 1500 байтам.
010	Задает максимальный размер информационного поля, равный 2052 байтам.
011	Задает максимальный размер информационного поля, равный 4472 байтам.
100	Задает максимальный размер информационного поля, равный 8144 байтам.
101	Зарезервировано.
110	Зарезервировано.
111	Используется при рассылке кадров по всем маршрутам.

Например, если в поле управления маршрутизацией задан максимальный размер информационного кадра, равный 2052, то размер блока MTU должен быть равен 2044. Это относится только к сетевым интерфейсам Token-Ring.

Примечание: При использовании **iptrace** файл вывода *не* должен находиться в сетевой файловой системе (NFS).

Неполадки TCP/IP при соединении с удаленным хостом

Если вы не можете установить соединение с удаленным хостом, попытайтесь выполнить следующие действия.

- Для проверки работы локального сетевого интерфейса выполните команду **ping** для локального хоста.
- С помощью команды **ping** последовательно проверяйте соединения со все более удаленными хостами до обнаружения точки сбоя соединения.

Если наблюдается потеря пакетов или задержки при передаче данных, то попробуйте предпринять следующие меры:

- Для отслеживания пакетов на уровне сокетов воспользуйтесь командой **trpt**.
- Для отслеживания работы протоколов всех уровней можно воспользоваться командой **iptrace**.

Если вам не удается установить соединение между сетями Token-Ring и Ethernet с помощью моста и вы проверили исправность моста, то выполните следующие действия:

- Проверьте значения MTU для обоих сетевых адаптеров. Для правильной передачи данных по соединению эти значения должны быть согласованы. Система уничтожает пакеты в том случае, если размер поступающего пакета (включая заголовки) превышает размер блока MTU для адаптера. Например, к отправленному через мост пакету размером 1500 байт будет добавлен 8-байтовый заголовок LLC, что приводит к увеличению размера пакета до 1508 байт. Если размер MTU в принимающей системе равен 1500, то пакет размером 1508 байт будет уничтожен.

Неполадки TCP/IP, связанные с ответом snmpd на запросы

Если демон **snmpd** не отвечает на запросы, и при этом в протокол не заносятся сообщения, то причина может заключаться в том, что размер пакета слишком велик для обработчика пакетов **UDP** ядра.

В этом случае увеличьте значения переменных ядра **udp_sendspace** и **udp_recvspace** с помощью следующих команд:

```
no -o udp_sendspace=64000
no -o udp_recvspace=64000
```

Максимальный размер пакета **UDP** равен 64 Кб. Если размер запроса превышает 64 Кб, то запрос будет аннулирован. Для устранения этой неполадки пакет должен быть разбит на несколько небольших пакетов.

Неполадки TCP/IP с протоколом динамической настройки хостов

Если вам не удалось получить данные конфигурации, попробуйте использовать одно из следующих решений.

Если вашему хосту не удается получить IP-адрес или другие параметры конфигурации:

- Проверьте, указали ли вы настраиваемый интерфейс. Для этого можно использовать команду быстрого доступа SMIT `smit dhcp`.
- Проверьте, существует ли в сети сервер или промежуточный агент, настроенный на передачу запросов от вашего хоста за пределы локальной сети.
- Проверьте, запущен ли демон **dhcpcd**. Если это не так, введите команду **startsrc -s dhcpcd**.

Команды TCP/IP

Протокол **TCP/IP** - это часть структуры протоколов, лежащей в основе вашей системы. Протокол TCP/IP позволяет установить соединение с другой рабочей станцией или системой, выполнив команду или программу.

Протокол **TCP/IP** - это часть структуры протоколов, лежащей в основе вашей системы. Протокол TCP/IP позволяет установить соединение с другой рабочей станцией или системой, выполнив команду или программу. Все остальное система сделает сама.

Элемент	Описание
chnamsv	Изменяет конфигурацию службы имен **Протокола управления передачей/Протокола Internet (TCP/IP)** на хосте.
chprtsv	Изменяет конфигурацию службы печати в компьютере-клиенте или сервере.
hostent	Предназначена для прямого управления записями преобразования адресов в базе данных конфигурации системы.
ifconfig	Предназначена для настройки или просмотра параметров сетевого интерфейса сети **TCP/IP**.
mknamsv	Настраивает службу имен **TCP/IP** хоста для работы в качестве клиента.
mkprtsv	Настраивает службу печати **TCP/IP** хоста.
mktcpip	Устанавливает значения, необходимые для запуска **TCP/IP** на хосте.
no	Настраивает опции сети.
rmnamsv	Удаляет конфигурацию службы имен **TCP/IP** на хосте.
rmprtsv	Удаляет конфигурацию службы печати в компьютере-клиенте или сервере.
slattach	Подключает последовательные линии связи в качестве сетевых интерфейсов.
arp	Предназначена для просмотра или изменения таблиц преобразования IP-адресов в аппаратные адреса, которые применяются **Протоколом преобразования адресов (ARP)**.
gettable	Получает от хоста таблицы хостов NIC (Сетевой информационный центр).
hostid	Предназначена для настройки или просмотра идентификатора текущего локального хоста.
hostname	Предназначена для настройки или просмотра имени текущего хоста.
htable	Преобразует файлы хоста в формат, применяемый сетевыми библиотечными процедурами.
ipreport	Создает отчет о трассировке пакетов с помощью указанного файла трассировки пакетов.
iptrace	Предназначена для трассировки пакетов протоколов Internet уровня сетевого интерфейса.
lsnamsv	Предназначена для просмотра информации службы имен, хранящейся в базе данных.
lsprtsv	Предназначена для просмотра информации службы печати, хранящейся в базе данных.
mkhosts	Создает файл таблицы хостов.
namerslv	Обеспечивает прямой доступ к записям сервера имен доменов из базы данных конфигурации системы для локальных процедур преобразования.
netstat	Предназначена для просмотра состояния сети.
route	Предназначена для изменения таблиц маршрутизации вручную.

Элемент	Описание
ruser	Предназначена для прямой настройки трех различных баз данных системы, которые управляют доступом внешних хостов к программам.
ruptime	Предназначена для просмотра состояний всех хостов сети.
securetcpip	Подключает средства защиты сети.
setclock	Устанавливает дату и время на хосте или в сети.
timedc	Возвращает информацию о демоне **timed**.
trpt	Выполняет трассировку пакетов на уровне сокетов **Протокола управления передачей (TCP)**.

Команды SRC

Команды SRC могут выполняться для одного демона, группы демонов, или для заданного демона и всех демонов, которыми он управляет (т.е. для подсистемы и всех ее субсерверов).

Кроме того, некоторые демоны **TCP/IP** не реагируют на все команды SRC. Ниже приведен список команд SRC, которые могут использоваться для управления демонами **TCP/IP**, а также исключения из этого списка.

Элемент	Описание
startsrc	Запускает все подсистемы **TCP/IP** и субсерверы **inetd**. Команда **startsrc** применяется для всех подсистем **TCP/IP** и субсерверов **inetd**.
stopsrc	Завершает все подсистемы **TCP/IP** и субсерверы **inetd**. Эта команда носит также название **обычного завершения**. Команда **обычного завершения** разрешает подсистемам обработать все оставшиеся задания. Субсерверу **inetd** разрешается установить все ожидающие соединения и завершить все существующие соединения. Команда **обычного завершения** выполняется для всех подсистем **TCP/IP** и субсерверов **inetd**.
stopsrc -f	Завершает все подсистемы **TCP/IP** и субсерверы **inetd**. Эта команда носит также название команды **принудительного завершения**. Выполнение команды **принудительного завершения** приводит к немедленному завершению работы всех подсистем. Что касается субсерверов **inetd**, то все существующие соединения, так же как и находящиеся в состоянии ожидания, немедленно завершаются.
refresh	Обновляет следующие подсистемы и субсерверы: **inetd**, **syslogd**, **named**, **dhcpsd** и **gated**.
lssrc	Выдает краткие сведения о состоянии подсистемы (активна или неактивна). Также выдает краткие сведения о субсерверах **inetd**. Сведения для субсерверов **inetd** включают имя субсервера, описание его состояния, имя команды и параметры, с которыми он был запущен.
lssrc -l	Выдает краткие сведения и дополнительную информацию (полное состояние) для следующих подсистем: **gated** Состояние отладки или трассировки, используемые протоколы маршрутизации, таблицы маршрутизации, принимаемые сигналы и их действие. **inetd** Состояние отладки, список активных субсерверов и краткие сведения об их состоянии; принимаемые сигналы и их действие. **named** Состояние отладки, информация из файла `named.conf`. **dhcpsd** Состояние отладки, все контролируемые IP-адреса и их текущее состояние. **routed** Состояние отладки и трассировки, состояние маршрутизации, таблицы маршрутизации. **syslogd** Информация о конфигурации **syslogd**. С помощью команды **lssrc -l** можно получить подробные сведения о состоянии субсерверов **inetd**. Подробные сведения включают краткую информацию о состоянии и информацию об активных соединениях. Для некоторых субсерверов выдается дополнительная информация. Она включает: **ftpd** Состояние отладки и занесения в протокол **telnetd** Тип эмулируемого терминала **rlogind** Состояние отладки **fingerd** Состояние отладки и занесения в протокол Для субсерверов **rwhod** и **timed** подробные сведения не выводятся.
traceson	Включает отладку на уровне сокетов. Для форматирования вывода используйте команду **trpt**. Подсистемы **timed** и **iptraced** команду **traceson** не поддерживают.
tracesoff	Выключает отладку на уровне сокетов. Для форматирования вывода используйте команду **trpt**. Подсистемы **timed** и **iptraced** команду **tracesoff** не поддерживают.

Примеры вызова этих команд можно найти в тех разделах, где данные команды описаны. Дополнительная информация о Контроллере системных ресурсов содержится в разделе Контроллер системных ресурсов в *Управление операционной системой и устройствами*.

Команды передачи файлов

Здесь приведены краткие описания команд передачи файлов.

Элемент	Описание
ftp *хост*	Обеспечивает передачу файлов между локальным и удаленным хостами.
rcp*файл хост:файл*	Обеспечивает передачу файлов между локальным и удаленным хостом, либо между двумя удаленными хостами.
tftp	Обеспечивает передачу файлов между хостами.

Команды удаленного входа в систему

Здесь приведены краткие описания команд **TCP/IP** для удаленного входа в систему.

Элемент	Описание
rexec *хост команда*	Выполняет команду на удаленном хосте (в качестве параметра можно указать не более одной команды).
rlogin*удаленный-хост*	Устанавливает соединение между локальным и удаленным хостом.
rsh и **remsh***удаленный-хост команда*	Позволяет выполнить указанную команду или войти в систему на удаленном хосте.
telnet, **tn** и **tn3270***имя-хоста*	Устанавливает соединение между локальным и удаленным хостом с помощью интерфейса **TELNET**.

Команды состояния

Здесь приведены краткие описания команд состояния **TCP/IP**.

Элемент	Описание
finger или **f** *пользователь@хост*	Показывает информацию о пользователе.
host*имя-хоста*	Преобразует имя хоста в его IP-адрес или наоборот.
ping*имя-хоста*	Отправляет эхо-запрос хосту сети.
rwho	Показывает список пользователей, которые в данный момент работают на хостах локальной сети.
whois*имя*	Идентифицирует пользователя по его ИД или псевдониму.

Команда работы с удаленными системами

Команда **TCP/IP** работы с удаленными системами, **talk** *User@Host*, позволяет вам общаться с другими пользователями.

Элемент	Описание
talk *Пользователь@Хост*	Инициализирует диалог с другим пользователем.

Команды печати

Здесь приведены краткие описания команд печати **TCP/IP**.

Элемент	Описание
enq *файл*	Помещает файл в очередь печати удаленного устройства.
refresh	Запрашивает обновление информации о подсистеме или группе подсистем.
smit	Обеспечивает управление системой.

Демоны TCP/IP

Подсистемами называются демоны, работой которых управляет SRC. *Субсервер* - это демон, который управляется подсистемой. (Команды и имена программ-демонов обычно обозначаются буквой **d** в конце имени.)

Понятия подсистемы и субсервера являются взаимоисключающими. Это значит, что демон не может быть одновременно и подсистемой, и субсервером. Единственной подсистемой **TCP/IP**, которая управляет другими программами-демонами, является демон **inetd**. Все субсерверы **TCP/IP** являются также субсерверами **inetd**.

SRC управляет работой следующих демонов **TCP/IP**:

Подсистемы

Элемент	Описание
gated	Обеспечивает функции маршрутизации шлюза и поддерживает **Протокол информации о маршрутизации (RIP)**, **Протокол информации о маршрутизации следующего поколения (RIPng)**, **Протокол внешних шлюзов (EGP)**, **Протокол граничных шлюзов (BGP)** и **BGP4+**, **протокол HELLO**, **Протокол кратчайшего пути (OSPF)**, **протоколы IS-IS**, **ICMP** и **ICMPv6/Router Discovery**. Кроме того, демон **gated** поддерживает **Простой протокол управления сетью (SNMP)**. Демон **gated** - один из двух демонов маршрутизации, которые могут применяться при маршрутизации сетевых адресов. Использование демона **gated** более предпочтительно, чем использование **routed**, так как **gated** поддерживает большее количество протоколов шлюза.
inetd	Вызывает и планирует запуск других демонов при получении запроса к службам демонов. Этот демон может также запускать другие демоны. Демон **inetd** иногда называют супердемоном.
iptrace	Выполняет трассировку пакетов на уровне интерфейса для протоколов Internet.
named	Применяется протоколом **Сервер имен доменов** (DOMAIN) для преобразования имен.
routed	Управляет таблицами маршрутизации; поддерживает **Протокол информации о маршрутизации (RIP)**. Использование демона **gated** более предпочтительно, чем использование **routed**, так как **gated** поддерживает большее количество протоколов шлюза.
rwhod	Каждые три минуты отправляет оповещения на все другие хосты и хранит информацию о пользователях, вошедших в систему, и о состоянии сети. При работе с демоном **rwhod** учтите, что он может занимать значительный объем ресурсов системы.
timed	Выполняет функции сервера времени.

Примечание: Демоны **routed** и **gated** относятся к подсистемам TCP/IP. Не выполняйте команду **startsrc -g tcpip**, которая запускает оба эти демона, с другими подсистемами **TCP/IP**. Одновременный запуск обоих демонов на одной машине может привести к непредсказуемым результатам.

Подсистема **inetd** управляет следующими демонами TCP/IP:

Субсерверы inetd

Элемент	Описание
comsat	Уведомляет пользователей о поступлении почты.
fingerd	Создает отчет о состоянии всех пользователей, вошедших в систему, и о состоянии сети на удаленном хосте. Этот демон применяет протокол **Finger** .
ftpd	Предоставляет процессу клиента функции передачи файлов, основанные на **Протоколе передачи файлов (FTP)**.
rexecd	Выполняет функцию сервера внешнего хоста для команды **rexec**.
rlogind	Выполняет функцию удаленного устройства регистрации для команды **rlogin**.
rshd	Обеспечивает выполнение команд удаленного сервера **rcp** и **rsh**.
talkd	Обеспечивает режим диалога для команды **talk**.
syslogd	Читает системные сообщения и заносит их в протокол. Этот демон относится к группе подсистем **Службы удаленного доступа** (RAS).
telnetd	Выполняет функцию сервера для протокола **TELNET**.
tftpd	Выступает в роли сервера для **Упрощенного протокола передачи файлов (TFTP)**.
uucpd	Управляет процессом взаимодействия между Основными сетевыми утилитами (BNU) и **TCP/IP**.

Методы устройств

Методы устройств - это программы, связанные с устройствами и выполняющие основные операции по настройке этих устройств.

Дополнительные сведения о методах **TCP/IP** приведены в разделе List of TCP/IP Programming References книги *Communications Programming Concepts*.

Запросы на получение комментариев

В системе AIX поддерживаются следующие запросы на комментарии (RFC) **TCP/IP**.

Список RFC, поддерживаемых данной операционной системой, приведен в разделе List of TCP/IP Programming References книги *Communications Programming Concepts*.

- RFC 1359 *Connecting to the Internet: What connecting institutions should anticipate*
- RFC 1325 *FYI on questions and answers: Answers to commonly asked 'new Internet user' questions*
- RFC 1244 *Site Security Handbook*
- RFC 1178 *Choosing a Name for Your Computer*
- RFC 1173 *Responsibilities of host and network managers: A summary of the `oral tradition' of the Internet*

Основные сетевые утилиты

BNU являются группами программ, каталогов и файлов для установки соединения между компьютерными системами в лока льных и удаленных сетях. Их можно использовать для соединения с любой системой UNIX, на которой выполняется версия программы копирования UNIX-UNIX. BNU входят в группу программ расширенных служб (extended services), устанавливаемых по желанию заказчика вместе с базовой операционной системой.

В состав BNU входит группа команд, относящихся к программе UUCP, а также сама программа UUCP, разработанная фирмой AT&T и модифицированная как часть программного обеспечения Berkeley Software Distribution (BSD). В BNU предусмотрены команды, процессы и базы данных, позволяющие устанавливать связь с локальной и удаленной системами. Для локального соединения систем применяются сети Token-Ring и Ethernet. Локальная сеть может быть соединена с удаленной системой через выделенное соединение или модем. После установления соединения можно обмениваться командами и файлами между локальной и удаленной системами.

Перед началом работы с командами и программами BNU необходимо установить и настроить утилиты BNU.

Управление BNU осуществляется с помощью файлов конфигурации, которые определяют, какие удаленные системы могут подключаться к локальной системе, и какие действия они могут выполнять после этого. Эти файлы конфигурации необходимо настроить в соответствии с требованиями, предъявляемыми к системе, и имеющимися ресурсами.

Обслуживание BNU заключается в периодическом просмотре и удалении файлов протоколов, а также просмотре содержимого очередей BNU, который необходим для проверки правильности передачи данных в удаленные системы. Кроме того, вам необходимо время от времени обновлять файлы конфигурации, отражая изменения, произошедшие в вашей и удаленных системах.

Как работает BNU

BNU устанавливают связь между системами с помощью специального аппаратного обеспечения и компонентов программного обеспечения.

Информация об операциях, выполняемых BNU, отражена в структуре каталогов и файлов BNU. В нее входят набор общих каталогов, группа административных каталогов и файлов, файлы конфигурации и файлы блокировки. Большинство каталогов BNU создается во время установки. Некоторые административные каталоги и файлы создаются программами BNU.

За исключением команд удаленного входа в систему, все операции BNU выполняются в пакетном режиме. Когда пользователь запрашивает отправку задания в удаленную систему, BNU сохраняет информацию, необходимую для выполнения задания. Такая операция называется *постановкой задания в очередь*. В запланированные моменты времени или по команде пользователя BNU обращается к различным удаленным системам, передает накопившиеся в очереди задания и принимает поступившие задания. Эти операции регулируются с помощью файлов конфигурации, находящихся в вашей и удаленных системах.

Поддержка национальных языков для команд BNU

Все команды BNU, за исключением **uucpadm**, обеспечивают поддержку национальных языков.

Имена пользователей не обязательно указывать символами ASCII. Однако все системные имена должны быть заданы символами ASCII. Если пользователь попытается запланировать передачу информации или выполнение удаленной команды с использованием не-ASCII символов в системных именах, BNU возвратит сообщение об ошибке.

Структура файлов и каталогов BNU

Информация об операциях, выполняемых BNU, отражена в структуре каталогов и файлов BNU.

Структура содержит общие каталоги, файлы конфигурации, административные каталоги и блокирующие файлы.

Большинство каталогов BNU создается во время установки. Некоторые административные каталоги и файлы создаются программами BNU во время выполнения.

Общие каталоги BNU

Если общий каталог BNU (`/var/spool/uucppublic`) задан, то в нем хранятся файлы, полученные локальной системой от других систем.

Эти файлы находятся в общем каталоге до тех пор, пока пользователи их не затребуют. Общий каталог создается при установке BNU. Кроме того, в общем каталоге создается подкаталог для каждой удаленной системы, отправляющей файлы в локальную систему.

Файлы конфигурации BNU

Файлы конфигурации BNU, называемые также базой данных поддержки BNU, хранятся в каталоге `/etc/uucp`. Эти файлы необходимо специально настроить для конкретной системы.

Они принадлежат пользователю uucp и могут быть изменены только пользователем с правами доступа root. Файлы конфигурации содержат информацию о следующих объектах и значениях:

- Доступные удаленные системы
- Устройства, применяемые при подключении к удаленным системам
- Периоды времени, когда разрешено подключаться к удаленным системам
- Действия, которые удаленные системы могут выполнять в локальной системе.

Кроме того, в некоторых файлах конфигурации заданы ограничения на операции BNU, предохраняющие систему от перегрузки.

Ниже приведен список файлов конфигурации BNU:

Элемент	Описание
Devices	Содержит информацию о доступных устройствах, включая как модемы, так и кабели для прямого подключения.
Dialcodes	Содержит коды набора номера, позволяющие указывать сокращенные номера телефонов в файле Systems.
Dialers	Задает синтаксис команды вызова для конкретного типа модема ("программу набора номера").
Maxuuscheds	Ограничивает число одновременно запланированных заданий.
Maxuuxqts	Ограничивает число одновременно выполняемых удаленных команд.
Permissions	Содержит коды прав доступа. Этот файл играет основную роль при работе с защитой в BNU.
Poll	Указывает, когда программа BNU должна опрашивать удаленные системы на предмет инициализации задач.
Sysfiles	Содержит список файлов, применяемых вместо файлов Systems, Devices и Dialers в конфигурации BNU. Если этот файл не используется, то по умолчанию применяются файлы /etc/uucp/Systems, /etc/uucp/Devices и /etc/uucp/Dialers.
Systems	Содержит список доступных удаленных систем и информацию, необходимую для установления соединения с ними, в частности, применяемое устройство, имя пользователя и пароль для входа в удаленную систему. Кроме того, в этом файле указаны моменты времени, когда можно обращаться к системам.

При работе BNU файлы конфигурации ссылаются друг на друга. Например:

- Файл Devices содержит поле *Флаг*, ссылающееся на записи из файла Dialers.
- Файл Systems содержит запись *Класс*. Устройство каждого *класса*, указанного в файле Systems, должно быть определено в файле Devices.
- Файл Poll содержит информацию о системах, к которым может обращаться локальная система. Каждая из этих систем должна быть определена в файле Systems.

Записи в файлах конфигурации BNU зависят от того, какие типы соединений применяются для связи между локальной и удаленными системами. Например, для соединений Протокола управления передачей/Протокола Internet (TCP/IP) и для прямых соединений создаются специальные записи. Если соединения с удаленными системами устанавливаются через модемы, то эти модемы должны быть определены в файле Dialers.

Для установления соединений с удаленными системами с помощью BNU, необходимо настроить файлы Systems, Devices и Permissions. Остальные файлы конфигурации позволяют вам пользоваться возможностями BNU, например, функцией автоматического опроса. Многие файлы конфигурации необходимо периодически обновлять с целью отразить в них изменения, произошедшие в вашей системе, и в системах, к которым вы обращаетесь. В файле Sysfiles можно указать файлы, которые будут применяться вместо файлов по умолчанию: Systems, Devices и Dialers.

Каталоги и файлы администрирования BNU

Административные каталоги и файлы BNU расположены в подкаталогах каталога /var/spool/uucp.

Эти каталоги и файлы содержат информацию двух типов:

- Данные, ожидающие передачи в другие системы
- Протоколы и информация об ошибках BNU.

В каталоге /var/spool/uucp BNU создает следующие подкаталоги:

Элемент	Описание
.Admin	Содержит четыре административных файла: • audit • Foreign • errors • xferstats В этих файлах хранятся протоколы и информация об ошибках BNU.
.Corrupt	Содержит копии файлов, которые не может обработать программа BNU.
.Log и .Old	Содержит файлы протоколов транзакций BNU.
.Status	Указывает, когда в последний раз демон **uucico** пытался установить связь с удаленными системами.
.Workspace	Хранит временные файлы, применяемые программами передачи файлов.
.Xqtdir	Содержит выполняемые файлы со списками команд, доступных в удаленных системах.
SystemName	Содержит файлы, применяемые программами передачи. Это следующие файлы: • Команда (**C.***) • Данные (**D.***) • Исполняемые (**X.***) • Временные (**TM.***) BNU создает каталог *Имя-системы* для каждой удаленной системы, с которой было установлено соединение.

Каталоги, имена которых начинаются с точки, являются *скрытыми*. Они не указываются в выводе команд **ls** и **li**, если не указан флаг **-a** . При запуске демона **uucico** из подкаталогов каталога /var/spool/uucp передаются все рабочие файлы, не являющиеся скрытыми. Демону **uucico** доступен только каталог *Имя-системы*. Остальные административные каталоги недоступны.

Файлы в скрытых каталогах принадлежат пользователю uucp. Они доступны только пользователю с правами доступа root, а также пользователю с UID 5.

За дополнительной информацией об обслуживании административных каталогов BNU обратитесь к разделу “Обслуживание BNU” на стр. 454.

Файлы блокировки BNU

Файлы блокировки BNU хранятся в каталоге /var/locks. Когда BNU обращается к устройству для подключения к удаленному компьютеру, он помещает файл блокировки для этого устройства в каталог /var/locks.

Если другой программе BNU или любой другой программе необходимо это устройство, она проверяет наличие соответствующего файла блокировки в каталоге /var/locks. Если файл блокировки существует, то программа ждет, пока устройство не освободится, или использует для связи другое устройство.

Кроме того, демон **uucico** помещает в каталог /var/locks файлы блокировки для удаленных систем. Перед подключением к удаленной системе демон **uucico** проверяет наличие соответствующего файла блокировки в каталоге /var/locks. Описанный механизм предотвращает создание нескольких соединений с одной и той же удаленной системой различными экземплярами демона **uucico**.

Примечание: Каталог /var/locks используется и другими программами, в том числе эмулятором асинхронного терминала (ATE) и протоколом TCP/IP.

Настройка BNU

Рассмотрена процедура настройки Основных сетевых утилит (BNU) для различных типов соединений, включая прямые соединения, связь по модему и соединения TCP/IP.

Предварительные требования

- В системе должно быть установлено программное обеспечение BNU.
- Для редактирования файлов конфигурации BNU необходимы права доступа root.
- Если в BNU применяются прямые соединения, то локальная система должна быть напрямую к удаленным системам.
- Если соединения BNU устанавливаются через модем, то следует установить и настроить все модемы.
- Если вы планируете применять соединения TCP/IP, то в локальной системе и удаленных системах должны быть установлены и запущены службы TCP/IP.
- Соберите информацию, необходимую для настройки BNU (обратитесь к приведенному ниже списку). Эта информация должна включать список удаленных систем и списки устройств и модемов, применяемых для соединения с системами.

Сбор необходимой информации о системе

Перед настройкой BNU соберите следующую информацию:

- Для каждой *удаленной системы*, к которой будет обращаться ваша система, соберите следующие данные:
  - Имя системы
  - Имя для входа в систему, применяемое системой в удаленной системе.
  - Пароль для входа в систему.
  - Приглашения для ввода имени и пароля для входа в удаленную систему.
  - Тип соединения, который будет применяться для подключения к удаленной системе (прямое, TCP/IP или через модем).

  Для прямого соединения соберите следующую информацию:
  - Скорость передачи данных по соединению в битах
  - Порт локальной системы, через который будет установлено соединение.

  Для соединения через модем (телефонная линия) соберите следующую информацию:
  - Номер телефона для подключения к удаленной системе.
  - Скорость передачи данных, которую поддерживают локальный и удаленный модемы.

  Примечание: Если удаленные системы будут обращаться к вашей системе, то убедитесь, что у администраторов BNU этих систем есть вся указанная выше информация о вашей системе.
- О каждом *локальном модеме*, применяемом для установления соединений BNU, нужно собрать следующую информацию:
  - Сценарий установления связи (обратитесь к документации по модему).

    Примечание: Для некоторых модемов сценарий установления связи задан в файле `/etc/uucp/Dialers`.
  - Локальный порт модема.

Создание списка системных устройств

С учетом собранной информации составьте список для каждого устройства, применяемого для соединения с удаленной системой. Ниже приведен пример списка для локальной системы `morgan`:

```
direct:
hera 9600 tty5
zeus& 2400 tty2
ariadne 2400 tty1
hayes modem (tty3):  apollo, athena
TCP/IP:  merlin, arthur, percy
```

В предыдущем примере для подключение к системе hera применяется прямое соединение direct со скоростью 9600 через порт tty5. Для обращения к системе apollo используется модем фирмы Hayes, подключенный к порту tty3. Соединение с системами merlin, arthur и percy установлено по протоколу TCP/IP.

Настройка средств удаленной связи

Для правильной работы BNU на вашем узле следует настроить средства связи таким образом, чтобы они обеспечивали выполнение следующих функций:

- Просмотр списка устройств, применяемых для установления прямого соединения или соединения через модем.
- Просмотр списка модемов, применяемых для подключения к удаленным системам по телефонной линии.
- Просмотр списка доступных удаленных систем.
- Просмотр списка символьных сокращений, которые используются в качестве префиксов телефонных номеров, применяемых для подключения к указанным удаленным системам (необязательно).
- Задание прав доступа, указывающих, каким образом локальная система может подключаться к удаленным системам.
- Отслеживание работы удаленных систем (необязательно).

Для создания этих списков, прав доступа и расписаний выполните следующие действия:

- Отредактируйте файлы конфигурации BNU.
- В файле /var/spool/cron/crontabs/uucp удалите символы комментария (#) из строк, в которых описаны процедуры автоматического обслуживания.

Примечание: Для того чтобы обеспечить правильную работу BNU на вашем узле, настройте файлы Systems, Devices и Permissions. Эти файлы можно редактировать в произвольном порядке.

После выполнения предыдущих действий можно настроить BNU в системе.

Настройка BNU в системе

Для того чтобы настроить BNU, выполните следующие действия:

1. Убедитесь, что утилиты BNU установлены. Для этого введите следующую команду:

   ```
   lslpp -h bos.net.uucp
   ```

   Если утилиты BNU установлены, в выводе команды будет указана строка bos.net.uucp. Если она отсутствует, установите BNU с магнитной ленты.
2. Задайте ИД и пароли для входа в удаленные системы, которые будут обращаться к вашей системе, и сообщите их сотрудникам, отвечающим за администрирование BNU и программы копирования UNIX-UNIX (UUCP) в этих системах. Для этого нужно отредактировать файлы /etc/passwd, /etc/group, /etc/security/login.cfg и /etc/security/passwd.

   Внимание: Предоставление удаленным системам права входить в локальную систему от имени пользователя UUCP значительно ослабляет защиту системы. Такие пользователи могут просматривать и, возможно, изменять локальные файлы Systems и Permissions. Эти действия удаленной системы зависят от прав доступа, указанных в записи LOGNAME из файла Permissions. Рекомендуется создавать другие ИД входа в систему BNU для удаленных систем и выделить специальный ИД входа в систему UUCP для специалиста, отвечающего за BNU в локальной системе. Надежная защита достигается в том случае, когда каждой удаленной системе выделен свой идентификатор для входа в локальную систему, с которым связан уникальный номер UID. Идентификатор группы (GID), связанный с этим ИД входа в систему, должен быть равен 5. По умолчанию пользователю с идентификатором NUUCP разрешено передавать файлы.

   a. Если вам нужно настроить для компьютеров разные права доступа, то задайте для каждого компьютера отдельный ИД для входа в систему, а также записи MACHINE и LOGNAME в файле

Permissions. Вы можете решить, следует ли указывать уникальные имена пользователей при установлении различных соединения BNU, или можно использовать одно и то же имя при установлении всех соединений. Ниже приведено несколько примеров записей из файла /etc/passwd:

```
Umicrtk:!:105:5:micrtk uucp:/usr/spool/uucppublic:/usr/sbin/uucp/uucico
Ufloydl:!:106:5:floydl uucp:/usr/spool/uucppublic:/usr/sbin/uucp/uucico
Uicus:!:107:5:icus uucp:/usr/spool/uucppublic:/usr/sbin/uucp/uucico
Urisctkr:!:108:5::/usr/spool/uucppublic:/usr/sbin/uucp/uucico
```

b. Если для всех компьютеров можно задать общие права доступа, и вы не планируете управлять каждым соединением UUCP в отдельности, то для всех компьютеров можно задать одно имя для входа в систему. Пример записи для такого сценария:

```
nuucp:!:6:5::/usr/spool/uucppublic:/usr/sbin/uucp/uucico
```

Примечание:

- UID (третье поле, разделенное двоеточиями) должно быть уникальным во избежание угроз безопасности.
- GID (четвертое поле, разделенное двоеточиями) должно содержать значение 5, которое соответствует группе, к которой относится UUCP.
- В качестве домашнего каталога (шестое поле, разделенное двоеточием) можно указать любой допустимый каталог.
- В качестве начальной оболочки (седьмое поле, разделенное двоеточиями) должно быть указано значение /usr/sbin/uucp/uucico.

c. Убедитесь, что новые пользователи добавлены в файл /etc/group. Ниже приведен пример такой записи:

```
uucp:!:5:uucp,uucpadm,nuucp,Umicrtk,Uicus,Urisctakr
```

d. Добавьте в группу UUCP пользователей, использующих модем для подключения с помощью программ, отличных от команды **cu**.

e. После изменения этих файлов (от имени пользователя root) задайте пароль для новых пользователей с помощью команды **passwd** *имя-пользователя*.

Примечание: Если пароль пользователя был изменен пользователем root, то в файле /etc/security/passwd для этого пользователя будут установлены следующие флаги:

```
flags = ADMCHG
```

Измените предыдущую строку следующим образом:

```
flags =
```

В противном случае, если удаленный процесс **uucico** войдет в систему, ему будет предложено ввести новый пароль. Поскольку это действие недопустимо, при попытке входа в систему возникает ошибка.

f. Во избежание прерываний процесса входа в систему, вызываемых процессом **uucico**, добавьте символы комментария в раздел по умолчанию (со звездочками) и создайте раздел для терминала, как это показано в следующем примере:

```
 /dev/tty0:
             herald = "\nrisc001 login:"
```

g. С помощью текстового редактора или команды **uucpadm** отредактируйте файл Poll. Добавьте запись для каждой опрашиваемой системы.

Примечание: Системы, перечисленные в файле Poll, должны быть указаны и в файле /etc/uucp/Systems.

h. С помощью текстового редактора измените файл /var/spool/cron/crontabs/uucp. Удалите символы комментария (#) в строках с командами **uudemon.hour** и **uudemon.poll**. Вы можете изменить число вызовов этих команд. Однако команда **uudemon.poll** должна вызываться примерно на пять минут раньше, чем команда **uudemon.hour**.

i. Убедитесь, что изменения успешно применены. Для этого введите следующую команду:

```
crontab -l uucp
```

j. Задайте следующие файлы данных BNU: `Systems`, `Permissions`, `Devices`, `Dialers` и `Sysfiles`. Сначала следует задать файлы с помощью команды `/usr/sbin/uucp/uucpadm`, а потом изменить их по мере необходимости. С помощью файла `Sysfiles` можно задать в качестве файлов конфигурации BNU файлы, отличные от `/etc/uucp/Systems`, `/etc/uucp/Devices` и `/etc/uucp/Dialers`. Дополнительная информация приведена в разделе `Sysfile`.

3. Если такие сокращения телефонных номеров будут применяться в файлах `Systems`, задайте эти сокращения в файле `Dialcodes`. Дополнительная информация приведена в разделе Формат файла Dialcodes для BNU.

   Если для установления соединений BNU применяется TCP/IP, введите следующую команду **netstat** для проверки состояния демона **uucpd**:

   ```
   netstat -a
   ```

   Для запуска демона **uucpd** применяется демон **inetd**. Если демон **uucpd** не будет запущен, настройте демон **inetd** для запуска демона **uucpd**. Дополнительная информация приведена в разделе “Настройка демона inetd” на стр. 360).

4. С учетом списка устройств, созданного перед началом выполнения этой процедуры, измените файл `Devices` в своей системе. Создайте запись для каждого модема и каждого прямого соединения. Если вы используете TCP/IP, то убедитесь, что с записи TCP/IP в файле `Devices` сняты символы комментария. В файле `/etc/uucp/Sysfiles` вы можете указать другие файлы, в которых будет задана конфигурация устройств. Дополнительная информация о файле Devices приведена в разделе Формат файла Devices для BNU.

   Кроме того, в случае применения TCP/IP убедитесь, что файл `/etc/services` содержит следующую строку:

   ```
   uucp        540/tcp          uucpd
   ```

   При необходимости добавьте эту строку в файл.

5. С учетом информацией об удаленных системах, собранной перед началом выполнения этой процедуры, измените файл `Systems` в своей системе. Примеры конфигурации приведены в виде комментариев в файле `Systems`. При работе с TCP/IP, убедитесь, что таблица хостов в файле `/etc/hosts` содержит имя удаленной системы, к которой вы планируете подключаться. В файле `/etc/uucp/Sysfiles` вы можете указать другие файлы, в которых будет задана конфигурация систем.

6. С учетом информацией об устройствах и модемах перед началом выполнения этой процедуры, измените добавьте в `Dialers` записи для каждого модема. Если вы работаете с TCP/IP и прямыми соединениями, то убедитесь, что в файле представлены соответствующие записи. В файле `/etc/uucp/Sysfiles` вы можете указать другие файлы, в которых будет задана конфигурация номеронабирателей.

7. Решите, какие права доступа будут предоставлены удаленным системам, которые самостоятельно подключаются к локальной системе и с которыми устанавливается соединение. Задайте соответствующие записи для всех систем и имен, применяемых для входа в систему, в файле `Permissions`.

8. С помощью команды **uucheck** проверьте правильность каталогов, программ и файлов:

   ```
   /usr/sbin/uucp/uucheck -v
   ```

   Команда **uucheck** проверяет, все ли каталоги, программы и файлы заданы верно, и все ли записи файла `Permissions` согласованы между собой. Если команда **uucheck** обнаружит какие-либо ошибки, исправьте их.

9. Необязательно: Настройте автоматический контроль за операциями BNU и автоматический опрос удаленных систем. Дополнительная информация приведена в разделах “Настройка автоматического отслеживания BNU” и “Настройка опроса BNU удаленных систем” на стр. 446).

Настройка автоматического отслеживания BNU

Для запуска демонов BNU и отслеживания их работы применяется демон **cron**.

Предварительные требования

- Выполните инструкции из раздела “Настройка BNU” на стр. 441.
- Для изменения файла /var/spool/cron/crontabs/uucp необходимы права доступа пользователя root.

Демон **cron** считывает инструкции по запуску функций BNU из файла /var/spool/cron/crontabs/uucp.

Для настройки автоматического мониторинга BNU выполните следующие действия:

1. Войдите в систему как пользователь root.
2. С помощью текстового редактора измените файл /var/spool/cron/crontabs/uucp.
3. Удалите символ комментария из строк, предназначенных для запуска процедур обслуживания BNU uudemon.admin и uudemon.cleanup. В том случае, если для вашей системы необходимо более частое или более редкое обслуживание, вы можете изменить количество этих процедур. Рекомендуется вызывать команду uudemon.admin по крайней мере раз в день, а команду uudemon.cleanup - по крайней мере раз в неделю.
4. С помощью файла crontabs/uucp можно запланировать запуск других команд обслуживания BNU, в том числе **uulog**, **uuclean** и **uucleanup**. Кроме того, с помощью файла crontabs/uucp можно задать для демона **cron** время запуска демонов **uucico**, **uuxqt** или **uusched**.

Настройка опроса BNU удаленных систем

Для того чтобы программы BNU при поиске заданий выполняли опрос удаленных систем, перечислите эти системы в файле /etc/uucp/Poll.

Предварительные требования

- Выполните инструкции из раздела “Настройка BNU” на стр. 441.
- Для изменения файлов /var/spool/cron/crontabs/uucp и /etc/uucp/Poll необходимы права доступа пользователя root.

Помимо перечисления систем в файле /etc/uucp/Poll рекомендуется периодически выполнять команды **uudemon.hour** и **uudemon.poll**.

Для настройки опроса BNU удаленных систем выполните следующие действия:

1. Выберите удаленные системы, которые будут опрашиваться автоматически. Определите, как часто будет выполняться опрос. В файле Poll задайте для каждой системы опрос не реже раза в день.
2. Войдите в систему как пользователь root.
3. С помощью текстового редактора или команды **uucpadm** отредактируйте файл Poll. Добавьте запись для каждой опрашиваемой системы.

   Примечание: Системы, перечисленные в файле Poll, должны быть указаны и в файле /etc/uucp/Systems.
4. С помощью текстового редактора измените файл /var/spool/cron/crontabs/uucp. Удалите символы комментария (#) в строках с командами **uudemon.hour** и **uudemon.poll**. Вы можете изменить частоту вызова этих команд. Однако команда **uudemon.poll** должна вызываться примерно на пять минут раньше, чем команда **uudemon.hour**.

В указанное время BNU выполнит автоматический опрос систем, перечисленных в файле Poll.

Файл /etc/uucp/Systems

Удаленные системы перечислены в файлах /etc/uucp/Systems.

Файл /etc/uucp/Systems - это файл Systems по умолчанию. Системный администратор может задать дополнительные файлы с помощью файла **/etc/uucp/Sysfiles**.

Каждая запись файла Systems содержит следующие элементы:

- Имя удаленной системы
- Время, когда пользователи могут устанавливать соединения с удаленной системой.

- Способ подключения (напрямую или через модем)
- Скорость передачи данных по линии связи
- Информация, необходимая для входа в удаленную систему

Каждая запись файла Systems соответствует одной удаленной системе. Для того чтобы было возможно установить связь, удаленная система должна быть указана в файле Systems. Файл Systems должен существовать в каждой системе, применяющей BNU. Как правило, к файлу Systems может обращаться только пользователь root. Однако любой пользователь может просмотреть список имен удаленных систем BNU с помощью команды **uuname**.

Редактирование файла Devices для прямого соединения

Для изменения файла Devices для прямых соединений требуется наличие прав доступа root, также как и для редактирования файла /etc/uucp/Devices или другого файла, указанного в списке /etc/uucp/Sysfiles в качестве файла Devices.

Для настройки прямого соединения с указанием порта и имени удаленной системы задайте запись следующим образом:

1. В поле **Тип**, расположенном во второй строке записи, укажите имя удаленной системы, с которой вы хотите установить соединение с помощью прямой линии.
2. В поле **Линия** в обеих строках записи укажите имя устройства, применяемого для установления прямого соединения.
3. Введите символ **-** в поле **Линия2** в обеих строках.
4. В поле **Скорость передачи** в обеих строках записи укажите скорость передачи данных по прямому соединению в вашей системе.
5. Введите direct в поле **Программа набора номера-флаг** в обеих строках.

   Например:

   ```
   type device - speed direct
   ```

Добавьте в файл Devices записи для всех устройств, применяемых для подключения к удаленной системе.

Для настройки прямого соединения между двумя системами, использующими постоянное асинхронное последовательное соединение, создайте следующую запись:

1. В поле **Тип** укажите имя удаленной системы.
2. Во поле **Линия** укажите имя устройства терминала.
3. Введите символ **-** в поле **Линия2**.
4. В поле **Класс** укажите скорость передачи данных по прямому соединению в вашей системе.
5. Введите direct в поле **Программа набора номера-флаг**. Например:

   ```
   type device - speed direct
   ```

Добавьте в файл Devices записи для всех устройств, применяемых для подключения к удаленной системе.

Редактирование файла Devices для соединений с автоматическим набором номера

При редактировании файла /etc/uucp/Devices выполните следующие действия.

Для изменения файла /etc/uucp/Devices или любого другого файла, заданного в списке /etc/uucp/Sysfiles в качестве файла Devices, необходимы права доступа пользователя root.

В записях соединений, устанавливаемых с помощью телефонных линий, значение поля **Тип** должно соответствовать блоку автоматического набора номера (ACU). Введите ACU вместо поля **Тип** для всех удаленных соединений, установленных по телефонной линии. Для того, чтобы задать в файле Devices записи о соединениях с автоматическим набором номера, добавьте для каждого модема следующую строку:

1. В поле **Тип** введите ACU.
2. В поле **Линия** введите имя устройства, подключенного к модему.
3. В поле **Линия2** введите дефис -), если устройство автоматического набора номера - это не стандартное устройство 801. В противном случае укажите 801.
4. В поле **Класс** укажите скорость передачи в бодах, соответствующую вашему модему и линии или класс модема (например, D2400). Возможные значения скорости передачи в бодах: 300, 1200, 2400 и выше.

   Примечание: Если модем поддерживает несколько скоростей передачи данных, задайте отдельную запись для каждой скорости в файле Devices. Если модем может работать с любой скоростью, введите слово Any в поле **Скорость**.
5. В поле **Программа набора номера - Флаг** укажите имя модема в качестве **Программы набора номера**. Если в файле /etc/uucp/Systems или другом файле Systems, заданном в файле /etc/uucp/Sysfiles, номера телефонов будут указываться полностью, не заполняйте поле **Флаг**. В этом случае программа BNU будет применять флаг по умолчанию - \D. Если вы собираетесь использовать сокращенный телефонный код, заданный в файле /etc/uucp/Dialcodes, то введите ключ \T.

   Например:

   ```
   type line - speed dialer - token pair
   ```

Добавьте в файл Devices записи для всех соединений с удаленными системами, которые будут устанавливаться через модем.

Изменение файла Devices для TCP/IP

При редактировании файла /etc/uucp/Devices выполните следующие действия.

Для изменения файла /etc/uucp/Devices или любого другого файла, заданного в списке /etc/uucp/Sysfiles в качестве файла Devices, необходимы права доступа пользователя root.

Если для подключения к системам на сайте применяется протокол TCP/IP, то внесите соответствующую запись TCP/IP в файл Devices. Для работы с TCP/IP добавьте в файл Devices следующую строку:

```
TCP  -  -  -  TCP
```

Примеры: настройка BNU для соединения TCP/IPE

Эта группа примеров настраивает BNU для соединения TCP/IP.

Следующие файлы задают конфигурацию соединения TCP/IP между системами zeus и hera, причем zeus - локальная система, а hera - удаленная.

Файлы BNU для записей соединений TCP/IP в файле локальной системы:

Эти файлы BNU являются записями в локальной системе zeus.

- **Файл Systems:** Файл Systems в системе zeus содержит следующую запись, чтобы разрешить системе zeus обращаться к системе hera:

  ```
  hera Any TCP,t - - in:--in: uzeus word: birthday
  ```

  В этом примере показано, что система zeus может вызывать систему hera в любое время с помощью протокола **t** для связи с системой hera. Пользователь системы zeus входит в систему hera под именем uzeus с паролем birthday.

  Примечание: Протокол **t** поддерживает протокол **TCP**. По этой причине, в соединениях BNU типа TCP/IP всегда следует применять протокол **t**. Однако протокол **t** неприменим, если в поле **Тип** указано ACU (устройство автоматического вызова), или соединение устанавливается через модем.

  BNU выбирает устройство для соединения, используя поля **Тип** и **Класс** файла Systems. В файле Devices выполняется поиск записи типа TCP.

- **Файл Devices:** Файл Devices, применяемый демоном **uucico** в системе zeus, содержит следующую запись для соединений TCP/IP:

  ```
  TCP  -  -  -  TCP
  ```

  Поскольку задан тип устройства TCP, то записи *Класс*, *Линия* или *Линия2* отсутствуют. В поле *Программа набора номера* задано TCP. В файле Dialers выполняется поиск записи TCP.

- **Файл Dialers:** Файл Dialers, применяемый демоном **uucico** в системе zeus, содержит следующую запись TCP/IP:

  ```
  TCP
  ```

  Эта запись означает, что задавать конфигурацию программы набора номера не нужно.

  Примечание: Для соединений TCP/IP никогда не требуется задавать конфигурацию программы набора номера.

- **Файл Permissions:** Файл Permissions в системе zeus содержит следующую запись, предоставляющую системе hera доступ к системе zeus:

  ```
  LOGNAME=uhera SENDFILES=yes REQUEST=yes \
  MACHINE=zeus:hera VALIDATE=uhera \
  READ=/var/spool/uucppublic:/home/hera \
  WRITE=/var/spool/uucppublic:/home/hera COMMANDS=ALL
  ```

  Записи LOGNAME и MACHINE предоставляют системе hera следующие права доступа, если системы zeus и hera подключены:

  – Система hera может запрашивать и отправлять файлы независимо от того, какая система являлась инициатором установления соединения.

  – Система hera может считывать и записывать данные в общий каталог и в каталог /home/hera системы zeus.

  – Система hera может выполнять любые команды в системе zeus.

  – Пользователь системы hera должен входить в систему zeus под именем uhera и не может выполнять транзакции BNU под другим именем пользователя.

    Примечание: Так как права доступа не зависят от того, какая система является инициатором установления соединения, приведенные выше записи LOGNAME и MACHINE объединены. Если для систем hera и zeus настроена разные права доступа, то записи LOGNAME и MACHINE будут выглядеть следующим образом:

    ```
    LOGNAME=uhera VALIDATE=hera SENDFILES=yes REQUEST=yes \
    READ=/var/spool/uucppublic:/home/hera \
    WRITE=/var/spool/uucppublic:/home/hera

    MACHINE=zeus:hera REQUEST=yes COMMANDS=ALL\
    READ=/var/spool/uucppublic:/home/hera \
    WRITE=/var/spool/uucppublic:/home/hera
    ```

Файлы BNU для записей соединений TCP/IP в файле удаленной системы:

Эти файлы расположены в удаленной системе hera.

- **Файл Systems:** Файл Systems в системе hera содержит содержать следующую запись, почему чтобы разрешить системе hera доступ к системе zeus:

  ```
  zeus Any TCP,t - - ogin:--ogin: uhera ord: lightning
  ```

  В этом примере показано, что система hera может подключаться к системе zeus в любое время по протоколу **t**. Пользователь системы hera входит в систему zeus под именем uhera с паролем lightning. Соответственно, в файле Devices выполняется поиск записи типа TCP.

  Примечание: Протокол **t** поддерживает протокол **TCP**. По этой причине, в соединениях BNU типа TCP/IP всегда следует применять протокол **t**. Однако протокол **t** неприменим, если в поле *Тип* указано ACU (устройство автоматического вызова) или соединение устанавливается через модем.

- **Файл Devices:** Файл Devices, применяемый демоном **uucico** в системе hera, содержит следующую запись для соединений TCP/IP:

  ```
  TCP  -  -  -  TCP
  ```

  Поскольку задан тип устройства TCP, то записи *Тип*, *Линия* или *Линия2* отсутствуют. В поле *Программа набора номера* задано TCP. В файле Dialers выполняется поиск записи TCP.
- **Файл Dialers:** Файл Dialers, применяемый демоном **uucico** в системе hera, содержит следующую запись TCP/IP:

  ```
  TCP
  ```

  Эта запись означает, что задавать конфигурацию программы набора номера не нужно.

  Примечание: Для соединений TCP/IP никогда не требуется задавать конфигурацию программы набора номера.
- **Файл Permissions:**: Файл Permissions в системе hera содержит следующую запись, предоставляющую системе zeus доступ к системе hera:

  ```
  LOGNAME=uzeus SENDFILES=yes REQUEST=yes \
  MACHINE=hera:zeus VALIDATE=zeus COMMANDS=rmail:who:uucp
  ```

  Записи LOGNAME и MACHINE предоставляют системе zeus следующие права доступа, если системы zeus и hera подключены:
  - Система zeus может запрашивать и отправлять файлы независимо от того, какая система выступила инициатором соединения.
  - Система zeus может считывать и записывать данные только в общий каталог (каталог по умолчанию).
  - Система zeus может выполнять только команды **rmail**, **who** и **uucp**.
  - Пользователь системы zeus должен входить в систему hera под именем uzeus и не может выполнять транзакции BNU под другим именем пользователя.

    Примечание: Если для систем hera и zeus настроена разные права доступа, то записи LOGNAME и MACHINE будут выглядеть следующим образом:

    ```
    LOGNAME=uzeus VALIDATE=zeus SENDFILES=yes REQUEST=yes
    MACHINE=hera:zeus COMMANDS=rmail:who:uucp REQUEST=yes
    ```

Примеры: настройка BNU для соединения по телефонной линии

Следующие файлы задают конфигурацию модемного соединения по телефонной линии между системами venus и merlin.

venus - локальная система, а merlin - удаленная.

В обеих системах устройства tty1 подключены к модемам Hayes с быстродействием 1200 бод. ИД системы venus для входа в систему merlin - uvenus, пароль - mirror. ИД системы merlin для входа в систему venus - umerlin, пароль - oaktree. Номер телефона модема, подключенного к системе venus - 9=3251436; номер телефона модема, подключенного к системе merlin - 9=4458784. Оба компьютера хранят неполные номера телефонов в своих файлах Systems и телефонные коды в файлах Dialcodes.

Следующие файлы задают конфигурацию соединения между системами venus и merlin:

- **Файл Systems:** Файл Systems в системе venus содержит следующую запись для системы merlin, включающую номер телефона и код:

  ```
  merlin Any ACU 1200 local8784 "" in:--in: uvenus word: mirror
  ```

  Система venus может обратиться к системе merlin в любое время через устройство ACU с быстродействием 1200 бод, используя для входа в систему имя uvenus с паролем mirror. К номеру телефона добавляется код local в файле Dialcodes; применяемое устройство определяется на основе записей *Тип* и *Класс*. В файле Devices выполняется поиск устройства типа ACU и класса 1200.
- **Файл Dialcodes:** Файл Dialcodes в системе venus содержит следующий префикс для номера телефона в файле Systems:

```
local 9=445
```

Вместе с этим кодом номер телефона системы merlin в файле Systems выглядит как 9=4458784.

- **Файл Devices:** Файл Devices в системе venus содержит следующую запись для соединения с системой merlin:

  ```
  ACU  tty1  -  1200  hayes \T
  ```

  Будет применяться порт tty1, а для парамтера *Программа набора номера* в поле *Программы набора номера-флаг* указано значение hayes. Запись *Флаг*, \T, указывает, что к номеру телефона нужно добавить код из файла Dialcodes. В файлах Dialers выполняется поиск типа программы набора номера hayes.
- **Файл Dialers:** Файл Dialers, применяемый демоном **uucico** в системе venus, содержит следующую запись для модема hayes:

  ```
  hayes =,-, "" \dAT\r\c OK \pATDT\T\r\c CONNECT
  ```

  Примечание: Символы ожидания передачи определяются в файле Dialers.
- **Файл Permissions:** Файл Permissions в системе venus определяют способы, которыми система merlin может выполнять транзакции **uucico** и **uuxqt** в системе venus:

  ```
  LOGNAME=umerlin REQUEST=yes SENDFILES=yes \
  READ=/var/spool/uucppublic:/home/merlin \
  WRITE=/var/spool/uucppublic:/home/merlin
  MACHINE=venus:merlin VALIDATE=umerlin REQUEST=yes SENDFILES=yes    \
  COMMANDS=ALL \
  READ=/var/spool/uucppublic:/home/merlin \
  WRITE=/var/spool/uucppublic:/home/merlin
  ```

  Система merlin подключается к системе venus от имени пользователя umerlin. Система merlin может запрашивать и отправлять файлы независимо от того, какая система являлась инициатором установления соединения. Кроме того, система merlin может считывать и записывать данные в каталоги /var/spool/uucppublic и /home/merlin системы venus. Система merlin может выполнять в системе venus любые команды из набора команд по умолчанию.

Файлы BNU с записями для телефонных соединений в локальной системе:

Эти файлы в локальной системе venus содержат записи для установки телефонных соединений.

- **Файл Systems:** файл Systems в системе venus содержит следующую запись для системы merlin, включающую номер телефона и код:

  ```
  merlin Any ACU 1200 local8784 "" in:--in: uvenus word: mirror
  ```

  Система venus может обратиться к системе merlin в любое время через устройство ACU с быстродействием 1200 бод, используя для входа в систему имя uvenus с паролем mirror. К номеру телефона добавляется код local в файле Dialcodes; применяемое устройство определяется на основе записей *Тип* и *Класс*. В файле Devices выполняется поиск устройства типа ACU и класса 1200.
- **Файл Dialcodes:** Файл Dialcodes в системе venus содержит следующий префикс для номера телефона в файле Systems:

  ```
  local 9=445
  ```

  Вместе с этим кодом номер телефона системы merlin в файле Systems выглядит как 9=4458784.
- **Файл Devices:** Файл Devices в системе venus содержит следующую запись для соединения с системой merlin:

  ```
  ACU  tty1  -  1200  hayes \T
  ```

  Будет применяться порт tty1, а для парамтера *Программа набора номера* в поле **Программы набора номера-флаг** указано значение hayes. Запись *Флаг*, \T, указывает, что к номеру телефона нужно добавить код из файла Dialcodes. В файлах Dialers выполняется поиск типа программы набора номера hayes.
- **Файл Dialers:** Файл Dialers, применяемый демоном **uucico** в системе venus, содержит следующую запись для модема hayes:

  ```
  hayes =,-, "" \dAT\r\c OK \pATDT\T\r\c CONNECT
  ```

  Примечание: Символы ожидания передачи определяются в файле Dialers.

- **Файл Permissions:** Файл Permissions в системе venus определяют способы, которыми система merlin может выполнять транзакции **uucico** и **uuxqt** в системе venus:

  ```
  LOGNAME=umerlin REQUEST=yes SENDFILES=yes \
  READ=/var/spool/uucppublic:/home/merlin \
  WRITE=/var/spool/uucppublic:/home/merlin
  MACHINE=venus:merlin VALIDATE=umerlin REQUEST=yes SENDFILES=yes    \
  COMMANDS=ALL \
  READ=/var/spool/uucppublic:/home/merlin \
  WRITE=/var/spool/uucppublic:/home/merlin
  ```

  Система merlin подключается к системе venus от имени пользователя umerlin. Система merlin может запрашивать и отправлять файлы независимо от того, какая система являлась инициатором установления соединения. Кроме того, система merlin может считывать и записывать данные в каталоги /var/spool/uucppublic и /home/merlin системы venus. Система merlin может выполнять в системе venus любые команды из набора команд по умолчанию.

Файлы BNU с записями для телефонных соединений в удаленной системе:

Эти файлы в удаленной системе merlin содержат записи для установки телефонных соединений.

- **Файл Systems:** файл Systems в системе merlin содержит следующую запись для системы venus, включающую номер телефона и код:

  ```
  venus Any ACU 1200 intown4362 "" in:--in: umerlin word: oaktree
  ```

  Система merlin может обратиться к системе venus в любое время через устройство ACU с быстродействием 1200 бод, войдя под именем umerlin с паролем oaktree. К номеру телефона добавляется код intown в файле Dialcodes; применяемое устройство определяется на основе записей *Тип* и *Класс*. В файле Devices выполняется поиск устройства типа ACU и класса 1200.
- **Файл Dialcodes:** Файл Dialcodes в системе merlin содержит следующий префикс для номера телефона в файле Systems:

  ```
  intown 9=325
  ```

  Таким образом, полным номером телефона системы venus будет 9=3254362.
- **Файл Devices:** Файл Devices в системе merlin содержит следующую запись для соединения с системой venus:

  ```
  ACU  tty1  -  1200  hayes \T
  ```

  Устройство ACU подключено к порту tty1; применяется программа набора номера hayes. К номеру телефона добавляется информация из файла Dialcodes. В файлах Dialers выполняется поиск записи для модема hayes.
- **Файл Dialers:** Файл Dialers, применяемый демоном **uucico** в системе merlin, содержит следующую запись для модема:

  ```
  hayes =,-, "" \dAT\r\c OK \pATDT\T\r\c CONNECT
  ```
- **Файл Permissions:** Файл Permissions в системе merlin содержит следующие записи, предоставляющие системе venus права доступа к системе merlin:

  ```
  LOGNAME=uvenus SENDFILES=call REQUEST=no \
  WRITE=/var/spool/uucppublic:/home/venus \
  READ=/var/spool/uucppublic:/home/venus
  MACHINE=merlin:venus VALIDATE=uvenus  \
  READ=/ WRITE=/ COMMANDS=ALL REQUEST=yes \
  NOREAD=/etc/uucp:/usr/etc/secure \
  NOWRITE=/etc/uucp:/usr/etc/secure
  ```

Примеры: настройка BNU для прямого соединения

Файлы из следующего примера задают конфигурацию прямого соединения между системами zeus и hera, причем zeus - локальная система, а hera - удаленная.

В системе zeus применяется устройство с прямым подключением tty5. Система hera содержит прямое устройство tty1. Скорость передачи данных составляет 1200 бод. ИД системы zeus для входа в систему hera - uzeus, пароль - thunder. ИД системы hera для входа в систему zeus - uhera, пароль - portent.

Файлы BNU с прямым соединением в файлах локальной системы:

Эти файлы в локальной системе zeus содержат записи для установки телефонных соединений.

- **Файл Systems:** Файл Systems в системе zeus содержит следующую запись для соединения с системой hera:

  ```
  hera Any hera 1200 - "" \r\d\r\d\r in:--in: uzeus word:  thunder
  ```

  Эта запись означает, что система hera может обратиться к системе zeus в любое время по прямому соединению, указанному в файле Devices. Поиск записи в файле Devices выполняется согласно третьему и четвертому полям записи Systems. Таким образом, в файле Devices выполняется поиск записи, для которой в поле *Тип* указано значение hera, а в поле *Класс* - значение 1200. Пользователь системы zeus входит в систему hera под именем uzeus с паролем thunder.
- **Файл Devices:** Файл Devices в системе zeus содержит следующую запись для соединения с системой hera:

  ```
  hera    tty5  -  1200  direct
  ```

  Эта запись означает, что система zeus устанавливает соединение с системой hera через устройство tty5 с быстродействием 1200 бит/с. Учтите, что значение *Программа набора номера* в обоих полях **пары Программа набора номера-флаг** равно direct. При подключении к системе hera в файле Dialers выполняется поиск записи direct.
- **Файл Dialers:** Файл Dialers в системе zeus содержит следующую запись для выделенного соединения:

  ```
  direct
  ```

  Эта запись указывает, что при установлении прямого соединения квитирование не требуется.
- **Файл Permissions:** Файл Permissions в локальной системе zeus определяют способы, которыми система hera может выполнять транзакции **uucico** и **uuxqt** в системе zeus:

  ```
  LOGNAME=uhera MACHINE=hera VALIDATE=uhera REQUEST=yes \
  SENDFILES=yes MACHINE=zeus READ=/ WRITE=/ COMMANDS=ALL
  ```

  Эта запись указывает, что пользователи системы hera входят в другую систему под именем uhera. Так как задана опция VALIDATE=uhera, система hera не может войти в систему zeus под другим ИД, и любая другая удаленная система не может использовать ИД uhera. Система hera может считывать и записывать данные в любой каталог в системе zeus, а также запрашивать и отправлять файлы независимо от того, какая система инициализировала соединение. Система hera может также выполнять любые команды в системе zeus.

  Примечание: Так как предоставляемые права доступа не зависят от того, какая система является инициатором соединения, записи LOGNAME и MACHINE объединены. Если для систем hera и zeus настроена разные права доступа, то записи LOGNAME и MACHINE будут выглядеть следующим образом:

  ```
  LOGNAME=uhera REQUEST=yes SENDFILES=yes READ=/ WRITE=/
  MACHINE=zeus:hera VALIDATE=uhera READ=/ WRITE=/ REQUEST=yes \
  COMMANDS=ALL
  ```

  Внимание: Предоставление прав доступа в предыдущем примере эквивалентно предоставлению каждому пользователю удаленной системы ИД для входа в локальную систему. Столь либеральный подход может привести к нарушению защиты, поэтому такие права доступа следует предоставлять только проверенным удаленным системам, находящимся рядом с вашей.

Файлы BNU с прямым соединением в файлах удаленной системы:

Эти файлы в удаленной системе hera содержат записи для установки телефонных соединений.

- **Файл Systems:** Файл Systems в системе hera содержит следующую запись для системы zeus:

  ```
  zeus Any zeus 1200 - "" \r\d\r\d\r in:--in: uhera word: portent
  ```

  Эта запись означает, что система hera может обратиться к системе zeus в любое время по прямому соединению, указанному в файле Devices. Поиск записи в файле Devices выполняется согласно третьему и четвертому полям записи Systems. Таким образом, в файле Devices выполняется поиск записи, для которой в поле **Тип** указано значение zeus, а в поле **Класс** - значение 1200. Пользователи системы hera входят в систему zeus под именем uhera с паролем portent.

- **Файл Devices:** файл Devices в системе hera содержит следующую запись для взаимодействия с системой zeus:

  ```
  zeus    tty1  -  1200  direct
  ```

  Эта запись указывает, что система hera устанавливает соединение с системой zeus с помощью устройства tty1 с быстродействием 1200 бит/с. Так как в поле *Программа набора номера* указано direct, в файлах Dialers выполняется поиск записи direct.
- **Файл Dialers:** Файл Dialers в системе hera содержит следующую запись для выделенного соединения:

  ```
  direct
  ```

  Эта запись указывает, что при установлении прямого соединения задавать конфигурацию программы набора номера не требуется.
- **Файл Permissions:** Файл Permissions в системе hera определяет способы которыми система zeus может выполнять транзакции **uucico** и **uuxqt** в системе hera:

  ```
  LOGNAME=uzeus REQUEST=yes SENDFILES=yes READ=/ WRITE=/
  MACHINE=hera:zeus VALIDATE=uzeus REQUEST=yes COMMANDS=ALL READ=/\
  WRITE=/
  ```

  Эти записи означают, что система zeus входит в систему hera под именем uzeus. Так как в записях указан параметр VALIDATE=uzeus, система zeus не может войти в систему hera под другим ИД, и любая другая удаленная система не может использовать ИД uzeus. Система zeus может считывать и записывать данные в любой каталог в системе hera, и может запрашивать и отправлять файлы независимо от того, какая система инициализировала соединение. Система zeus может также выполнять любые команды в системе hera.

  Внимание: Предоставление прав доступа в предыдущем примере эквивалентно предоставлению каждому пользователю удаленной системы ИД для входа в локальную систему. Столь либеральный подход может привести к нарушению защиты, поэтому такие права доступа следует предоставлять только проверенным удаленным системам, находящимся рядом с вашей.

Обслуживание BNU

Для того чтобы утилиты BNU работали правильно, необходимо регулярно выполнять процедуры обслуживания.

Обслуживание BNU заключается в следующем:

- Периодическое чтение и удаление файлов протоколов.
- Просмотр содержимого очередей BNU с помощью команд **uuq** и **uustat** для проверки правильности передачи заданий в удаленные системы.
- Планирование автоматического выполнения команд, опрашивающих удаленные системы при поиске заданий, возвращающих неотправленные файлы пользователям, и выдающих сообщения о состоянии BNU.
- Периодическое обновление файлов конфигурации для отражения изменений, происходящих в системе.

Кроме того, вы должны время от времени обмениваться информацией с администраторами удаленных систем, чтобы узнавать об изменениях, которые могут потребовать внесения изменений в конфигурацию вашей системы. Например, если администратор системы venus изменил пароль вашей системы, то вы должны поместить новый пароль в файл /etc/uucp/Systems (или альтернативный файл, указанный в /etc/uucp/Sysfiles) до подключения к системе venus.

Файлы протокола BNU

Для отслеживания выполняемых операций BNU создает файлы протоколов и файлы сообщений об ошибках.

Эти файлы необходимо периодически просматривать и удалять с целью экономии места на диске. В BNU предусмотрены следующие команды очистки файлов протоколов:

- uulog
- uuclean

- uucleanup
- uudemon.cleanu.

Вы можете запускать эти команды вручную, либо создать записи в файле /var/spool/cron/crontabs/uucp для запуска команд с помощью демона **cron**.

Файлы протокола в каталогах .Log и .Old:

BNU размещает файлы протоколов в каталоге /var/spool/uucp/.Log.

Эти файлы создаются для всех удаленных систем с помощью команд **uucp**, **uucico**, **uux** и **uuxqt**. BNU помещает информацию о состоянии каждой транзакции в соответствующий файл протокола при любом обращении к BNU в системе. Если выполняется несколько процессов BNU, то система не может обратиться к файлу протокола. Информация о состоянии помещается в отдельный файл с префиксом .LOG.

Команда **uulog** выдает краткую информацию о запросах **uucp** или **uux**, отсортированную по именам пользователей или систем. Команда **uulog** показывает список файлов. Однако вы можете выбрать и автоматическое объединение файлов протоколов в основной файл протокола. Такую процедуру называют *сжатием* файлов протоколов. Ее можно выполнить с помощью команды **uudemon.cleanu**, обычно запускаемой демоном **cron**.

Демон **cron** запускает команду **uudemon.cleanu**. Команда **uudemon.cleanu** объединяет файлы протокола **uucico** и **uuxqt** в локальной системе и помещает их в каталог /var/spool/uucp/.Old. Одновременно с этим команда удаляет старые файлы протоколов, которые ранее хранились в каталоге .Old. По умолчанию команда **uudemon.cleanu** хранит файлы протокола два дня.

Если в системе недостаточно памяти, попробуйте уменьшить продолжительность хранения файлов. Если же вам необходима информация о транзакциях BNU за больший период времени, попробуйте увеличить эту продолжительность. Для изменения продолжительности хранения файлов протокола по умолчанию отредактируйте сценарий оболочки для команды **uudemon.cleanu**. Сценарий хранится в каталоге /usr/sbin/uucp; для его изменения необходимы права доступа root.

Файлы протокола BNU /.Admin:

BNU собирает разного рода информацию и помещает ее в каталог /var/spool/uucp/.Admin. В этом каталоге хранятся файлы errors, xferstats, Foreign и audit.

Эти файлы необходимо время от времени просматривать и удалять с целью экономии места на диске. BNU создает каждый файл при необходимости.

Если к локальной системе обращается удаленная система, в которой работает демон **uucico** в режиме отладки, то в локальной системе также запускается демон **uucico** в режиме отладки. Сообщения отладки, создаваемые демоном в локальной системе, записываются в файл audit. Размер этого файла постоянно увеличивается. Рекомендуется периодически просматривать и удалять файл audit.

В файле errors хранится информация об ошибках, обнаруженных демоном **uucico**. Этот файл применяется при устранении ошибок, например, в случае неверных прав доступа в рабочих файлах BNU.

Файл xferstats содержит информацию о состоянии каждой операции передачи файлов. Время от времени просматривайте и удаляйте этот файл.

Файл Foreign играет важную роль в защите системы. Каждый раз, когда неизвестная система пытается войти в локальную систему, BNU вызывает процедуру оболочки remote.unknown. Эта процедура регистрирует попытку входа в систему в файле Foreign. Таким образом, файл Foreign содержит имена систем, которые попытались вызвать локальную систему, но получили отказ. Если система часто пытается вызвать вашу систему, учтите это при решении вопроса о том, следует ли разрешить доступ данной системе.

Системные файлы протокола, используемые BNU:

Так как для работы со многими функциями BNU необходимы права доступа пользователя root, BNU часто создает записи в файле протокола `/var/spool/sulog log`.

Аналогично, при планировании задач BNU с помощью демона **cron** большое число записей заносится в файл `/var/spool/cron/log`. При работе с BNU периодически просматривайте и очищайте эти файлы.

Команды обслуживания BNU

Основные сетевые утилиты предоставляют несколько команд, предназначенных для отслеживания операций BNU, а также очистки каталогов и файлов BNU.

Команды очистки BNU:

В BNU предусмотрены три команды очистки каталогов и удаления неотправленных файлов.

Элемент	Описание
uuclean	Удаляет все файлы, возраст которых превышает заданное количество часов, из административных каталогов BNU. В команде **uuclean** вы можете указать каталог для очистки или тип удаляемого файла. Кроме того, вы можете указать опцию уведомления владельцев удаленных файлов. Команда **uuclean** - это эквивалент команды **uucleanup** в программном обеспечении BSD.
uucleanup	Выполняет функции, аналогичные команде **uuclean**. Однако команда **uucleanup** подсчитывает возраст файлов в *днях*, а не в часах. С помощью команды **uucleanup** вы можете отправить предупреждающее сообщение пользователям, файлы которых не были отправлены, и сообщить им о том, что файлы по-прежнему находятся в очереди. Кроме того, команда **uucleanup** удаляет файлы, относящиеся к указанной удаленной системе.
uudemon.cleanu	Это процедура оболочки, выполняющая команды **uulog** и **uucleanup** для сжатия файлов протоколов BNU и удаления файлов протоколов и рабочих файлов, возраст которых превышает три дня. Команда **uudemon.cleanu** запускается демоном **cron**.

Команды проверки состояния BNU:

В BNU предусмотрены команды проверки состояния передачи и файлов протоколов.

Элемент	Описание
uuq	Показывает задания, находящиеся в очереди заданий BNU. С помощью команды **uuq** вы можете узнать состояние указанного задания или всех заданий. Обладая правами доступа root, вы можете с помощью команды **uuq** удалить задание из очереди.
uustat	Предоставляет ту же информацию, что и команда **uuq**, но в другом формате. Команда **uustat** применяется для просмотра информации о состоянии заданий и удаления собственных заданий. Обладая правами доступа root, вы можете удалить и задания, принадлежащие другим пользователям.
uulog	Выдает краткую информацию о запросах **uucp** или **uux** по всем пользователям или системам. Команда **uulog** показывает имена файлов. См. раздел “Файлы протокола BNU” на стр. 454.
uupoll	Опрашивает удаленную систему. Применяется в случае, когда в очереди накопилась информация для обработки, которую необходимо передать в удаленную систему, а время автоматического вызова удаленной системы по расписанию пока не наступило.
uusnap	Выдает очень краткую информацию о состоянии BNU. Для каждой удаленной системы эта команда показывает количество файлов, ожидающих передачи. Однако она не указывает, сколько времени продолжается ожидание. В BSD команда **uusnap** служит эквивалентом команды **uustat**.

Процедуры оболочки BNU:

BNU поставляется с двумя процедурами оболочки, применяемыми в целях обслуживания:

Элемент	Описание
uudemon.cleanu	Эта процедура описана в разделе “Команды очистки BNU” на стр. 456.
uudemon.admin	Запускает команду **uustat**, которая сообщает о состоянии заданий BNU. Результаты передаются пользователю uucp как почтовое сообщение. Вы можете изменить процедуру оболочки **uudemon.admin** так, чтобы почта рассылалась всем пользователям, или с помощью программы передачи почты перенаправить всю почту, предназначенную пользователю uucp, пользователю, отвечающему за администрирование BNU.

Указанные процедуры оболочки хранятся в каталоге `/usr/sbin/uucp`. Если вы хотите изменить их, скопируйте их в другой каталог и внесите изменения в копии. Эти процедуры можно запустить из командной строки, либо добавить в расписание демона **cron**.

Если вы хотите, чтобы команды **uudemon.cleanu** и **uudemon.admin** запускались автоматически, удалите символы комментария (#) из соответствующих строк файла `/var/spool/cron/crontabs/uucp`.

Имена каталогов BNU

Имена каталогов в основных сетевых утилитах (BNU) можно указывать различными способами.

Имя каталога может быть задано либо полностью, начиная с корневого каталога, либо кратко, с помощью имени удаленной системы. Для каждого способа определены свои собственные требования.

Полное имя файла

Полное имя целевого файла или каталога задается начиная с корневого каталога и включает перечень всех каталогов более низкого уровня.

Например, имя `/etc/uucp/Devices` соответствует файлу `Devices`, находящемуся в подкаталоге `uucp` каталога `etc`, который в свою очередь, находится в корневом каталоге.

Перед именем корневого каталога всегда должна быть указана косая черта (/). Все элементы имени также должны разделяться символами косой черты (/).

Относительные пути к файлам

В относительном имени каталоги перечисляются начиная с текущего каталога.

Например, если текущий каталог - `/usr/bin`, а целевой - `/usr/bin/reports`, то относительным именем целевого каталога будет `reports` (без косой черты).

Относительные имена могут применяться с командами **cu**, **uucp** и **uux**, а также с именем исходного файла в команде **uuto**.

> **Примечание:** Не все команды BNU могут работать с относительными именами. Если после ввода относительного имени вы не добились желаемого результата, то введите команду еще раз, указав в ней полное имя.

~ [опция] путь к файлу

Значение `~ [опция]` задает домашний каталог указанного пользователя.

Тильда (~) может применяться для быстрого доступа к определенным каталогам.

Например, имя `~jane` задает домашний каталог пользователя `jane`. Имя `~uucp` или `~` (одна тильда) задает общий каталог BNU в удаленной системе. Полный путь к общему каталогу BNU - `/var/spool/uucppublic`.

> **Примечание:** Не путайте функцию тильды, описанную в этом примере, с другими функциями, применяемыми в программе BNU. При работе с удаленной системой с помощью команды **cu** тильда указывается перед командами, если их необходимо выполнить в локальной системе.

имя-системы! путь к файлу
Имя-системы! задает путь к файлу в другой системе.

Например, значение distant!/account/march указывает файл march, находящийся в каталоге account удаленной системы distant.

имя-системы!имя-системы! путь к файлу
Имя-системы!*имя-системы*! задает путь через несколько систем.

Например, если соединение с системой distant можно установить только через систему near, то путь к файлу будет следующим: near!distant!/account/march.

Имена систем должны разделяться восклицательными знаками (!). Если путь содержит несколько систем, то правило разделения элементов символами косой черты (/) неприменимо к именам систем. Тем не менее, это правило остается в силе для каталогов и файлов целевой системы.

> **Примечание:** При работе с оболочкой bourne имена систем следует разделять восклицательными знаками (!). При работе с BNU в оболочках C и korn перед восклицательным знаком нужно указывать обратную косую черту (\). Обратная косая черта - это управляющий символ, означающий, что следующий символ следует рассматривать как обычный, а не как специальный.

Демоны BNU
В состав программы BNU входят четыре демона, расположенные в каталоге /usr/sbin/uucp.

Элемент	Описание
uucico	Обеспечивает передачу файлов (обратитесь к разделу “Демон uucico”)
uusched	Обеспечивает планирование обработки файлов, помещенных в локальный каталог буферизации (обратитесь к разделу “Демон uusched” на стр. 459)
uuxqt	Обеспечивает удаленное выполнение команд (обратитесь к разделу “Демон uuxqt” на стр. 459)
uucpd	Обеспечивает работу с соединениями TCP/IP (обратитесь к разделу “Демон uucpd” на стр. 459)

Демоны **uucico**, **uusched** и **uuxqt** запускаются демоном **cron** в соответствии с расписанием, составленным администратором BNU. При наличии прав доступа root вы также можете запускать эти демоны вручную. Демон **uucpd** следует запускать с помощью демона TCP/IP **inetd**.

Демон uucico
Демон **uucico** передает фалы из одной системы в другую.

Команды **uucp** и **uux** запускают демон **uucico** для передачи командных файлов, файлов данных и выполняемых файлов в указанную систему. Кроме того, демон **uucico** периодически запускается планировщиком BNU - демоном **uusched**. Будучи запущенным демоном **uusched**, демон **uucico** пытается установить соединение с другими системами и выполнить инструкции из командных файлов.

Перед выполнением инструкций из командных файлов демон **uucico** считывает информацию о целевой системе из файла /etc/uucp/Systems (либо одного или нескольких альтернативных файлов, заданных /etc/uucp/Sysfiles). Затем демон проверяет допустимость времени вызова по записи в файле Systems. Если время допустимо, демон **uucico** находит значения *Тип* и *Класс* и ищет подходящее устройство в файле /etc/uucp/Devices (или в одном или нескольких файлах, указанных в /etc/uucp/Sysfiles).

Обнаружив устройство, демон **uucico** проверяет, нет ли в каталоге /var/locks файла блокировки устройства. Если такой файл существует, демон проверяет, нет ли другого устройства с подходящими типом и быстродействием.

Если других подходящих устройств нет, демон возвращается к файлу Systems и ищет в нем другую запись удаленной системы. Если такая запись существует, демон повторяет поиск устройства. Если нет, демон создает запись в файле /var/spool/uucp/.Status/*имя_системы* для этой удаленной системы и переходит к обработке следующего запроса. Командный файл остается в очереди. Демон **uucico** повторяет попытку передачи позже. Такая попытка называется *повтором*.

Когда демон **uucico** подключается к удаленной системе, он входит в нее, руководствуясь инструкциями из файла Systems. В частности, при этом в удаленной системе запускается экземпляр демона **uucico**.

Пара демонов **uucico**, работающих в обеих системах, обеспечивает передачу данных. Демон **uucico** в исходной системе управляет каналом передачи данных и отправляет запросы на обработку. Демон **uucico** в удаленной системе проверяет локальные права доступа, выясняя, допускают ли они обработку запроса. Если да, то начинается передача файла.

После того, как демон **uucico** вызывающей системы заканчивает передачу всех запросов, он отправляет запрос на окончание связи. Если у демона **uucico** в удаленной системе есть транзакции, которые необходимо отправить в исходную систему, он отклоняет запрос об окончании связи, после чего два демона меняются ролями.

Примечание: Как в файле /etc/uucp/Permissions локальной системы, так и в файле /etc/uucp/Permissions удаленной системе может быть установлен запрет на изменение ролей демонов. В этом случае удаленная система должна отложить передачу файлов до того момента, когда она вызовет локальную систему.

Когда файлов для передачи в обоих направлениях не остается, демоны **uucico** завершают связь. После этого для обработки запросов на выполнение удаленных команд вызывается демон **uuxqt** (“Демон uuxqt”).

В течение всего процесса передачи демоны **uucico** в обеих системах заносят сообщения в протокол BNU и в файлы сообщений об ошибках.

Демон uusched

Демон **uusched** планирует передачу файлов, находящихся в очереди в каталоге буферизации

/var/spool/uucppublic локальной системы. При инициализации демона **uusched** он находит командные файлы в каталоге буферизации, располагает их в случайной последовательности и запускает демон **uucico**. Затем демон **uucico** передает файлы.

Демон uuxqt

Когда пользователь вводит команду **uux** для выполнения заданной команды в удаленной системе, эту команду выполняет демон **uuxqt**.

Создав необходимые файлы, команда **uux** запускает демон **uucico**, передающий эти файлы в общий каталог буферизации в удаленной системе.

Демон **uuxqt** периодически просматривает каталог буферизации, выясняя, нет ли в нем запросов на выполнение команд, поступивших из подключенных систем. При обнаружении такого запроса демон **uuxqt** проверяет наличие необходимых файлов и прав доступа. Если права доступа есть, демон запускает указанную команду.

Демон uucpd

Для того, чтобы программа BNU могла установит связь с удаленным компьютером с помощью **Протокола управления передачей/Протокола Internet (TCP/IP)**, должна существовать возможность запуска на удаленном компьютере демона **uucpd**.

Демон **uucpd** - это субсервер демона TCP/IP **inetd** и запускается им.

По умолчанию строка вызова демона **uucpd** закомментирована в файле `inetd.conf`. Для работы с этим демоном нужно удалить символ комментария и перезапустить приложение **inetd**. Однако, если этот параметр изменялся в локальной системе, то вам может потребоваться заново настроить демон **inetd** на запуск демона **uucpd**.

Защита BNU

Поскольку другие системы устанавливают связь с вашей системой, передают в нее файлы и выполняют в ней команды, в BNU предусмотрен механизм защиты.

Он позволяет ограничить набор действий, которые пользователи удаленных систем могут выполнять в локальной системе (в свою очередь, пользователи удаленных систем могут ограничить набор действий, разрешенных вам в их системах). Защитные действия выполняются несколькими демонами BNU, которые хранят необходимые им файлы в административных каталогах. Кроме того, в BNU ведется протокол выполняемых операций.

Защита в BNU реализована на нескольких уровнях. При настройке BNU вы можете задать следующие параметры:

- Каким пользователям в вашей системе будет разрешен доступ к файлам BNU.
- С какими удаленными системами будет разрешено устанавливать соединение.
- Каким образом пользователи удаленных систем будут входить в вашу систему.
- Какие действия пользователи удаленных систем смогут выполнять в вашей системе.

ИД пользователей uucp

При установке BNU все файлы конфигурации и демоны, а также многие команды и процедуры оболочки принадлежат пользователю uucp.

ИД пользователя uucp соответствует ИД пользователя 5 и ИД группы 5. Демон **cron** считывает содержимое файла `/var/spool/cron/crontabs/uucp` и на его основе планирует автоматическое выполнение заданий BNU.

Обычно вход в систему с именем uucp запрещен. Для изменения файлов, принадлежащих пользователю uucp, войдите в систему с правами доступа root.

Внимание: Разрешая удаленным пользователям входить в локальную систему под именем uucp, вы серьезно ослабляете защиту локальной системы. Такие пользователи могут просматривать и, возможно, изменять локальные файлы `Systems` и `Permissions`, в зависимости от других прав доступа, указанных в записи LOGNAME. Настоятельно рекомендуется создать в BNU другие ИД пользователей для удаленных систем, зарезервировав ИД uucp для администратора BNU локальной системы. Надежная защита достигается в том случае, когда каждой удаленной системе выделен свой идентификатор для входа в локальную систему, с которым связан уникальный номер UID.

Для передачи файлов по умолчанию применяется идентификатор nuucp.

ИД входа в систему BNU

Начальной оболочкой для пользователей BNU является демон **uucico** (`/usr/sbin/uucp/uucico`).

Когда удаленные системы обращаются к вашей системе, они автоматически запускают в ней демон **uucico**. Пользователям BNU присвоен ИД группы uucp 5.

Идентификаторам входа в систему, применяемым удаленными системами, необходимы пароли. Для того чтобы у нового пользователя BNU из удаленной системы при входе в вашу систему не запрашивался новый пароль, вы должны задать пароль сразу после создания учетной записи. Для этого введите команду **passwd**, а затем - команду **pwdadm**. Например, для того чтобы задать пароль для пользователя `nuucp`, войдите в систему как пользователь root и вызовите следующие команды:

```
passwd nuucp
pwadm -f NOCHECK
nuucp
```

Появится приглашение указать пароль для пользователя nuucp. После того как это будет сделано, удаленная система сможет установить связь с вашей системой без необходимости немедленно указывать новый пароль (работающий в пакетном режиме пользователь nuucp просто не сможет это сделать).

Создав ИД пользователя для удаленной системы, сообщите ИД и пароль администратору BNU этой удаленной системы.

Пользователь root может задать ИД администратора BNU. Это полезно, если вы хотите предоставить функции администратора BNU пользователю без прав доступа root. У административного пользователя BNU должен быть пароль, ИД пользователя 5 и ИД группы uucp 5. Начальной оболочкой для административного пользователя должна быть программа /usr/bin/sh (а не демон **uucico**). ИД пользователя 5 предоставляет пользователю BNU такие же права доступа, что и у ИД пользователя **uucp**. По этой причине, удаленным системам не следует разрешать входить в систему с ИД администратора BNU, иначе может произойти нарушение защиты.

Защита данных, файлы Systems и remote.unknown

В большинстве систем BNU входить в локальную систему могут только удаленные системы, перечисленные в файле /etc/uucp/Systems или в заменяющем его файле, указанном в Sysfiles.

Если к локальной системе обращается неизвестная система, то во всех случаях выполняется сценарий /usr/sbin/uucp/remote.unknown. Данный сценарий запрещает вход в систему неизвестной системе и заносит в файл /var/spool/uucp/.Admin/Foreign запись о времени попытки входа в систему.

Пользователь root или администратор BNU может изменить сценарий remote.unknown для сохранения большего объема информации или для записи этой информации в другой файл. Например, вы можете сделать так, чтобы при каждой попытке неизвестной системы установить связь с вашей системой процедура оболочки отправляла сообщение администратору BNU.

Если вы запретите выполнение процедуры remote.unknown, то неизвестные компьютеры смогут подключаться к вашей системе. В этом случае вам следует добавить запись MACHINE=OTHER в файл /etc/uucp/Permissions, задав права доступа для неизвестных компьютеров.

Ваша система может устанавливать связь только с удаленными системами, перечисленными в файле Systems. Это позволяет предотвратить подключение пользователей вашей системы к неизвестным системам.

Защита и файл Permissions

При работе с файлом Permissions обратите внимание не следующие замечания, касающиеся защиты.

Файл /etc/uucp/Permissions определяет следующие параметры:

- Имена удаленных пользователей, которым разрешен вход в систему
- Разрешенные команды и права доступа для удаленных систем, подключающихся к локальной системе.

Файл /etc/uucp/Permissions содержит записи двух типов:

Элемент	Описание
LOGNAME	Задает пользователей, которым разрешен вход в систему, и предоставляемые им права доступа. Записи LOGNAME применяются в тот момент, когда удаленная система вызывает локальную систему и пытается в нее войти.
MACHINE	Определяет имена компьютеров и предоставляемые им права доступа. Записи MACHINE применяются в тот момент, когда удаленная система пытается выполнить команды в локальной системе.

Опции файла `Permissions` позволяют устанавливать различные уровни защиты для каждой удаленной системы. Например, если несколько удаленных систем используют один и тот же ИД для входа в локальную систему, то с помощью опции VALIDATE вы можете указать, что каждая удаленная система должна применять уникальный ИД входа в систему. Опции SENDFILES, REQUEST и CALLBACK указывают, какая система является управляющей, и с их помощью можно при необходимости передать управление транзакциями локальной системе.

Опции READ, WRITE, NOREAD и NOWRITE определяют доступ к конкретным каталогам локальной системы. Кроме того, они управляют каталогом локальной системы, в котором удаленные пользователи размещают свои данные. Опция COMMANDS ограничивает число команд, которые пользователи удаленных систем могут выполнять в локальной системе. Опция COMMANDS=ALL предоставляет полные права доступа к системам, тесно связанным с вашей.

Внимание: Применение опции COMMANDS=ALL значительно ослабляет защиту вашей системы.

Связь между локальной и удаленной системой

Для установления связи между удаленной и локальной системой удаленная система должна иметь подключение через модем или выделенное соединение с локальной системой, установленную операционную систему на базе UNIX, а также BNU или другую версию программы копирования UNIX-UNIX (UUCP).

> **Примечание:** BNU может применяться для установления соединений с операционными системами, отличными от UNIX, но для этого может потребоваться дополнительное аппаратное или программное обеспечение.

В BNU предусмотрено две команды, позволяющие устанавливать соединения с удаленными системами. Команда **cu** устанавливает соединение через специальный канал связи или телефонную линию. Команда **ct** устанавливает соединение только через телефонную линию, с помощью модема.

Если известен номер телефона или имя целевой системы, то для установления соединения между сетями можно применять команду **cu**. Для применения команды **ct** вы *обязательно* должны знать номер телефона целевой системы.

> **Примечание:** Третья команда, **tip**, работает аналогично команде **cu**. Однако она входит в версию Berkeley Software Distribution (BSD) программы UUCP. Для установки этой версии с BNU требуется специальная настройка.

Связь с другой системой по выделенному каналу или через модем

После ввода команды **cu** в локальной системе можно выполнять следующие операции:

- Устанавливать соединение с удаленной системой
- Входить в удаленную систему
- Выполнять различные задачи в удаленной системе
- Переключаться между системами, работая в них одновременно

Если на удаленном и локальном компьютерах применяются одинаковые операционные системы, то можно вводить стандартные команды прямо в командной строке локального компьютера. Например, вы можете вводить команды смены каталога, просматривать содержимое каталогов и файлов или передавать файлы в

очередь печати в удаленной системе. Для выполнения команд в локальной системе, а также для запуска удаленных команд и передачи файлов вы можете использовать специальные локальные команды **cu**, указав перед ними тильду (~).

Связь с другой системой через модем

Команда **ct** применяется для установления соединения с другим компьютером с помощью модема.

Введите команду **ct**, указав после нее номер телефона удаленной системы. Когда соединение будет установлено, на экране появится приглашение для входа в систему на удаленном компьютере.

Команда **ct** применяется лишь в некоторых ситуациях. Более подробная информация о команде BNU **ct** приведена в следующих разделах:

- “Набор номера до установления соединения”
- “Набор нескольких номеров до установления соединения”

Набор номера до установления соединения

В этом примере описана процедура автоматического набора номера с помощью команды **ct** до установления соединения или до истечения определенного времени.

В вызываемой системе должны быть запущены основные сетевые утилиты (BNU) или другая версия программы копирования UNIX-UNIX (UUCP).

В командной строке локальной системы введите:

```
ct -w3 5550990
```

При этом будет набран номер телефона удаленной системы `555-0990`. Флаг **-w3** указывает, что команда **ct** должна набирать номер с интервалом в одну минуту до установления соединения или до истечения трех минут.

> **Примечание:** Номер телефона удаленной системы в командной строке **ct** можно указывать как до, так и после флага.

Набор нескольких номеров до установления соединения

В этом примере описана процедура автоматического набора нескольких номеров телефонов с помощью команды **ct** до установления соединения или до истечения определенного времени.

В вызываемой системе должны быть запущены основные сетевые утилиты (BNU) или другая версия программы копирования UNIX-UNIX (UUCP).

В командной строке локальной системы введите:

```
ct -w6 5550990 5550991 5550992 5550993
```

При этом будут последовательно набираться номера `555-0990`, `555-0991`, `555-0992` и `555-0993`. Флаг **-w6** указывает, что команда **ct** должна набирать номер с интервалом в одну минуту до установления соединения или до истечения шести минут.

> **Примечание:** Номера телефонов удаленной системы можно указывать как перед, так и после флага.

Обмен файлами между локальной и удаленной системой

Чаще всего основные сетевые утилиты (BNU) применяются для передачи файлов между системами. Для передачи файлов между локальной и удаленной системами BNU применяет четыре команды: **uucp**, **uusend**, **uuto** и **uupick**.

Команда **uucp** - основное средство, применяемое BNU для передачи данных. Команда **uusend** - это встроенная в BNU команда передачи данных из пакета Berkeley Software Distribution (BSD). Команды **uuto** и **uupick** - это специализированные команды для передачи и получения данных, работающие совместно с командой **uucp**.

Команды BNU **uuencode** и **uudecode** используются для передачи файлов. Они позволяют кодировать и декодировать файлы, передаваемые с помощью почтовой программы BNU.

Отправка и получение файлов

К командам отправки и получения файлов через соединение BNU относятся команды **uucp** и **uusend**.

Команда **uucp** позволяет передавать файлы в локальной системе, между локальной и удаленной системами, а также между удаленными системами. С помощью опций команды **uucp** можно, например, создать каталоги для хранения файлов в получающей системе или разрешить отправку почтовых сообщений об удачном или неудачном завершении передачи файлов.

Команда **uusend** позволяет отправлять файлы в удаленную систему, если она подключена к отправляющей системе не напрямую, а через несколько соединений BNU. Несмотря на то, что у команды **uusend** меньше опций, чем у **uucp**, она также включена в утилиты BNU и может быть полезна для пользователей программы BSD UUCP.

Отправка файлов конкретным пользователям

Для отправки файлов конкретным пользователям в отправляющей и получающей системах должны быть запущены основные сетевые утилиты (BNU) или другая версия программы копирования UNIX-UNIX (UUCP).

Команда **uuto** отправляет файлы из одной системы в другую. Она представляет собой часть команды **uucp** и упрощает передачу файлов как для отправителей, так и для получателей. Команда **uuto** отправляет файлы определенному пользователю и размещает их непосредственно в личном подкаталоге этого пользователя, в общем каталоге BNU. Затем она сообщает получателю о приеме файла. Для обработки этого файла получатель может воспользоваться командой **uupick**.

Отправка файла с помощью команды uuto:

При использовании для отправки файла команды **uuto** необходимо указать отправляемый файл, целевую удаленную систему и пользователя в этой системе.

Например:

```
uuto /home/bin/file1 distant!joe
```

Эта команда отправляет файл `file1` из локального каталога `/home/bin` пользователю `joe` удаленной системы `distant`.

Команда **uuto** применяется совместно с командой **uucp**. Файл перемещается в каталог `/var/spool/uucppublic` в удаленной системе. Файл размещается в каталоге удаленной системы `/var/spool/uucppublic/receive/`*пользователь*`/System`. Если целевой каталог не существует, он создается при обмене файлами.

Затем команда BNU **rmail** уведомляет получателя о приеме файла.

> **Примечание:** Для отправки файла пользователю *локальной* системы введите команду **uuto** и укажите отправляемый файл, целевую локальную систему, а также пользователя в этой системе. Например:
>
> ```
> uuto /home/bin/file2 near!nick
> ```
>
> Эта команда отправляет файл `file2` из локального каталога `/home/bin` пользователю `nick` локальной системы `near`.

Получение файлов

Для получения и обработки файлов в отправляющей и получающей системах должны быть запущены основные сетевые утилиты (BNU) или другая версия программы копирования UNIX-UNIX (UUCP).

Команда **uupick** предназначена для получения файлов, отправленных с помощью команды **uuto**, и работы с ними. В ней предусмотрены опции обработки файлов, с помощью которых получатель может искать отправленные файлы, перемещать файлы в определенный каталог, выполнять команды и удалять файлы.

Получение файла с помощью команды uupick:

Команда **uupick** позволяет получить файл.

Например:

```
uupick
```

Команда **uupick** ищет в общем каталоге файлы, в именах которых присутствует ИД удаленного пользователя. Затем команда **uupick** показывает на экране удаленной системы примерно следующее сообщение:

```
из системной базы данных: файл file1?
```

Символ **?** (вопросительный знак) во второй строке сообщения означает, что пользователь должен указать опции **uupick** для размещения файлов в общем каталоге BNU.

Для просмотра списка всех доступных опций введите звездочку (*) после вопросительного знака (?) в строке запроса. Опции просмотра, сохранения и выхода:

Элемент	Описание
p	Показывает содержимое файла.
m *[Каталог]*	Сохраняет файл в каталоге, указанном в переменной *[Каталог]*. Если каталог в опции **m** не указан, то файл помещается в текущий рабочий каталог.
q	Завершение работы с файлом с помощью **uupick**.

Кодирование и декодирование файлов для передачи

Команды **uuencode** и **uudecode** предназначены для подготовки файлов к передаче с помощью модема.

Эти команды используются в паре. Команда **uuencode** преобразует двоичные файлы в файлы ASCII. Затем эти файлы можно отправить по электронной почте в удаленную систему.

С помощью команды **uudecode** файлы ASCII вновь преобразуются в двоичный формат.

Отчеты о состоянии передачи файла и команды

Отчеты о состоянии передачи файла и команды можно просмотреть с помощью команд **uusnap**, **uuq** и **uustat**.

Просмотр состояния систем, с которыми установлены соединения BNU

Команда **uusnap** показывает таблицу с информацией обо всех системах, с которыми установлены соединения.

В строках этой таблицы показана следующая информация о каждой из систем: имена и номера командных файлов, файлов данных, а также число запросов на выполнение удаленных команд, находящихся в системных очередях. В конце строки показано сообщение о состоянии. В этом сообщении говорится либо об успешном установлении соединения BNU, либо указаны причины, из-за которых не удалось установить связь.

Обратитесь к описанию команды **uusnap**.

Просмотр очереди заданий BNU
Команда **uuq** перечисляет все задания, находящиеся в очереди заданий BNU.

Формат списка такой же, как у данных, выдаваемых командой **ls**. Для каждой записи указывается номер задания и его краткое описание, включающее имя системы, число заданий для этой системы и общий объем данных для передачи. Пользователи с правами доступа root могут применять команду **uuq** для идентификации заданий в очереди по их номерам.

Дополнительная информация приведена в описании команды **uuq** в *Справочник по командам, том 5*.

Состояние операций BNU
Команда **uustat** позволяет просмотреть информацию о состоянии определенной команды или файла, передаваемого в систему BNU.

После ввода команды **uustat** без флагов на экране будет показана следующая информация обо всех заданиях, запрошенных текущим пользователем:

- Идентификационный номер задания
- Дата и время
- Состояние (отправка или получение)
- Имя системы
- ИД пользователя, который ввел команду.
- Размер и имя файла задания

Кроме того, у команды **uustat** есть флаги, позволяющие просмотреть информацию о заданиях, которые были запрошены другими пользователями и в настоящее время находятся в очереди, а также информацию о заданиях, запрошенных другими системами.

Команда **uustat** предоставляет возможность ограниченного управления заданиями, находящимися в очереди и ожидающими запуска на удаленном компьютере. Вы можете просмотреть информацию о состоянии соединений BNU и проверить, как файлы или команды передаются в другую систему. Затем вы можете, например, отменить запросы на копирование, запущенные командой **uucp**.

Обратитесь к описанию команды **uustat**.

Обмен командами между локальной и удаленной системой
Основные сетевые утилиты (BNU) позволяют передавать команды между локальной и удаленной системами.

Команда **uux** запускает команды в удаленной системе. Команда **uupoll** обеспечивает синхронизацию при выполнении команд.

Запросы на выполнение команд в удаленной системе
Команда **uux** отправляет запрос на выполнение команды в удаленной системе.

Команда **uux** не выполняет команды в удаленной системе. Эта команда подготавливает необходимые файлы управления и данных в каталоге `/var/spool/uucp` . Для их передачи вызывается демон **uucico**. После завершения передачи демон удаленной системы **uucico** создает исполняемый файл в буферном каталоге.

Когда два демона **uucico** согласовали завершение соединения, демон **uuxt** проверяет наличие невыполненных запросов в буферном каталоге, права доступа и необходимость дополнительной информации. Затем он отдает команду на выполнение запроса.

Примечание: Команду **uux** можно применять в любой системе, в которой разрешено выполнение указанных команд. Тем не менее, в целях защиты применение отдельных команд на некоторых серверах может быть запрещено. Например, на некоторых серверах разрешено выполнять только команду **mail**.

После получения файлов удаленной системой демон **uuxqt** запускает в этой системе указанную команду. Демон **uuxqt** периодически проверяет общий буферный каталог удаленной системы и проверяет, нет ли в нем файлов, переданных командой **uux**. Затем демон **uuxqt** проверяет, существуют ли в удаленной системе данные, к которым будут обращаться полученные файлы. Кроме того, он проверяет, есть ли у отправляющей системы права доступа к этим данным. После выполнения всех этих проверок демон **uuxqt** либо выполняет команду, либо сообщает отправляющей системе, что команда не выполнена.

Отслеживание удаленных соединений BNU

Используйте следующую процедуру для отслеживания удаленного соединения BNU.

- В системе должна быть установлена программа BNU.
- Между вашей и удаленной системами должен быть настроен канал передачи данных (выделенный, телефонный или TCP/IP).
- В файлах конфигурации BNU, в том числе `Systems`, `Permissions`, `Devices` и `Dialers` (а также, при необходимости, `Sysfiles`), должны быть заданы параметры соединения между локальной и удаленной системами.

  Примечание: Изменять файлы конфигурации BNU разрешено только пользователю root.

При возникновении неполадок, связанных с передачей файлов, можно воспользоваться командой **Uutry**, обеспечивающей контроль работы демона **uucico**.

1. С помощью команды **uustat** определите состояние заданий передачи в текущей очереди:

   ```
   uustat -q
   ```

   Появится отчет о состоянии примерно следующего вида:

   ```
   venus 3C (2) 05/09-11:02 Нет доступа к устройству
   hera 1C 05/09-11:12 SUCCESSFUL
   merlin 2C 5/09-10:54 NO DEVICES AVAILABLE
   ```

   Этот отчет указывает, что три командных файла (`C.*`) предназначенных для удаленной системы `venus`, находятся в очереди в течение двух дней. Возможны различные причины такой задержки. Например, система `venus` могла быть выключена для выполнения обслуживания, либо не включен модем.
2. Перед выполнением расширенных процедур устранения неполадок вызовите команду **Uutry**, как показано ниже, чтобы проверить, может ли локальная система подключиться к системе `venus`:

   ```
   /usr/sbin/uucp/Uutry -r venus
   ```

   Эта команда запускает демон **uucico** со средним уровнем отладки и переопределяет интервал повтора по умолчанию. Команда **Uutry** записывает данные отладки во временный файл `/tmp/venus`.
3. Если локальной системе удастся установить соединение с системой `venus`, то результаты отладки будут содержать большое количество информации. Наиболее важна последняя строка в этом сценарии. Она приведена ниже:

   ```
   Диалог завершен: Успешно
   ```

   Если соединение успешно установлено, будем предполагать, что временные неполадки передачи файлов устранены. Вызовите команду **uustat** еще раз, чтобы убедиться, что файлы в буферном каталоге успешно переданы в удаленную систему. Если переданы не все файлы, выполните инструкции из раздела “Отслеживание передачи файлов BNU” на стр. 468 для обнаружения неполадок, возникающих при передаче файлов из локальной системы в удаленную.
4. Если локальной системе не удалось подключиться к удаленной системе, то вывод команды **Uutry** будет содержать примерно следующую информацию:

```
Вызов mchFind (venus)
conn (venus)
getto ret -1
Вызов не обработан: устройство недоступно
код завершения: 101
Диалог завершен: Сбой
```

Сначала проверьте физическое соединение между локальной и удаленной системами. Убедитесь, что удаленный компьютер включен, и все кабели правильно подсоединены, порты включены или отключены (по необходимости) в обеих системах, а нужные модемы находятся в рабочем состоянии.

Если физическое соединение исправно и надежно, проверьте все необходимые файлы конфигурации как в локальной, так и в удаленной системах:

- Убедитесь, что файлы `Devices`, `Systems` и `Permissions` (а также, при необходимости, `Sysfiles`) из каталога **/etc/uucp** не содержат ошибок в локальной и удаленной системах.
- При работе с модемом убедитесь, что файл `/etc/uucp/Dialers` (или альтернативный файл, указанный в **/etc/uucp/Sysfiles**) содержит правильную запись. Если вы применяете сокращенные коды, убедитесь, что они определены в файле `/etc/uucp/Dialcodes`.
- Если применяется соединение TCP/IP, то убедитесь, что записи TCP в файлах конфигурации не содержат ошибок, а в удаленной системе можно запустить демон **uucpd**.

5. После проверки физического соединения и файлов конфигурации вызовите команду **Uutry** еще раз. Если в результатах отладки по-прежнему сообщается о сбое соединения, рекомендуется обратиться в службу поддержки. Сохраните вывод команды **Uutry**. Он может понадобиться при диагностике неполадки.

Передача файла в удаленную систему для печати

Команда **uux** передает файл в удаленную систему для печати.

Для передачи файла удаленной системе для печати должны быть выполнены следующие предварительные условия:

- С целевой удаленной системой должно быть установлено соединение BNU.
- У вас должны быть права доступа на выполнение операций в удаленной системе.

Введите в командной строке локальной системы:

```
uux remote!/usr/bin/lpr local!имя-файла
```

При этом локальный файл *имя-файла* будет напечатан в удаленной системе.

Отслеживание передачи файлов BNU:

Ниже приведена процедура отслеживания передачи файлов в удаленную систему.

- В системе должна быть установлена и настроена программа BNU.
- Установите соединение с удаленной системой, выполнив инструкции из раздела “Отслеживание удаленных соединений BNU” на стр. 467.

Она применяется в случае, когда при передаче файлов в удаленную систему происходит сбой по неизвестным причинам. Для поиска причин неполадок можно воспользоваться отладочной информацией, созданной демоном **uucico** (этот демон вызывается командой **Uutry**).

Команда **Uutry** позволяет контролировать процесс передачи файлов:

1. Подготовьте файл для передачи командой **uucp** с флагом **-r**:

   ```
   uucp -r test1 venus!~/test2
   ```

   Флаг **-r** сообщает программе UUCP о том, что нужно создать и поместить в очередь файл, но *не* нужно запускать демон **uucico**.
2. Запустите демон **uucico** в режиме отладки с помощью команды **Uutry** с флагом **-r**:

```
/usr/sbin/uucp/Uutry -r venus
```

Эта команда указывает демону **uucico** установить соединение с удаленной системой `venus` и переопределяет интервал повтора по умолчанию. Демон подключается к системе `venus`, входит в нее и передает файл, в то время как команда **Uutry** создает отладочный вывод, позволяющий отслеживать работу демона **uucico**. Для прекращения создания отладочного вывода и возврата в режим командной строки нажмите клавишу прерывания.

Команда **Uutry** сохраняет отладочный вывод в файле `/tmp/`*`SystemName`*. Если вы прервете создание результатов отладки до окончания установки соединения, то можете просмотреть файл вывода и узнать результат установки соединения.

Передача буферизированных заданий:

Команда **uupoll** начинает передачу заданий, сохраненных в общем буферном каталоге локальной системы.

Команда **uupoll** создает в общем каталоге пустое задание для удаленной системы и запускает демон **uucico**. При этом демон **uucico** немедленно устанавливает соединение с удаленной системой и передает задания из очереди.

Идентификация совместимых систем

Команда **uuname** позволяет просмотреть список всех систем, доступных из локальной системы.

Например, если ввести команду:

```
uuname
```

то на экране будет показан примерно следующий список:

```
arthur
hera
merlin
zeus
```

Этой информацией можно воспользоваться для определения имени доступной системы перед копированием файла. Команда **uuname** применяется и для идентификации локальной системы. Команда **uuname** получает информацию из файла `/etc/uucp/systems`.

Обращение к системам UNIX, соединенным с вашей системой, с помощью команды tip

С помощью команды **tip** можно обращаться к любым компьютерам с операционной системой UNIX, которые соединены с вашей системой.

Команда **tip** устанавливается вместе с продуктом BNU. Для ее работы могут применяться асинхронные соединения, созданные BNU.

Для управления соединением в команде **tip** применяются переменные и Escape-сигналы, а также флаги. Флаги можно вводить в командной строке. Escape-сигналы используются в соединении с удаленной системой для начала и прекращения передачи файлов, изменения направления передачи файлов и выхода в оболочку следующего уровня.

Переменные команды tip:

Переменные команды **tip** задают такие параметры, как символ конца строки, сигнал прерывания и режим передачи файлов.

Значения переменных задаются во время выполнения с помощью файла `.tiprc`. Кроме того, их можно изменять во время выполнения с помощью Escape-сигнала `~s`. Некоторые переменные, например, символ конца строки, можно задать для отдельной системы в записи об этой системе в файле `remote`.

Для определения начальных значений переменных команда **tip** считывает три файла: phones, remote, .tiprc. Файл .tiprc должен располагаться в домашнем каталоге пользователя. Имена и расположение файлов remote и phones могут быть произвольными. Имена файлов remote и phones можно задать с помощью следующих переменных среды:

Элемент	Описание
PHONES	Указывает имя пользовательского файла phones. Файлу может быть присвоено любое имя, однако его формат должен совпадать с форматом файла /usr/lib/phones-file. Имя файла по умолчанию - etc/phones. Файл, указанный в переменной PHONES, применяется вместо файла /etc/phones (а не в дополнение к нему).
REMOTE	Указывает имя пользовательского файла с определением удаленной системы. Файлу может быть присвоено любое имя, однако его формат должен совпадать с форматом файла /usr/lib/remote-file file. Имя файла по умолчанию - /etc/remote. Файл, указанный в переменной **REMOTE**, применяется вместо файла /etc/remote (а не в дополнение к нему).

Для использования переменной среды задайте ее перед запуском команды **tip**. Кроме того, имена альтернативных файлов phones и remote можно задать с помощью переменных phones и remote команды **tip** в файле .tiprc.

Примечание: Команда **tip** считывает только *последние* имена файлов remote или phones. Таким образом, если вы укажете файл remote или phones в переменной среды, то этот файл будет применяться вместо всех указанных ранее файлов (а не в дополнение к ним).

Команда **tip** использует заданные переменные в следующем порядке:

1. Вначале команда проверяет, заданы ли файлы **PHONES** и **REMOTE** в переменных среды phones и remote.
2. Команда считывает содержимое файла .tiprc и присваивает соответствующие значения всем переменным. Если в файле .tiprc заданы переменные phones и remote, то их значения переопределяют значения переменных среды.
3. При инициализации соединения с удаленной системой команда считывает запись файла remote, соответствующую этой системе. Значения из файла remote переопределяют значения из файла .tiprc.
4. Если в команде - указан флаг BaudRate, то указанное значение переопределяет все предыдущие значения скорости в бодах.
5. Значение, введенное с помощью Escape-сигнала ~s, переопределяет все предыдущие значения соответствующей переменной.

   Примечание: Любой пользователь команды **tip** может создать файл .tiprc и указать в нем начальные значения переменных **tip**. Файл .tiprc нужно поместить в свой домашний каталог $HOME.

Файлы конфигурации команды tip:

Для того чтобы команда **tip** могла подключаться к удаленной системе, необходимо настроить параметры в файлах /etc/remote и /etc/phones.

Элемент	Описание
/etc/remote	Определяет атрибуты удаленных систем, например порт и тип устройства, которые будут применяться при установлении соединений, а также сигналы, которые будут указывать начало и конец передачи.
/etc/phones	Содержит список номеров телефона для связи с удаленными системами через модем.

Примеры файлов remote и phones предусмотрены в пакете bos.net.uucp. Пример файла remote называется /usr/lib/remote-file. Пример файла phones называется /usr/lib/phones-file. Скопируйте файл /usr/lib/remote-file в файл /etc/remote, а затем отредактируйте файл /etc/remote. Для настройки одного из этих файлов создайте копию файла-образца под нужным именем и внесите в нее необходимые изменения.

Пользователь команды **tip** может также создать собственные файлы `remote` и `phones`. Пользовательский файл `remote` должен быть задан в формате файла `/usr/lib/remote-file` и указан в переменной `remote` или переменной среды `REMOTE`. Пользовательский файл `phones` должен быть задан в формате файла `/usr/lib/phones-file` и указан в переменной `phones` или переменной среды `PHONES`. Если в какой-либо из этих переменных указан пользовательский файл `phones` или `remote`, то он будет применяться вместо файла `/etc/phones` или `/etc/remote` (а не в дополнение к нему).

В команде **tip** могут применяться различные файлы `phones` и `remote`. Например, пользователь может применять файл `remote` по умолчанию, `/etc/remote`, но указать собственный файл `phones` в переменной `phones`.

Отмена удаленного задания

С помощью команды **uustat** можно отменить процесс BNU, запущенный в удаленной системе.

Для отмены удаленной задачи необходимо выполнить следующие предварительные требования:

- С целевой удаленной системой должно быть установлено соединение (BNU)
- Удаленное задание должно быть запущено из локальной системы

1. Определите идентификационный номер процесса в удаленной очереди. Введите в командной строке локальной системы:

   ```
   uustat -a
   ```

   Опция **-a** предназначена для просмотра всех заданий, находящихся в очереди удаленной системы, а также запросов всех остальных пользователей BNU.

   На экране будет показано примерно следующее сообщение:

   ```
   heraC3113   11/06-17:47  S hera   you   289 D.venus471afd8
   merlinC3119 11/06-17:49  S merlin jane  338 D.venus471bc0a
   ```

2. Затем введите:

   ```
   uustat -k heraC3113
   ```

   Опция **-k** отменяет запрос на обработку задания `heraC3113`.

Устранение неполадок BNU

Сообщения об ошибках BNU можно вставить в определенную фазу диалога. Для определения причин неполадок BNU воспользуйтесь рисунком "Диаграмма диалога BNU" и приведенными после него описаниями ошибок.

Некоторые из приведенных ниже сообщений могут не применяться BNU. Они приведены на тот случай, если в системе установлена другая версия UUCP.

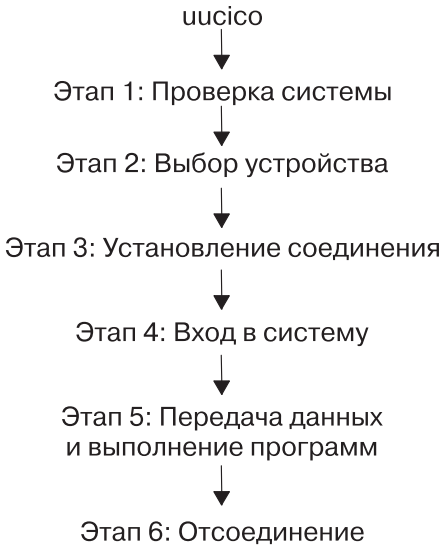


Рисунок 26. Диаграмма диалога BNU

На этом рисунке показана схема и различные этапы диалога BNU. После отправки данных демоном uucico выполняется следующая последовательность действий: этап 1 - Проверка системы, этап 2 - Выбор устройства, этап 3 - Установление соединения, этап 4 - Вход в систему, этап 5 - Передача данных и выполнение программ и этап 6 - Отключение.

Сообщение состояния BNU PHASE 1

Существует пять сообщений о состоянии BNU PHASE 1. Он описаны в следующей таблице.

Элемент	Описание
`Ошибка захвата`	Неполадки в локальном системном блоке. Список возможных причин ошибки приведен в отчете об ошибках, который можно просмотреть с помощью команды `errpt -a | pg`.
`Система не указана в файле Systems`	Это сообщение отправляется в том случае, если указанное имя удаленной системы не было найдено в файлах `Systems`. После этого работа BNU завершается. Введите команду **uuname** для выяснения имени системы.
`Неверное время вызова`	В файле `Systems` заданы ограничения на моменты времени, в которые разрешены вызовы удаленных систем. BNU будут продолжать попытки вызова до тех пор, пока не наступит подходящее время. Просмотрите файл `Systems`.
`Необходим обратный вызов`	Связь по сети ограничена из-за слишком высокой цены обслуживания или по причинам, связанным с защитой, и в данный момент доступ запрещен.
`Невозможно вызвать Нет вызова`	Эти сообщения об ошибках означают, что BNU попытались вызвать удаленную систему, но неудачно. BNU не будут немедленно повторять попытку. Другая возможная причина ошибки - устаревший файл с информацией о состоянии системы, не позволяющий демону **uucico** повторить попытку.

Сообщение состояния BNU PHASE 2

Существует четыре сообщения о состоянии BNU PHASE 2. Он описаны в следующей таблице.

Элемент	Описание
Сбой сценария Dialer	Сценарий файла Dialers выполнен неудачно.
Нет доступных устройств Не удалось обратиться к устройству	Модем или телефонная линия для вызовов из локальной системы заняты. Проверьте правильность записи об устройстве в файле Systems. Кроме того, проверьте файлы Devices и Dialers, чтобы убедиться, что с логическими устройствами связаны физические устройства. Возможно, в файле /etc/uucp/Sysfiles заданы альтернативные файлы Systems, Devices или Dialers, в которых задана неверная конфигурация. Не используется ли устройство другой программой? Убедитесь, что в каталоге /var/locks нет файла блокировки порта. Если файл блокировки существует (например, LCK..TTY0), проверьте, активен ли процесс, номер которого указан в файле блокировки. Если нет, то файл блокировки можно удалить (например, с помощью команды rm /var/locks/LCK..TTY0). Кроме того, проверьте права доступа к порту.
Сбой вызова Сбой вызова системы	Эти сообщения об ошибках появляются, когда ваша система успешно вызвала другую систему, но та не отвечает. Кроме того, они могут свидетельствовать об ошибке в файлах Devices. Введите команду uucico -r1 -x6 -s *имя-системы*. Возможно, BNU не получил строку, которую ожидал. Установите соединение вручную и выясните, какие записи необходимо добавить в файл Systems для выполнения запроса. При этом должен быть учтен фактор времени: возможно, в строке набора номера из конфигурации модема нужно предусмотреть некоторые задержки. Кроме того, это может означать, что порт занят, вы набрали неверный номер или BNU перестал быть владельцем порта.
OK Автоматический вызов	Это информационные сообщения, не указывающие на ошибку.

Сообщение состояния BNU PHASE 3

Существует пять сообщений о состоянии BNU PHASE 3. Он описаны в следующей таблице.

Элемент	Описание
Сбой квитирования (LCK)	Устройство занято, процессу не удалось создать файл LCK. Иногда администратору следует вручную удалять файлы LCK. Если несколько попыток окажутся неудачными, обратитесь к системному администратору. Выясните, не работает ли с портом другой процесс (например, другой экземпляр демона **uucico**).
Сбой входа в систему	Вход в систему выполнить не удалось из-за неудачного соединения или недостаточного быстродействия компьютера.
Тайм-аут	Удаленная система не ответила за предопределенный период времени. Другая возможная причина - ошибка в сценарии диалога.
Успешный вызов системы	Вызов успешно выполнен.
BNU (продолжение)	Это информационные сообщения, не указывающие на ошибку.

Сообщение состояния BNU PHASE 4

Существует шесть сообщений о состоянии BNU PHASE 4. Он описаны в следующей таблице.

Элемент	Описание
Запуск не удался Отказ в удаленной системе после входа	После входа в удаленную систему в ней был запущен демон **uucico**. Такие сообщения появляются, если инициализировать диалог между двумя системами не удалось. Такая ошибка возникает и в том случае, если пользователь BNU вошел в систему под неправильным именем, либо если не удалось выполнить начальное согласование параметров.
Неправильное имя компьютера	Имя компьютера указано неверно, либо оно изменилось.
Неверное сочетание ИД пользователя и компьютера	Вход в удаленную систему не выполнен. Причиной могут быть неправильный номер телефона, неверный ИД входа в систему или пароль, ошибка в сценарии диалога.
В удаленной системе есть файл LCK для локальной системы	Обе системы попытались вызвать друг друга одновременно. В течение некоторого времени запрос из локальной системы выполняться не будет.
OK Выполняется диалог	Это информационные сообщения, не указывающие на ошибку.

Элемент	Описание
LOGIN: PASSWORD:	Если приглашение на ввод имени пользователя и пароля указано прописными буквами, то модем, скорее всего, работает в режиме эхоповтора (значение E1 в Hayes-совместимых модемах). Это приводит к тому, что при получении вызова модем отправляет эхоповтор RING в вашу систему. Команда **getty** получает строку и изменяет буквы в приглашениях login: и password: на прописные. Выключите режим эхоповтора (укажите ATE0 для Hayes-совместимых модемов). **Примечание:** Учтите, что после внесения этого изменения вы должны указать значение ATE1 в сценарии диалога в файлах Dialers; в противном случае от модема не будет получен ожидаемый ответ OK. Если для удаленного порта задано delay или getty -r, и сценарий диалога рассчитан на ввод с клавиатуры, то при работе с портами, для которых задано delay, нужно ввести один или несколько символов возврата каретки перед продолжением входа в систему. Попробуйте начать сценарий диалога в исходной системе со следующих символов: `"" \r\d\r\d\r\d\r in:--in: ...` Они означают следующее: не ожидать сообщения, отправить символ возврата каретки, задержка, символ возврата каретки, задержка, символ возврата каретки, задержка, символ возврата каретки.

Сообщение состояния BNU PHASE 5

Существует пять сообщений о состоянии BNU PHASE 5. Он описаны в следующей таблице.

Элемент	Описание
Предупреждение	Демону **uucico** не удалось установить соединение. Либо соединение неисправно, либо параметру "xon/xoff" на модеме присвоено значение "yes".
Удаленный доступ к пути/файлу запрещен Сбой копирования	Эти сообщения указывают на неполадку с правами доступа; проверьте права доступа к файлу и пути.
Ошибка чтения	В удаленной системе недостаточно памяти, скорее всего, в буферной области, либо демону **uucico** не удалось прочесть или записать информацию на устройство.
Сбой диалога	Модем не отвечает. Возможно, модем отключен, не подсоединен кабель, удаленная система закрыта или в ней произошел сбой. Возможен также сбой на телефонной линии.
Запрошено Копирование выполнено успешно	Это информационные сообщения, не указывающие на ошибку.

Сообщение состояния BNU PHASE 6

Существует два сообщения о состоянии BNU PHASE 6. Он описаны в следующей таблице.

Элемент	Описание
OK (Диалог завершен)	Удаленная система может аннулировать запрос "положить трубку" и поменяться ролями с локальной (это означает, что ей необходимо передать информацию локальной системе). После того, как демоны **uucico** обменяются всей накопившейся информацией, диалог будет завершен.
Успешный диалог	Это информационное сообщение, не указывающее на ошибку.

Отладка сбоя входа в систему BNU с помощью демона uucico

Для отладки сбоев входа в систему BNU используйте демон **uucico**.

- В системе должно быть установлено программное обеспечение BNU.
- Между вашей и удаленной системами должен быть настроен канал передачи данных (выделенный, телефонный или TCP/IP).
- В файлах конфигурации BNU, в том числе Sysfiles (если он есть), Systems, Permissions, Devices и Dialers, должны быть заданы параметры соединения между локальной и удаленной системой.

  Примечание: Изменять файлы конфигурации BNU разрешено только пользователю root.
- Для запуска демона **uucico** в режиме отладки необходимы права доступа root.

1. Для создания отладочной информации о неработающем соединении между локальной и удаленной системами запустите демон **uucico** с флагом **-x**, как показано ниже:

   ```
   /usr/sbin/uucp/uucico -r 1 -s venus -x 9
   ```

где `-r 1` - режим главной системы (вызов); `-s venus` - имя удаленной системы; `-x 9` - уровень отладки, на котором собирается наиболее подробная информация.

2. Если в записи expect-send sequence в файле `Systems` в формате `/etc/uucp/Systems` указано:

```
venus Any venus 1200 - "" \n in:--in: uucp1 word:
 mirror
```

то демон **uucico** подключает локальную систему к удаленной системе `venus`. Результаты отладки будут примерно следующими:

```
expect: ""
got it
sendthem (^J^M)
expect (in:)^
M^Jlogin:got it
sendthem (uucp1^M)
expect (word:)^
M^JPassword:got it
sendthem (mirror^M)
imsg >^M^J^PShere^@Login Successful: System=venus
```

где:

Элемент	Описание
`expect: ""`	Указывает, что локальная система не будет ожидать информации от удаленной системы.
`got it`	Подтверждает прием сообщения.
`sendthem (^J^M)`	Указывает, что локальная система отправит в удаленную систему символы возврата каретки и перехода на новую строку.
`expect (in:)`	Указывает, что локальная система должна получить приглашение на вход в удаленную систему, которое заканчивается символами `in:`.
`^M^Jlogin:got it`	Подтверждает, что локальная система получила приглашение на вход в удаленную систему.
`sendthem (uucp1^M)`	Указывает, что локальная система отправит в удаленную систему ИД пользователя `uucp1`.
`expect (word:)`	Указывает, что локальная система должна получить приглашение на ввод пароля для удаленной системы, которое заканчивается символами `word:`.
`^M^JPassword:got it`	Подтверждает, что локальная система получила приглашение на ввод пароля для удаленной системы.
`sendthem (mirror^M)`	Указывает, что локальная система отправит в удаленную систему пароль для ИД пользователя `uucp1`.
`imsg >^M^J^PShere^@Login Successful: System=venus`	Подтверждает, что локальная система успешно вошла в удаленную систему `venus`.

Примечание:

1. Отладочный вывод команды **uucico** основывается на информации из файла `/etc/uucp/Dialers` или `/etc/uucp/Systems`. Информация об обмене сигналами с модемом берется из файла `Dialers`, а информация о соединении с удаленной системой - из файла `Systems`. (Учтите, что `/etc/uucp/Systems` и `/etc/uucp/Dialers` - это файлы конфигурации BNU по умолчанию. Вместо них в файле `/etc/uucp/Sysfiles` можно указать альтернативные файлы.)
2. Для настройки соединения с удаленной системой вам должна быть известна последовательность приглашений, выдаваемых при входе в эту систему.

SNMP для сетевого управления

Управление сетью обеспечивает всестороннее управление системными сетями посредством применения **Простого протокола управления сетью (SNMP)**, обеспечивающего обмен административной информацией между хостами в сети.

SNMP - это межсетевой протокол, предназначенный для сетей **TCP/IP**.

Вместе с операционной системой AIX по умолчанию устанавливается версия **SNMPv3** без шифрования. Она запускается во время загрузки системы. Если в вашем файле конфигурации `/etc/snmpd.conf` уже заданы необходимые взаимодействия, прерывания и записи SMUX, вам придется вручную перенести их в файл `/etc/snmpdv3.conf`. Информация о переносе групп приведена в разделе “Переход от SNMPv1 к SNMPv3” на стр. 485.

Также, возможно, вам пригодится информация из раздела Обзор SNMP для программистов в *Communications Programming Concepts*.

Управление сетью с помощью **SNMP** основывается на стандартной модели клиент/сервер, которая широко используется в сетевых приложениях, использующих **TCP/IP**. На каждом хосте, на который распространяется управление, выполняется процесс-*агент*. Агент - это обслуживающий процесс, который работает с Базой информации управления (MIB) хоста. На хостах, применяемых для управления сетью, может выполняться процесс-диспетчер. *Диспетчер* - это приложение-клиент, которое запрашивает информацию MIB и обрабатывает ответы. Кроме того, диспетчер может направлять запросы на агентские серверы с целью изменения информации MIB.

SNMP в системе AIX обеспечивает поддержку следующих RFC:

Элемент	Описание
RFC 1155	Структура и идентификация информации управления для глобальных сетей на основе **TCP/IP**.
RFC 1157	**Простой протокол управления сетью (SNMP)**
RFC 1213	База информации управления для управления глобальными сетями на основе **TCP/IP**: MIB-II
RFC 1227	**Простой протокол управления сетью (SNMP)** и База информации управления (MIB) для Простого протокола управления сетью (SNMP)
RFC 1229	Расширения общего интерфейса Базы информации управления (MIB)
RFC 1231	База информации управления (MIB) для IEEE 802.5 Token-Ring
RFC 1398	Определения объектов управления для интерфейсов Ethernet
RFC 1512	Информационная база управления FDDI
RFC 1514	MIB ресурсов хоста
RFC 1592	**Простой протокол управления сетью** - Интерфейс распределенных программ версии 2
RFC 1905	Операции протокола для Простого протокола управления сетью версии 2 (SNMPv2)
RFC 1907	Информационная база управления для Простого протокола управления сетью версии 2 (SNMPv2)
RFC 2572	Диспетчеризация и обработка сообщений для **Простого протокола управления сетью (SNMP)**
RFC 2573	Приложения **SNMP**
RFC 2574	Модель защиты на основе пользователей (USM) для **Простого протокола управления сетью версии 3 (SNMPv3)**
RFC 2575	Модель управления доступом на основе представлений (VACM) для **Простого протокола управления сетью (SNMP)**

SNMPv3

В предыдущих версиях AIX поддерживалась только версия **SNMPv1** протокола **SNMP**. **SNMPv3**, реализованный в AIX, обладает мощными и гибкими инструментами для обеспечения защиты сообщений и управления доступом.

Информация данного раздела относится только к **SNMPv3**.

Защита сообщений предоставляет следующие возможности:

- Проверка целостности данных, которая удостоверяет отсутствие изменений в переданной информации.

- Проверка источника данных, которая позволяет убедиться в том, что запрос или ответ поступил из указанного в сообщении источника.
- Проверка своевременности доставки сообщений и, по выбору, защита конфиденциальности данных.

В архитектуре **SNMPv3** для защиты сообщений применяется Модель защиты на основе пользователей (USM), а для управления доступом - Модель управления доступом на основе представлений (VACM). Эта архитектура позволяет одновременно использовать различные модели защиты, управления доступом и обработки сообщений. Так, например, при необходимости, модель защиты на основе связей может применяться одновременно с USM.

В USM применяется понятие пользователя, для которого в системах агента и менеджера настроены параметры защиты (уровни защиты, протоколы идентификации и защиты данных и ключи). Для сообщений, отправляемых с помощью USM, обеспечивается более надежная защита, чем для сообщений, отправляемых с применением модели защиты на основе связей, которая предполагает передачу паролей без шифрования с возможностью их просмотра средствами трассировки. В модели USM для всех сообщений, предаваемых между агентом и диспетчером, выполняются функции проверки целостности данных и идентификации источника. Для защиты от задержек и повторов сообщений (за исключением тех, которые могут быть вызваны применением протокола передачи без соединений) в этой модели предусмотрены индикаторы времени и ИД запросов. Для защиты конфиденциальности данных и их шифрования предназначен отдельный продукт, который можно при необходимости установить в системе. Версия **SNMP** с шифрованием входит в состав пакета AIX Expansion Pack.

Применение VACM предполагает определение наборов данных (называемых представлениями), групп пользователей данных и операторов доступа, определяющих, к каким представлениям у группы есть права на чтение, запись или получение уведомлений.

SNMPv3 также позволяет динамически изменять конфигурацию агента **SNMP** с помощью команд **SNMP** SET для объектов MIB, представляющих конфигурацию агента. С помощью этой динамической функции можно добавлять, удалять и изменять записи конфигураций как из локальной, так и из удаленной системы.

Стратегии доступа и параметры защиты **SNMPv3** определяются в файле `/etc/snmpdv3.conf` в системе агента **SNMP** и в файле `/etc/clsnmp.conf` в системе диспетчера **SNMP**. Сценарий, демонстрирующий настройку этих файлов, приведен в разделе “Создание пользователей в SNMPv3” на стр. 489. Также см. описание форматов файлов `/etc/snmpdv3.conf` и `/etc/clsnmp.conf` в книге *Справочник по файлам* .

Архитектура SNMPv3

Архитектура **SNMPv3** состоит из четырех основных компонентов.

На следующей иллюстрации объясняется, как эти компоненты взаимодействуют и обмениваются данными.

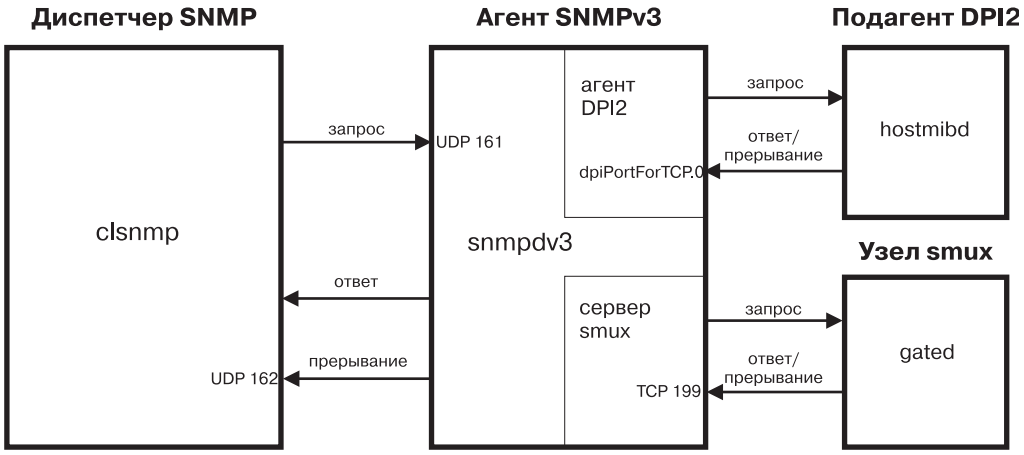


Рисунок 27. Основные компоненты архитектуры SNMPv3

На этой схеме показан пример архитектуры **SNMPv3**. Субагент DPI2, равноправный узел smux, диспетчер **SNMP** и агент **SNMP**. Кроме того, на схеме показано взаимодействие этих компонентов.

Агент SNMP:

Агент **SNMP** получает запросы от диспетчера **SNMP** и передает ему ответы.

Кроме того, агент **SNMP** осуществляет обмен данными со всеми субагентами DPI2 и узлами SMUX в системе. Агент **SNMP** управляет некоторыми переменными MIB, и все субагенты DPI2 и узлы SMUX регистрируют свои переменные MIB на агенте **SNMP**.

Запрос **clsnmp** (диспетчера **SNMP**) передается через порт UDP 161 агенту **SNMP**. Для запросов **SNMPv1** и **SNMPv2c** агент **SNMP** проверяет имя связи и выполняет запрос. Получив запрос **SNMPv3**, агент **SNMP** идентифицирует пользователя, запрашивающего данные, и проверяет наличие у этого пользователя прав доступа, необходимых для выполнения запроса. Для этого применяются идентификационные ключи и, в версии SNMP с поддержкой шифрования, ключи защиты данных. Если агенту **SNMP** не удается идентифицировать пользователя или у пользователя нет необходимых прав доступа для выполнения запроса, то агент **SNMP** не выполняет этот запрос. Инструкции по созданию пользователей в **SNMPv3** приведены в разделе “Создание пользователей в SNMPv3” на стр. 489.

Если пользователь идентифицирован и обладает необходимыми правами доступа, то агент **SNMP** выполняет его запрос. Агент **SNMP** находит запрошенные переменные MIB. Если сам агент **SNMP** управляет запрошенными переменными MIB, то он выполняет запрос и передает диспетчеру **SNMP** ответ. Если запрошенными переменными MIB управляет субагент DPI2 или узел SMUX, то агент **SNMP** пересылает запрос субагенту DPI2 или узлу SMUX, на которых находятся эти переменные MIB, разрешает им выполнить данный запрос и возвращает ответ диспетчеру **SNMP**.

Субагенты DPI2:

Субагент DPI2, например, **hostmibd**, осуществляет обмен данными с агентом DPI2, функции которого в **SNMPv3** выполняет агент SNMP.

Субагент DPI2 передает ответы и уведомления о прерываниях агенту DPI2 через порт dpiPortForTCP.0. Так как это нестандартный порт, то субагент DPI2 сначала запрашивает порт dpiPortForTCP.0. Этот запрос отправляется через порт UDP 161 в системе агента **SNMP**, после чего агент **SNMP** сообщает субагенту DPI2 номер порта dpiPortForTCP.0. Получив номер порта, субагент DPI2 устанавливает через этот порт соединение с агентом DPI2. После этого субагент DPI2 регистрирует свои ветви MIB на агенте DPI2.

Примечание: Для того чтобы агент **SNMP** мог использовать порт, отличный от порта UDP 161, необходимо задать переменную среды **SNMP_PORT**. Есть два способа задать эту переменную:

- **Способ 1**: Завершите работу субагента DPI2 и введите следующие команды:
  - `SNMP_PORT=<`*номер_порта*`> /usr/sbin/aixmibd -d 128`
  - `SNMP_PORT=<`*номер_порта*`> /usr/sbin/hostmibd -d 128`
  - `SNMP_PORT=<`*номер_порта*`> /usr/sbin/snmpmibd -d 128`

  где *номер_порта* - это нужный номер порта.

  После выполнения этих команд вновь запустите агент DPI2.
- **Способ 2**: Включите переменную **SNMP_PORT** в файл `/etc/environment` и присвойте ей новое значение порта. Разрешите демона **aixmibd**, **hostmibd**, **snmpmibd** и **snmpd** запускаться из `/etc/rc.tcpip` "как есть". Таким образом вы можете избежать запуска команд **aixmibd**, **hostmibd** и **snmpmibd** из командной строки.

Установив соединение и зарегистрировав ветви MIB, субагент DPI2 может отвечать на запросы, полученные от агента DPI2. Субагент DPI2 получает запрос и возвращает необходимые данные.

Кроме того, при необходимости, субагент DPI2 может отправлять уведомления о прерываниях. При получении уведомления агент **SNMP** считывает из файла `/etc/snmpdv3.conf` один или несколько IP-адресов для отправки уведомлений и пересылает уведомление по этим адресам.

Узлы SMUX:

Узлы **TCP**, например, **gated**, устанавливают при запуске соединение с портом TCP 199 и инициализируют связь SMUX.

После этого узел SMUX регистрирует ветви MIB, которыми он будет управлять.

Выполнив регистрацию, узел SMUX может отвечать на входящие запросы с сервера SMUX. Получив запрос с сервера, узел SMUX обрабатывает его и возвращает на сервер ответ.

Кроме того, узел SMUX может передавать на сервер SMUX уведомления о прерываниях. При получении уведомления агент **SNMP** считывает из файла `/etc/snmpdv3.conf` один или несколько IP-адресов для отправки уведомлений и пересылает уведомление по этим адресам.

Диспетчер SNMP:

Диспетчер **SNMP** запускает команду **clsnmp**, совместимую с **SNMPv1**, **SNMPv2c**и **SNMPv3**.

С помощью команды **clsnmp** можно передавать запросы, такие как `get`, `get-next`, `get-bulk` или `set`. Запрос передается через порт UDP 161 агента **SNMP**, после чего команда ожидает ответа от агента **SNMP**.

Примечание: Для того чтобы диспетчер **SNMP** мог применять порт, отличный от порта UDP 161, вам потребуется указать нужный номер порта и IP-адрес в поле **targetAgent** файла `/etc/clsnmp.conf`. Информация о файле `/etc/clsnmp.conf` приведена в главе clsnmp.conf File книги *Справочник по файлам* .

Диспетчер также может принимать уведомления о прерываниях **SNMP** через порт UDP. Диспетчер **SNMP** получает уведомления, если его IP-адрес указан в файле `/etc/snmpdv3.conf` агента **SNMP**.

Переменные MIB:

Сведения о переменных MIB можно получить из следующих источников.

Информация о переменных MIB находится в разделах Management Information Base, Terminology Related to Management Information Base Variables, Working with Management Information Base Variables и Management Information Base Database книги *Communications Programming Concepts*.

Примеры настройки агента DPI2 и узла smux находятся в каталогах /usr/samples/snmpd/smux и /usr/samples/snmpd/dpi2.

Ключи идентификации SNMPv3

Для обработки запросов **SNMPv3**, как правило, необходима идентификация (если не задан уровень защиты noAuth).

При идентификации запроса агент **SNMP** проверяет, можно ли с помощью идентификационного ключа, переданного в запросе **SNMPv3**, создать описатель сообщения, совпадающий с описателем сообщения, который был создан с помощью идентификационного ключа, заданного пользователем.

При создании запроса диспетчером **SNMP** команда **clsnmp** применяет идентификационный ключ, указанный в одной из записей в файле /etc/clsnmp.conf диспетчера **SNMP**. Он должен соответствовать идентификационному ключу, указанному в записи USM_USER для данного пользователя в файле конфигурации агента **SNMP** /etc/snmpdv3.conf. Для создания идентификационных ключей предназначена команда **pwtokey**.

Идентификационный ключ создается на основе двух элементов:

- Указанного пароля.
- Идентификатора агента **SNMP**, на котором будет применяться этот ключ. В случае агента IBM, идентификатор engineID которого был создан по формуле вендора, в качестве идентификатора может применяться IP-адрес или имя хоста агента. В противном случае в качестве идентификатора должен быть указан engineID.

Ключ, содержащий идентификатор агента, на котором он будет применяться, называется локальным. Этот ключ может применяться только данным агентом. Ключ, не содержащий идентификатора агента, на котором он будет применяться, называется нелокальным.

Ключи в файле конфигурации команды **clsnmp** - /etc/clsnmp.conf должны быть нелокальными. Ключи в файле конфигурации агента **SNMP**, /etc/snmpdv3.conf, могут быть как локальными, так и нелокальными, однако, считается, что применение локальных ключей обеспечивает более надежную защиту.

Вместо хранения в файле конфигурации клиента идентификационных ключей команда **clsnmp** позволяет хранить пароли пользователей. Если команда **clsnmp** настроена для работы с паролями, то она создает для пользователя идентификационных ключ (а если в системе установлена версия с шифрованием и заданы соответствующие параметры - то и ключ защиты данных). Значения, созданные с помощью этих ключей, должны совпадать со значениями, созданными с помощью ключей, заданных в записи USM_USER в файле агента /etc/snmpdv3.conf, либо созданных динамически с помощью команд **SNMP** SET. Однако считается, что хранение ключей в файлах конфигурации клиентов обеспечивает более надежную защиту, по сравнению с хранением паролей.

Ключи защиты данных SNMPv3

Функция шифрования в качестве отдельного продукта может входить в состав пакета AIX Expansion Pack, если это разрешено экспортным законодательством. Применяемые для шифрования ключи создаются с помощью того же алгоритма, что и идентификационные ключи.

Однако длина ключей может отличаться. Так, например, длина идентификационного ключа HMAC-SHA составляет 20 байт, а длина локального ключа шифрования HMAC-SHA - только 16 байт.

Версия SNMP с поддержкой шифрование автоматически активируется после установки. Переключиться на версию без шифрования можно с помощью команды **snmpv3_ssw**.

Создание ключей SNMPv3

В операционной системе AIX для создания идентификационных ключей и ключей защиты данных (если они поддерживаются) применяется команда **pwtokey**.

Команда **pwtokey** выполняет преобразование паролей в локальные и нелокальные идентификационные ключи и ключи шифрования. Она создает ключи на основе пароля и идентификатора агента. Так как в командах **pwtokey** и **clsnmp** применяется один и тот же алгоритм, то администратор, настраивающий агент **SNMP**, может создать для пользователя необходимые идентификационные ключи (и ключи защиты данных) для занесения в файл `/etc/clsnmp.conf` в системе диспетчера **SNMP**, при наличии пароля и IP-адреса системы агента.

Создав идентификационные ключи (и ключи защиты данных, если применяется версия с поддержкой шифрования), их необходимо сохранить в файле `/etc/snmpdv3.conf` агента **SNMP** и в файле `/etc/clsnmp.conf` диспетчера **SNMP**.

В версии **SNMPv3** существует девять возможных конфигураций пользователя. Ниже описаны все возможные конфигурации и приведены соответствующие примеры. Приведенные ниже ключи были созданы на основе пароля `defaultpassword` и IP-адреса `9.3.149.49` . Ключи были созданы с помощью следующей команды:

```
pwtokey -u all -p all defaultpassword 9.3.149.49
```

Ниже приведен список созданных идентификационных ключей и ключей защиты данных:

```
Display of 16 byte HMAC-MD5 authKey:
  18a2c7b78f3df552367383eef9db2e9f

Display of 16 byte HMAC-MD5 localized authKey:
  a59fa9783c04bcbe00359fb1e181a4b4

Display of 16 byte HMAC-MD5 privKey:
  18a2c7b78f3df552367383eef9db2e9f

Display of 16 byte HMAC-MD5 localized privKey:
  a59fa9783c04bcbe00359fb1e181a4b4

Display of 20 byte HMAC-SHA authKey:
  754ebf6ab740556be9f0930b2a2256ca40e76ef9

Display of 20 byte HMAC-SHA localized authKey:
  cd988a098b4b627a0e8adc24b8f8cd02550463e3

Display of 20 byte HMAC-SHA privKey:
  754ebf6ab740556be9f0930b2a2256ca40e76ef9

Display of 16 byte HMAC-SHA localized privKey:
  cd988a098b4b627a0e8adc24b8f8cd02
```

Эти записи должны храниться в файле `/etc/snmpdv3.conf`. Ниже перечислены девять возможных конфигураций:

- Локальные идентификационные ключи и ключи защиты данных на основе протокола HMAC-MD5:

  ```
  USM_USER user1 - HMAC-MD5 a59fa9783c04bcbe00359fb1e181a4b4 DES a59fa9783c04bcbe00359fb1e181a4b4 L - -
  ```

- Нелокальные идентификационные ключи и ключи защиты данных на основе протокола HMAC-MD5:

  ```
  USM_USER user2 - HMAC-MD5 18a2c7b78f3df552367383eef9db2e9f DES 18a2c7b78f3df552367383eef9db2e9f N  - -
  ```

- Локальный идентификационный ключ на основе протокола HMAC-MD5:

  ```
  USM_USER user3 - HMAC-MD5 a59fa9783c04bcbe00359fb1e181a4b4 - - L -
  ```

- Нелокальный идентификационный ключ на основе протокола HMAC-MD5:

  ```
  USM_USER user4 - HMAC-MD5 18a2c7b78f3df552367383eef9db2e9f - - N -
  ```

- Локальные идентификационные ключи и ключи защиты данных на основе протокола HMAC-SHA:

  ```
  USM_USER user5 - HMAC-SHA cd988a098b4b627a0e8adc24b8f8cd02550463e3 DES
  cd988a098b4b627a0e8adc24b8f8cd02 L -
  ```

- Нелокальные идентификационные ключи и ключи защиты данных на основе протокола HMAC-SHA:

  ```
  USM_USER user6 - HMAC-SHA 754ebf6ab740556be9f0930b2a2256ca40e76ef9 DES
  754ebf6ab740556be9f0930b2a2256ca40e76ef9 N -
  ```

- Локальный идентификационный ключ на основе протокола HMAC-SHA:
  ```
  USM_USER user7 - HMAC-SHA cd988a098b4b627a0e8adc24b8f8cd02550463e3 - - L -
  ```
- Нелокальный идентификационный ключ на основе протокола HMAC-SHA:
  ```
  USM_USER user8 - HMAC-SHA 754ebf6ab740556be9f0930b2a2256ca40e76ef9 - - N -
  ```
- Ключ не применяется (**SNMPv1**)
  ```
  USM_USER user9 - none - none - - -
  ```

Для настройки пользователей в м необходимо добавлять записи как в файл `/etc/snmpdv3.conf`, так и в файл `/etc/clsnmp.conf`. Сценарий, описывающий создание пользовательских ключей и изменение файлов конфигурации, приведен в разделе “Создание пользователей в SNMPv3” на стр. 489. Кроме того обратитесь к описаниям команды **pwtokey** в книге *Справочник по командам, том 4* и **clsnmp** в книге *Справочник по командам, том 1*, а также к описанию формата файла `/etc/clsnmp.conf` и `/etc/snmpdv3.conf` в книге *Справочник по файлам* . Кроме того в каталоге `/usr/samples/snmpdv3` находятся примеры файлов `snmpdv3.conf` и `clsnmp.conf`.

Обновление ключей SNMPv3

SNMPv3 позволяет динамически изменять ключи пользователей на основе новых паролей.

Для этого с помощью команды **pwchange** создаются новые пользовательские ключи на основе изменившегося пароля, с помощью команды **clsnmp** динамически изменяются пользовательские ключи в файле `/etc/snmpdv3.conf` и с помощью текстового редактора новые ключи заносятся в файл `/etc/clsnmp.conf`. При этом новый пароль ни разу не передается между системами.

Пошаговые инструкции по изменению пользовательских ключей приведены в разделе “Динамическое обновление ключей идентификации и защиты данных в SNMPv3”. Кроме того обратитесь к описаниям команды **pwchange** в книге *Справочник по командам, том 4* и команды **clsnmp** в книге *Справочник по командам, том 1*, а также к описанию формата файла `/etc/clsnmp.conf` и `/etc/snmpdv3.conf` в книге *Справочник по файлам* .

Динамическое обновление ключей идентификации и защиты данных в SNMPv3

В этом сценарии продемонстрировано динамическое изменение идентификационных ключей пользователя в **SNMPv3**.

В данном сценарии пользователь `u4` изменяет идентификационные ключи для пользователя `u8`. У обоих пользователей, `u4` и `u8`, уже есть идентификационные ключи, созданные на основе `defaultpassword` и IP-адреса `9.3.149.49`, и все работает нормально.

В ходе выполнения сценария для пользователя `u8` будут созданы новые ключи, а файл `/etc/snmpdv3.conf` будет динамически изменен. После этого потребуется вручную изменить в файле `/etc/clsnmp.conf` системы диспетчера идентификационный ключ пользователя `u8` в соответствии с внесенными изменениями.

Перед тем, как начать эту операцию, создайте резервную копию файла `/etc/snmpdv3.conf` в агенте **SNMP** и резервную копию файла `/etc/clsnmp.conf` в диспетчере **SNMP**.

Ниже приведено содержимое файла `/etc/snmpdv3.conf`, который будет динамически изменен:

```
USM_USER u4 - HMAC-MD5 18a2c7b78f3df552367383eef9db2e9f - - N -
USM_USER u8 - HMAC-SHA 754ebf6ab740556be9f0930b2a2256ca40e76ef9 - - N -

VACM_GROUP group1 SNMPv1  public  -
VACM_GROUP group2 USM u4 -
VACM_GROUP group2 USM u8 -

VACM_VIEW defaultView        internet                    - included -

VACM_ACCESS  group1 - - noAuthNoPriv SNMPv1  defaultView - defaultView -
VACM_ACCESS  group2 - - noAuthNoPriv USM defaultView defaultView defaultView -
VACM_ACCESS  group2 - - AuthNoPriv USM defaultView defaultView defaultView -
```

```
VACM_ACCESS  group2 - - AuthPriv USM defaultView defaultView defaultView -

NOTIFY notify1 traptag trap -

TARGET_ADDRESS Target1 UDP 127.0.0.1    traptag trapparms1 - - -
TARGET_ADDRESS Target2 UDP 9.3.149.49   traptag trapparms2 - - -
TARGET_ADDRESS Target3 UDP 9.3.149.49   traptag trapparms3 - - -
TARGET_ADDRESS Target4 UDP 9.3.149.49       traptag trapparms4 - - -

TARGET_PARAMETERS trapparms1 SNMPv1  SNMPv1  public  noAuthNoPriv -
TARGET_PARAMETERS trapparms3 SNMPv2c  SNMPv2c  publicv2c  noAuthNoPriv -
TARGET_PARAMETERS trapparms4 SNMPv3  USM     u4 AuthNoPriv -
```

Ниже приведено содержимое файла `/etc/clsnmp.conf`, который будет изменен для пользователя `u8`:

```
testu4 9.3.149.49 snmpv3 u4 - - AuthNoPriv HMAC-MD5 18a2c7b78f3df552367383eef9db2e9f - -
testu8 9.3.149.49 snmpv3 u8 - - AuthNoPriv HMAC-SHA 754ebf6ab740556be9f0930b2a2256ca40e76ef9 - -
```

Рассмотрите следующие вопросы:

- Описанная ниже процедура была протестирована в отдельных версиях AIX. Результаты, которые вы можете получить, в значительной степени зависят от конкретных версии и уровня AIX.

Обновите пароль и ключи идентификации

Имена связей из файла `/etc/snmpd.conf` становятся частью записей `VACM_GROUP` в файле `/etc/snmpdv3.conf`. Каждую связь необходимо поместить в группу. Затем для групп задаются представления и права доступа.

1. В системе агента **SNMP** введите команду **pwchange**. В этом сценарии запускается следующая команда:

   ```
   pwchange -u auth -p HMAC-SHA пароль-по-умолчанию новый-пароль 9.3.149.49
   ```

   Эта команда создает новый идентификационный ключ.

   - Параметр `-u auth` указывает, что необходимо создать только идентификационный ключ. Если необходимо также изменить и ключи защиты данных, укажите параметр `-u all`.
   - Параметр `-p HMAC-SHA` задает протокол, с помощью которого будет создан идентификационный ключ. Если необходимо изменить и ключи защиты данных, укажите параметр `-p all`.
   - *пароль-по-умолчанию* - пароль, на основе которого был создан последний идентификационный ключ (например, если предыдущий идентификационный ключ был бы создан на основе пароля `bluepen`, то здесь следовало бы указать `bluepen`)
   - *новый-пароль* - это новый пароль, на основе которого будет создан идентификационный ключ. Сохраните этот пароль для последующего использования.
   - `9.3.149.49` - IP-адрес системы, в которой работает агент **SNMP**.

   Вывод этой команды выглядит следующим образом:

   ```
   Dump of 40 byte HMAC-SHA authKey keyChange value:
     8173701d7c00913af002a3379d4b150a
     f9566f56a4dbde21dd778bb166a86249
     4aa3a477e3b96e7d
   ```

   Этот идентификационный ключ применяется на следующем шаге.

   Примечание: Храните заданные вами пароли в надежном месте. Они понадобятся вам при внесении изменений в будущем.

2. В системе диспетчера **SNMP** пользователь `u4` вводит следующую команду, чтобы изменить идентификационный ключ для пользователя `u8`:

   ```
   clsnmp -h testu4 set usmUserAuthKeyChange.12.0.0.0.2.0.0.0.0.9.3.149.49.2.117.56
    \'8173701d7c00913af002a3379d4b150af9566f56a4dbde21dd778bb166a862494aa3a477e3b96e7d\'h
   ```

   - Имя `testu4` применяется потому, что оно присвоено пользователю `u4` в файле `/etc/clsnmp.conf`.
   - ИД экземпляра *usmUserAuthKeyChange* включает в десятеричном формате ИД сервера агента SNMP, на котором изменяются данные, и имя пользователя, идентификационный которого ключ изменяется.

ИД сервера указан в файле /etc/snmpd.boots (файл /etc/snmpd.boots содержит две строки чисел. ИД сервера - первая строка. Не обращайте внимания на вторую строку).

ИД сервера необходимо преобразовать из шестнадцатеричного в десятичное значение. Каждые два знака шестнадцатеричного ИД сервера преобразуются в одно десятичное значение. Например, ИД сервера 00000002000000000039531 преобразуется в 00 00 00 02 00 00 00 00 09 03 95 31. Если каждую пару цифр преобразовать в десятичное значение, то получится 0.0.0.2.0.0.0.0.9.3.149.49 (Таблица преобразования приведена в разделе Таблица преобразования десятичных, шестнадцатеричных, восьмеричных и двоичных значений.). Первое число в строке обозначает количество байт в десятичной строке. В данном случае, это число 12 - 12.0.0.0.2.0.0.0.0.9.3.149.49.

Следующее число обозначает количество байт в имени пользователя, а за ним следуют значения самого имени пользователя. В данном случае, имя пользователя - u8. При преобразовании в десятичные значения u8 преобразуется в 117.56. Так как длина имени пользователя равна 2 байтам, то само имя будет представлено в виде 2.117.56 . Это значение нужно добавить в конец десятичного ИД сервера (Таблица преобразования приведена в разделе Таблица преобразования десятичных, шестнадцатеричных, восьмеричных и двоичных значений.).

В данном случае будет получено значение 12.0.0.0.2.0.0.0.0.9.3.149.49.2.117.56 .

- Следующее значение в команде - новый идентификационный ключ, созданный на предыдущем шаге с помощью команды **pwchange**.

  Примечание: Если для пользователя настроены и ключи защиты данных, то эту процедуру необходимо повторить для изменения ключей защиты данных. При изменении ключей защиты данных вместо значения usmUserAuthKeyChange следует указать значение usmUserPrivKeyChange.

  Применение значения usmUserOwnAuthKeyChange вместо usmUserAuthKeyChange позволит пользователю изменить собственный идентификационный ключ. Например, пользователь u4 мог бы изменить свой идентификационный ключ, указав значение usmUserOwnAuthKeyChange.

Вывод этой команды выглядит следующим образом:

```
1.3.6.1.6.3.15.1.2.2.1.6.12.0.0.0.2.0.0.0.0.9.3.149.49.2.117.56 = '8173701d7c00913af002a3379
d4b150af9566f56a4dbde21dd778bb166a862494aa3a477e3b96e7d'h
```

По завершении выполнения команды файл /etc/snmpdv3.conf автоматически обновляется в системе агента **SNMP** с задержкой в пять минут. Для обновления файла можно также остановить и снова запустить демон **SNMP**. Следующая запись для пользователя u8 автоматически изменяется в файле /etc/snmpdv3.conf:

```
USM_USER u8 00000002000000000039531 HMAC-SHA 4be657b3ae92beee322ee5eaeef665b338caf2d9
None - L nonVolatile
```

3. Введите команду **pwtokey** в системе диспетчера **SNMP**, чтобы создать новый идентификационный ключ на основе нового пароля для сохранения в файле /etc/clsnmp.conf. В этом сценарии запускается следующая команда:

   ```
   pwtokey -u auth -p HMAC-SHA  newpassword 9.3.149.49
   ```

   - Параметр -u auth указывает, что необходимо создать только идентификационный ключ. Если необходимо также изменить и ключи защиты данных, укажите параметр -u all.
   - Параметр -p HMAC-SHA задает протокол, с помощью которого будет создан идентификационный ключ. Если необходимо также изменить ключи защиты данных, укажите параметр -p all.
   - Применяемый пароль (в данном примере - *новый-пароль*) должен совпадать с паролем, который был указан для создания идентификационных ключей с помощью команды **pwchange**.
   - Указанный IP-адрес (в данном примере - 9.3.149.49) должен указывать на систему агента SNMP.

   В результате будут созданы локальный и нелокальный идентификационные ключи:

   ```
   Display of 20 byte HMAC-SHA authKey:
     79ce23370c820332a7f2c7840c3439d12826c10d

   Display of 20 byte HMAC-SHA localized authKey:
     b07086b278163a4b873aace53a1a9ca250913f91
   ```

4. Откройте файл /etc/clsnmp.conf в текстовом редакторе и укажите нелокальный идентификационный ключ в строке пользователя, для которого необходимо изменить ключи. В данном сценарии эта запись имеет следующий вид:

   ```
   testu8 9.3.149.49 snmpv3 u8 - - AuthNoPriv HMAC-SHA 79ce23370c820332a7f2c7840c3439d12826c10d - -
   ```

   Сохраните и закройте файл.
5. Проверьте измененную конфигурацию с помощью следующей команды:

   ```
   clsnmp -v -h testu8 walk mib
   ```

   где *mib* - переменная MIB, для которой у пользователя u8 есть права на чтение. В данном примере, пользователя u8 обладает правами на чтение internet.

Запросы SNMPv3

Команда **clsnmp** предназначена для отправки запросов **SNMP** агенту **SNMP** в локальной или удаленной системе.

С помощью этой команды можно отправлять запросы **SNMPv1**, **SNMPv2c** или **SNMPv3**. Для обработки запроса должен быть настроен файл /etc/clsnmp.conf .

Команда **clsnmp** позволяет передавать запросы get, getnext, getbulk, set, walk и findname. Ниже приведены краткие описания этих запросов:

get позволяет пользователю получить данные из одной переменной MIB

getnext
получает следующую переменную MIB из поддерева MIB

getbulk
получает все переменные MIB из нескольких ветвей MIB

set позволяет пользователю задать переменную MIB

walk получает все переменные MIB из одного поддерева

findname
преобразует идентификатор объекта в имя переменной

trap позволяет команде **clsnmp** получать прерывания через порт 162

Подробная информация о запросах **clsnmp** содержится в описании команды **clsnmp** в *Справочник по командам, том 1*.

Переход от SNMPv1 к SNMPv3

В этом сценарии описана типичная процедура перехода от **SNMPv1** к **SNMPv3**.

В операционной системе AIX агент **SNMP** по умолчанию, который запускается во время загрузки системы, является незашифрованной версией **SNMPv3**. Параметры **SNMPv3** определены в файле конфигурации /etc/snmpdv3.conf. Все параметры, настроенные в файле /etc/snmpd.conf, который используется **SNMPv1** в более ранних версиях операционной системы AIX, необходимо вручную перенести в файл /etc/snmpdv3.conf.

В этом сценарии связи и прерывания, настроенные в файле /etc/snmpd.conf переносятся в файл /etc/snmpdv3.conf. После завершения процедуры функции **SNMPv3** полностью заменят **SNMPv1**. Если связи и прерывания **SNMPv1** не были настроены, то выполнять эту процедуру не нужно.

Этот файл не содержит информацию о функциях **SNMPv1**. Информация о создании пользователей с применением новых функций **SNMPv3** приведена в разделе “Создание пользователей в SNMPv3” на стр. 489.

Ниже приведен пример файла /etc/snmpd.conf, информация из которого будет перенесена в новый файл конфигурации. Настроены следующие связи: daniel, vasu и david. Эти связи необходимо перенести вручную.

```
logging         file=/usr/tmp/snmpd.log         enabled
logging         size=0                          level=0

community       daniel       0.0.0.0     0.0.0.0         readWrite  1.17.35
community       vasu         9.3.149.49  255.255.255.255 readOnly   10.3.5
community       david        9.53.150.67 255.255.255.255 readWrite  1.17.35

view 1.17.35    udp  icmp  snmp 1.3.6.1.2.1.25
view 10.3.5     system interfaces tcp icmp

trap            daniel          9.3.149.49      1.17.35   fe
trap            vasu            9.3.149.49      10.3.5   fe
trap            david           9.53.150.67     1.17.35   fe

smux            1.3.6.1.4.1.2.3.1.2.3.1.1         sampled_password  # sampled
```

Для выполнения шагов этого сценария понадобится файл /etc/snmpd.conf. Перед тем, как приступить к выполнение описанной ниже процедуры, создайте копию этого файла.

Рассмотрите следующие вопросы:

- Описанная ниже процедура была протестирована в отдельных версиях AIX. Результаты, которые вы можете получить, в значительной степени зависят от конкретных версии и уровня AIX.

Шаг 1. Перенос сведений о сообществе

Имена связей из файла /etc/snmpd.conf становятся частью записей VACM_GROUP в файле /etc/snmpdv3.conf. Каждую связь необходимо поместить в группу. Затем для групп задаются представления и права доступа.

1. Войдите в систему от имени пользователя root и откройте файл /etc/snmpdv3.conf в текстовом редакторе. Найдите в этом файле записи VACM_GROUP.
2. Создайте запись VACM_GROUP для каждой связи, которую необходимо перенести. Если для нескольких связей необходимо задать одно представление и одинаковые права доступа, то создайте для них только одну группу. Имена связей из файла /etc/snmpd.conf становятся *именами* в записях VACM_GROUP. В этом сценарии для связей vasu, daniel и david были добавлены следующие записи:

   ```
   #--------------------------------------------------------------------
   # записи VACM_GROUP
   #    Задает группу защиты (состоящую из пользователей и подгрупп)
   #    для Модели управления доступом на основе представлений (VACM).
   # Формат записи:
   #  имя-группы модель-защиты имя тип-памяти
   VACM_GROUP group2 SNMPv1  vasu -
   VACM_GROUP group3 SNMPv1  daniel -
   VACM_GROUP group3 SNMPv1  david -
   #--------------------------------------------------------------------
   ```

   - *имя-группы* может быть любым значением, кроме group1.
   - *модель-защиты* - SNMPv1, так как выполняется перенос связей SNMPv1.
   - В этом сценарии для связей daniel и david в файле /etc/snmpd.conf задано одно представление и одинаковые права доступа. Поэтому они включены в группу group3 в файле /etc/snmpdv3.conf. Связь vasu добавлена в другую группу, так как ее представление и права доступа отличаются от соответствующих параметров связей david и daniel.

   Теперь связи добавлены в группы.

Шаг 2. Перенос сведений представления

Данные представления из файла /etc/snmpd.conf переносятся в записи COMMUNITY, VACM_VIEW и VACM_ACCESS в файле /etc/snmpdv3.conf. Эти записи определяют представление и права доступа для каждой группы.

1. Создайте записи COMMUNITY для связей daniel, vasu и david, сохранив IP-адреса, указанные в параметрах сетевой-адрес и маска-сети в файле /etc/snmpd.conf.

```
#--------------------------------------------------------------------
# COMMUNITY
#    Задает связь для модели защиты на основе связей.
# Формат записи:
#  имя-связи имя уровень-защиты сетевой-адрес маска-сети тип-памяти
COMMUNITY public    public     noAuthNoPriv 0.0.0.0     0.0.0.0         -
COMMUNITY daniel    daniel     noAuthNoPriv 0.0.0.0     0.0.0.0         -
COMMUNITY vasu      vasu       noAuthNoPriv 9.3.149.49  255.255.255.255 -
COMMUNITY david     david      noAuthNoPriv 9.53.150.67 255.255.255.255 -
#--------------------------------------------------------------------
```

2. Создайте записи VACM_VIEW для каждого объекта MIB или переменной, к которым у группы есть права доступа. В соответствии с данными файла /etc/snmpd.conf, связи daniel и david наделены правами доступа к объектам udp, icmp, snmp и 1.3.6.1.2.1.25 (поддерево хоста, определенное в RFC 1514), а vasu обладает правами доступа к объектам system, interfaces, tcp и icmp. Эти записи представления переносятся в файл /etc/snmpdv3.conf следующим образом:

```
#--------------------------------------------------------------------
# записи VACM_VIEW
#    Задает набор данных MIB, называемый представлением, для
#    модели управления доступом на основе представлений.
# Формат записи:
#  имя-представления поддерево-представления маска-представления
#  тип-представления тип-памяти

VACM_VIEW group2View        system                    - included -
VACM_VIEW group2View        interfaces                - included -
VACM_VIEW group2View        tcp                       - included -
VACM_VIEW group2View        icmp                      - included -

VACM_VIEW group3View        udp                       - included -
VACM_VIEW group3View        icmp                      - included -
VACM_VIEW group3View        snmp                      - included -
VACM_VIEW group3View        1.3.6.1.2.1.25            - included -
#--------------------------------------------------------------------
```

3. Определите права доступа к переменным MIB, заданным в записях VACM_VIEW , добавив записи VACM_ACCESS. В файле /etc/snmpd.conf пользователям daniel и david предоставлены права доступа readWrite к переменным MIB, а пользователю vasu - только права доступа readOnly.

   Задайте эти права доступа в записях VACM_ACCESS. В этом сценарии group2 (vasu) предоставляются права group2View для readView и - для writeView, поскольку vasu обладает только правами readOnly к файлу /etc/snmpd.conf. group3 (daniel и david) предоставлены права group3View для readView и writeView, поскольку они обладали правами readWrite в файле /etc/snmpd.conf. См. следующий пример.

```
#-----------------------------------------------------------------------------------------------------------
# записи VACM_ACCESS
#    Определяет права доступа к различным группам защиты
#    в модели управления доступом на основе представлений.
# Формат записи:
# имя-группы префикс-контекста учет-контекста уровень-защиты модель-защиты
# представление-чтения представление-записи представление-уведомления тип-памяти
VACM_ACCESS  group1 - - noAuthNoPriv SNMPv1  defaultView - defaultView -
VACM_ACCESS  group2 - - noAuthNoPriv SNMPv1  group2View - group2View -
VACM_ACCESS  group3 - - noAuthNoPriv SNMPv1  group3View group3View group3View -
#-----------------------------------------------------------------------------------------------------------
```

Шаг 3. Перенос записей прерывания

Записи прерываний из файла /etc/snmpd.conf переносятся в записи NOTIFY, TARGET_ADDRESS и TARGET_PARAMETERS в файле /etc/snmpdv3.conf. Однако переносить следует только записи TARGET_ADDRESS и TARGET_PARAMETERS.

1. IP-адреса, указанные в записях прерываний в файле /etc/snmpd.conf? переносятся в записи TARGET_ADDRESS в файле /etc/snmpdv3.conf. Эта строка определяет систему, в которую отправляются

уведомления. Можно задать записи параметры. В этом сценарии применяются значения trapparms1, trapparms2, trapparms3 и trapparms4, которые будут заданы в записях TARGET_PARAMETERS.

```
#-------------------------------------------------------------------------------
# TARGET_ADDRESS
#    Определяет адрес и параметры управляющего приложения,
#    применяемые при отправке уведомлений.
# Формат записи:
#  имя-целевого-адреса целевой-домен целевой-адрес список параметры тайм-аут
#  число-повторов тип-памяти
TARGET_ADDRESS Target1 UDP 127.0.0.1    traptag trapparms1 - - -
TARGET_ADDRESS Target2 UDP 9.3.149.49   traptag trapparms2 - - -
TARGET_ADDRESS Target3 UDP 9.3.149.49   traptag trapparms3 - - -
TARGET_ADDRESS Target4 UDP 9.53.150.67  traptag trapparms4 - - -
#-------------------------------------------------------------------------------
```

2. Имена связей, указанные в записях прерываний в файле /etc/snmpd.conf, переносятся в записи TARGET_PARAMETERS в файле /etc/snmpdv3.conf. Имена связей сопоставляются с конкретной записью TARGET_ADDRESS с помощью значений параметры. Например, связь daniel соответствует значению trapparms2, которое в записи TARGET_ADDRESS указывает на IP-адрес 9.3.149.49. daniel и IP-адрес 9.3.149.49 были указаны в записи trap в файле /etc/snmpd.conf. Ниже приведен пример вывода команды:

```
#-------------------------------------------------------------------------
# TARGET_PARAMETERS
#    Задает параметры обработки сообщений и защиты, применяемые при
#    отправке уведомлений в целевое управляющее приложение.
# Формат записи:
#  имя-параметров модель-протокола модель-защиты имя уровень-защиты тип-памяти
TARGET_PARAMETERS trapparms1 SNMPv1  SNMPv1  public  noAuthNoPriv -
TARGET_PARAMETERS trapparms2 SNMPv1  SNMPv1  daniel  noAuthNoPriv -
TARGET_PARAMETERS trapparms3 SNMPv1  SNMPv1  vasu    noAuthNoPriv -
TARGET_PARAMETERS trapparms4 SNMPv1  SNMPv1  david   noAuthNoPriv -
#-------------------------------------------------------------------------
```

3. Данные из параметра trapmask в файле /etc/snmpd.conf не переносятся в файл /etc/snmpdv3.conf.

Шаг 4. Перенос данных smux

Строки, содержащие данные smux, можно просто скопировать в новый файл конфигурации. В этом сценарии в файле /etc/snmpd.conf была задана запись smux sampled. Эту строку необходимо скопировать в файл /etc/snmpdv3.conf.

```
#-----------------------------------------------------------------------------------------
#       smux <client OIdentifier> <password> <address> <netmask>
smux            1.3.6.1.4.1.2.3.1.2.3.1.1         sampled_password  # sampled
#-----------------------------------------------------------------------------------------
```

Шаг 5. Перезапуск демона snmpd

После того, как данные из файла /etc/snmpd.conf будут перенесены в файл /etc/snmpdv3.conf, перезапустите демон **snmpd**. Перезапускать демон **snmpd** необходимо при каждом изменении файла /etc/snmpdv3.conf.

1. Введите следующую команду, чтобы остановить работу демона:

   ```
   stopsrc -s snmpd
   ```

2. Введите следующую команду, чтобы запустить демон:

   ```
   startsrc -s snmpd
   ```

Примечание: Простое обновление агента **SNMPv3** не действует, в отличие от **SNMPv1**. После внесения изменений в файле /etc/snmpdv3.conf необходимо перезапустить демон, как показано выше. Функция динамического изменения конфигурации **SNMPv3** не позволяет выполнять обновление.

Создание пользователей в SNMPv3

В этом сценарии продемонстрировано создание пользователей в SNMPv3 путем редактирования файлов `/etc/snmpdv3.conf` и `/etc/clsnmp.conf`.

В этом сценарии создается пользователь `u1`. Пользователь `u1` получит идентификационные ключи, но не получит ключей защиты данных (которые можно создать только при наличии установленного набора файлов `snmp.crypto`). Идентификационные ключи пользователя будут созданы с применением протокола HMAC-MD5. Когда пользователь u1 будет настроен, он будет помещен в группу, после чего для этой группы будут заданы представление и права доступа. И, наконец, для пользователя u1 будут заданы записи прерываний.

Размер каждого из значений в файлах `/etc/snmpdv3.conf` и `/etc/clsnmp.conf` ограничен 32 байтами.

Обратите внимание

- Описанная ниже процедура была протестирована в отдельных версиях AIX. Результаты, которые вы можете получить, в значительной степени зависят от конкретных версии и уровня AIX.

Шаг 1. Создание пользователя

1. Выберите необходимый протокол защиты: HMAC-MD5 или HMAC-SHA. В данном сценарии применяется протокол HMAC-MD5.
2. Создайте идентификационные ключи с помощью команды `pwtokey`. Вывод команды зависит от применяемого протокола идентификации и наличия ключей защиты данных. Эти ключи будут применяться в файлах `/etc/snmpdv3.conf` и `/etc/clsnmp.conf`. Ниже приведена команда, введенная для пользователя `u1`:

   ```
   pwtokey -p HMAC-MD5 -u auth любой-пароль 9.3.230.119
   ```

   В команде указан IP-адрес системы, в которой работает агент. Здесь можно указать любой пароль, но его следует обязательно сохранить в надежном месте, так как он вам понадобится в будущем. Вывод этой команды выглядит примерно следующим образом:

   ```
   Display of 16 byte HMAC-MD5 authKey:
     63960c12520dc8829d27f7fbaf5a0470

   Display of 16 byte HMAC-MD5 localized authKey:
     b3b6c6306d67e9c6f8e7e664a47ef9a0
   ```
3. Войдите в систему как пользователь root и откройте файл `/etc/snmpdv3.conf` в текстовом редакторе.
4. Создайте пользователя, добавив запись `USM_USER` в формате, указанном в этом файле. Вместо значения *authKey* необходимо указать локальный идентификационный ключ, созданный с помощью команды **pwtokey**. Запись для пользователя `u1` выглядит следующим образом:

   ```
   #-------------------------------------------------------------------------
   # записи USM_USER
   #    Определяет пользователя в Модели защиты на основе пользователей (USM).
   # Формат записи:
   #  имя-пользователя ИД-сервера протокол-идентификации идентификационный-ключ
   #  протокол-защиты-данных ключ-защиты-данных тип-ключа тип-памяти
   #
   USM_USER u1 - HMAC-MD5 b3b6c6306d67e9c6f8e7e664a47ef9a0 - - L -
   #-------------------------------------------------------------------------
   ```

   - *имя-пользователя* - имя определяемого пользователя. В данном сценарии имя пользователя - `u1`.
   - *протокол-идентификации* указывает протокол, с помощью которого были созданы ключи. В данном случае это протокол `HMAC-MD5`.
   - *идентификационный-ключ* - локальный идентификационный ключ, который был создан с помощью команды **pwtokey**.
   - *протокол-защиты-данных* и *ключ-защиты-данных* не указаны, так как в данном сценарии ключи защиты данных не использованы.
   - *тип-ключа* - `L`, так как применяется локальный идентификационный ключ.

5. Сохраните и закройте файл /etc/snmpdv3.conf.
6. Откройте в текстовом редакторе файл /etc/clsnmp.conf в системе диспетчера SNMP.
7. Добавьте нового пользователя в соответствии с указанным в файле форматом. Запись для пользователя u1 выглядит следующим образом:

```
#--------------------------------------------------------------------------------------------------
#
# Формат записей:
# имя-winSnmp целевой-агент администрация имя пароль контекст уровень-защиты
# протокол-идентификации идентификационный-ключ протокол-защиты-данных
# ключ-защиты-данных
#
user1  9.3.230.119  SNMPv3  u1  -  -  AuthNoPriv  HMAC-MD5  63960c12520dc8829d27f7fbaf5a0470  -  -
#--------------------------------------------------------------------------------------------------
```

   - *имя-winSnmp* может быть любым значением. Это значение применяется при создании запросов SNMP с помощью команды **clsnmp**.
   - *целевой-агент* - IP-адрес системы, в которой работает агент. Это тот же адрес, который был указан при создании идентификационный ключей.
   - *администрация* - SNMPv3, так как будут отправляться запросы SNMPv3.
   - *имя* - имя создаваемого пользователя. В данном сценарии имя пользователя - u1.
   - *уровень-защиты* - AuthNoPriv, так как для пользователя создаются идентификационные ключи, но не создаются ключи защиты данных (в этой связи значения *протокол-защиты-данных* и *ключ-защиты-данных* отсутствуют).
   - *протокол-идентификации* - протокол идентификации, который был указан при создании идентификационных ключей.
   - *идентификационный-ключ* - нелокальный ключ, созданный командой **pwtokey**.
8. Сохраните и закройте файл /etc/clsnmp.conf.

Шаг 2. Настрйка группы

Теперь этого пользователя необходимо поместить в группу. Если группа с представлением и всеми необходимыми правами доступа для этого пользователя уже существует, то можно включить пользователя в эту группу. Если представление и права доступа этого пользователя должны отличаться от соответствующих параметров других групп, или группы не настроены, то создайте группу и добавьте в нее пользователя.

Для добавления пользователя в новую группу внесите новую запись VACM_GROUP в файл /etc/snmpdv3.conf. Запись группы для пользователя u1 выглядит следующим образом:

```
#-------------------------------------------------------------
# записи VACM_GROUP
#    Задает группу защиты (состоящую из пользователей и подгрупп)
#    для Модели управления доступом на основе представлений (VACM).
# Формат записи:
#  имя-группы модель-защиты имя тип-памяти
VACM_GROUP group1 USM u1 -
#-------------------------------------------------------------
```

- *имя-группы* может быть любым именем. Это имя группы. В данном случае применяется имя group1.
- *модель-защиты* - USM. Эта модель применяет функции защиты SNMPv3.
- *имя* - имя пользователя. В данном сценарии имя пользователя - u1.

Шаг 3. Настройка представления и прав доступа

Для созданной группы необходимо задать представление и права доступа. Эти права доступа задаются путем добавления в файл /etc/snmpdv3.conf записей VACM_VIEW и VACM_ACCESS.

1. Выберите представление и права доступа для новой группы.

2. Добавьте записи VACM_VIEW, определяющие доступные этой группе объекты MIB, в файл /etc/snmpdv3.conf. В этом сценарии группе group1 будут доступны ветви MIB interfaces, tcp, icmp и system. Однако в поддереве MIB system группе group1 будет недоступна переменная MIB sysObjectID.

```
#----------------------------------------------------------------
# записи VACM_VIEW
#    Задает набор данных MIB, называемый представлением, для
#    модели управления доступом на основе представлений.
# Формат записи:
#  имя-представления поддерево-представления маска-представления
#  тип-представления тип-памяти
VACM_VIEW group1View  interfaces    - included -
VACM_VIEW group1View  tcp        - included -
VACM_VIEW group1View  icmp        - included -
VACM_VIEW group1View  system     - included -
VACM_VIEW group1View  sysObjectID    - excluded -
#----------------------------------------------------------------
```

   - *имя-представления* - имя данного представления. В этом сценарии применяется имя представления - group1View.
   - *поддерево-представления* - поддерево MIB, к которому необходимо предоставить доступ.
   - *тип-представления* определяет, включены ли заданные ветви MIB в определяемое представление. В данном примере все ветви включены в представление, а переменная MIB sysObjectID, входящая в поддерево system, исключена.

3. Добавьте запись VACM_ACCESS в файл /etc/snmpdv3.conf, чтобы определить права доступа группы к указанным выше объектам MIB. Группе group1 предоставляются только права на чтение.

```
#-------------------------------------------------------------------------------------------------------------
# записи VACM_ACCESS
#    Определяет права доступа к различным группам защиты
#    в модели управления доступом на основе представлений.
# Формат записи:
# имя-группы префикс-контекста учет-контекста уровень-защиты модель-защиты
# представление-чтения представление-записи представление-уведомления тип-памяти
VACM_ACCESS  group1 - - AuthNoPriv USM group1View - group1View -
#-------------------------------------------------------------------------------------------------------------
```

   - *имя-группы* - имя группы. В данном случае применяется имя group1.
   - *уровень-защиты* - применяемый уровень защиты. В этом сценарии применяются только идентификационные ключи, ключи защиты данных не применяются. В связи с этим задано значение AuthNoPriv.
   - *модель-защиты* - применяемая модель защиты (SNMPv1, SNMPv2c или USM). В этом сценарии указано значение USM для применения функций защиты SNMPv3.
   - *представление-чтения* определяет, для каких представлений VACM_VIEW у группы есть права на чтение. В этом сценарии задано представление group1View и группа group1 получает права на чтение для записей group1View VACM_VIEW .
   - *представление-записи* определяет, для каких представлений VACM_VIEW у группы есть права на запись. В этом сценарии группа group1 не получает прав на запись.
   - *представление-уведомления* указывает имя представления, применяемого при выполнении прерывания под управлением записи таблицы доступа.

   Примечание: В некоторых случаях для одной группы приходится задавать несколько записей VACM_ACCESS. Если группа состоит из пользователей с разными параметрами идентификации и защиты данных (noAuthNoPriv, AuthNoPriv или AuthPriv), то необходимо задать несколько записей VACM_ACCESS с соответствующими значениями параметра уровень-защиты.

Шаг 4. Задание записей прерываний для созданного пользователя

Записи прерываний в SNMPv3 задаются записями NOTIFY, TARGET_ADDRESS и TARGET_PARAMETERS в файле /etc/snmpdv3.conf. Запись TARGET_ADDRESS указывает целевой адрес для отправки уведомлений, а запись TARGET_PARAMETERS - связывает информацию TARGET_ADDRESS с группой group1.

Запись NOTIFY настроена по умолчанию. Ниже приведена запись NOTIFY:

```
NOTIFY notify1 traptag trap -
```

В этом сценарии применяется значение, указанное в записи по умолчанию, - traptag.

1. Добавьте запись TARGET_ADDRESS, чтобы задать целевой адрес для отправки уведомлений.

```
#-------------------------------------------------------------------------------
# TARGET_ADDRESS
#    Определяет адрес и параметры управляющего приложения,
#    применяемые при отправке уведомлений.
# Формат записи:
#  имя-целевого-адреса целевой-домен целевой-адрес список параметры тайм-аут
#  число-повторов тип-памяти
#-------------------------------------------------------------------------------
TARGET_ADDRESS Target1 UDP 9.3.207.107     traptag trapparms1 - - -
```

   - *имя-целевого-адреса* может быть любым именем. В этом сценарии применяется имя Target1.
   - *целевой-адрес* - IP-адрес системы, в которую следует отправлять уведомления для данной группы.
   - *список* - имя, заданное в записи NOTIFY. В этом сценарии применяется имя traptag.
   - *параметры* - может быть любым значением. Здесь указано значение trapparms1, которое будут задано в записи TARGET_PARAMETERS .

2. Добавьте запись TARGET_PARAMETERS.

```
#-------------------------------------------------------------------------
# TARGET_PARAMETERS
#    Задает параметры обработки сообщений и защиты, применяемые при
#    отправке уведомлений в целевое управляющее приложение.
# Формат записи:
#  имя-параметров модель-протокола модель-защиты имя уровень-защиты тип-памяти
#-------------------------------------------------------------------------
TARGET_PARAMETERS trapparms1 SNMPv3  USM     u1        AuthNoPriv -
```

   - *имя-параметра* - имя, указанное в значении параметры в записи TARGET_ADDRESS; в данном случае - trapparms1.
   - *модель-протокола* - применяемая версия SNMP.
   - *модель-защиты* - применяемая модель защиты (SNMPv1, SNMPv3 или USM). В этом сценарии указано значение USM для применения функций защиты SNMPv3.
   - *имя* - имя пользователя, указанное в записи USM_USER; в данном случае - u1.
   - *уровень-защиты* - AuthNoPriv, так как применяются только идентификационные ключи.

Шаг 5. Перезапуск демона snmpd

После внесения изменений в файл /etc/snmpdv3.conf остановите и снова запустите демон **snmpd**.

1. Введите следующую команду, чтобы остановить работу демона **snmpd**:

```
stopsrc -s snmpd
```

2. Введите следующую команду, чтобы запустить демон **snmpd**:

```
startsrc -s snmpd
```

После этого новые параметры вступят в силу.

Примечание: Простое обновление агента SNMPv3 с помощью команды refresh -s snmpd не действует, в отличие от SNMPv1. После внесения изменений в файле /etc/snmpdv3.conf необходимо перезапустить демон, как показано выше. Функция динамического изменения конфигурации SNMPv3 не позволяет выполнять обновление.

Шаг 6. Проверка конфигурации

Для того чтобы проверить правильность настроенной конфигурации, запустите в диспетчере SNMP следующую команду.

```
clsnmp -h user1 walk
mib
```

где *mib* - поддерево MIB, для которого у пользователя есть прав доступа. В этом сценарии, можно было бы указать `interfaces`, `tcp`, `icmp` или `system`. Если конфигурация задана правильно, то будет показана информация из указанного поддерева.

Если необходимый вывод получить не удалось, то просмотрите все шаги в этом документе и убедитесь в том, что все параметры были заданы верно.

Устранение неполадок SNMPv3

При работе с **SNMPv3** могут возникнуть неполадки.

- В процессе миграции записи связей и SMUX, заданные в файле `/etc/snmpd.conf`, необходимо перенести в файл `/etc/snmpdv3.conf`. Инструкции по переносу этой информации приведены в разделе “Переход от SNMPv1 к SNMPv3” на стр. 485.
- На запрос не поступает никаких ответов.

  Эта неполадка, скорее всего, вызвана ошибкой в файле конфигурации `/etc/snmpdv3.conf` или `/etc/clsnmp.conf`, либо в обоих файлах. Внимательно просмотрите эти файлы, чтобы убедиться в отсутствии ошибок. Инструкции по редактированию этих файлов для создания новых пользователей приведены в разделе “Создание пользователей в SNMPv3” на стр. 489.
- Был создан пользователь с идентификационными ключами и ключами защиты данных, но при его применении возвращается сообщение об ошибке.

  Вероятнее всего, применяется версия **SNMPv3** без шифрования. Выполните следующие действия, чтобы определить, какая версия установлена в системе:

  1. Введите команду `ps -e|grep snmpd`.
     - Если вывода не последовало, то, вероятно, необходимо запустить демон **snmpd**. Введите команду `startsrc -s snmpd`.
     - Если в выводе указано `snmpdv1`, то в системе установлена версия **SNMPv1**. При работе с этой версией можно передавать на выполнение запросы **SNMPv1**.
     - Если в выводе указано `snmpdv3ne`, то в системе установлена версия **SNMPv3** без шифрования. После установки операционной системы AIX эта версия будет применяться по умолчанию. Она не поддерживает применение ключей защиты данных.
     - Если в выводе указано `snmpdv3e`, то в системе была дополнительно установлена версия **SNMPv3** с шифрованием. Версия **SNMPv3** с шифрованием может входить в состав пакета AIX Expansion Pack, если это разрешено. Эта версия **SNMPv3** поддерживает применение ключей защиты данных.
  2. Убедитесь в том, что в системе запущена необходимая версия. Если это не так, то с помощью команды **snmpv3_ssw** измените версию:
     - Команда `snmpv3_ssw -1` включает **SNMPv1**
     - Команда `snmpv3_ssw -n` включает **SNMPv3** без шифрования
     - Команда `snmpv3_ssw -e` включает **SNMPv3** с шифрованием, если эта версия установлена в системе.
- После внесения изменений в файл `/etc/snmpdv3.conf` и обновления конфигурации демона изменения не вступили в силу.

  После внесения изменений в файл `/etc/snmpdv3.conf` необходимо перезапустить демон **SNMP**. Обновления конфигурации демона недостаточно. Выполните следующие действия:

  1. Остановите демон **SNMP** с помощью команды `stopsrc -s snmpd`.
  2. Запустите демон **SNMP** с помощью команды `startsrc -s snmpd`.
- Субагент DPI2 запущен, но не удается получить от него переменные MIB.

Наиболее вероятная причина - отсутствие записи связи `public` в файле `/etc/snmpdv3.conf`. По умолчанию субагент DPI2, поставляемый в составе операционной системы AIX, устанавливает соединение с агентом **SNMP** с помощью имени связи `public`. Связь `public` задана в файле `/etc/snmpdv3.conf` по умолчанию. Если связь `public` была удалена из файла `/etc/snmpd.conf`, то добавьте в этот файл следующие строки:

```
VACM_GROUP group1 SNMPv1  public  -
VACM_VIEW defaultView 1.3.6.1.4.1.2.2.1.1.1.0 - included -
VACM_ACCESS  group1 - - noAuthNoPriv SNMPv1  defaultView - defaultView -
COMMUNITY public    public     noAuthNoPriv 0.0.0.0     0.0.0.0         -
```

`1.3.6.1.4.1.2.2.1.1.1.0` - OID объекта dpiPortForTCP.0.

- Отправка запросов к переменным MIB, управляемым узлом SMUX более не поддерживается.

  Убедитесь в наличии записи SMUX в файле `/etc/snmpdv3.conf` и `/etc/snmpd.peers`. Добавляя узлы SMUX, необходимо заносить соответствующие записи в оба этих файла.

- Был реализован личный набор переменных MIB, но добавить или исключить переменные из представлений других пользователей не удалось.

  В записи VACM_VIEW в файле `/etc/snmpdv3.conf` необходимо указать OID переменной MIB вместо ее имени.

- Прерывания не принимаются.

  Проверьте правильность заданных записей прерываний в файле `/etc/snmpdv3.conf`. Кроме того, для прерывания **SNMPv3** необходимо также настроить файл `/etc/clsnmp.conf`. Инструкции по настройке прерываний приведены в разделе “Создание пользователей в SNMPv3” на стр. 489.

  Кроме того, убедитесь в том, что система, указанная в качестве получателя уведомлений (в файле `/etc/snmpdv3.conf`), настроена для их приема. Этот процесс можно запустить в принимающей системе с помощью команды `clsnmp trap`.

- Почему сервер DPI2 не запускается в среде **SNMPv3**?

  В архитектуре **SNMPv3** агент **SNMPv3** сам запускает сервер DPI2. Дополнительная информация приведена в разделе “Архитектура SNMPv3” на стр. 477.

SNMPv1

Эта информация относится к **SNMPv1**. При работе с **SNMPv1**, для определения того, каким станциям, применяющим **Простой протокол управления сетью (SNMP)**, разрешен доступ к переменным базы данных информации управления (MIB), агент **snmpd** использует простую схему идентификации.

Эта схема включает спецификацию стратегий доступа **SNMP** для **SNMPv1**. Стратегия доступа **SNMP** - это связь по управлению между группой взаимодействия **SNMP**, режимом доступа и представлением MIB.

Группа взаимодействия SNMP - это группа с заданным именем, состоящая из одного или нескольких хостов. Имя группы взаимодействия - это последовательность 8-битовых наборов, которую диспетчер **SNMP** вставляет в пакет запроса **SNMP** для идентификации.

Параметр *режим доступа* задает уровень доступа хостов группы взаимодействия к функциям получения и изменения переменных MIB из определенного агента **SNMP**. Существуют следующие режимы доступа: *нет*, *только чтение*, *чтение-запись* и *только запись*.

Представление MIB - определяет одно или несколько поддеревьев MIB, к которым разрешен доступ определенной группе взаимодействия **SNMP**. Представлением MIB может быть все дерево MIB или его ограниченное подмножество.

Когда агент **SNMP** получает запрос, он проверяет, входит ли IP-адрес отправившего запрос хоста в указанную связь. Если хост входит в эту связь **SNMP**, то агент **SNMP** определяет, разрешен ли хосту доступ в заданном режиме к соответствующим переменным MIB согласно стратегии доступа для данной связи. Если запрос удовлетворяет всем критериям, агент **SNMP** попытается выполнить его. В противном случае агент **SNMP** вызовет прерывание *authenticationFailure* или возвратит соответствующее сообщение об ошибке хосту, отправившему запрос.

Пользователь может задавать стратегии доступа **SNMPv1** для агента **snmpd** в файле `/etc/snmpd.conf`. Инструкции по настройке стратегий доступа **SNMP** для агента **snmpd** приведены в описании файла `/etc/snmpd.conf` в книге *Справочник по файлам*.

Настройка демона SNMP

Демон **Простого протокола управления сетью (SNMP)** - это фоновый процесс сервера, который может выполняться на любой рабочей станции с установленным **Протоколом управления передачей/протоколом Internet (TCP/IP)**.

Программа-демон в качестве агента **SNMP** принимает запросы **SNMP** от приложений-диспетчеров, выполняет их идентификацию и обработку. Дополнительные сведения об агентах и диспетчерах приведены в разделах Функции простого протокола управления сетью, Как работает диспетчер и Как работает агент в книге *Communications Programming Concepts*.

Примечание: Термины демон **SNMP**, агент **SNMP** и просто агент эквивалентны.

Для активации демона **snmpd** с минимальной конфигурацией необходим циклический интерфейс **TCP/IP**. Перед запуском **TCP/IP** введите следующую команду:

```
ifconfig lo0 loopback up
```

Демон **SNMP** пытается связать сокеты со стандартными портами **Протокола пользовательских дейтаграмм (UDP)** и **Протокола управления передачей (TCP)**, которые должны быть определены в файле `/etc/services` следующим образом:

```
snmp            161/udp
snmp-trap       162/udp
smux            199/tcp
```

Согласно RFC 1157, службе `snmp` должен быть выделен порт с номером `161`. В файле `/etc/services` этим службам присвоены порты с номерами 161, 162 и 199. Если файл `/etc/services` обслуживается с другого компьютера, то перед запуском демона `/etc/services` указанные порты необходимо освободить в обслуживаемом файле **SNMP** на сервере.

Демон **SNMP** считывает информацию из файла конфигурации текущей версии **SNMP** при запуске, при вызове команды **refresh** (если **snmpd** выполняется под управлением контроллера системных ресурсов) и при получении сигнала **kill -1**.

Файл /etc/snmpd.conf:

В файле конфигурации `/etc/snmpd.conf` задаются имена связей, соответствующие права доступа и представления, хосты для уведомления о прерываниях, атрибуты протокола, особые параметры **snmpd** параметры автономного мультиплексора (SMUX), необходимые для работы демона **SNMP SNMPv1**.

Дополнительная информация приведена в файле /etc/snmpd.conf в книге *Справочник по файлам*.

Работа демона SNMP

Демон **простого протокола управления сетью (SNMP)** обрабатывает запросы **SNMP**, поступающие от приложений-диспетчеров.

Дополнительные сведения об агентах и диспетчерах приведены в разделах Простой протокол управления сетью (SNMP), Как работает диспетчер и Как работает агент книги *Communications Programming Concepts*.

Обработка и идентификация сообщений SNMP:

Все запросы, прерывания и ответы передаются в виде сообщений в формате ASN.1.

Согласно RFC 1157, сообщение имеет следующую структуру:

Версия Группа_взаимодействия PDU

где *Версия* - это версия SNMP (в настоящее время - версия 1), *Группа_взаимодействия* - это имя группы взаимодействия, а *PDU* - блок данных протокола, содержащий запрос, ответ или данные прерывания **SNMP**. PDU также закодирован в соответствии с правилами ASN.1.

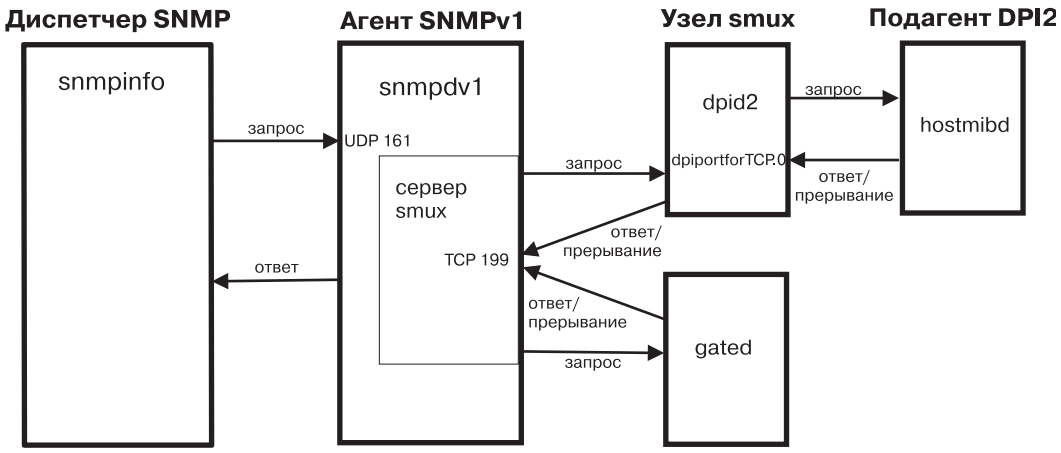


Рисунок 28. Основные компоненты архитектуры SNMPv1

На этой схеме показан пример архитектуры **SNMPv1**. Субагент DPI2, равноправный узел smux, диспетчер **SNMP** и агент **SNMP**. Кроме того, на схеме показано взаимодействие этих компонентов.

Демон **SNMP** принимает и передает сообщения **SNMP** с помощью протоколов **(TCP/IP) и UDP**. Запросы принимаются через стандартный порт 161. Прерывания передаются хостам, указанным в соответствующем списке в файле `/etc/snmpd.conf`. Сообщения о прерываниях принимаются через порт 162.

При получении запроса IP-адрес источника и имя группы взаимодействия проверяются по списку, содержащему IP-адреса, имена групп взаимодействия, права доступа и представления, заданные в файле `/etc/snmpd.conf`. Агент **snmpd** читает этот файл при запуске, при вызове команды **refresh** и при отправке сигнала **kill -1**. Если соответствующая запись в файле не обнаружена, запрос будет проигнорирован. Если же запись обнаружена, то доступ будет разрешен в соответствии с правами доступа, заданными для группы взаимодействия и представления, отвечающих данному IP-адресу, имени группы взаимодействия и представлению в файле `/etc/snmpd.conf`. И сообщение, и PDU должны быть закодированы в соответствии с правилами ASN.1.

Данная схема идентификации не предназначена для обеспечения полной защиты. Если демон **SNMP** применяется только для отправки запросов get и get-next, то дополнительные меры защиты не нужны. Если же разрешены запросы set, то права доступа для запросов set следует ограничить.

Дополнительная информация приведена в файле `/etc/snmpd.conf` в книге *Справочник по файлам*. Кроме того, более подробную информацию можно найти в разделе База информации управления (MIB) в *Communications Programming Concepts*.

Обработка запросов SNMP:

Демон **SNMP** получает PDU запросов трех типов.

Эти типы запросов определены в RFC 1157, а все PDU соответствуют следующему формату:

Таблица 83. Формат PDU запроса

ИД_запроса	состояние_ошибки	индекс_ошибки	значения_переменных
GET	0	0	*Значения переменных*
GET-NEXT	0	0	*Значения переменных*
SET	0	0	*Значения переменных*

Поле ИД_запроса определяет причину запроса; поля состояние_ошибки и индекс_ошибки не используются и должны иметь значения 0 (нуль); поле значения_переменных представляет собой список переменной длины, содержащий числовые ИД экземпляров, значения которых запрашиваются. Если значение поля ИД-запроса равно SET, то поле значения-переменных представляет собой список пар ИД экземпляра - значение экземпляра.

Дополнительная информация о типах запросов приведена в разделе Работа с базой информации управления (MIB) в *Communications Programming Concepts*.

Обработка ответов SNMP:

Формат PDU ответов почти совпадает с форматом PDU запросов.

Таблица 84. Формат PDU ответа

ИД_запроса	состояние_ошибки	индекс_ошибки	значения_переменных
GET-RESPONSE	*состояние_ошибки*	*индекс_ошибки*	*Значения переменных*

Если запрос был обработан успешно, значения полей состояние_ошибки и индекс_ошибки равны 0 (нулю), а поле значения-переменных представляет собой полный список пар ИД и значений экземпляров.

Если какой-либо ИД экземпляра в поле значения_переменных в PDU запроса не был обработан успешно, агент SNMP прекращает работу, записывает индекс ИД экземпляра, вызвавшего сбой, в поле индекс_ошибки, записывает код ошибки в поле состояние_ошибки и копирует неполный список в поле значения_переменных.

В RFC 1157 для поля состояние_ошибки определены следующие значения:

Таблица 85. Значения поля состояние-ошибки

Значение	Значение	Описание
noError	0	Обработка завершена успешно (индекс-ошибки равен 0).
tooBig	1	Размер PDU ответа превышает ограничение, накладываемое реализацией (индекс-ошибки равен 0).
noSuchName	61 см	ИД экземпляра не существует в соответствующем представлении MIB для типов запросов GET и SET, или у него нет потомков в дереве MIB в соответствующем представлении MIB для типов запросов GET-NEXT (ненулевое значение индекса-ошибки).
badValue	3	Только для запросов SET: заданное значение синтаксически несовместимо с типом соответствующего ИД экземпляра (ненулевое значение индекса-ошибки).
readOnly	4	Не определено.

Таблица 85. Значения поля состояние-ошибки (продолжение)

Значение	Значение	Описание
genErr	5	Произошла ошибка, связанная с ограничениями реализации (ненулевое значение индекса-ошибки); например, попытка присвоить значение, превышающее максимальное значение для данной реализации.

Обработка прерываний SNMP:

Согласно RFC 1157, PDU прерываний представляются в следующем формате, указанном в данной таблице.

Таблица 86. Формат PDU прерываний

объект	адрес-агента	Шаблон-прерывания	конкретное-прерывание	системное-время	значения-переменных
ИД объекта	*Целое*	*Целое*	*Целое*	*Время в тактах*	*Значения переменных*

Информация о полях:

Элемент	Описание
объект	Идентификатор объекта, присвоенный разработчику данной реализации агента. Это значение переменной **sysObjectID**, которое уникально для каждой реализации агента **SNMP**. Данной реализации агента присвоено значение **1.3.6.1.4.1.2.3.1.2.1.1.3** или **risc6000snmpd.3**.
адрес-агента	IP-адрес объекта, вызывающего прерывание.
Шаблон-прерывания	Целое, принимающее одно из следующих значений: **0** *coldStart* **1** *warmStart* **61 см** *linkDown* **3** *linkUp* **4** *authenticationFailure* **5** *egpNeighborLoss* **6** *enterpriseSpecific*
Конкретное прерывание	Не используется, зарезервировано для применения в будущем.
Системное время	Время в сотых долях секунды, прошедшее с момента последней инициализации объекта до момента возникновения события, которое привело к вызову прерывания.
Значения переменных	Дополнительная информация, зависящая от типа *общего-прерывания*.

Ниже приведены значения шаблонов прерываний, которые уведомляют об обнаружении того или иного системного события:

Элемент	Описание
coldStart	Повторная инициализация объекта. Возможно, были изменены данные конфигурации или значения переменных MIB. Перезапустите службу Measurement epochs.
warmStart	Происходит повторная инициализация объекта, но данные конфигурации и значения переменных MIB не изменились. В данной реализации агента **SNMP** прерывание *warmStart* вызывается при повторном чтении файла `/etc/snmpd.conf`. Информация о конфигурации в файле `/etc/snmpd.conf` предназначена для настройки агента без побочного эффекта для баз данных диспетчера SNMP. Повторный запуск Measurement epochs не требуется.
linkDown	Агент обнаружил, что интерфейс связи был отключен.
linkUp	Агент обнаружил, что интерфейс связи был подключен.
authenticationFailure	Не удалось идентифицировать полученное сообщение.

Элемент	Описание
egpNeighborLoss	Связь с маршрутизатором по **Протоколу внешних шлюзов (EGP)** прервана. Это прерывание генерируется только в том случае, если агент работает в той же системе, что и демон **gated**, применяющий **EGP**.
enterpriseSpecific	Не реализовано; зарезервировано для использования в будущем.

В списке значений переменных для прерываний *linkDown* и *linkUp* предусмотрена единственная пара ИД экземпляра - значение. ИД экземпляра определяет **ifIndex** отключенного или включенного адаптера, значение - это значение **ifIndex**. Кроме того, прерывание для *egpNeighborLoss* содержит сведения о связи между ИД экземпляра и значением *egpNeighAddr* для узла, связь с которым была прервана.

Поддержка демоном SNMP переменных MIB семейства EGP

Если на хосте агента работает демон **gated** и применяется **Протокол внешних шлюзов (EGP)**, то в группе **EGP** есть несколько переменных MIB, поддерживаемых демоном **gated** и доступных для агента **snmpd**.

Все перечисленные ниже переменные MIB **EGP** существуют в единственном экземпляре:

Элемент	Описание
egpInMsgs	Число сообщений **EGP**, полученных без ошибок.
egpInErrors	Число сообщений **EGP**, полученных с ошибкой.
egpOutMsgs	Общее число сообщений **EGP**, переданных программой-демоном **gated**, которая выполняется на хосте агента.
egpOutErrors	Число сообщений **EGP**, которые демону **gated** на хосте агента не удалось отправить из-за ограничений на ресурсы.
egpAs	Номер автономной системы, связанный с демоном **gated** на хосте агента.

Следующие переменные MIB **EGP** создаются для каждого узла и соседа **EGP**, зарегистрированного демоном **gated** хоста агента:

Элемент	Описание
egpNeighState	Состояние данного узла **EGP**: **1** простой **61 см** регистрация **3** down **4** up **5** отказ.
egpNeighAddr	IP-адрес данного узла **EGP**.
egpNeighAs	Номер автономной системы данного узла **EGP**. Нулевое значение (0) означает, что номер этого узла неизвестен.
egpInNeighMsgs	Число сообщений **EGP**, полученных от данного узла **EGP** без ошибок.
egpNeighInErrs	Число сообщений **EGP**, полученных данного узла **EGP** с ошибкой.
egpNeighOutMsgs	Число локально созданных сообщений **EGP** для данного узла **EGP**.
egpNeighOutErrs	Число локальных сообщений **EGP**, не отправленных данному узлу **EGP** из-за ограничений на ресурсы.
egpNeighInErrMsgs	Число сообщений об ошибках **EGP**, полученных от данного узла **EGP**
egpNeighOutErrMsgs	Число сообщений об ошибках **EGP**, отправленных на данный узел **EGP**
egpNeighStateUp	Число переходов **EGP** в состояние АКТИВЕН для данного узла **EGP**.
egpNeighStateDowns	Число переходов **EGP** из состояния АКТИВЕН в любое другое состояние для данного узла **EGP**.
egpNeighIntervalHello	Интервал между повторами передачи команд Hello **EGP** в сотых долях секунды.
egpNeighIntervalPoll	Интервал между повторами передачи команд опроса **EGP** в сотых долях секунды.
egpNeighMode	Режим опроса данного узла **EGP**. Активный (1) или пассивный (2).
egpNeighEventTrigger	Переменная, управляющая событиями запуска и останова для данного узла **EGP**. Эта переменная MIB может принимать следующие значения: запуск (1) и останов (2).

Если демон **gated** не запущен, не настроен для взаимодействия с агентом **snmpd** или не настроен для работы с **EGP**, то при обращении к этим переменным с помощью запросов get и set будет получен код ошибки *noSuchName*.

Файл конфигурации демона **gated**, `/etc/gated.conf`, должен содержать в себе следующий параметр:

```
snmp      yes;
```

Демон **gated** разработан для работы в режиме узла или промежуточного агента протокола одиночного мультиплексора (SMUX) демона **snmpd** простого протокола управления сетью (SNMP). При запуске программы-демона **gated** происходит регистрация дерева переменных MIB *ipRouteTable* с помощью агента **snmpd**. Если программа-демон **gated** настроена для работы с **EGP**, то она также регистрирует дерево переменных MIB **EGP**. После того как регистрация выполнена, диспетчер SNMP может запросить у агента **snmpd** сведения о *ipRouteTable* и о переменных MIB **EGP**, поддерживаемых программой-демоном **gated** на хосте этого агента. Таким образом, если демон **gated** запущен, то всю информацию о маршрутизации MIB можно получить с помощью этого демона. В этом случае не разрешается отправлять запросы set к таблице *ipRouteTable*.

Соединение SMUX между демонами **gated** и **snmpd** осуществляется с применением стандартного порта 199 Протокола управления передачей (TCP). При завершении работы демона **gated** демон **snmpd** немедленно отменяет регистрацию деревьев, зарегистрированных ранее демоном **gated**. Если демон **gated** был запущен раньше, чем демон **snmpd**, то **gated** периодически проверяет наличие **snmpd**, пока не удастся установить связь SMUX.

Для того, чтобы агент **snmpd** определял и разрешал установление связи с SMUX клиентом демона **gated**, пользователь должен включить запись SMUX в файл `/etc/snmpd.conf`. Идентификатор и пароль клиентского объекта, заданные в этой записи SMUX для программы-демона **gated**, должны совпадать со значениями, заданными в файле `/etc/snmpd.peers`.

Агент **snmpd** поддерживает запросы set для следующих переменных MIB I и MIB II, предназначенных для чтения и записи:

sysContact
: Текстовое описание лица, ответственного за хост этого агента. Сюда включается имя и контактная информация, например: "Bob Smith, тел. 555-5555, доб. 5." Длина записи не должна превышать 256 символов. Если длина значения этой переменной MIB в запросе set превышает 256 символов, агент **snmpd** возвращает код ошибки *badValue*, а операция set не выполняется. Начальное значение переменной*sysContact* определено в файле `/etc.snmp.conf`. Если значение не задано, используется пустая строка.

Экземпляр	Значение	Действие
0	"строка"	Переменной MIB присвоено значение "строка".

sysName
: Имя хоста этого агента. Как правило, указывается полное имя хоста. Длина записи не должна превышать 256 символов. Если длина значения этой переменной MIB в запросе set превышает 256 символов, агент **snmpd** возвращает код ошибки *badValue*, а операция set не выполняется.

Экземпляр	Значение	Действие
0	"строка"	Переменной MIB присвоено значение "строка".

sysLocation

Строка текста, определяющая физическое расположение компьютера, на котором установлен демон **snmpd**, например: "Здание 802, лаборатория 3C-23." Длина записи не должна превышать 256 символов. Если длина значения этой переменной MIB в запросе set превышает 256 символов, агент **snmpd** возвращает код ошибки *badValue*, а операция set не выполняется. Начальное значение переменной *sysLocation* определено в файле `/etc/snmp.conf`. Если значение не задано, используется пустая строка.

Экземпляр	Значение	Действие
0	"строка"	Переменной MIB присвоено значение "строка".

ifAdminStatus

Требуемое состояние адаптера интерфейса на хосте агента. Поддерживаются состояния "активен" и "неактивен". Вы можете задать состояние "тестирование", однако это не повлияет на рабочее состояние интерфейса.

Экземпляр	Значение	Действие
f	1	Адаптер интерфейса с **ifIndex f** включен.

Примечание: Для переменной *ifAdminStatus* может быть установлено значение up или down несмотря на то, что фактически состояние интерфейса не было изменено из-за ошибки. В этом случае запрос get для *ifAdminStatus* может вернуть значение *up*, тогда как переменная *ifOperStatus* для этого интерфейса равна *down*. В этом случае администратор сети должен еще раз попытаться изменить состояние, установив для переменной *ifAdminStatus* значение up с помощью запроса set.

atPhysAddress

Часть записи таблицы адресов на хосте агента, содержащая аппаратный адрес (запись в таблице протокола преобразования адресов, ARP). Эта переменная MIB эквивалентна *ipNetToMediaPhysAddress*.

Экземпляр	Значение	Действие
f.1.n.n.n.n	hh:hh:hh:hh:hh:hh	Для интерфейса с **ifIndex f** все существующие записи таблицы ARP об IP-адресе n.n.n.n заменяются на (n.n.n.n, hh:hh:hh:hh:hh:hh). Если такая связь не существует, добавляется новая запись. hh:hh:hh:hh:hh:hh - это аппаратный адрес, состоящий из двенадцати шестнадцатеричных цифр.

atN0etAddress

IP-адрес, соответствующий аппаратному или физическому адресу, указанному в *atPhysAddress*. Эта переменная MIB эквивалентна *ipNetToMediaNetAddress*.

Экземпляр	Значение	Действие
f.1.n.n.n.n	m.m.m.m	Для интерфейса с **ifIndex f** IP-адрес n.n.n.n в записи таблицы ARP заменяется на IP-адрес m.m.m.m.

ipForwarding

Указывает, выполняет ли хост агента пересылку дейтаграмм.

Таблица 87. ipforwarding

Экземпляр	Значение	Действие
0	1	Если на хосте агента установлено несколько активных интерфейсов, то ядро **TCP/IP** будет настроено для пересылки пакетов. Если на хосте агента установлен один активный интерфейс, запрос set выполнен не будет.
0	61 см	Ядро **TCP/IP** на хосте агента не настроено для пересылки пакетов.

ipDefaultTTL

Значение максимального числа участков (TTL), которое включается в IP-заголовки дейтаграмм, создаваемых хостом агента.

Экземпляр	Значение	Действие
0	n	Значение TTL для средств поддержки протокола IP устанавливается равным целому числу n.

ipRouteDest

IP-адрес назначения маршрута в таблице маршрутизации.

Экземпляр	Значение	Действие
n.n.n.n	m.m.m.m	IP-адрес назначения для маршрута n.n.n.n принимает значение m.m.m.m.

ipRouteNextHop

Шлюз, через который должны отправляться пакеты получателю с указанным IP-адресом (запись в таблице маршрутизации).

Экземпляр	Значение	Действие
n.n.n.n	m.m.m.m	В таблицу маршрутизации добавляется запись о том, что сеть n.n.n.n достижима через шлюз m.m.m.m. Части IP-адреса n.n.n.n, определяющей хост, необходимо присвоить нулевое значение для обозначения адреса сети.

ipRouteType

Состояние записи таблицы маршрутизации на хосте агента (используется для удаления записей).

Экземпляр	Значение	Действие
h.h.h.h	1	Все маршруты с IP-адресами хоста h.h.h.h удаляются.
n.n.n.n	61 см	Все маршруты с IP-адресами хоста n.n.n.n удаляются.

ipNetToMediaPhysAddress

Часть записи таблицы адресов на хосте агента, содержащая аппаратный адрес (запись таблицы ARP). Эта переменная MIB эквивалентна *atPhysAddress*.

Экземпляр	Значение	Действие
f.1.n.n.n.n	hh:hh:hh:hh:hh:hh	Для интерфейса с **ifIndex f** все существующие записи таблицы ARP об IP-адресе n.n.n.n заменяются на (n.n.n.n, hh:hh:hh:hh:hh:hh). Если такая связь не существует, добавляется новая запись. hh:hh:hh:hh:hh:hh - это аппаратный адрес, состоящий из 12 шестнадцатеричных цифр.

ipNetToMediaNetAddress

IP-адрес, соответствующий аппаратному или физическому адресу, указанному в *ipNetToMediaPhysAddress*. Эта переменная MIB эквивалентна *atNetAddress*.

Экземпляр	Значение	Действие
f.1.n.n.n.n	m.m.m.m	Для интерфейса с **ifIndex f** IP-адрес n.n.n.n в записи таблицы ARP заменяется на IP-адрес m.m.m.m.

ipNetToMediaType

Тип преобразования IP-адресов в физические адреса.

Экземпляр	Значение	Действие
f.1.n.n.n.n	1	Для записи о связи IP-адреса интерфейса с **ifIndex f** и физического адреса устанавливается тип преобразования 1 - "другой".
f.1.n.n.n.n	61 см	Для записи таблицы ARP о связи IP-адреса интерфейса с **ifIndex f** и физического адреса устанавливается тип преобразования 2 - "недопустимый". Кроме того, как недопустимое определяется значение соответствующей записи в **ipNetMediaTable**; т.е. связь интерфейса с этой записью **ipNetToMediaTable** прекращается.
f.1.n.n.n.n	3	Для записи таблицы ARP о связи IP-адреса интерфейса с **ifIndex f** и физического адреса устанавливается тип преобразования 3 - "динамическое".
f.1.n.n.n.n	4	Для записи таблицы ARP о связи IP-адреса интерфейса с **ifIndex f** и физического адреса устанавливается тип преобразования 4 - "статическое".

snmpEnableAuthenTraps

Указывает, настроен ли агент **snmpd** для вызова прерываний *authenticationFailure*.

Экземпляр	Значение	Действие
0	1	Агент **snmpd** будет вызывать прерывания, связанные с ошибками идентификации.
0	61 см	Агент **snmpd** не будет вызывать прерывания, связанные с ошибками идентификации.

smuxPstatus

Состояние узла протокола SMUX (используется для удаления узлов SMUX).

Экземпляр	Значение	Действие
n	1	Агент **snmpd** не выполняет никаких действий.
n	61 см	Агент **snmpd** прекратит обмен данными с узлом SMUX n.

smuxTstatus

Состояние дерева MIB SMUX (используется для удаления точек монтирования дерева MIB).

Экземпляр	Значение	Действие
l.m.m.m._ _ _ .p	1	Агент **snmpd** не выполняет никаких действий.
l.m.m.m._ _ _ .p	61 см	Отменяет монтирование SMUX дерева MIB m.m.m... где *l* - это длина экземпляра дерева MIB, а *p* - значение smuxTpriority.

Ниже перечислены доступные для изменения переменные, определенные в RFC 1229. Программа-демон **snmpd** предоставляет пользователю возможность устанавливать значения этих переменных. Базовое устройство может не поддерживать установку значений этих переменных. Узнайте, настройку каких переменных поддерживают различные устройства.

ifExtnsPromiscuous

Состояние смешанного режима для заданного устройства. Используется для включения и выключения смешанного режима для заданного устройства. Этот режим полностью контролируется **snmpd**. При отключении **snmpd** смешанный режим полностью выключается, независимо от того, какие приложения выполняются в системе.

Экземпляр	Значение	Действие
n	1	Включает смешанный режим для устройства n.
n	61 см	Выключает смешанный режим для устройства n.

ifExtnsTestType

Переменная начала тестирования. При установке значения этой переменной для данного устройства запускается соответствующий тест. Значением переменной является Идентификатор объекта. Конкретное значение зависит от типа устройства и теста, который требуется выполнить. В настоящий момент **snmpd** поддерживает только тест testFullDuplexLoopBack.

Экземпляр	Значение	Действие
n	oid	Выполнить тест, заданный с помощью oid.

ifExtnsRcvAddrStatus

Переменная состояния адреса. При присвоении этой переменной значения создается указанный адрес с соответствующим временем жизни. **snmpd** позволяет устанавливать только временные адреса, так как он не может изменять записи Администратора объектных данных (ODM) для устройства. Можно задавать только адреса многоцелевой рассылки или оповещения.

Экземпляр	Значение	Действие
n.m.m.m.m.m.m	1	Добавить адрес, не являющийся временным или постоянным.
n.m.m.m.m.m.m	61 см	Удалить адрес, прекратив его использование.
n.m.m.m.m.m.m	3	Добавить временный адрес.
n.m.m.m.m.m.m	4	Добавить постоянный адрес.

Значения приведенным ниже переменным можно присваивать в соответствии с RFC 1231. Программа-демон **snmpd** предоставляет пользователю возможность устанавливать значения этих переменных. Базовое устройство может не поддерживать установку значений этих переменных. Узнайте, настройку каких переменных поддерживают различные устройства.

dot5Commands

Команды устройства Token-Ring.

Экземпляр	Значение	Действие
n	1	Не выполняет действий. Возвращает управление.
n	61 см	Открывает устройство Token-Ring.
n	3	Выполняет сброс устройства Token-Ring.
n	4	Закрывает устройство Token-Ring.

dot5RingSpeed

Текущее быстродействие или пропускная способность Token-Ring.

Экземпляр	Значение	Действие
n	1	Неизвестное быстродействие.
n	61 см	Быстродействие 1 Мбит/с.
n	3	Быстродействие 4 Мбит/с.
n	4	Быстродействие 16 Мбит/с.

dot5ActMonParticipate

Этот объект определяет, будет ли данное устройство задействовано при выборе активного монитора.

Экземпляр	Значение	Действие
n	1	Будет задействовано.
n	61 см	Не будет задействовано.

dot5Functional

Функциональная маска, позволяющая устройству Token-Ring указывать адреса хостов, от которых он получает кадры.

Экземпляр	Значение	Действие
n	m.m.m.m.m.m	Устанавливаемая функциональная маска.

Согласно RFC, приведенные ниже переменные для работы с таймером изменять нельзя, однако вы можете как просматривать их значения, так и изменять их. Для получения полной информации об этих переменных ознакомьтесь с документом RFC. Демон **snmpd** предоставляет возможность изменить значения этих переменных, однако устройство может не поддерживать эту функцию. Для получения дополнительной информации обратитесь к документации по драйверу устройства. Список переменных:

- dot5TimerReturnRepeat
- dot5TimerHolding
- dot5TimerQueuePDU
- dot5TimerValidTransmit
- dot5TimerNoToken
- dot5TimerActiveMon
- dot5TimerStandbyMon
- dot5TimerErrorReport
- dot5TimerBeaconTransmit
- dot5TimerBeaconReceive.

Демон SNMP позволяет пользователю изменять значения следующих переменных. Для получения информации применяется протокол Управления станцией (SMT) FDDI 7.2, реализованный на уровне микрокода. Просмотрите информацию о микрокоде в документации по FDDI и убедитесь, что используется микрокод SMT 7.2.

fddimibSMTUserData

32-разрядная переменная, содержащая с пользовательскую информацию.

Экземпляр	Значение	Действие
n	строка	Хранит 32 байта пользовательской информации.

fddimibSMTConfigPolicy

Состояние стратегий конфигурации, в частности, использование стратегии блокировки.

Экземпляр	Значение	Действие
n	0	Не использовать стратегию блокировки.
n	1	Использовать стратегию блокировки.

fddimibSMTConnectionPolicy

Состояние стратегий связи на узле FDDI. Дополнительная информация о конкретных переменных, допускающих установку значения, содержится в RFC 1512.

Экземпляр	Значение	Действие
n	k	Определяет стратегии связи.

fddimibSMTTNotify

Интервал в секундах, используемый протоколом уведомления соседей. Принимает значения от 2 до 30 секунд; значение по умолчанию - 30 секунд.

Экземпляр	Значение	Действие
n	k	Определяет значение интервала.

fddimibSMTStatRptPolicy

Состояние процесса создания кадров, содержащих отчет о состоянии.

Экземпляр	Значение	Действие
n	1	Указывает, что узел будет создавать кадры с информацией о состоянии для реализованных событий.
n	61 см	Указывает, что узел не будет создавать кадры с информацией о состоянии.

fddimibSMTTraceMaxExpiration

Эта переменная определяет максимальное значение времени окончания работы таймера для трассировки.

Экземпляр	Значение	Действие
n	k	Задает максимальное значение времени окончания работы таймера в миллисекундах.

fddimibSMTStationAction

Эта переменная отвечает за выполнение объектом SMT определенных действий. Для получения более подробной информации об этой переменной обратитесь к RFC.

Экземпляр	Значение	Действие
n	k	Определяет действие для объекта SMT. Принимает значения от 1 до 8.

fddimibMACRequestedPaths

Определяет пути для настройки управления доступом к среде передачи данных (MAC).

Экземпляр	Значение	Действие
n.n	k	Определяет запрошенный путь для MAC.

fddimibMACFrameErrorThreshold

Пороговое значение для создания отчета о состоянии MAC. Определяет, сколько ошибок должно произойти для того, чтобы система начала создавать отчет.

Экземпляр	Значение	Действие
n.n	k	Определяет, сколько ошибок должно быть обнаружено, чтобы был создан отчет о состоянии MAC.

fddimibMACMAUnitdataEnable

Эта переменная определяет значение флага **MA_UNITDATA_Enable** в RMT. Значение по умолчанию и начальное значение этого флага равно true (1).

Экземпляр	Значение	Действие
n.n	1	Присваивает флагу MA_UNITDATA_Enable значение true.
n.n	61 см	Присваивает флагу MA_UNITDATA_Enable значение false.

fddimibMACNotCopiedThreshold

Пороговое значение для создания отчета о событиях MAC.

Экземпляр	Значение	Действие
n.n	k	Определяет, сколько ошибок должно быть обнаружено, чтобы был создан отчет о событиях MAC.

Три следующие переменные - это переменные таймера, взаимодействующие между собой. Прежде чем изменять значения этих переменных, рекомендуется ознакомиться с их описанием в **RFC 1512**.

- fddimibPATHTVXLowerBound
- fddimibPATHTMaxLowerBound
- fddimibPATHMaxTReq

fddimibPORTConnectionPolicies

Задает стратегии связи для указанного порта.

Экземпляр	Значение	Действие
n.n	k	Задает стратегии связи для указанного порта.

fddimibPORTRequestedPaths

Данная переменная содержит дерево разрешенных путей, причем каждый лист дерева задает разрешенные пути к порту. Первые восемь разрядов соответствуют 'нет', вторые - 'дереву', третьи - 'узлу'.

Экземпляр	Значение	Действие
n.n	ccc	Задает пути для порта.

fddimibPORTLerCutoff

Оценка частоты возникновения ошибок на линии связи, при которой связь прерывается. Возможны значения от 10**-4 до 10**-15; значение задается как абсолютная величина десятичного логарифма (по умолчанию - 7).

Элемент	Описание	
Экземпляр	**Значение**	**Действие**
n.n	k	Определяет значение LerCutoff для порта.

fddimibPORTLerAlarm

Оценка частоты возникновения ошибок на линии связи, при которой будет отправлено предупреждение. Возможны значения от 10**-4 до 10**-15; значение задается как абсолютная величина десятичного логарифма (по умолчанию - 8).

Экземпляр	Значение	Действие
n.n	k	Определяет значение LerAlarm для порта.

fddimibPORTAction

Эта переменная отвечает за выполнение портом определенных действий. Для получения более подробной информации об этой переменной обратитесь к RFC.

Экземпляр	Значение	Действие
n	k	Определяет действие для определенного порта. Принимает значения от 1 до 6.

Примечание: Согласно RFC 1213, все переменные в таблицах *atEntry* и *ipNetToMediaEntry* предназначены для чтения и записи. Поддержка метода set реализована только для переменных *atPhysAddress* и *atNetAddress* в таблице *atEntry* и переменных *ipNetToMediaEntry* variables *ipNetToMediaPhysAddress, ipNetToMediaNetAddress* и *ipNetToMediaType.* Для приема запросов set, связанных с неподдерживаемыми атрибутами этих двух таблиц, применяются запросы set для других переменных: *atIfIndex* и *ipNetToMediaIfIndex.* Функции, отправившей запрос, не возвращается сообщение об ошибке, но при следующем запросе get будет выяснено, что исходные значения не изменились.

Согласно RFC 1213, все поля в таблице *ipRouteEntry*, за исключением *ipRouteProto*, доступны как для чтения, так и для записи. Как было сказано выше, поддержка метода set реализована только для переменных *ipRouteDest*, *ipRouteNextHop* и *ipRouteType*. Для поддержки запросов set, в которых можно указать нереализованные атрибуты маршрутизации, применяются запросы set для других переменных в таблице *ipRouteEntry*: *ipRouteIfIndex*, *ipRouteMetric1*, *ipRouteMetric2*, *ipRouteMetric3*, *ipRouteMetric4*, *ipRouteMetric5*, *ipRouteAge* и *ipRouteMask*. Функции, отправившей запрос, не возвращается сообщение об ошибке, но при следующем запросе get будет выяснено, что исходные значения не изменились. Демон **snmpd** не согласовывает действия по маршрутизации с демоном **routed**. Если демон **gated** запущен, и таблица *ipRouteTable* зарегистрирована этим демоном для **snmpd**, то запросы set к *ipRouteTable* не допускаются.

Переменные, изменение которых поддерживается демоном **snmpd**, перечислены в RFC 1229. Исключения из этого правила описаны выше.

Ниже приведены примеры применения команды **snmpinfo**. Предполагается, что для имени группы взаимодействия **snmpinfo** по умолчанию (public) разрешены чтение и запись для соответствующего поддерева.

```
snmpinfo -m set sysContact.0="Primary contact: Bob Smith, office phone: 555-5555,
beeper: 9-123-4567. Secondary contact: John Harris, phone: 555-1234."
```

Эта команда присваивает переменной `sysContact.0` указанную строку. Если для `sysContact.0` уже существует запись, она заменяется новой.

```
snmpinfo -m set sysName.0="bears.austin.ibm.com"
```

Эта команда присваивает переменной `sysName.0` указанную строку. Если для `sysName.0` уже существует запись, она заменяется новой.

```
snmpinfo -m set sysLocation.0="Austin site, building 802, lab 3C-23, southeast
corner of the room."
```

Эта команда присваивает переменной `sysLocation.0` указанную строку. Если для `sysLocation.0` уже существует запись, она заменяется новой.

```
snmpinfo -m set ifAdminStatus.2=2
```

Эта команда отключает адаптер сетевого интерфейса, значение ifIndex которого равно 2. Если значение равно 1, адаптер интерфейса включается.

```
snmpinfo -m set atPhysAddress.2.1.192.100.154.2=02:60:8c:2e:c2:00
snmpinfo -m set ipNetToMediaPhysAddress.2.1.192.100.154.2=02:60:8c:2e:c2:00
```

Эти команды изменяют аппаратный адрес для записи `192.100.154.2` в таблице ARP на `02:60:8c:2e:c2:00`. Обе команды выполняют действия над одной и той же записью в таблице ARP. Переменная MIB *atPhysAddress* является устаревшей, и вместо не используется переменная MIB *ipNetToMediaPhysAddress*. Таким образом, переменные *atPhysAddress* и *ipNetToMediaPhysAddress* обращаются к одной структуре в таблице ARP ядра TCP/IP.

```
snmpinfo -m set atNetAddress.2.1.192.100.154.2=192.100.154.3
snmpinfo -m set ipNetToMediaNetAddress.2.1.192.100.154.2=192.100.154.3
```

Эти команды изменяют IP-адрес для записи `192.100.154.2` в таблице ARP на `192.100.154.3`. Обе команды выполняют действия над одной и той же записью в таблице ARP. Переменная MIB *atNetAddress* является устаревшей, и вместо нее используется переменная MIB *ipNetToMediaNetAddress*. Таким образом, переменные *atNetAddress* и *ipNetToMediaNetAddress* обращаются к одной структуре в таблице ARP ядра TCP/IP.

```
snmpinfo -m set ipForwarding.0=1
```

Эта команда позволяет настроить ядро **TCP/IP** для пересылки пакетов, если на хосте агента установлено несколько активных интерфейсов. Если на хосте агента установлен один активный интерфейс, то запрос set выполнен не будет, и программа-демон **snmpd** возвратит код ошибки *badValue*.

```
snmpinfo -m set ipDefaultTTL=50
```

Эта команда задает для IP-дейтаграмм ограничение на число участков (TTL) по умолчанию, равное 50. После прохождения через указанное число шлюзов, дейтаграмму уничтожаются. Каждый шлюз при обработке дейтаграммы уменьшает на 1 значение поля Максимальное число участков. Кроме того, прежде чем отправить дейтаграмму в следующую точку маршрута, каждый шлюз уменьшает значение поля Максимальное число участков на число секунд, которое дейтаграмма провела в ожидании обслуживания.

```
snmpinfo -m set ipRouteDest.192.100.154.0=192.100.154.5
```

Эта команда присваивает целевому IP-адресу маршрута, связанного с адресом `192.100.154.0`, значение `192.100.154.5`, при условии что маршрут уже `192.100.154` существует.

```
snmpinfo -m set ipRouteNextHop.192.100.154.1=129.35.38.47
```

Эта команда устанавливает маршрут к хосту 192.100.154.1 через шлюз 129.35.38.47 при условии, что маршрут 192.100.154.1 уже существует.

```
snmpinfo -m set ipRouteNextHop.192.100.154.0=192.100.154.7
```

Эта команда устанавливает маршрут к сети класса C 192.100.154 через шлюз 192.100.154.7 при условии, что маршрут 192.100.154.0 уже существует. Обратите внимание, что для обозначения адреса сети часть IP-адреса, определяющая хост, должна быть равна 0.

```
snmpinfo -m set ipRouteType.192.100.154.5=2
```

Эта команда удаляет все маршруты к хосту 192.100.154.5 .

```
snmpinfo -m set ipRouteDest.129.35.128.1=129.35.128.1
               ipRouteType.129.35.128.1=3
            ipRouteNextHop.129.35.128.1=129.35.128.90
```

Эта команда создает новый маршрут от хоста 129.35.128.90 к хосту 129.35.128.1 в качестве шлюза.

```
snmpinfo -m set ipNetToMediaType.2.1.192.100.154.11=4
```

Эта команда присваивает записи 192.100.154.11 в таблице ARP атрибут "статический".

```
snmpinfo -m set snmpEnableAuthenTraps=2
```

Эта команда запрещает агенту **snmpd** на заданном хосте вызывать прерывания *authenticationFailure*.

```
snmpinfo -m set smuxPstatus.1=2
```

Эта команда определяет узел SMUX 1 как неидентифицированный. При этом соединение между агентом **snmpd** и этим узлом SMUX разрывается.

```
snmpinfo -m set smuxTstatus.8.1.3.6.1.2.1.4.21.0=2
```

Эта команда отменяет монтирование дерева SMUX 1.3.6.1.2.1.4.21 в таблице *ipRoute* или определяет его как недопустимое. Первое число экземпляра показывает количество уровней для идентификатора дерева SMUX. Последнее число показывает значение smuxTpriority. В данном примере для идентификатора дерева SMUX существует 8 уровней: 1.3.6.1.2.1.4.21. Значение приоритета 0 означает наивысший приоритет.

```
snmpinfo -m set ifExtnsPromiscuous.1=1 ifExtnsPromiscuous.2=2
```

Эта команда включает смешанный режим для первого устройства и выключает смешанный режим для второго устройства в таблице интерфейсов.

```
snmpinfo -m set ifExtnsTestType.1=testFullDuplexLoopBack
```

Эта команда запускает тест testFullDuplexLoopBack для интерфейса 1.

```
snmpinfo -m set ifExtnsRcvAddrStatus.1.129.35.128.1.3.2=2
```

Эта команда удаляет физический адрес 129.35.128.1.3.2 из списка адресов интерфейса 1.

```
snmpinfo -m set dot5Commands.1=2
```

Вызов этой команды приводит к открытию первого интерфейса.

```
snmpinfo -m set dot5RingSpeed.1=2
```

Вызов этой команды устанавливает быстродействие сети Token-Ring первого интерфейса равным 1 Мбит/с.

```
snmpinfo -m set dot5ActMonParticipate.1=1
```

Вызов этой команды приводит к выбору первого интерфейса для работы с активным монитором.

```
snmpinfo -m set dot5Functional.1=255.255.255.255.255.255
```

Эта команда устанавливает для маски функционального адреса значение "разрешить все".

```
snmpinfo -m set fddimibSMTUserData.1="Greg's Data"
```

Эта команда записывает в поле пользовательских данных первого экземпляра SMT значение "Greg's Data".

```
snmpinfo -m set fddimibMACFrameErrorThreshold.1.1=345
```

Эта команда устанавливает пороговое число ошибок кадров для первого MAC первого объекта SMT равным 345.

Примечание: Всем перечисленным выше переменным можно присвоить значение одним из описанных ранее способов.

Дополнительная информация о протоколах и IP-адресах приведена в разделе “Протокол преобразования адресов” на стр. 142 и “Адреса Internet” на стр. 168.

Устранение неполадок SNMP

Советы по устранению неполадок для демона **SNMP**, включая сведения об устранении неполадок завершения работы, неполадки доступа к переменным MIB, неполадки доступа к переменным MIB записи взаимодействия, неполадки noSuchName, неполадки отсутствия ответа от агентов и сбои демона.

Неполадки при завершении работы демона

Ниже приведены рекомендации по локализации и устранению неполадок в работе агента **snmpd**. Настоятельно рекомендуется разрешить заносить в протокол некоторые сообщения агента **snmpd**. Если ошибка возникает при вызове демона **snmpd**, настоятельно рекомендуется разрешить демону **syslogd** заносить в протокол сообщения демонов с уровнем серьезности DEBUG. Дополнительная информация о создании протоколов агента **snmpd** содержится в описании команды **snmpd** в *Справочник по командам, том 5* и в файле **snmpd.conf** в книге *Справочник по файлам*.

Ниже приведены возможные причины завершения работы демона **snmpd** сразу после его запуска, а также перечислены способы устранения этой неполадки:

- Причина завершения работы демона **snmpd** будет занесена в файл протокола демона **snmpd** или в указанный файл протокола демона **syslogd**. Найдите в файле протокола сообщения об ошибках **FATAL**.

  Исправление: Устраните неполадку и повторите запуск демона **snmpd**.
- Аргументы командной строки демона **snmpd** указаны неправильно. Если команда **snmpd** была вызвана без помощи Контроллера системных ресурсов (SRC), то описание формата команды будет выведено на экран. В противном случае описание формата вызова **snmpd** не будет показано на экране. Вместо этого оно будет записано в файл протокола.

  Исправление: Вызовите команду **snmpd** в правильном формате.
- Демон **snmpd** разрешено запускать только пользователю root.

  Исправление: Войдите в систему как пользователь root и повторите запуск демона **snmpd**.
- Владельцем файла **snmpd.conf** должен быть пользователь root. Агент **snmpd** проверяет, кто является владельцем файла конфигурации. Если владелец файла отличен от root, то такая ошибка считается неисправимой, и агент **snmpd** завершает свою работу.

  Исправление: Убедитесь, что вы являетесь пользователем root, измените владельца файла конфигурации на root и повторите запуск программы-демона **snmpd**.
- Не найден файл **snmpd.conf**. Если в параметрах команды **snmpd** не указан флаг **-c**, то не найден файл **/etc/snmpd.conf**. Если файл **/etc/snmpd.conf** был случайно удален, повторно установите образ **bos.net.tcp.client** или восстановите файл **snmpd.conf** по описанию его записей. Если с флагом **-c** в командной строке **snmpd** был указан альтернативный файл конфигурации, убедитесь в том, что он принадлежит пользователю root. Имя файла конфигурации должно быть указано полностью, иначе будет применяться файл по умолчанию, **/etc/snmpd.conf**.

Исправление: Убедитесь, что указанный файл конфигурации существует и его владельцем является пользователь root. Повторите запуск демона **snmpd**.

- Порт `udp 161` уже занят. Убедитесь, что программа-демон **snmpd** еще не запущена. Для этого вызовите команду **ps -eaf | grep snmpd**. С **портом udp 161** может работать только один агент `snmpd`.

  Исправление: Либо удалите существующий агент **snmpd**, либо не запускайте еще один процесс **snmpd**.

Неполадки, связанные со сбоем демона

Ниже перечислены возможные причины и способы устранения ошибок, связанных со сбоями демона **snmpd**, возникающими при вызове команды **refresh** или при отправке сигнала **kill -1**:

- Причина завершения работы демона **snmpd** записывается в файл протокола демона **snmpd** или в указанный файл протокола демона **syslogd**. Найдите в файле протокола сообщения об ошибках FATAL.

  Исправление: Устраните неполадку и повторите запуск демона **snmpd**.

- Убедитесь в том, что при вызове демона **snmpd** было указано полное имя файла конфигурации. При запуске демон **snmpd** создает процесс-потомок и изменяет свой текущий каталог на корневой. Если полный путь к файлу конфигурации не задан, то агент **snmpd** не сможет найти этот файл после обновления. Эта ошибка не может быть исправлена и вызывает завершение работы агента **snmpd**.

  Исправление: Укажите полное имя файла конфигурации **snmpd**. Убедитесь, что владельцем этого файла является пользователь root. Повторите запуск демона **snmpd**.

- Убедитесь в том, что файл конфигурации **snmpd** существует. Возможно, файл был случайно удален после запуска агента **snmpd**. Если агенту **snmpd** не удается открыть файл, то его работа завершается.

  Исправление: Создайте файл конфигурации **snmpd**, проверьте, что его владельцем является пользователь root, и повторите запуск демона **snmpd**.

Неполадка доступа к переменной MIB

Если агенту **snmpd** не удается получить доступ к переменным Базы информации управления (MIB), или если при работе **snmpd** в **Диспетчере простого протокола управления сетью (SNMP) snmpd**, выполните следующие действия:

- С помощью команды **netstat -in** проверьте правильность конфигурации сети, заданной на хосте агента **snmpd**. Убедитесь в том, что устройство lo0 работает. Если устройство отключено, слева от имени lo0 будет показан символ "*" (звездочка). Для того чтобы агент **snmpd** мог обслуживать запросы, lo0 должно быть активно.

  Исправление: Для активации циклического интерфейса введите следующую команду:

  ```
  ifconfig lo0 inet up
  ```

- Убедитесь в том, что для демона **snmpd** определен маршрут к хосту, отправившему запрос.

  Исправление: В системе демона **snmpd** добавьте маршрут к хосту, вызвавшему команду **route add**. Дополнительная информация приведена в описании команды **route** в *Справочник по командам, том 4*.

- Проверьте правильность имени и IP-адреса хоста.

  Исправление: Измените имя хоста, чтобы оно соответствовало его IP-адресу.

- Убедитесь в том, что значение *localhost* совпадает с IP-адресом lo0.

  Исправление: Укажите для хоста *localhost* IP-адрес, совпадающий с адресом lo0 (как правило, `127.0.0.1`).

Неполадки доступа к переменной MIB в записи группы взаимодействия

Если запись группы взаимодействия в файле конфигурации задана с помощью имени представления MIB, но вы не можете обратиться к переменным MIB, выполните следующие действия:

- Проверьте правильность записи связи. Если запись группы взаимодействия содержит имя представления, то необходимо задать значения всех полей группы взаимодействия.

  Исправление: Задайте значения для всех полей группы взаимодействия в файле конфигурации. Обновите агент **snmpd** и повторите запрос.

- Убедитесь, что режим доступа, указанный в записи связи, соответствует типу запроса. При отправке запроса **get** или **get-next** убедитесь, что для данной группы взаимодействия разрешены операции только чтение или чтение-запись. При отправке запроса **set** убедитесь, что данной связи предоставлены права на чтение и запись.

  Исправление: В записи группы взаимодействия укажите правильное значение режима доступа. Обновите агент **snmpd** и повторите запрос.
- Убедитесь в том, для имени представления, заданного в записи связи в файле конфигурации, указана запись представления. Если запись группы взаимодействия содержит имя представления, для которого не существует соответствующей записи представления, то агент **snmpd** не разрешит доступ к этой группе взаимодействия. Для имени представления, заданного в записи группы взаимодействия в файле конфигурации, необходимо наличие записи представления.

  Исправление: Добавьте запись для представления, указанного записи группы взаимодействия. Обновите агент **snmpd** и повторите запрос.
- Если в качестве поддерева MIB для записи представления указано значение `iso`, убедитесь в том, что задано именно значение `iso.3`. Экземпляр 3 необходим агенту **snmpd** для доступа к фрагменту `org` дерева `iso`.

  Исправление: Укажите в записи представления значение `iso.3` в качестве поддерева MIB. Обновите агент **snmpd** и повторите запрос.
- Проверьте правильность *IP-адреса* и *маски подсети*, указанных в записи связи. Убедитесь в том, что хост, отправивший запрос SNMP, входит в указанную группу взаимодействия.

  Исправление: Измените значения полей *IP-адрес* и *маска подсети* в записи группы взаимодействия в файле конфигурации таким образом, чтобы эта группа взаимодействия включала хост, с которого отправляется запрос SNMP.

Отсутствие ответа от агента

Если в группе взаимодействия указан *IP-адрес* 0.0.0.0, но ответ от агента **snmpd** отсутствует, выполните следующие действия:

- Проверьте правильность *маски подсети*, указанной в записи связи. Для того чтобы группа взаимодействия была общедоступной, укажите *маску подсети* **0.0.0.0**. Если в *маске подсети* указано значение `255.255.255.255`, агент **snmpd** не будет принимать никакие запросы с данной группой взаимодействия.

  Исправление: Укажите *маску подсети* в записи группы взаимодействия 0.0.0.0. Обновите агент **snmpd** и повторите запрос.
- Убедитесь, что режим доступа, указанный в записи связи, соответствует типу запроса. При отправке запроса**get** или **get-next** убедитесь, что данной группе взаимодействия предоставлены права только на чтение или на чтение и запись. При отправке запроса **set** убедитесь, что данной связи предоставлены права на чтение и запись.

  Исправление: В записи группы взаимодействия укажите правильное значение режима доступа. Обновите агент **snmpd** и повторите запрос.

Ошибка noSuchName

При попытке присвоить значение переменной MIB, которая должна поддерживаться агентом **snmpd**, может быть получено сообщение об ошибке `noSuchName`. Ниже перечислены возможные причины этой ошибки.

В отправленном запросе set не указано имя группы взаимодействия с правами доступа на запись. Согласно протоколу **SNMP**, в ответ на запрос set, в котором указана группа взаимодействия с неправильными правами доступа, отправляется сообщение об ошибке `noSuchName`.

Исправление: Укажите в запросе set имя группы взаимодействия, у которой есть права на запись и которая содержит хост, с которого отправлен запрос set.

Сетевая файловая система (NFS)

Сетевая файловая система (NFS) обеспечивает хранение файлов в сети. Сетевая файловая система (NFS) - это распределенная файловая система. NFS обеспечивает пользователям доступ к файлам, расположенным на удаленных компьютерах, и позволяет работать с этими файлами точно так же, как и с локальными.

Например, с помощью команд операционной системы пользователи могут создавать, удалять, считывать, записывать и изменять атрибуты удаленных файлов и каталогов.

Пакет NFS содержит наборы команд и программ-демонов, предназначенных для работы с NFS, Службой информации о сети (NIS) и другими службами. Несмотря на то, что NFS и NIS устанавливаются совместно, как один пакет, они независимы друг от друга, и их настройка и администрирование выполняются раздельно.

AIX 5.3 и более поздние версии поддерживают NFS версии 2, 3 и 4. NFS версии 4 является последней версией NFS. Она описана в RFC 3530. Дополнительные сведения о поддержке NFS версии 4 в AIX приведены ниже в этом разделе. Клиенты NFS по умолчанию используют протокол NFS версии 3.

Службы NFS

Службы NFS реализованы по принципу клиент-сервер.

Компьютеры, предоставляющие свои *файловые системы*, *каталоги* и другие ресурсы для удаленного доступа называются *серверами*. Сам процесс предоставления доступа к файловым системам называется *экспортом*. Компьютеры и процессы, работающие с ресурсами сервера, называются *клиентами*. Смонтировав файловую систему, экспортированную сервером, клиент получает доступ к отдельным файлам сервера (доступ к экспортированным каталогам для отдельных клиентов может быть ограничен).

NFS выполняет следующие основные функции:

Таблица 88. Службы NFS

Служба	Описание
Монтирование	Монтирует из демона **/usr/sbin/rpc.mountd** на сервере и из команды **/usr/sbin/mount** на клиенте. Эта служба доступна только в NFS версии 2 и 3.
Удаленный доступ к файлам	Доступ из демона **/usr/sbin/nfsd** на сервере и из команды **/usr/sbin/biod** на клиенте.
Поддержка удаленного выполнения	Исполняется из демона **/usr/sbin/rpc.rexd** на сервере и команды **/usr/bin/on** на клиенте.
Удаленный доступ к статистическим данным о системе	Компилируется из демона **/usr/sbin/rpc.rstatd** на сервере и команды **/usr/bin/rup** на клиенте.
Создание списка удаленных пользователей	Создает список из демона **/usr/lib/netsvc/rusers/rpc.rusersd** на сервере и команды **/usr/bin/rusers** на клиенте.
Предоставление параметров загрузки	Демон **/usr/sbin/rpc.bootparamd**, работающий на сервере, передает параметры загрузки бездисковым клиентам с операционной системой Sun.
Удаленная служба Wall	Защищает из демона **/usr/lib/netsvc/rwall/rpc.rwalld** на сервере и команды **/usr/sbin/rwall** на клиенте.
Служба рассылки	Отправляет односторонний поток пакетов Вызова удаленных процедур (RPC) с помощью демона **/usr/lib/netsvc/spray/rpc.sprayd** на сервере и команды **/usr/sbin/spray** на клиенте.
Служба идентификации PC	Выполняет идентификацию пользователей для службы PC-NFS с помощью демона **/usr/sbin/rpc.pcnfsd** на сервере.
Расширенная служба защиты	Предоставляет серверу и клиенту дополнительные возможности по обеспечению защиты, например Kerberos 5. Демон **/usr/sbin/gssd** обеспечивает NFS доступ к службам защиты, предлагаемым Службами сетевой идентификации. Должны быть установлены службы сетевой идентификации и наборы файлов криптографической библиотеки (`krb5.client.rte`, **krb5.server.rte** и **modcrypt.base**). Эти наборы файлов можно установить из пакета расширения для AIX.

Таблица 88. Службы NFS (продолжение)

Служба	Описание
Служба преобразования идентификаторов	Выполняет преобразование между субъектами защиты, строками идентификаторов NFS версии 4 и их соответствующими числовыми ИД. Кроме того, предлагается преобразование информации идентификаторов из внешних доменов NFS версии 4. Этими службами управляет демон **/usr/sbin/nfsrgyd**.

Примечание: Компьютер может быть одновременно и сервером, и клиентом NFS.

Серверы NFS версии 2 и 3 работают *без сохранения состояния*, что означает, что они не сохраняют информацию о транзакциях своих клиентов. Под транзакцией NFS понимается выполнение одной операции над файлом. Все данные, необходимые для дальнейшей работы, должны храниться клиентом NFS.

Сервер NFS версии 4 работают с сохранением состояния благодаря указанным в протоколе NFS версии 4 операциям открытия файла и блокирования файла.

Поддержка списков управления доступом NFS

Реализация NFS версии 4 в AIX поддерживает два типа ACL: NFS4 и AIXC.

Права доступа проверяются базовой файловой системой, экспортированной сервером NFS. Файловая система принимает во внимание списки управления доступом (ACL или биты прав доступа) файла, разрешения вызывающего пользователя и другие возможные ограничения локальной системы. Приложения и пользователи должны учитывать, что решение о предоставлении доступа не принимается на основе только битов режима UNIX или только ACL.

Командами **aclget**, **aclput** и**acledit** можно пользоваться в клиентской системе для управления NFS или AIX ACL. Дополнительная информация приведена в разделе Access Control Lists книги *Защита*.

NFS RBAC

NFS поддерживает ролевое управление доступом (RBAC). Команды клиента и сервера NFS поддерживают RBAC.

Такой подход позволяет обычным пользователям выполнять команды NFS, если им присвоена роль команды RBAC. Для просмотра списка прав доступа, связанных с командами NFS, обратитесь к файлу /etc/security/privcmds.

NFS4 ACL

NFS4 ACL - это ACL, задаваемые протоколом NFS версии 4.

NFS4 ACL не зависят от платформы и могут поддерживаться серверами и клиентами других производителей. Для поддержки NFS4 ACL не требуются серверы и клиенты NFS версии 4.

Если на сервере AIX экземпляр базовой физической файловой системы поддерживает NFS4 ACL, то сервер AIX NFS4 поддерживает NFS4 ACL для этого экземпляра файловой системы. Большинство типов физических файловых систем AIX не поддерживают NFS4 ACL. К таким файловым системам относятся CFS, UDF, JFS, JFS2 с расширенным атрибутом версии 1 и другие. Все экземпляры JFS2 с расширенным атрибутом версии 2 поддерживают NFS4 ACL.

Файловые системы клиентов NFS версии 4 могут читать и записывать NFS4 ACL, если экспортированный экземпляр файловой системы NFS версии 4 на сервере поддерживает NFS4 ACL.

AIX ACL

AIXC ACL является списком управления доступом, принадлежащим серверам AIX.

Они не определяются протоколом NFS версии 4 и распознаются только серверами и клиентами AIX.

На сервере NFS версии 4 AIXC ACL поддерживаются, когда экземпляр базовой файловой системы поддерживает AIXC ACL. Все экземпляры JFS и JFS2 поддерживают AIXC ACL.

Все клиенты NFS версии 4 предусматривают опцию монтирования, которая включает или выключает поддержку AIX ACL. По умолчанию AIXC ACL не поддерживается. Пользователь клиентской файловой системы NFS версии 4 может читать и записывать AIXC ACL, когда и на клиенте, и на сервере запущен AIX, базовая физическая файловая система на сервере поддерживает AIXC ACL, а клиент AIX смонтировал экземпляр файловой системы с включенной поддержкой AIXC ACL. Поддержка AIXC ACL в NFS версии 4 реализована так же, как и поддержка AIXC ACL в NFS версий 2 и 3 в AIX.

Все экземпляры файловой системы JFS2 с расширенным атрибутом версии 2 поддерживают AIXC ACL и NFS4 ACL. Файл в такой файловой системе может находиться в следующих режимах: только биты (без ACL), NFS4 ACL или AIXC ACL. Однако NFS4 ACL и AIXC ACL не могут применяться одновременно в файле.

С помощью команды **aclgettypes** можно определить типы ACL, доступные для чтения и записи в экземпляре файловой системы. Результаты выполнения этой команды могут различаться при ее запуске из физической файловой системы на сервере NFS версии 4 локально или при запуске из такой же файловой системы на клиенте NFS версии 4. Например, экземпляр файловой системы NFS версии 4 на сервере NFS версии 4 поддерживает NFS4 ACL и AIXC ACL, но клиент настроен только для отправки и приема NFS4 ACL. В этом случае при запуске команды **aclgettypes** из файловой системы клиента NFS версии 4 выдаются только результаты NFS4. Если пользователь клиента запросит AIXC ACL, будет выдано сообщение об ошибке.

Поддержка кэширующей файловой системы

Кэширующая файловая система (CacheFS) представляет собой механизм кэширования общего назначения, увеличивающий производительность и масштабируемость NFS за счет уменьшения нагрузки на сервер и сеть.

Обладая многослойной структурой, CacheFS дает возможность кэшировать одну файловую систему в другой. Использование CacheFS в среде NFS увеличивает соотношение количества клиентов к количеству серверов, уменьшает нагрузку на сервер и сеть, и увеличивает производительность клиентов, соединенных через медленные каналы связи, например по двухточечному протоколу (PPP).

Создание кэша в системе клиента ускоряет доступ к смонтированным файловым системам, поскольку для получения доступа к данным не нужно отправлять запрос по сети. Данные помещаются в кэш при первом обращении пользователя к этим данным. До тех пор, пока пользователь не запросит какие-либо файлы с сервера, кэш будет оставаться пустым. Первое обращение к файлу может занять много времени, но последующие запросы к тем же файлам будут выполняться значительно быстрее.

Примечание:

1. Кэширование файловых систем / (корневая) и `/usr` невозможно.
2. Монтировать можно только общие файловые системы. (См. команду **exportfs** в *Справочник по командам, том 2*.)
3. Кэширование локальной дисковой журналируемой файловой системы (JFS) не дает никакого выигрыша в производительности.
4. Для выполнения задач, перечисленных в приведенной ниже таблице, необходимы системные права доступа или права доступа root.

Таблица 89. Задачи CacheFS

Процедура	Команды быстрого доступа SMIT	Команда или файл
Создать кэш	`cachefs_admin_create`	**cfsadmin -c** *каталог-монтирования*[1].
Указать файлы, предназначенные для монтирования	`cachefs_mount`	**mount -F cachefs -o backfstype=***тип-файловой-системы*, **cachedir=***каталог-кэша*[,*options*] резервная-файловая-система *каталог-монтирования*[2] или измените файл `/etc/`**filesystems**.
Изменить кэш	`cachefs_admin_change`	удалите кэш, а затем снова создайте его командой **mount** с нужными параметрами.
Показать данные о кэшировании	`cachefs_admin_change`	**cfsadmin -l***каталог-монтирования*
Удалить кэш	`cachefs_admin_remove`	1. Размонтировать файловую систему: **umount** *каталог-монтирования* 2. Определить ИД кэша: **cfsadmin -l** *каталог-монтирования* 3. Удалить файловую систему: **cfsadmin -d** *ИД-кэша каталог-кэша*
Проверить целостность файловой системы	`cachefs_admin_check`	**fsck_cachefs***каталог-кэша*[3].

Notes:

1. После того как кэш будет создан, выполнять какие-либо операции непосредственно в самом каталоге кэша (`cachedir`) не рекомендуется. Это может вызвать конфликты с программным обеспечением CacheFS.
2. Если монтируемые каталоги задаются в команде **mount**, то эту команду нужно заново выполнять после каждой перезагрузки системы.
3. Для того чтобы только проверить целостность файловых систем (не внося при этом исправлений), используйте опции **-m** или **-o** команды **fsck_cachefs**.
4. После миграции прежних версий на версию AIX 6.1 или более позднюю необходимо удалить и снова создать все старые файловые системы кэша.

Поддержка отображений файлов NFS

Функция отображения файлов NFS позволяет программам в клиентской системе работать с файлом так, как будто он загружен в оперативную память.

Клиенты могут отображать фрагменты файла в адресное пространство с помощью функции **shmat**. В момент, когда программа считывает и записывает данные в эту области памяти, происходит считывание данного файла с сервера и запись его в память, либо необходимое обновление файла на сервере.

Отображение файлов в системе NFS имеет следующие три недостатка:

- Оно не обеспечивает в достаточной степени возможности совместного использования данных клиентами.
- Изменения, внесенные в файл на одном клиенте, не будут видны остальным клиентам.
- Поочередная блокировка и разблокирование областей файлов не обеспечивает эффективного согласования данных между клиентами.

Если файл NFS будет применяться несколькими программами, работающими в разных клиентских системах, то необходимо использовать функцию блокирования записей в сочетании с обычными функциями **read** и **write**.

Использование отображения файлов несколькими программами на одном клиенте обеспечивает повышение производительности. Согласовать обновления файла на клиенте можно путем поочередного блокирования всего файла. Несколько клиентов могут совместно использовать только такие отображения файлов, данные которых никогда не изменяются (например, статические базы данных).

Обслуживание посредника NFS

AIX поддерживает обслуживание посредника сетевой файловой системы (NFS). Сервер AIX может одновременно выполнять экспорт локально-доступных файловых систем и экспорт посредника. Экспортированный посредник может быть смонтирован клиентами NFS.

Посредник AIX NFS сохраняет в кэше диска данные для последующего локального обслуживания схожих запросов для снижения объема потока данных, передаваемых на базовый сервер. Применение посредника может существенно расширить доступ к данным NFS по низкоскоростным или менее надежным сетям и повысить производительность и снизить объем передаваемых данных к основному серверу, на котором хранятся данные. В зависимости от доступности и требований к управлению информацией применение посредника может помочь расширить доступ NFS к сетевым граням без необходимости копировать данные. Настроить обслуживание посредника AIX NFS можно с помощью команды **mknfsproxy**.

Кэширование посредника можно использовать с протоколами NFS версии 3 и 4. Между посредником и подключенными клиентами может использоваться протокол NFS версии 3 или версии 4, если между посредником и базовым сервером используется протокол NFS версии 4. Тем не менее, если используется протокол NFS версии 3, то между посредником и подключенными клиентами должен использоваться протокол NFS версии 3. Помимо блокировки диапазона байтов поддерживаются также операции чтения и записи данных.

Между сервером proxy и подключенными клиентами могут использоваться методы защиты krb5, krb5i и krb5p. Эти же методы можно использовать между сервером proxy и главным сервером. Используя технологию передачи паспортов через proxy, можно зарегистрироваться в системе клиента и быть зарегистрированным на главном сервере. Для того, чтобы воспользоваться преимуществами этой технологии, при выполнении процедуры предоставления прав доступа Kerberos введите команду **kinit** с опцией **-f**. Если между proxy и базовым сервером действует защита **auth_sys**, то при входе на базовый сервер, сервер proxy преобразовывает права доступа клиента Kerberos в атрибуты **auth_sys**. Для достижения лучших результатов сервер proxy и базовый сервер должны иметь общие определения пользователя и группы.

К обслуживанию посредника NFS применяются следующие ограничения:

- Клиенты должны быть подключены по протоколу TCP.
- Поскольку обслуживание proxy позволяет клиентам NFS версии 3 перемещаться в экспортированном пространстве имен NFS версии 4, не прибегая к командам **mount** и **unmount**, то при создании файловой системы proxy следует воспользоваться командой **mknfsproxy** с опцией **mfsid**.
- Файловая система кэширования, используемая с сервером proxy, должна являться расширенной файловой системой JFS (JFS2).
- Сервер proxy выполняет CacheFS поверх клиента AIX, смонтированного на базовом сервере NFS. Функция параллельного ввода-вывода (CIO) доступна для клиента AIX NFS, она расширяет производительность CacheFS. Попытки напрямую обратиться к основному смонтированному клиенту NFS потерпят крах из-за конфликта с попытками открыть CIO.

Типы монтирования NFS

В NFS существует три типа монтирования: предопределенное, ручное и автоматическое.

Предопределенные каталоги, монтируемые по умолчанию, перечислены в файле `/etc/filesystems`. В этом файле каждый раздел (или отдельная запись) задает монтируемый каталог. Такой раздел содержит данные об имени хоста, пути к удаленному каталогу, локальном каталоге и всех остальных параметрах монтирования. Предопределенное монтирование удобно в тех случаях, когда для работы клиента всегда требуются одни и те же каталоги.

Монтирование вручную обычно выполняется пользователем root. Вручную смонтированные каталоги обычно устанавливают на небольшой срок, когда возникает необходимость в каких-либо ранее неучтенных файловых системах. Кроме того, каталоги могут монтироваться в ручную для выполнения разовых операций. Все параметры монтирования обычно задают в командной строке команды **mount**. При

монтировании вручную изменение файла /etc/filesystems не требуется. Каталоги, смонтированные вручную, размонтировать можно также только вручную командой **umount**. Неразмонтированные каталоги сохраняются до перезагрузки системы.

Для управления *автоматическим* монтированием служит команда **automount**. Расширение ядра **AutoFS** отслеживает обращения к каталогам, указанным в этой команде. Если программа или пользователь пытаются обратиться к каталогу, который еще не смонтирован, **AutoFS** перехватывает запрос и монтирует необходимую файловую систему.

Экспорт и монтирование NFS

Для успешного администрирования NFS необходимо понимать, для чего предназначены процедуры экспортирования и монтирования каталогов.

Сервер NFS должен экспортировать файл или каталог, после чего клиент NFS может смонтировать этой файл или каталог. В этом разделе приведены более подробные сведения об этих понятиях.

Экспорт каталогов NFS

Экспортирование каталога осуществляется на сервере NFS. Экспортирование каталога означает, что каталог в пространстве имен сервера становится доступным для клиентов.

Экспортированный каталог называется *экспорт*. Он содержит все файлы, расположенные в файловой системе экспортированного каталога.

Каждый экспорт также задает права доступа. Например, могут применяться следующие ограничения:

- какие клиенты могут обращаться к экспортированному каталогу
- с помощью каких версий NFS клиенты должны обращаться к этому каталогу
- может ли клиент сохранять файлы в экспорте
- какими средствами защиты должен пользоваться клиент для обращения к каталогам и файлам экспорта

Полное описание ограничений экспорта и используемых форматов приведено в описании команды **exportfs** в книге *Справочник по командам, том 2* и в описании файла /etc/exports в книге *Справочник по файлам* .

Примечание: После изменения атрибутов экспорта следует повторно экспортировать каталог, чтобы изменения вступили в силу. Кроме того, повторно экспортировать каталог может потребоваться при появлении изменений в других файлах или изменений, внешних по отношению к серверу. Например, если имя клиента, указанное в списке доступа, является сетевой группой, указанной в файле /etc/netgroup, то при изменении определения клиентской группы все экспорты, использующие эту группу в списке доступа, следует экспортировать повторно, чтобы изменения вступили в силу.

Аналогично, если IP-адрес клиента изменился, то все экспорты, у которых в списке доступа указан этот клиент, следует повторно экспортировать. Причина в том, что сервер NFS сохраняет кэш с правами доступа клиентов в каждом экспорте. Кэш выгружается при каждой операции экспортирования или отмены экспортирования. Если права доступа экспорта изменяются, например, если меняется IP-адрес клиента или клиент удаляется из списка доступа, то нужно выполнить повторное экспортирование или отмену экспортирования, чтобы права доступа клиента были правильно указаны в кэше. Сервер NFS запускает демон **rpc.mountd** для получения информации о правах доступа каждого клиента, поэтому демон **rpc.mountd** должен быть запущен на сервере, даже если сервер экспортирует файловые системы только для доступа к NFS версии 4.

Монтирование каталога NFS

Клиент NFS может смонтировать каталог, экспортированный сервером NFS. В результате монтирования каталога файлы, находящиеся на сервере NFS, становятся доступными клиенту NFS.

Клиент может обращаться к файлам на сервере, если файлы были экспортированы сервером и обращение допускается экспортными ограничениями. После того как клиент успешно смонтировал экспорт сервера в

точке монтирования в своем пространстве имен, файлы сервера для этого экспорта будут находиться в пространстве имен клиента и выглядеть как файлы локальной файловой системы.

Например, предположим, что вы хотите экспортировать каталог /tmp с сервера diamond и смонтировать его на клиенте clip в виде каталога /mnt. На сервере введите следующую команду:

```
exportfs -i -o access=clip /tmp
```

В результате каталог /tmp стал доступен клиенту.

На клиенте введите следующую команду:

```
mount diamond:/tmp /mnt
```

Каталоги и файлы каталога /tmp на сервере теперь появились в клиентской системе под именем каталога /mnt.

Примечание:

1. Монтирование в NFS версий 2 и 3 и в NFS версии 4 выполняется не совсем одинаково. В NFS версий 2 и 3 сервер экспортирует каталоги, которые нужно сделать доступными для монтирования. В NFS версий 2 или 3 клиенту затем нужно явно смонтировать каждый экспорт для получения доступа к нему.

   В NFS версии 4 сервер так же указывает параметры управления экспортом каждого каталога, подлежащего экспортированию. С помощью этих элементов управления экспортом сервер создает единое дерево всех экспортируемых данных. Это дерево называется псевдофайловой системой, начало которой находится в псевдокорневом каталоге сервера NFS версии 4. Псевдофайловая система NFS версии 4 позволяет клиенту NFS версии 4, в зависимости от его реализации, обойтись лишь одним монтированием псевдокорневого каталога сервера, чтобы получить доступ ко всем данным. Клиенты AIX NFS поддерживают эту функцию. Реальное содержимое, которое доступно клиенту, зависит от элементов сервера, управляющих экспортом.
2. NFS версии 4 не поддерживает монтирование файлов по отдельности.

Монтирование NFS

Для работы с хранящимися на сервере файлами клиенты должны смонтировать каталоги, экспортированные сервером. Причем монтирование - это не копирование каталога. Процедура монтирования представляет собой серию вызовов удаленных процедур, обеспечивающих клиентам прямой доступ к каталогам сервера.

Ниже описана процедура монтирования :

1. При запуске сервера сценарий /etc/rc.nfs вызывает команду **exportfs**, ко торая считывает файл /etc/exports и передает ядру информацию о том, какие каталоги с какими правами доступа необходимо экспортировать.
2. В этом случае запускается демон **rpc.mountd** и несколько демонов **nfsd** с помощью сценария /etc/rc.nfs.
3. После этого сценарий /etc/rc.nfs выполняет команду **mount**, которая считывает список файловых систем, находящийся в файле /etc/filesystems.
4. Команда **mount** находит сервер, экспортировавший данные, которые были запрошены клиентом, и устанавливает с ним связь. Этот процесс называется *связыванием*.
5. После этого команда **mount** запрашивает у одного или нескольких серверов разрешение на доступ каталогам, указанным в файле /etc/filesystems на клиенте.
6. Демон на сервере получает от клиентов запросы на монтирование и либо разрешает выполнение операции, либо отказывает в доступе. Если клиенту разрешен доступ к запрошенному каталогу, демон передает ядру клиента идентификатор, называемый *описателем файла*.
7. Затем ядро клиента связывает описатель файла с точкой монтирования (каталогом), занося соответствующие данные в *запись монтирования*.

Связь клиента с демоном **rpc.mountd** не устанавливается при выполнении монтирования в NFS версии 4. Операции клиента по монтированию обслуживаются операциями ядра протокола NFS версии 4. Однако реализация сервера NFS версии 4 использует поддержку демона **rpc.mountd** для управления доступом в NFS версии 4.

Файл /etc/exports

В файле `/etc/exports` перечислены все каталоги, экспортируемые с сервера.

Одна строка задает один каталог. В файле `/etc/exports` один каталог может быть указан дважды: один раз для NFS версии 2 или NFS версии 3 и один раз - для NFS версии 4. Сервер автоматически экспортирует указанные каталоги каждый раз при запуске сервера NFS. Затем клиенты могут смонтировать экспортированные каталоги. Синтаксис строк файла `/etc/exports` следующий:

каталог `-`*опция*`[,`*опция*`]`

Параметр *каталог* - это полный путь к каталогу. В опциях может быть задан флаг, например, **ro**, или список имен хостов. Для ознакомления с полным перечнем опций и с их описаниями обратитесь к специальной документации по файлу `/etc/exports` в книге *Справочник по файлам* и команде **exportfs** в *Справочник по командам, том 2*. Если файл `/etc/exports` отсутствует, то сценарий `/etc/rc.nfs` не запустит демоны **nfsd** и **rpc.mountd**.

Ниже приведен пример записей файла `/etc/exports`:

```
/usr/games    -ro,access=ballet:jazz:tap
/home         -root=ballet,access=ballet
/var/tmp
/usr/lib      -access=clients
/accounts/database -vers=4,sec=krb5,access=accmachines,root=accmachine1
/tmp          -vers=3,ro
/tmp          -vers=4,sec=krb5,access=accmachines,root=accmachine1
```

Первая запись означает, что каталог `/usr/games` могут монтировать клиенты `ballet`, `jazz` и `tap`. Эти клиенты могут считывать данные и запускать программы из данного каталога, но не могут выполнять запись данных.

Вторая запись означает, что каталог `/home` можно смонтировать в системе `ballet`, и что к ней разрешен доступ с правами root.

Третья запись примера означает, что каталог `/var/tmp` может быть смонтирован любым клиентом. (Обратите внимание на отсутствие списка доступа.)

Четвертая запись - это список доступа для сетевой группы `clients`. Иначе говоря, компьютеры, объединенные в сетевую группу `clients`, могут монтировать с сервера каталог `/usr/lib`. (*Сетевая группа* это группа сетевых компьютеров, обладающих одинаковыми правами доступа к определенным ресурсам сети. Группа создается в организационных целях или для обеспечения необходимой защиты. Сетевыми группами можно управлять с помощью NIS.

Пятая запись разрешает доступ к каталогу `/accounts/database` только для клиентов из сетевой группы `accmachines`, применяющих NFS версии 4 и имеющей доступ к каталогу посредством идентификации Kerberos 5. Доступ к корневому каталогу разрешен только из `accmachine1`.

Шестая и седьмая записи экспортируют каталог `/tmp` с помощью различных версий и опций. Если в файле `/etc/exports` имеется две записи одного и того же каталога, отличающиеся только версиями NFS, то с помощью команды **exportfs** будет выполнен экспорт в соответствии с обеими записями. Если для одного каталога заданы одинаковые опции для NFS версии 4 и NFS версии 3, то в файл `/etc/exports` можно внести одну запись с указанием `-vers=3:4`.

Файл /etc/xtab

Формат файла `/etc/xtab` схож с форматом файла `/etc/exports`, в нем перечислены текущие экспортированные каталоги.

Файл `/etc/xtab` изменяется при каждом вызове команды **exportfs**. Это позволяет временно экспортировать каталог, не внося изменений в файл `/etc/exports`. При отмене экспорта временно экспортированного каталога его имя удаляется из файла `/etc/xtab`.

Примечание: Файл `/etc/xtab` обновляется автоматически. Его не нужно изменять вручную.

Файл /etc/nfs/hostkey

Этот файл используется сервером NFS для указания субъектов хоста Kerberos и расположения файла `keytab`.

Информация о настройке и использовании этого файла содержится в описании команды **nfshostkey** в книге *Справочник по командам, том 4*.

Файл /etc/nfs/local_domain

В этом файле содержится локальный домен NFS системы.

Подразумевается, что системы, использующие общий локальный домен NFS, используют и общие реестры пользователей и групп. Информация о настройке и использовании этого файла содержится в описании команды **chnfsdom** в книге *Справочник по командам, том 1*.

Файл /etc/nfs/realm.map

Этот файл используется демоном реестра NFS для преобразования входящих субъектов Kerberos из формата *имя@домен_kerberos* в формат *имя@домен_nfs*.

Затем он может предоставить *имя@домен_nfs* одноразовое разрешение на вход в защищенную среду UNIX. Этот файл предлагает простой способ преобразования субъектов Kerberos в реестр пользователей сервера. Это удобно, когда к серверу обращаются клиенты из разных областей Kerberos, но одного глобального пространства имен. В файле должны содержаться записи следующего формата:

```
область1 домен_nfs
область2 домен_nfs
```

для всех поддерживаемых сервером областей Kerberos. Если имя области Kerberos всегда совпадает с именем домена сервера NFS, то этот файл не требуется. Если вам необходимы дополнительные функции при работе с преобразованием *пользовательA@область_kerberos* в *пользовательB@домен_nfs*, воспользуйтесь службой Преобразования идентификаторов в рамках предприятия (EIM). Дополнительная информация приведена в разделе “Преобразование идентификаторов” на стр. 539.

Для добавления, удаления или изменения записей этого файла воспользуйтесь командой **chnfsrtd**. Дополнительные сведения приведены в описании команды **chnfsrtd** в книге *Справочник по командам, том 1*.

Файл /etc/nfs/princmap

Преобразует имена хостов в субъекты Kerberos, когда субъект не является полным именем домена сервера.

Он может состоять из любого числа строк следующего формата:

```
<часть хоста субъекта> псевдоним1 псевдоним2 ...
```

Для добавления, удаления или изменения записей этого файла воспользуйтесь командой **nfshostmap**. Дополнительные сведения приведены в описании команды **nfshostmap** в книге *Справочник по командам, том 4*.

Файл /etc/nfs/security_default

Файл `/etc/nfs/security_default` содержит список опций защиты, которые можно использовать с клиентом NFS, расположенных в порядке их использования.

Для управления этим файлом воспользуйтесь командой **chnfssec**. Дополнительные сведения приведены в описании команды **chnfssec** в книге *Справочник по командам, том 1*.

Протокол вызова удаленной процедуры

NFS реализована для многих типов компьютеров, операционных систем и сетевых архитектур. Подобная универсальность NFS достигается благодаря использованию протокола **Вызова удаленной процедуры (RPC)**.

RPC представляет собой библиотеку процедур. Процедуры позволяют процессам клиента запускать процессы сервера и выполнять с их помощью различные действия точно так же, как если бы процесс клиента выполнял запросы в своем собственном адресном пространстве. Так как сервер и клиент - это два отдельных процесса, они могут физически находиться в разных системах.

NFS реализована в виде набора вызовов **RPC**, которые сервер выполняет по запросу клиентов. В соответствии с операциями над файлами, выполняемыми процессом клиента, клиент передает необходимые запросы серверу. Таким образом, NFS можно рассматривать как приложение RPC.

Поскольку сервер и клиент могут располагаться в разных системах с различной архитектурой, то и способ представления данных может оказаться различным. Поэтому в **RPC** применяется внешнее представление данных **(XDR)**.

Протокол внешнего представления данных

Протокол внешнего представления данных (XDR) - это протокол стандартного представления различных типов данных.

Благодаря применению этого протокола даже те данные, которые поступают от компьютеров совершенно другой архитектуры, интерпретируются программой правильно.

Однако в действительности большинство программ не используют протокол **XDR** непосредственно. Чаще всего программы представляют данные в формате, характерном для архитектуры своего компьютера. Если возникает необходимость во взаимодействии с какой-либо другой программой, перед передачей она преобразует данные в формат **XDR**. И наоборот: при получении данных извне программа преобразует их в свой тип представления.

Демон portmap

Демон **portmap** помогает клиентам привязать пары номер программы и номер версии к номеру порта сервера.

Каждому приложению RPC присвоен определенный номер программы и номер версии. Эти номера необходимы для правильного взаимодействия с приложением сервера. Для передачи запроса серверу клиент должен знать номер порта сервера. Номер этого порта связан с применяемым **Протоколом пользовательских дейтаграмм (UDP)** или **Протоколом управления передачей (TCP)**. Клиенту известны номер программы, номер версии и имя системы (имя хоста), на которой находится нужная служба. Клиент должен связать номера программы и версии с номером порта приложения сервера. Для этой цели применяется демон **portmap**.

Демон **portmap** и приложение NFS работают в одной системе. При запуске сервер регистрируется у демона **portmap**. При этом сервер передает демону данные о номере программы, номере версии, и номере порта **UDP** или **TCP**. Программа-демон **portmap** хранит таблицу приложений сервера. Перед отправкой запроса серверу клиент обращается к демону **portmap** и узнает номер порта сервера. Демон **portmap** возвращает клиенту номер порта запрашиваемого сервера. Все последующие запросы клиент отправляет напрямую приложению сервера.

Приложения и управление NFS

Демонами NFS NIS управляет Контроллер системных ресурсов (SRC).

Это означает, что для запуска, завершения работы и получения информации о состоянии демонов NFS и NIS должны применяться такие команды SRC, как **startsrc** , **stopsrc** и **lssrc**.

Некоторыми демонами NFS, а именно **rpc.rexd**, **rpc.rusersd**, **rpc.rwalld** и **rpc.rsprayd**, нельзя управлять с помощью SRC. Запуском и завершением работы этих демонов управляет демон **inetd**.

В следующей таблице приведен список демонов и подсистем, которыми можно управлять с помощью SRC.

Таблица 90. Программы-демоны и их подсистемы

Путь к файлу	Имя подсистемы	Имя группы
/usr/sbin/nfsd	**nfsd**	nfs
/usr/sbin/biod	**biod**	nfs
/usr/sbin/rpc.lockd	**rpc.lockd**	nfs
/usr/sbin/rpc.statd	**rpc.statd**	nfs
/usr/sbin/rpc.mountd	**rpc.mountd**	nfs
/usr/sbin/nfsrgyd	**nfsrgyd**	nfs
/usr/sbin/gssd	**gssd**	nfs
/usr/lib/netsvc/yp/ypserv	**ypserv**	yp
/usr/lib/netsvc/yp/ypbind	**ypbind**	yp
/usr/lib/netsvc/rpc.yppasswdd	**yppasswdd**	yp
/usr/lib/netsvc/rpc.ypupdated	**ypupdated**	yp
/usr/sbin/keyserv	**keyserv**	keyserv
/usr/sbin/portmap	**portmap**	portmap

Информация, связанная с данной:

Обзор контроллера системных ресурсов

Изменение числа демонов biod и nfsd

С помощью команды **chnfs** можно изменить максимальное число демонов **biod** или **nfsd**, запускаемых в системе.

Например, для того чтобы в системе работало 1000 демонов **nfsd** и 4 демона **biod**, вызовите следующую команду:

```
chnfs -n 1000 -b 4
```

Примечание: Эта команда прекратит работу текущих демонов, обновит параметры SRC и вновь запустит демоны. В результате ее выполнения служба NFS будет временно недоступна.
Максимальное число демонов **biod** следует указывать в зависимости от операций монтирования. Для этого служит опция `biods=`*n*.

Примечание: Если число демонов **nfsd** оказывается недостаточным для обслуживания клиента, возвращается сообщение об ошибке неидемпотентной операции. Например, если клиент удаляет каталог, появляется сообщений об ошибке `ENOENT` несмотря на то, что каталог сервера был удален.

Изменение аргументов командной строки для демонов, управляемых SRC

Многие демоны NFS и NIS допускают указание при запуске различных параметров. Поскольку демоны нельзя запустить непосредственно из командной строки, для их правильной работы необходимо обновить базу данных SRC.

Для этого служит команда **chssys**. Ниже описан формат вызова команды **chssys**:

```
chssys -s демон -a 'новый-параметр'
```

Например:

```
chssys -s nfsd -a '10'
```

Эта команда изменяет подсистему **nfsd** таким образом, чтобы команда запуска демона выглядела следующим образом: `nfsd 10`. Изменения, внесенные с помощью команды **chssys**, вступают в силу только после завершения работы и перезагрузки системы.

Запуск демона NFS

Значение максимального размера файла на сервере NFS определяется параметрами среды, действовавшими на момент запуска **nfsd**.

Другое значение можно указать в файле `/etc/rc.nfs`. Вызовите команду **ulimit**, указав в ней необходимое ограничение, перед вызовом команды **startsrc** с параметром **nfsd**.

Вы можете запустить все демоны NFS или только один из них. Для запуска отдельного демона NFS вызовите следующую команду:

```
startsrc -s демон
```

где *демон* - любой демон, управляемый SRC. Например, для запуска демонов **nfsd** введите:

```
startsrc -s nfsd
```

Для запуска всех демонов NFS введите:

```
startsrc -g nfs
```

Примечание: Если файл `/etc/exports` отсутствует, то демоны **nfsd** и **rpc.mountd** запущены не будут. С помощью команды `touch /etc/exports` можно создать пустой файл `/etc/exports`. Эта операция позволит запустить демоны **nfsd** и **rpc.mountd**, хотя никакие файловые системы при этом экспортированы не будут.

Остановка демонов NFS

Вы можете завершить работу всех демонов NFS или одного из них.

Для завершения работы отдельного демона NFS введите:

```
stopsrc -s демон
```

где *демон* - любой демон, управляемый SRC. Например, для завершения работы демона **rpc.lockd** нужно ввести:

```
stopsrc -s rpc.lockd
```

Для того чтобы завершить работу всех демонов NFS, введите:

```
stopsrc -g nfs
```

Просмотр текущего состояния демонов NFS

Вы можете узнать текущее состояние отдельного демона NFS или всех демонов сразу.

Для того чтобы узнать текущее состояние отдельного демона NFS, введите:

```
lssrc -s демон
```

где *демон* - любой демон, управляемый SRC. Например, для того чтобы узнать текущее состояние демона **rpc.lockd** нужно ввести:

```
lssrc -s rpc.lockd
```

Для того чтобы получить информацию о текущем состоянии всех демонов NFS, введите:

```
lssrc -a
```

Поддержка NFS версии 4

Протокол NFS версии 4 поддерживается начиная с AIX 5.3.

Обязательные функции протокола поддерживаются, как описано в RFC 3530, со следующими исключениями:

- Механизмы защиты LIPKEY и SPKM-3 не поддерживаются при идентификации RPCSEC-GSS RPC. Поддерживается только Kerberos V5.
- Требования UTF-8 не поддерживаются полностью. Не гарантируется, что передача имен файлов и строк файловой системы, например содержание символьной связи и имена записей каталогов, будет соответствовать формату UTF-8. Передача строк атрибутов NFS, например имя владельца и группы владельца, всегда соответствует формату UTF-8. Сервер и клиент NFS осуществляют проверку UTF-8 при получении строк данных, как описано в RFC 3530. Эту проверку можно отключить с помощью команды **nfso**. Отключение проверки UTF-8 может потребоваться для использования NFS версии 4 в среде, в которой формат данных и конфигураций отличен от UTF-8.
- Бездисковый клиент, NIM и UDP не поддерживаются в NFS версии 4.

Поддерживаются следующие необязательные функции NFS версии 4:

- Списки ACL NFS версии 4 поддерживаются как клиентом, так и сервером NFS. Клиент NFS поддерживает управление ACL NFS версии 4 с помощью утилит **acledit**, **aclget** и **aclput**. Сервер NFS может хранить и извлекать списки ACL NFS версии 4 в базовых файловых системах, поддерживающих модель ACL NFS версии 4. Дополнительная информация приведена в разделе “Поддержка списков управления доступом NFS” на стр. 515.
- Также поддерживается преобразование субъектов и атрибутов принадлежности файлов из одного домена NFS версии 4 в другой. Эта функция предназначена для использования в основном на серверах AIX NFS. Она требует развертывания LDAP. Для управления преобразованиями NFS служит утилита **chnfsim**.

Если вы используете параллельный доступ с NFS версий 2 и 3 и NFS версии 4, то учтите следующее. Доступ NFS версии 3 может вызвать ошибку из-за состояния готовности NFS версии 4. Кроме того, быстродействие NFS версии 3 может снизиться при экспортировании данных для доступа NFS версии 4.

Период отсрочки сервера NFS

Протокол NFS версии 4 (NFSv4) позволяет системным администраторам использовать период отсрочки для сервера NFSv4 для особой обработки определенных операций.

Во время этого периода отсрочки администраторы могут управлять блокировкой, использовать операции чтения и записи в течение всего срока аренды сервера. Блокировка и связанные с ней состояния могут быть восстановлены клиентами с помощью запросов восстановления блокировки.

Примечание: Не все восстанавливаемые пользователями состояния во время периода отсрочки могут быть обязательно состояниями, принадлежащими серверу в предыдущем экземпляре. Восстановленное во время периода отсрочки состояние обязательно является корректным, согласно определению NFSv4 RFC.

Администраторы могут использовать период отсрочки для серверов NFSv4 начиная с AIX 5L версии 5.3 с технологическим уровнем обслуживания 5300-05. По умолчанию период отсрочки отключен. Для того чтобы включить период отсрочки на сервере используйте меню SMIT или интерфейс командной строки **chnfs**.

По истечении периода отсрочки сервер NFSv4 записывает сведения о состоянии на диск в файл `/var`. При перезапуске сервера сохраненное состояние восстанавливается автоматически.

Поддержка DIO и CIO в NFS

AIX 5L версии 5.3 с рекомендуемым пакетом обслуживания 5300-03 поддерживает прозрачный и параллельный ввод-вывод в клиенте NFS для протоколов версий 3 и 4. DIO и CIO работают только с клиентом.

При использовании DIO и CIO производительность при выполнении задач центра данных, таких как приложения базы данных и приложения, выполняющие большое количество вычислений, может увеличиться, а также может снизиться потребление ресурсов CPU и памяти системы. При этом сохраняются преимущества централизованного хранения файлов и связанного с ним управления базовыми системами.

Ввод-вывод часто не является последовательным, и приложения не получают преимуществ от кэширования данных на клиенте NFS, либо приложения выполняют расширенное кэширование. Такие приложения выигрывают при отсутствии кэширования в NFS, выполняют упреждающее чтение либо используют механизмы отложенной записи. Кроме этого, некоторые приложения (например, базы данных) не зависят от односторонней семантики POSIX, в которой операции чтения и записи выполняются последовательно. Такие приложения выполняют параллельные операции чтения и записи, однако также отвечают за согласованность и координацию таких операций.

Прозрачный ввод-вывод для NFS

DIO позволяет приложениям выполнять операции чтения и записи непосредственно на сервер NFS без обращения к слою кэширования на клиенте NFS (Администратор виртуальной памяти), то есть избегая дополнительной нагрузки, связанной с кэшированием данных.

При использовании DIO запросы приложения на ввод-вывод обслуживаются с помощью Вызова удаленных процедур (RPC) на сервере NFS. DIO (Прозрачный ввод-вывод) можно включить с помощью опции монтирования AIX *dio*. Без применения этой опции монтирования можно также включить DIO для файла, установив флаг **AIX O_DIRECT open()**.

Для обслуживания операций прозрачного ввода-вывода в NFS может потребоваться несколько вызовов RPC на сервере, в зависимости от необходимого объема ввода-вывода и максимального размера проводника, разрешенного сервером и клиентом. Дополнительная информация о DIO приведена в опции **-o** для команды **mount**.

Параллельный ввод-вывод для NFS

При использовании CIO операции чтения и записи, запущенные одновременно, выполняются параллельно без блокирования операций чтения при выполнении операций записи или наоборот.

Несколько операций записи также выполняются параллельно. Гарантии атомарности POSIX не предоставляются. Использование CIO предполагает прозрачный ввод-вывод. CIO можно установить с помощью опции монтирования AIX *cio* либо установив флаг **O_CIO open()**. Дополнительная информация о CIO приведена в опции **-o** для команды **mount**.

Начиная с AIX версии 6.1 с технологическим пакетом обслуживания 6100-04, с помощью команд **mount** и **nfs4cl**, а также процедуры **open()** можно открывать файлы, доступные только для чтения, которые уже открыты в CIOR. Опцию монтирования **cior** и флаг **O_CIOR open ()** можно использовать только совместно с CIO.

Информация, связанная с данной:

команда mount

Взаимосвязи между DIO, CIO, операциями открытия и привязанных файлов для NFS

При использовании DIO и CIO между различными режимами доступа существуют следующие взаимосвязи.

Если выполняются существующие операции открытия DIO:

- Обычная операция открытия выключает DIO до тех пор, пока выполняются обычные операции открытия. Когда операция закрытия уменьшает число открытых обычным способом файлов до 0, работа DIO возобновляется, если существуют ожидающие обработки операции открытия DIO.
- Запись преобразования файла shmat() или mmap() выключает DIO для этого файла до тех пор, пока число записей преобразования не уменьшается до 0. Затем если существуют операции открытия DIO, работа DIO возобновляется.
- При попытке открытия файла для CIO возникает ошибка **EINVAL**.

Если выполняются обычные операции открытия (не CIO или DIO):
- Попытки открытия для DIO будут удачными, однако DIO не активируется до тех пор, пока число файлов, открытых обычных образом, не станет равным 0.
- При попытке открытия файлов для CIO возникает ошибка **EINVAL**.

Если выполняются операции открытия для CIO:
- При попытке выполнения обычных операций ввода-вывода, DIO и преобразований файла возникает ошибка **EINVAL**.

Если выполняются операции открытия для CIO|CIOR:
- При попытке выполнения обычных операций ввода-вывода, DIO и преобразований файла возникает ошибка **EINVAL** за исключением операций открытия CIO|CIOR и только для чтения.

Примечание: При переходе к DIO или CIO кэшированные клиентом изменения сначала записываются обратно на сервер NFS, а затем вся кэшированная информация удаляется.

Репликация NFS и глобальное пространство имен

Протокол NFS версии 4 (NFSv4) предоставляет функции, позволяющие системному администратору распределять данные между несколькими серверами доступным для пользователей данных способом.

Можно использовать две функции, предоставляемые начиная с AIX 5L версии 5.3 с рекомендуемым пакетом обслуживания 5300-03. Первая из них - функция глобального пространства имен, называемая *referral*. Вторая - средство указания адресов хранения копий данных, называемое *replica*.

referral - специальный объект, который можно создать в пространстве имен сервера, к которому подключена информация о расположении. Сервер применяет функции протокола NFSv4 для перенаправления клиентов на сервер, указанный в информации о расположении. Переадресация создает строительный блок для интеграции данных нескольких серверов NFS в едином файле дерева пространства имен, по которому могут перемещатся предупрежденные о переадресации клиенты NFSv4.

replica является копией файловой системы на одном из серверов NFS, размещенной на нескольких других серверах NFS (либо по альтернативному адресу, например, на другом диске этого же сервера). Если копия, используемая клиентом NFSv4, становится недоступной по одному из адресов, клиент переключится на другую доступную копию. Дополнительная информация о копиях приведена в разделе “Копии NFS” на стр. 530.

Переадресация NFS

В следующих примерах приведены сценарии, которые помогут понять принципы переадресации.

В приведенных ниже примерах рассматривается четыре сервера:
- На сервере `publications` содержатся файлы документации.
- На сервере `projects` - пользовательские рабочие каталоги.
- На сервере `data` размещены информационные базы данных.
- Сервер `account1` является главным сервером NFS, экспортирующим все остальные файлы, а также сервером, к которому обращаются все клиенты.

Предоставление всем клиентам доступа к файлам на главном сервере NFS

Сервер account1 экспортирует каталог /work на все клиенты с помощью следующего оператора в файле /etc/exports:

```
/work -vers=4
```

Все клиенты могут получить доступ к файлам в удаленном каталоге /work, смонтировав / с сервера account1 в каталог /mnt с помощью следующей команды:

```
mount -o vers=4 account1:/ /mnt
```

Когда пользователь клиента просматривает содержимое каталога /mnt, он видит удаленный каталог work по пути /mnt/work. Содержимое каталога /mnt/work на клиенте аналогично содержимому каталога /work на сервере account1.

Предоставление клиентам доступа к файлам на определенном сервере

Пользователю клиента необходим также доступ к каталогу /usr/doc на сервере publications.

В предыдущих выпусках следовало экспортировать каталог с сервера и смонтировать его на клиенте.

Применение переадресации для создания распределенных пространств имен

Можно настроить сервер таким образом, что клиенты могут получить доступ к данным на других серверах без оповещения клиента о размещении данных. Информация о размещении данных необходима только администратору сервера, с которого происходит переадресация. Сервер, с которого осуществляется переадресация, может перенаправить клиентов к расположению каталога /usr/doc с помощью функции переадресации. На сервере publications каталог /usr/doc можно экспортировать, добавив следующий оператор в файл экспорта:

```
/usr/doc -vers=4
```

В результате этого каталоги становятся доступными для клиентов NFSv4.

Сервер account1 теперь может использовать переадресацию, чтобы сделать эти каталоги доступными для клиентов, добавив следующий оператор в файл экспорта:

```
/usr/doc -vers=4,refer=/usr/doc@publications
```

Затем каталог экспортируется. На этом этапе клиент, смонтировавший каталог /mnt из каталога / на сервере account1, получает доступ к каталогу usr при просмотре каталога /mnt. Клиенту не требуется проводить операции монтирования на других серверах. Также не существует необходимости информировать пользователя клиента о том, что файлы предоставляются не сервером account1. Например, можно сделать каталог /databases/db на сервере data и каталог /home/accts на сервере projects доступными с сервера account1, путем экспорта каталогов с серверов data и projects и создания на сервере account1 переадресацию к этим каталогам.

Поскольку пользователь клиента не предупрежден о действительном расположении данных, администратор может перенаправить клиентов с одного сервера на другой, просто изменив оператор переадресации в файле экспорта на сервере. Администратор отвечает за размещение и корректность данных, на которые указывает переадресация.

Администраторам следует убедиться, что второй сервер не переадресует запрос обратно первому серверу, создав таким образом круговую переадресацию. Если администратор в приведенном выше примере создал переадресацию на сервер publications в каталоге /usr/doc, который ссылается на каталог /usr/doc на сервере account1, возникшая круговая переадресация будет нежелательной.

Хотя переадресация создается с помощью exportfs, она отличается от экспорта данных. Адреса, указанные для переадресации, должны соответствовать корневым каталогам файловых систем, экспортированных NFSv4. Можно создать переадресацию с экспортированным либо неэкспортированным пространством имен. В приведенном выше примере переадресацию /usr/doc можно создать на сервере account1 даже в том случае, если каталог /usr не экспортирован. Переадресация будет размещена в псевдопространстве NFSv4. Если на сервере account1 имеется экспортированный каталог /usr, переадресация будет по-прежнему разрешена в отличие от экспорта каталога doc, который оказывается невозможным, если каталог располагался в той же файловой системе. С другой стороны переадресация невозможна, если файл или каталог существует в каталоге /usr/doc. Не существует ограничений на число переадресаций, которые можно создать либо в псевдопространстве NFSv4 сервера либо в экспортированной файловой системе.

Поскольку при переадресации никакие данные не экспортируются, и переадресация имеет значение только для протокола NFSv4, переадресация доступна только в NSFv4. Экспортирующая переадресация без опции vers=4 невозможна. Хотя в данном примере указан только один адрес, можно задать до 8 адресов.

При создании переадресации создается специальный объект переадресации по адресу, указанному в параметре каталога. Поскольку доступ клиента к объекту определяется доступом к родительскому каталогу объекта, большинство оставшихся параметров переадресации не имеет значения и игнорируется. Единственным исключением является параметр exname, имеющий ожидаемое поведение. Например, если сервером создается переадресация /n4root/special/users -vers=4,exname=/exported/users,refer=/restricted/users@secrethost, клиенты, монтирующие каталог / с этого сервера увидят путь /mnt/exported/users, который перенаправит клиентов к каталогу /restricted/users на сервере secrethost. На сервере, выполняющем экспорт, объект переадресации будет в действительности создан в локальном пространстве имен в каталоге /n4root/special/users. Таким образом, никакие файлы или каталоги не могут существовать в нем после выполнения экспорта. На сервере создан специальный объект gets для хранения информации о расположении переадресации. Вдоль пути к переадресации также будут созданы все необходимые каталоги (если они не существуют). Если переадресация не экспортирована, информация о переадресации будет удалена из объекта, однако сам объект удален не будет. Сервер NSFv4 не разрешит клиентам доступ к получившемуся объекту переадресации *stale* или *orphan*. Клиентам, осуществляющим попытку доступа к объекту, будет возвращено сообщение об ошибке. При необходимости объект можно удалить с помощью **rm**. Переадресацию можно экспортировать повторно с созданием новой информации о переадресации. Частое применение подобной процедура не рекомендуется, поскольку для клиентов, получающих доступ к переадресации, требуется некоторое время, чтобы понять, что информация о расположении изменилась. Сервер обращается к родительскому каталогу переадресации, чтобы указать, что информация в каталоге была изменена. Это помогает клиентам обнаружить, что вся информация о каталоге (и о переадресации в каталоге), кэшированная клиентом, изменилась, и ее необходимо внести повторно, однако неизвестно, сколько времени для этого понадобится клиентам.

Информация о применении опции **refer** для изменения порядка расположения путей, указанных в списке расположений файловой системы, содержится в разделе Изменение порядка сортировки списка расположений файловой системы с помощью опции **scatter** .

Копии NFS

Репликация позволяет администратору NFSv4 размещать копии данных на нескольких серверах NFSv4 и сообщать клиентам NFSv4 информацию о расположении этих копий.

В том случае, если первичный сервер данных становится недоступным для клиентов, для продолжения работы с файловой системой, копия которой была создана, эти клиенты могут использовать один из серверов-копий. Предполагается, что файловые системы-копии являются точными копиями данных первичного сервера. Можно указать до 8 адресов копий. Сервер AIX не указывает, каким образом создаются файловые системы-копии первичной файловой системы и как осуществляется синхронизация данных. Если необходимо сделать копии доступными и для чтения, и для записи, следует обеспечить синхронизацию копий и первичной файловой системы.

Сервер-копия представляет собой сервер, который содержит копию каталога или каталогов другого сервера. Если первичный сервер недоступен для клиента, клиент может получить доступ к тем же файлам копии. Ниже приведен пример подобного сценария:

Если файлы в каталоге `/data` на сервере `account1` доступны также в каталоге `/backup/data` на сервере `inreserve`, можно оповестить об этом клиентов NSFv4, указав в экспорте адреса копий. Добавив в файл экспорта оператор, аналогичный приведенному далее, можно экспортировать каталог `/data` и указать адрес копии:

```
/data -vers=4,replicas=/data@account1:/backup/data@inreserve
```

Если сервер `account1` становится недоступным, пользователи клиента, работающие с файлами в каталоге `/data` на сервере `account1`, могут продолжить работу с файлами в каталоге `/backup/data` на сервере `inreserve` без уведомления о том, что клиент переключился на другой сервер.

Информация о применении опции **replicas** для изменения порядка расположения путей, указанных в списке расположений файловой системы, содержится в разделе Изменение порядка сортировки списка расположений файловой системы с помощью опции **scatter** .

Требования к конфигурации NFS для создания копий:

Для включения, выключения и указания копий корневых субъектов необходимо иметь права доступа администратора.

Для включения, выключения и указания копий корневых субъектов используйте следующую команду:

```
chnfs -R {on|off|host[+host]}
```

Для указания копий сервер необходимо настроить с использованием опции **chnfs -R** (**chnfs -R** on) для создания изменяющихся описателей файлов NFSv4. Описатель файла является идентификатором, который серверы NFS выдают клиентам для идентификации файла или каталога на сервере. По умолчанию сервер выпускает постоянные описатели файлов. Переключение типов описателей файлов может вызвать ошибки в работе приложений на клиентах NFSv4, которые обращаются к серверу в то время, когда происходит переключение. При изменении режима описателя файлов с помощью **chnfs -R** файловые системы нельзя экспортировать для доступа к NFSv4. Указание размещения описателя файла следует выполнять на новом сервере NFS либо в то время, когда активность NFS может быть снижена до минимума либо приостановлена. Для клиентов, которые подключаются к серверу при изменении режима, может возникнуть необходимость размонтировать и затем смонтировать NFSv4 повторно. Для минимизации этого процесса можно уменьшить число монтирований клиента до небольшого количества монтирований каталогов верхнего уровня экспортированного файлового пространства сервера NFSv4.

Клиент NFSv4 не может переключаться между копиями с различными параметрами доступа при экспорте. Администраторы должны убедиться, что для всех копий указаны одни и те же права доступа и режим доступа (только для чтения либо для чтенния/записи). Предполагается, что все копии данных будут экспортироваться только для чтения (исключение может составлять файловая система GPFS). В обязанности администратора входит также обслуживание данных во всех копиях. Деревья каталогов и все данные должны оставаться идентичными. Обновление данных следует производить способом, соответствующшим требованиям приложений, работающих с этими данными.

При работе с копиями можно использовать опцию экспорта **exname**, чтобы скрыть от клиентов NFSv4 сведения о локальном пространстве имен файловой системы сервера. Дополнительные сведения приведены в описании команды **exportfs** в книге *Справочник по командам, том 2* и файла `/etc/exports` в книге *Справочник по файлам* .

При экспорте кластерных файловых систем (например, распараллеленной файловой системы General Parallel File System (GPFS)) можно использовать опцию **replicas** для указания нескольких узлов серверов NFS с одним и тем же представлением GPFS. В этой конфигурации допустим экспорт данных с правами доступа на чтение/запись. Однако в этом случае если операции записи выполняются в то время, когда происходит

автоматический перенос ресурсов, в работе приложений, выполняющих запись, могут возникнуть неисправимые ошибки. Аналогично при выполнении операции **mkdir** или создания файла исключений во время автоматического переноса ресурсов может возникнуть ошибка **EXISTS**.

При экспорте копии необходимо экспортировать всю файловую систему. Это означает, что экспортируемый каталог должен являться корневым каталогом локальной файловой системы. Сервер, экспортирующий файловую систему-копию, должен быть указан в качестве одного из адресов для экспорта. Для серверов, имеющих несколько интерфейсов, следует указать первичное имя хоста сервера. Если сервер, экспортирующий файловую систему-копию, не указан в качестве одного из адресов для экспорта, он будет по умолчанию добавлен первым к списку адресов копий. Порядок адресов копий в этом списке указывает порядок, в котором клиентам следует использовать копии при автоматическом переносе ресурсов. Например, если пользователь сервера `serverA` должен экспортировать каталог `/webpages`, и на сервере `serverB` существует копия каталога `/webpages` в каталоге `/backup/webpages`, можно экспортировать каталог `/webpages` с сервера `serverA` и оповестить клиенты о существующей копии файловой системы на сервере `serverB` в каталоге `/backup/webpages`, добавив следующую запись в файл `/etc/exports`:

```
/webpages -vers=4,ro,replicas=/webpages@serverA:
/backup/webpages@serverB
```

Предполагается, что каталоги `/webpages` на сервере `serverA` и `/backup/webpages` на сервере `serverB` являются корневыми каталогами соответствующих файловых систем. Если сервер `serverA` не был внесен в список экспорта, он будет по умолчанию добавлен первым в список адресов копий. Это происходит по той причине, что сервер, экспортирующий данные, считается предпочитаемым для экспортируемых данных.

Копии применяются только протоколом NFSv4. При экспорте можно указать NFSv3 (`vers=3:4`), однако информация о копировании будет недоступна для клиентов NFSv3. Клиенты, работающие с NFSv3, могут получить доступ к информации в каталоге `/webpages` на сервере `serverA`, однако если сервер `serverA` становится недоступным, их автоматическое переключение на копию невозможно.

Клиентская поддержка NFS для нескольких расположений:

Если клиент не может получить доступ к данным, копия которых создана, с текущего сервера, клиент осуществляет попытки получить доступ к этим данным на следующем сервере из списка предпочитаемых серверов.

Порядок копий в списке адресов копий, к которому обращается клиент, является порядком предпочтений.

Администратор клиента может предпочитаемый порядок копий с помощью команды **prefer** в команде **nfs4cl**. Команда **nfs4cl** выводит на клиенте всю информацию о файловой системе либо изменяет параметры файловой системы и показывает либо изменяет текущую статистику и свойства NFSv4.

Общие замечания об NFS для копий и переадресаций:

Если клиентом обнаружены два пути к одним и тем же данным (файловой системе), второй путь рассматривается клиентом в качестве символьной связи с файлом.

Например, экспорт сервера `server A` имеет вид:

```
/tmp/a              -vers=4,replicas=/tmp/a@B:/tmp/a@A
/tmp/b              -vers=4,refer=/tmp/a/b@B
```

А экспорт сервера `server B` имеет вид:

```
/tmp/a              -vers=4
/tmp/a/b            -vers=4
```

В этом примере клиент смонтирует каталог `/` на сервере `server A` в каталог `/mnt` с помощью команды `mount -o vers=4 A:/ /mnt`. Клиент обращается к каталогу `/tmp/a/b` на сервере `server B` через `cd /mnt/tmp/a/b` или `cd /mnt/tmp/b`. Если вначале пользователь перейдет в каталог `cd /mnt/tmp/a/b`, то путь `/mnt/tmp/b`

выполняет роль символической ссылки на /mnt/tmp/a/b. В данном сценарии если пользователь находится в каталоге /mnt/tmp/b и использует команду /bin/pwd, /bin/pwd > вернет результат /mnt/tmp/a/b.

Примечание: Не рекомендуется использовать подобную практику. Администратору следует установить параметры экспорта таким образом, чтобы в пространстве имен существовал единственный путь к экспортированным данным.

В переадресации можно указать несколько адресов, если целевые данные переадресации скопированы. Клиенты применяют адреса переадресаций только для поиска цели переадресации на доступном сервере. Как только клиент получит доступ к цели переадресации, он получит информацию о новом расположении для поиска данных.

Поскольку клиенты не могут немедленно обнаружить изменения в информации о расположении переадресации, не рекомендуется часто удалять или изменять эту информацию. При изменении целевого адреса переадресации подразумевается, что данные будут размещены по новому адресу одновременно с изменением информации о расположении в спецификации экспорта переадресации. Данные по прежнему адресу следует хранить в течение нескольких часов или даже дней, чтобы обеспечить клиентам необходимое время для обнаружения и использования нового адреса.

Репликацию и переадресацию можно выполнять только на серверах с 64-разрядным ядром. Клиенты могут работать как с 32-, так и с 64-разрядным ядром.

Если необходимо сделать копии доступными и для чтения, и для записи, следует обеспечить синхронизацию копий и первичного набора файлов.

Автоматический перенос ресурсов клиентов NFS:

Автоматический перенос ресурсов происходит, когда клиент переключается с одной копии на другую после того, как обнаруживает, что текущий сервер более недоступен.

Следующие переменные влияют на осуществление автоматического переключения ресурсов для клиента NFS:

Опция монтирования NFS *timeo*
: Эта опция монтирования задает время ожидания уровнем TCP/IP перед отправкой сообщения о тайм-ауте.

Опция монтирования NFS *retrans*
: Эта опция монтирования указывает, сколько раз уровень NFS RPC должен повторить запрос клиента перед возвратом ошибки тайм-аута RPC (**ETIMEDOUT**).

Опция nfso *nfs_v4_fail_over_timeout*
: Эту опцию **nfso** можно использовать для задания минимального промежутка времени, в течение которого клиент должен ожидать перед автоматическим переходом к копии. Эта опция является глобальной для клиента NFS и переопределяет поведение при монтировании, заданное по умолчанию. По умолчанию опция **nfs_v4_fail_over_timeout** отключена. Ее значение равно 0.

Если опция **nfs_v4_fail_over_timeout** отключена, пороговое значение времени ожидания перед автоматическим переносом данных устанавливается равным удвоенному значению опции **timeo**. Если в течение этого промежутка времени все вызовы RPC оказываются неудачными, клиент начнет автоматический поиск другой доступной копии. Однако на этот промежуток времени ожидания оказывает влияние опция **retrans**. Если значение **retrans** больше 2, клиент будет ожидать до получения тайм-аута RPC на основании произведения значений **retrans** и **timeo** (retrans × timeo). Следовательно, можно управлять проведением автоматического переключения ресурсов, изменяя опции **timeo** и **retrans**. Можно также устанавливать эти опции более дискретно с помощью команды **nfs4cl**.

Если задано отличное от нуля значение **nfso nfs_v4_fail_over_timeout**, оно представляет собой время в секундах, в течение которого клиент ожидает перед началом автоматического переключения с недоступного

сервера на другую копию. Если значения **timeo** и **retrans** дают значение тайм-аута RPC, которое оказывается меньше значения параметра **nfso**, автоматический перенос ресурсов не начинается до возникновения тайм-аута RPC.

Дополнительные сведения об опциях **retrans**, **timeo** и **nfs_v4_fail_over_timeout** обратитесь к описанию относящихся к NFS вопросов применения команд **mount**, **nfs4cl** и **nfso**.

Кроме автоматического переключения копий в случае, если сервер становится недоступным, существуют ситуации, когда клиент добровольно переключается с одной копии на другую. Одна из таких ситуаций - указание предпочитаемой копии с помощью команды **nfs4cl**. В этом случае клиент осуществляет переключение на предпочитаемый сервер (если клиент в данный момент не работает с этим сервером). Клиент также повторно получает информацию о размещении копии с сервера NFS с интервалом примерно в 30 минут, если происходили обращения к связанным данным. Если порядок адресов был изменен, клиент пытается изменить первый адрес в списке, если он отличается от текущего сервера и если не установлено предпочтение копий с помощью команды **nfs4cl**.

Слабое монтирование NFS и алгоритм передачи управления:

По умолчанию применяется сильное монтирование NFS, и описанное выше проведение автоматического переключения копий относится к случаю сильного монтирования. Если применяется слабое монтирование NFS, проведение автоматического переноса ресурсов будет иным.

Если параметры слабого монтирования приводят к тайм-ауту RPC до истечения периода ожидания перед автоматическим переключением копий, тайм-аут вызовет ошибку **ETIMEDOUT** в работе осуществляющего вызов приложения. Применение слабого монтирования одновременно с репликацией данных не рекомендуется. Если применяется слабое монтирование, и установлено значение опции **nfso nfs_v4_fail_over_timeout**, следует выбрать такие значения опций монтирования **retrans** и **timeo**, чтобы превысить значение параметра **nfso**. Это позволит избежать возникновения ошибки **ETIMEDOUT** для приложений, работающих с копиями данных.

Переупорядочение списка расположений файловой системы с помощью опции scatter

Опция **scatter** команды **exportfs** позволяет изменить порядок перечисления путей, указанных в списке расположений файловой системы. Первоначально этот порядок задает опция **refer** либо **replicas** команды **exportfs**.

С помощью данной опции можно получить разнообразные сочетания расположений серверов, то есть в разных списках будут указаны различные серверы, перечисленные в предпочтительном порядке. Следовательно, различные клиенты получат разные списки расположений сервера. Такое переупорядочение позволяет сбалансировать нагрузку, поскольку в списках разных клиентов на первом месте будут указаны разные серверы. К тому же, если сервер выйдет из строя, то при аварийном переключении нагрузка будет распределена на несколько серверов, потому что в разных списках под вторым номером будут указаны разные серверы. Опция **scatter** применяется только к каталогам, экспортированным для доступа согласно протоколу NFS версии 4.

Для опции **scatter** предусмотрены следующие значения:

- **full** - Полное переупорядочение. Все серверы указаны в различном порядке с целью создания комбинаций для альтернативных расположений. Общее число комбинаций ограничено либо 12, либо количеством серверов, в зависимости от того, какое из этих значений больше.
- **partial** - Частичное переупорядочение. Первое расположение для всех генерируемых комбинаций серверов закреплено за первым сервером из списка серверов. Остальные расположения перечисляются так же, как и в случае применения полного переупорядочения.
- **none** - Нет переупорядочения. Переупорядочение списка расположений файловой системы не выполняется. Это значение по умолчанию для опции **option**. Оно используется для отмены всех прежних переупорядочений списка.

Примечание: Если при применении команды **exportfs** не отмечен флаг **noauto**, это значит, что список расположений включает в себя имя основного хоста в качестве одного из расположен-копий. Дополнительная информация о флаге **noauto** содержится в описании команды **exportfs** в*Справочник по командам, том 2*.

Для того, чтобы указать ссылки на каталог `/common/documents` на хостах s1, s2 и s3, а затем переупорядочить их с помощью опции **full**, добавьте следующую строку в файл `/etc/exports` и после этого экспортируйте каталог `/common/documents`:

```
/common/documents -ver=4, refer=/common/documents@s1:/common/document@s2a:/common/
documents@s3,scatter=full
```

Для того чтобы указать реплики для каталога `/common/documents` на хостах s1, s2 и s3 и затем переупорядочить их с помощью опции partial (s1 будет первым сервером для всех сочетаний), добавьте в файл `/etc/exports` следующую строку и затем экспортируйте каталог `/common/documents`:

```
/common/documents -vers=4, replicas=/common/documents@s1:/common/documents@s2:/common/
   documents@s3:/common/documents@s4,scatter=partial
```

Делегирование сервер-клиент NFS

Делегирование представляет собой возможность для сервера передавать ряд полномочий клиенту.

Начиная с AIX 5L версии 5.3 с рекомендуемым пакетом обслуживания 5300-03 вы можете использовать делегирование. Когда сервер делегирует файл клиенту, клиент гарантирует соблюдение определенной семантики в том, что касается совместного использования этого файла вместе с другими клиентами. При открытии файла сервер может делегировать клиенту право на чтение. В этом случае другие клиенты не могут осуществлять запись в этот файл в течение периода делегирования. Если клиенту делегировано право на запись, другие клиенты не могут осуществлять запись в этот файл или чтение файла. Сервер AIX может делегировать только право на чтение. Сервер AIX поддерживает делегирование только при работе с 64-разрядным ядром AIX. Клиент AIX поддерживает делегирование и прав на чтение, и прав на запись.

Для того чтобы сервер предоставил делегирование клиенту, этот клиент должен сообщить серверу адрес обратного вызова. При повторном запросе делегирования сервер отправляет запрос по указанному адресу. По умолчанию клиент указывает IP-адрес, который применяется при обычном соединении с сервером. Для клиентов, имеющих несколько сетевых интерфейсов, конкретный адрес можно указать в файле `/etc/nfs/nfs4_callback.conf`. В этом файле применяется следующий формат записей:

сервер-хост ip-адрес-клиента

Здесь *сервер-хост* - имя или адрес сервера NFSv4, а *ip-адрес-клиента* - адрес клиента для предоставления серверу информации для обратного вызова. Если указан адрес *сервер-хост* 0.0.0.0 (IPv4) или 0::0 (IPv6), заданный *ip-адрес-клиента* будет использован для всех серверов, не перечмсленных в файле. Если данный файл не существует, либо не найдена запись для данного сервера (или запись по умолчанию), клиент выбирает адрес на основании существующего соединения с сервером.

Делегирование может быть отозвано сервером. Если другой клиент запрашивает доступ к файлу таким образом, что этот доступ вызывает конфликт с предоставленным делегированием, сервер может послать уведомление первому клиенту и отозвать делегирование. Для этого между сервером и клиентом должен существовать путь для обратного вызова. Если такой путь не существует, делегирование не может быть предоставлено. Если делегирование было предоставлено, отзыв делегирования может быть вызван доступом к файлу других клиентов NFSv4, клиентов NFS версий 2 и 3, а также локальным доступом к файлу на файловом сервере. Если GPFS экспортирована NFSv4 доступ к сети с узла GPFS может вызвать отзыв делегирования.

Сущность делегирования заключается в том, что оно позволяет клиенту локально выполнять такие операции, как OPEN, CLOSE, LOCK, LOCKU, READ и WRITE без немедленного взаимодействия с сервером.

Делегирование со стороны сервера включено по умолчанию. Делегирование можно выключить с помощью команды `nfso -o server_delegation=0`. С помощью опции **exportfs deleg=yes | no** администратор может включать и выключать делегирование отдельных файлов системы. Эта опция переопределяет значение параметра **nfso**.

Делегирование со стороны клиента можно выключить с помощью команды `nfso -o client_delegation=0`. Делегирование со стороны клиента следует установить до выполнения монтирования на клиенте.

При экспорте файловой системы, в которой предполагается проведение записи в многие файлы со стороны большого числа клиентов, администратор может отключить делегирование для данной файловой системы.

Если связь с клиентом отсутствует (например, в случае простоя сети или клиента), может возникнуть задержка в доступе к данным со стороны других клиентов.

Создание основных субъектов хоста для защищенных Kerberos путей обратного вызова

Вы можете настроить путь обратного вызова для Службы сетевой идентификации IBM (Kerberos).

Получающий делегирование клиент должен являться обычным клиентом с собственными субъектами хоста. Однако можно установить общие субъекты хоста для всех клиентов для использования обратных вызовов.

Для этого выполните следующие действия:

1. Для создания служебного субъекта (например, `nfs/client`) таким же образом, что и субъектов хоста, обратитесь к разделу Создание субъекта Kerberos в *Защита*.
2. Создайте запись ключей для данного служебного субъекта. Например, для создания ключа с именем slapd_krb5.keytab выполните следующие действия:

   ```
   kadmin.local: ktadd -k /etc/security/slapd_krb5.keytab ldap/plankton.austin.ibm.com
   Запись для субъекта ldap/plankton.austin.ibm.com с kvno 2,
   тип шифрования режим Triple DES cbc с HMAC/sha1 будет добавлен в ключ
   WRFILE:/etc/security/slapd_krb5.keytab.
   Запись для субъекта ldap/plankton.austin.ibm.com с kvno 2,
   тип шифрования ArcFour с HMAC/md5 будет добавлен в ключ WRFILE:/etc/security/slapd_krb5.keytab.
   Запись для субъекта ldap/plankton.austin.ibm.com с kvno 2,
   тип шифрования - режим AES-256 CTS с 96-разрядной SHA-1 HMAC добавлен в ключ
   WRFILE:/etc/security/slapd_krb5.keytab.
   Запись для субъекта ldap/plankton.austin.ibm.com с kvno 2,
   тип шифрования - режим DES cbc с RSA-MD5 добавлен в ключ WRFILE:/etc/security/slapd_krb5.keytab.
   kadmin.local:
   ```

3. Передайте этот ключ всем клиентам, которые будут использовать его.
4. Настройте клиенты с помощью команды **nfshostkey**.

Этот процесс аналогичен процессу настройки сервера для применения с Kerberos, но общие субъекты нельзя использовать для серверов, у каждого сервера должны иметься собственные субъекты в виде `nfs/`*`имя-хоста`*.

Краткосрочные файловые системы STNFS

Краткосрочная файловая система (STNFS) - это файловая система на основе сетевой файловой системы (NFS), разрешающая локальное изменение файлов. Изменения не сохраняются на сервере.

Примечания::

1 Многие клиенты STNFS используют один и тот же образ файловой системы сервера, однако изменения доступны только тем клиентам, которые их внесли.

61 см Все изменения, внесенные клиентом, теряются после размонтирования файловой системы или перезагрузки клиента.

3 Операции записи STNFS запрещаются после уменьшения объема свободной системной памяти ниже заданного порогового значения. Это внутреннее пороговое значение STNFS, которое нельзя настроить извне.

Монтирование краткосрочной файловой системы NFS

Команда **mount** позволяет выполнить краткосрочное монтирование файловой системы NFS. Например, введите следующую команду:

```
mount -v stnfs -o options server:/удаленный-путь
/локальный-путь
```

Доступные параметры:

vers=3 Использовать NFS версии 3 для взаимодействия с сервером.

vers=4 Использовать NFS версии 4 для взаимодействия с сервером.

rsize=size
Задать байты только для чтения.

proto=udp
Использовать UDP для взаимодействия с сервером NFS.

proto=tcp
Использовать TCP для взаимодействия с сервером NFS.

hard Разрешить сильное монтирование NFS.

soft Разрешить слабое монтирование NFS.

sec Использовать указанную классификацию безопасности.

Значения по умолчанию:

vers=3

rsize=32768

proto=tcp

hard

sec=sys

Справочная таблица по настройке NFS

После установки программного обеспечения NFS в системе вы можете перейти к его настройке. Для настройки NFS выполните следующие действия.

Прежде чем настраивать NFS для использования перечисленных ниже типов защиты, необходимо установить библиотеку ядра CryptoLite in C (CLiC):

- krb5
- krb5i
- krb5p

Каждый шаг подробно описан ниже.

1. Определите, какие системы, подключенные к сети, будут серверами, и какие - клиентами (система может быть одновременно и сервером, и клиентом).
2. Выберите версию NFS, которую вы будете использовать.
3. Решите, будете ли вы использовать защиту RPCSEC-GSS. Если да, то обратитесь к разделу “Настройка сети для RPCSEC-GSS” на стр. 541.
4. Для каждой системы (независимо от того, клиент это или сервер) выполните инструкции из раздела “Запуск демонов NFS при запуске системы” на стр. 538.

5. Для каждого сервера NFS выполните инструкции из раздела “Настройка сервера NFS”.
6. Для каждого клиента NFS выполните инструкции из раздела “Настройка клиента NFS”.
7. Для того чтобы персональным компьютерам, подключенным к сети, был предоставлен доступ к серверам NFS (а не только возможность монтирования файловых систем), настройте функцию PC-NFS, следуя инструкциям из раздела “PC-NFS” на стр. 552.
8. Если вы собираетесь использовать NFS версии 4, обратитесь к разделу “Поддержка NFS версии 4” на стр. 526.

Запуск демонов NFS при запуске системы

По умолчанию демоны NFS не запускаются во время установки.

После установки все необходимые файлы уже находятся в системе, но запуск NFS не выполняется. Запустить демоны NFS при загрузке системы можно с помощью:

- Команды быстрого доступа `smit mknfs` SMIT.
- Команды **mknfs**.

В любом случае в файл `inittab` добавляется запись, указывающая, что сценарий `/etc/rc.nfs` должен запускаться при каждой загрузке системы. Этот сценарий, в свою очередь, запускает все необходимые в данной системе программы-демоны NFS.

Настройка сервера NFS

Выполните эту процедуру для настройки сервера NFS.

Для того чтобы настроить сервер NFS, выполните следующие действия:

1. Создайте файл `/etc/exports`. См. раздел “Файл /etc/exports” на стр. 521.
2. Если вы используете Kerberos, установите сервер NFS как клиент Kerberos. См. раздел “Настройка сети для RPCSEC-GSS” на стр. 541.
3. Если вы используете NFS версии 4, установите домен NFS версии 4 с помощью команды **chnfsdom**. Дополнительные сведения приведены в описании команды **chnfsdom** в книге *Справочник по командам, том 1*.

   Изначально можно указать Internet-домен сервера в файле. Можно задать домен NFS версии 4, отличный от Internet-домена сервера. За более подробными сведениями по этому вопросу обратитесь к документации по демону реестра NFS **nfsrgyd** в *Справочник по командам, том 4*.
4. Если вы используете NFS версии 4 с Kerberos, возможно, вам потребуется создать файл `/etc/nfs/realm.map`. См. раздел “Файл /etc/nfs/realm.map” на стр. 522.
5. Если вы хотите использовать идентификацию Kerberos на сервере, вам следует включить расширенную защиту на сервере. Включить расширенную защиту можно с помощью SMIT или с помощью команды **chnfs -S -B**. Дополнительные сведения о **chnfs** приведены в описании команды **chnfs** в книге *Справочник по командам, том 1*.

Настройка клиента NFS

Выполните эту процедуру для настройки клиента NFS.

1. Запустите NFS, выполнив инструкции из раздела “Запуск демона NFS” на стр. 525.
2. Создайте локальную точку монтирования командой **mkdir**. Для того чтобы NFS могла успешно выполнить монтирование, каталог-точка монтирования NFS уже должен быть создан. Необходимо, чтобы каталог был пустым. Точка монтирования создается как обычный каталог, никакие специальные атрибуты указывать не нужно.

Примечание: Во всех случаях, за исключением одного, точка монтирования должна создаваться перед монтированием файловой системы. При работе с демоном **automount** точки монтирования нужно создавать не всегда. Дополнительная информация приведена в описании демона **automount** в *Справочник по командам, том 1*.

3. Если используется Kerberos, выполните следующие действия:
   a. Настройте клиент NFS в области Kerberos. Это можно сделать с помощью команды **config.krb5**. За дополнительной информацией о настройке обратитесь к книге *IBM Network Authentication Service Administrator's and User's Guide*.
   b. Создайте субъекты Kerberos для всех пользователей клиента, которые будут обращаться к файлам через Kerberos. Это можно сделать с помощью команды **kadmin**. За дополнительной информацией о создании субъектов Kerberos обратитесь к книге *Network Authentication Service Administrator's and User's Guide*.
   c. Создавать субъекты Kerberos для клиентской системы необязательно. Клиент без субъекта называется *тонким клиентом*, а клиент с субъектом - *обычным клиентом*. Тонкие клиенты используют более слабую защиту NFS RPC при выполнении некоторых операций по управлению контекстом клиент-сервер в NFS версии 4, используемых для управления состоянием. Обычный клиент может, в зависимости от конфигурации, использовать более надежную защиту RPC, основанную на Kerberos. Конфигурация тонкого клиента проще с точки зрения администратора и вполне достаточна во многих случаях. Если же среда предъявляет высокие требования к защите, то можно настроить обычный клиент.
4. Если вы используете NFS версии 4, установите домен NFS версии 4 с помощью команды **chnfsdom**. Изначально можно указать Internet-домен клиента в файле. Можно задать домен NFS версии 4, отличный от Internet-домена клиента. Более подробное пояснение приведено в документации по демону реестра NFS, **nfsrgyd**.
5. Если вы хотите использовать идентификацию Kerberos на клиенте, вам следует включить расширенную защиту на клиенте. Включить расширенную защиту можно с помощью SMIT или с помощью команды **chnfs -S -B**. Дополнительная информация о команде **chnfs** приведена на странице справочника, посвященной **chnfs**.
6. Задайте и смонтируйте предопределенные файловые системы, выполнив инструкции из раздела “Установка предопределенных монтирований NFS” на стр. 547.

Преобразование идентификаторов

Преобразование идентификаторов предлагает для локальных сервера и клиента NFS способ преобразования внешних пользователей и групп в локальные пользователи и группы.

Для преобразования идентификаторов AIXиспользует технологию EIM, основанную на LDAP. Все данные о преобразовании идентификаторов NFS хранятся на сервере LDAP.

Для установки клиента EIM в системе должны быть установлены наборы файлов `bos.eim.rte` и `ldap.client`. Серверу EIM также требуется набор файлов `ldap.server`. После того как требуемые наборы файлов установлены, для настройки EIM используется `/usr/sbin/chnfsim`. Минимальные опции установки таковы:

```
/usr/sbin/chnfsim -c -a -t [тип] -h [сервер EIM] -e [домен LDAP/EIM] -f [суффикс LDAP] -w [пароль администратора]
```

Эта команда настраивает серверы и клиенты EIM для использования указанного сервера EIM для преобразования идентификаторов. Если указанное в команде имя хоста соответствует локальному хосту, то будет установлен также сервер LDAP.

После завершения настройки администратор EIM сможет заполнять сервер LDAP записями преобразования идентификаторов NFS. Идентификатором преобразования называют отдельного пользователя или группу, например Иван Петров. Строка владельца NFS этого пользователя, *ivan-petrov@austin.ibm.com*, называется записью преобразования идентификатора. Для ввода в сервер LDAP этих данных используется следующая команда:

```
/usr/sbin/chnfsim -a -u -i "Ivan Petrov" -n ivan-petrov -d austin.ibm.com
```

Идентификатор преобразования - это описательное имя пользователя или группы, а запись преобразования идентификатора - это строка владельца NFS *имя@домен*. На сервере LDAP также хранятся записи преобразования областей в домены. Для ввода записи, задающей преобразование области Kerberos `kerb.austin.ibm.com` в домен NFS `austin.ibm.com`, выполните следующую команду:

```
/usr/sbin/chnfsim -a -r kerb.austin.ibm.com -d austin.ibm.com
```

Для настройки NFS для использования записей преобразований в EIM следует перезапустить демон реестра NFS. Демон реестра NFS проверяет, доступен ли сервер EIM при запуске, и если да, то все функции преобразования будут выполняться через EIM, а все локальные преобразования использоваться не будут.

Подробные сведения об EIM приведены в разделе Enterprise identity mapping книги *Защита*.

Экспорт файловой системы NFS

Для экспорта файловой системы NFS воспользуйтесь выполните одну из следующих процедур.

- Для того чтобы экспортировать файловую систему NFS с помощью SMIT, выполните следующие действия:
  1. С помощью команды `lssrc -g nfs` проверьте, запущена ли NFS. В выводе этой команды демоны **nfsd** и **rpc.mountd** должны быть помечены как активные. Если это не так, запустите NFS, выполнив инструкции из раздела “Запуск демона NFS” на стр. 525.
  2. В командной строке введите следующую команду и нажмите Enter:

     ```
     smit mknfsexp
     ```
  3. Укажите необходимые значения в параметрах Путь (путь к экспортируемому каталогу), Режим (режим экспорта) и Экспорт (экспортировать каталог немедленно и/или при повторном запуске системы).
  4. Заполните остальные поля или оставьте в них значения по умолчанию.
  5. После внесения всех необходимых изменений SMIT обновит файл `/etc/exports`. Если файл `/etc/exports` не существует, он будет создан.
  6. Повторите шаги 3-5 для каждого каталога, который нужно экспортировать.
- Для того чтобы экспортировать файловую систему NFS с помощью текстового редактора, выполните следующие действия:
  1. Откройте в любом текстовом редакторе файл `/etc/exports`.
  2. Создайте для каждого экспортируемого каталога запись, содержащую его полное имя. Начиная с первой колонки введите список всех каталогов, предназначенных для экспорта. Ни в одном из каталогов не должно быть каких-либо подкаталогов, экспортированных ранее. Описание полного формата записей файла `/etc/exports` содержится в файле `/etc/exports` в *Справочник по файлам* .
  3. Сохраните файл `/etc/exports` и закройте его.
  4. Если NFS запущена, введите следующую команду и нажмите Enter:

     ```
     /usr/sbin/exportfs -a
     ```

     Опция **-a** указывает команде **exportfs** отправлять все сведения файла `/etc/exports` ядру. Если NFS не запущена, запустите ее, выполнив инструкции из раздела “Запуск демона NFS” на стр. 525.
- Для того чтобы временно экспортировать файловую систему NFS, не изменяя файл `/etc/exports`, введите следующую команду и нажмите Enter:

  ```
  exportfs -i /имя-каталога
  ```

  где *имя-каталога* - имя экспортируемой файловой системы Команда **exportfs -i** не считывает информацию из файла `/etc/exports`, а получает все параметры из командной строки.

  NFS версии 4 в AIX позволяет администратору создавать и управлять альтернативным пространством имен, которое сервер NFS выдает клиентам. Для этой цели служит опция экспортирования **exname**. Кроме того, эта поддержка позволяет скрыть сведения о локальном пространстве имен файловой системы

сервера от клиентов NFS. Дополнительные сведения приведены в описании команды **exportfs** в книге *Справочник по командам, том 2* и файла `/etc/exports` в книге *Справочник по файлам* .

Настройка сети для RPCSEC-GSS

Настраиваемая в этом сценарии сеть содержит пять серверов и настроена для RPCSEC-GSS.

Сеть содержит следующие пять серверов:

- `kdc.austin.ibm.com`
- `alpha.austin.ibm.com`
- `beta.austin.ibm.com`
- `gamma.austin.ibm.com`
- `zeta.austin.ibm.com`

Система `kdc.austin.ibm.com` будет настроена как Центр рассылки ключей (KDC), также будет создана область Kerberos `AUSTIN.IBM.COM`, в которой все системы, кроме `kdc.austin.ibm.com` и `zeta.austin.ibm.com`, будут серверами NFS с файловыми системами, экспортированными с RPCSEC-GSS.

Между системами `alpha.austin.ibm.com` и `beta.austin.ibm.com` установлено дополнительное соединение; в этом соединении они распознают друг друга под именами `fast_alpha.test.austin.com` и `fast_beta.test.austin.ibm.com`. По этой причине, потребуется дополнительная настройка.

Кроме этого, в сети имеются следующие пользователи, которые имеют учетные записи в некоторых системах:

- `adam`
- `brian`
- `charlie`
- `dave`
- `eric`

Примечание: Нижеследующий образец настройки приведен только в качестве примера и не обязательно подходит к каждой среде. Перед установкой новой области Kerberos ознакомьтесь с Руководством пользователя и администратора для Службы сетевой идентификации.

Примечание: Kerberos требует, чтобы время во всех системах сети было примерно одинаковым. Перед началом процедуры настройте механизм автоматической синхронизации сети, например демон AIX **timed** или NTP.

1. Создайте сервер KDC.

   Примечание: Сервер KDC не рекомендуется использовать для иных целей; если сервер KDC будет подвергнут опасности, то все субъекты Kerberos также будут подвергнуты опасности.

   В этом сценарии `kdc.austin.ibm.com` будет настроен как сервер KDC. Ниже приведен пример настройки **des3**. Если из соображений производительности вы предпочтете команду **des**, добавьте аргумент `-e des-cbc-crc:normal` к вызовам `addprinc` и `ktadd` для команды **kadmin**.

   Для настройки сети с шифрованием **aes** encryption, добавьте аргумент `-e aes256-cts:normal` к вызовам `addprinc` и `ktadd` для команды **kadmin**.

   a. Установите набор файлов `krb5.server.rte` на `kdc.austin.ibm.com`.

   b. Создайте сервер KDC. В этом сценарии использовались следующие команды:

      ```
      config.krb5 -S -d austin.ibm.com -r AUSTIN.IBM.COM
      ```

      После выполнения этой команды система запросит пароль к главной базе и пароль к административному субъекту.

c. Создайте субъекты для каждого пользователя и хоста, выполнив команду /usr/krb5/sbin/kadmin.local на сервере KDC. Эта команда создает субъекты Kerberos, соответствующие именам пользователей в UNIX. Имя субъекта будет заменено NFS на имя пользователя для определения одноразового разрешения UNIX, связанного с субъектом. Более подробное описание преобразований субъектов в имена пользователей и обратно приведено в разделе “Преобразование идентификаторов” на стр. 539. Для этой сети созданы следующие субъекты:

- adam
- brian
- charlie
- dave
- eric
- nfs/alpha.austin.ibm.com
- nfs/beta.austin.ibm.com
- nfs/gamma.austin.ibm.com

Примечание: Выбранные имена субъектов пользователей должны совпадать с соответствующими именами пользователей в настроенном реестре пользователей системы /etc/passwd, **LDAP**, **NIS** и т.д.). NFS использует имя субъекта в качестве имени пользователя для получения ИД пользователей и групп в локальной системе. Если имена не совпадают, доступ к системе будет рассматривается как анонимный.

Теперь KDC настроен.

2. Теперь каждый клиент и сервер NFS будет настроен как клиент Kerberos с помощью команды **config.krb5**. Способ выполнения этого действия зависит от того, как настроен KDC. В этом сценарии в каждой системе NFS запускаются следующие программы:

```
config.krb5 -C -d austin.ibm.com -r AUSTIN.IBM.COM -c kdc.austin.ibm.com -s kdc.austin.ibm.com
```

Теперь можно выполнить команду **kinit** для любого субъекта пользователя в любой настроенной системе. Например, для выполнения команды **kinit** для пользователя adam введите:

```
/usr/krb5/bin/kinit adam
```

Необходимо указать пароль пользователя adam в Kerberos, а не в AIX.

В этом примере **kinit** используется для идентификации пользователя. Можно настроить AIX для использования идентификации Kerberos во время входа в систему. Дополнительная информация приведена в разделе Идентификация в AIX с использованием Kerberos в *Защита*.

3. Каждый сервер NFS теперь будет настроен с соответствующей записью keytab. В этом сценарии запись keytab настроена для alpha.austin.ibm.com в качестве примера; точно такие же операции выполняются для beta.austin.ibm.com и gamma.austin.ibm.com.

a. Из системы alpha.austin.ibm.com запустите команду **kadmin**. Затем запустите следующую команду:

```
ktadd nfs/alpha.austin.ibm.com
```

Она создает файл keytab.

b. Затем создайте демон **gssd** для использования только что созданного файла keytab с помощью команды **nfshostkey**. В этом сценарии запускается следующая команда:

```
nfshostkey -p nfs/alpha.austin.ibm.com -f /etc/krb5/krb5.keytab
```

c. Настройте демон **gssd** для автоматического запуска с помощью следующей команды:

```
chnfs -S -B
```

Повторите эту процедуру для каждой системы.

4. Теперь, когда сервер NFS заработал, все пользователи будут входить в систему как nobody. Рекомендуется, чтобы все пользователи присутствовали на всех серверах с одинаковыми uid и gid; любой отсутствующий пользователь будет получать доступ к экспортированному каталогу только как nobody. Для правильного отображения имен пользователей нужно настроить демон реестра NFS.

a. Создайте демон с помощью команды **chnfsdom**. В этом сценарии на всех серверах NFS была запущена следующая команда для настройки austin.ibm.com как домена:

```
chnfsdom austin.ibm.com
```

b. Создайте файл /etc/nfs/realm.map; он должен состоять из одной строки с именем области и локальным доменом. Для рассматриваемой сети эти два файла должны выглядеть следующим образом на всех серверах NFS:

```
realm.map AUSTIN.IBM.COM   austin.ibm.com
```

Запись области в этом файле указана без учета регистра, поэтому с технической точки зрения она не обязательна.

c. Для zeta.austin.ibm.com, который не будет сервером NFS, запустите демон **gssd** с помощью команды chnfs -S -B. Перед выполнением клиентских операций Kerberos пользователь должен получить разрешение с помощью команды **kinit**.

5. В этом сценарии настроена быстрая линия связи между alpha.austin.ibm.com и beta.austin.ibm.com. В этой линии beta.austin.ibm.com распознает alpha.austin.ibm.com как fast_alpha.test.austin.ibm.com, а alpha.autsin.ibm.com распознает beta.austin.ibm.com как fast_beta.test.austin.ibm.com. Так как ни nfs/fast_alpha.test.austin.ibm.com, ни nfs/fast_beta.test.austin.ibm.com не являются реальными субъектами, они не смогут использовать это соединение для монтирования.

   Для исправления этого следует использовать команду **nfshostmap**, которая подменит субъекты.

   a. В alpha.austin.ibm.com запускается следующая команда:

   ```
   nfshostmap -a beta.austin.ibm.com fast_beta.test.austin.ibm.com
   ```

   Она сообщает alpha.austin.ibm.com, что субъект fast_beta.test.austin.ibm.com соответствует beta.austin.ibm.com.

   b. В beta запускается следующая команда:

   ```
   nfshostmap -a alpha.austin.ibm.com fast_alpha.test.austin.ibm.com
   ```

   У серверов может быть несколько субъектов хоста. Допустим, IP-адрес fast_alpha - 10.0.0.1, а IP-адрес fast_beta - 10.0.0.2. Выполните следующие действия по добавлению нескольких субъектов хоста:

   a. Добавьте субъекты nfs/fast_alpha.test.austin.ibm.com и nfs/fast_beta.test.austin.ibm.com в соответствующие файлы ключей.

   b. Запустите команду **nfshostkey** на сервере alpha следующим образом:

   ```
   nfshostkey -a -p nfs/fast_alpha.test.austin.ibm.com -i 10.0.0.1
   ```

   c. Запустите команду **nfshostkey** на сервере beta следующим образом:

   ```
   nfshostkey -a -p nfs/fast_beta.test.austin.ibm.com -i 10.0.0.2
   ```

Отмена экспорта файловой системы NFS

Для отмены экспорта файловой системы NFS выполните одну из следующих процедур.

- Для того чтобы отменить экспорт каталога NFS с помощью SMIT, выполните следующие действия:
  1. Введите в командной строке следующую команду и нажмите Enter:

     ```
     smit rmnfsexp
     ```

  2. В поле Путь укажите имя экспортированного каталога.

     Имя каталога будет удалено из файла /etc/exports, а экспорт каталога будет отменен.

     Если каталог был экспортирован клиентам с помощью NFS версии 4, то возможно, что отменить экспорт не удастся из-за состояния файла на сервере. Состояние файла означает, что файлы в экспортированных каталогах открыты клиентом. Можно либо принять меры для того, чтобы прекратить использование этих данных приложениями, либо принудительно отменить экспорт (**exportfs -F**) данных, что может привести к сбою приложений, использующих эти данные.
- Для того чтобы отменить экспортирование каталога NFS с помощью текстового редактора, выполните следующие действия:

1. Откройте в любом текстовом редакторе файл /etc/exports.
2. Удалите из файла запись о каталоге, экспорт которого нужно отменить.
3. Сохраните файл /etc/exports и закройте его.
4. Если NFS в настоящее время работает, введите следующую команду:
   ```
   exportfs -u имя-каталога
   ```

   где *имя-каталога* - это полный путь к каталогу, который был только что удален из файла /etc/exports. Если отменить экспорт не удался из-за доступа клиентов NFS версии 4, то можно добавить опцию -F для принудительной отмены экспорта каталога.

Изменение экспортированной файловой системы

Экспортированную файловую систему NFS можно изменить одним из следующих способов.

- Для того чтобы изменить экспортированную файловую систему NFS с помощью SMIT, выполните следующие действия:
  1. Для отмена экспорта файловой системы введите:
     ```
     exportfs -u /имя-каталога
     ```

     где *имя-каталога* - имя изменяемой файловой системы.
  2. Введите:
     ```
     smit chnfsexp
     ```
  3. В поле Путь укажите полное имя ранее экспортированного каталога.
  4. Внесите все необходимые изменения.
  5. Завершите работу SMIT.
  6. Теперь снова экспортируйте данную файловую систему, введя следующую команду:
     ```
     exportfs /имя-каталога
     ```

     где *каталог* - это имя файловой системы, которую вы только что изменили.
- Для изменения экспортированной файловой системы NFS с помощью текстового редактора выполните следующие действия:
  1. Для отмена экспорта файловой системы введите:
     ```
     exportfs -u /dirname
     ```

     где *имя-каталога* - имя изменяемой файловой системы.
  2. Откройте в любом текстовом редакторе файл /etc/exports.
  3. Внесите все необходимые изменения.
  4. Сохраните и закройте файл /etc/exports.
  5. Теперь снова экспортируйте данную файловую систему, введя следующую команду:
     ```
     exportfs /имя-каталога
     ```

     где *каталог* - это имя файловой системы, которую вы только что изменили.

Доступ пользователя root к экспортированной файловой системе

У пользователя root по умолчанию нет доступа к экспортированной файловой системе.

При обращении пользователя root одного хоста к файлу другого хоста через NFS его идентификатор пользователя преобразуется системой NFS локального хоста в идентификатор пользователя nobody (имя пользователя nobody задано в файле /etc/password по умолчанию). Права доступа пользователя nobody совпадают с общими правами доступа к файлу (правами *прочих* пользователей). Например, если *общие права доступа* разрешают только запуск файла, пользователь nobody сможет только запустить файл.

Для того чтобы предоставить пользователю root доступ к экспортированной файловой системе, выполните инструкции из раздела “Изменение экспортированной файловой системы” на стр. 544. В случае экспорта с помощью SMIT укажите в поле ХОСТЫ имена тех хостов, пользователям root которых вы хотите предоставить доступ к файловой системе. При экспортировании с помощью текстового редактора добавьте к записи файловой системы флаг `-root=имя_хоста`. Пример:

```
/usr/tps -root=hermes
```

предоставляет пользователю root хоста `hermes` права доступа root к каталогу `/usr/tps`.

Монтирование файловой системы NFS вручную

Для монтирования файловой системы NFS выполните следующие действия:

1. Убедитесь в том, что каталог экспортирован сервером NFS:

   ```
   showmount -e имя-сервера
   ```

   где *имя-сервера* - имя сервера NFS. Эта команда показывает список всех каталогов, экспортированных с сервера NFS. Если каталог, который вы хотите смонтировать, отсутствует в этом списке, экспортируйте его с сервера.

   Примечание: Команда **showmount** не работает в файловых системах, экспортированных исключительно как файловые системы NFS версии 4. Для NFS версии 4 клиент может смонтировать корневую файловую систему для сервера и просматривать структуру экспортированного каталога. Отдельно экспортированные файловые системы не обязательно должны быть явно смонтированными, чтобы к ним могли получить доступ клиенты.
2. Создайте локальную точку монтирования командой **mkdir**. Для того чтобы каталог NFS был успешно смонтирован, необходимо заранее создать пустой каталог, играющий роль точки монтирования. Точка монтирования создается как обычный каталог, никакие специальные атрибуты указывать не нужно.
3. Введите:

   ```
   mount имя-сервера:/удаленный/каталог /локальный/каталог
   ```

   где *имя-сервера* - это имя сервера NFS, `/удаленный/`*каталог* - это монтируемый каталог сервера NFS, а `/локальный/`*каталог* - точка монтирования в системе клиента NFS.
4. В системе клиента введите следующую команду SMIT:

   ```
   smit mknfsmnt
   ```
5. Измените указанные ниже значения с учетом конфигурации вашей сети. В некоторых случаях нужно заполнить лишь некоторые поля.

   Примечание: При работе с интерфейсом SMIT для изменения значения поля нужно нажать клавишу Tab. *Не* нажимайте клавишу Enter до перехода к шагу 7.

   - Путь к точке монтирования.
   - Полное имя удаленного каталога.
   - Хост, на котором расположен удаленный каталог.
   - Смонтировать немедленно, добавить запись в файл `/etc/filesystems` или выполнить оба действия?
   - Если будет добавлена запись в файл `/etc/filesystems`, то каталог будет смонтирован при следующем запуске системы.
   - Режим работы файловой системы NFS.
6. В зависимости от конфигурации NFS задайте остальные параметры или оставьте для них значения по умолчанию.
7. После того, как вы внесете все необходимые изменения, инструмент SMIT смонтирует файловую систему NFS.
8. После того как в поле **Команда:** появится сообщение OK, завершите работу SMIT.

Файловая система NFS готова к работе.

Подсистема Automount

Подсистема **automount** позволяет пользователям, не имеющим прав root, монтировать удаленные файловые системы после указания начальных точек монтирования пользователем root.

Эта информация указывается в файле `/etc/auto_master`. Эти точки монтирования, также называемые ключами, имеют соответствующие отображения, определяющие, какая удаленная файловая система в них монтируется. Формат файла `/etc/auto_master` выглядит следующим образом:

`/ключ` *`отображение`*

Примечание: Система обращается к файлу `/etc/auto_master` при первоначальном запуске команды **automount**, поэтому сделанные изменения вступят в силу только при следующем выполнении команды **automount**.

Наиболее распространенными отображениями являются прямые отображения, косвенные отображения и отображения хостов.

Прямые отображения

Отображения хостов требуют наличия специального отображения (`/-`) в файле `/etc/auto_master`.

Отображение - этой файл в следующем формате:

`/directkey` `[-`*`опции`*`]` *`server`*`:`*`/dir`*

Когда пользователь обращается к каталогу */directkey*, демон **automount** смонтирует *server*:*/dir* поверх каталога *directkey*.

Косвенные отображения

Еще одним типом отображений, определяющим, какая файловая системе смонтирована в точке монтирования, являются косвенные отображения.

Косвенные отображения имеют следующий формат:

`indirectkey` `[-`*`опции`*`]` *`server`*`:`*`/dir`*

Когда пользователь обращается к каталогу `/key/`*indirectkey*, демон **automount** смонтирует *server*:*/dir* поверх `/key/`*indirectkey*.

Отображения хостов

Отображения хостов требуют наличия специального отображения (`-hosts`) в файле `/etc/auto_master`.

Демон **automount** создаст подкаталог в каталоге `/key` для каждого сервера, указанного в файле `/etc/hosts`. Когда пользователь обращается к каталогу `/key/server`, демон **automount** смонтирует экспортированные каталоги сервера поверх каталога `/key/server`.

Применение AutoFS для автоматического монтирования файловой системы

Расширение ядра **autoFS** применяет команду **automount** для получения информации о параметрах автоматического монтирования и запуска демона **automountd**.

В результате этой операции расширение ядра автоматически монтирует файловые системы вне зависимости от того, когда соответствующий файл или каталог будет открыт. Расширение ядра сообщает о запросах на монтирование демону **automountd**, который, в свою очередь, выполняет монтирование.

Поскольку взаимосвязь между именами и расположениями определяется демоном **automountd** автоматически, изменения, вносимые этим демоном в таблицу Службы информации о сети (NIS), прозрачны

для пользователя. Вам не требуется заранее монтировать файловые системы, на файлы и каталоги которых в программах есть явные ссылки, а также изменять соответствующие таким файловым системам записи.

AutoFS позволяет монтировать файловые системы по мере необходимости. Таким образом, в системе не требуется монтировать сразу все файловые системы; достаточно смонтировать только те из них, которые нужны в данный момент.

Например, для автоматического монтирования каталога NFS:

1. Убедитесь в том, что данный каталог был экспортирован сервером NFS. Для этого введите команду:

   ```
   showmount -e имя-сервера
   ```

   где *имя-сервера* - имя сервера NFS. Эта команда показывает список всех каталогов, экспортированных с сервера NFS.
2. Создайте главный файл и файл отображения **AutoFS**. **AutoFS** монтирует и размонтирует каталоги, указанные в этих файлах отображения. Допустим, вам необходимо, чтобы **AutoFS** смонтировал каталоги `/local/dir1` и `/local/dir2` сервера **serve1** в каталогах `/remote/dir1` и `/remote/dir2`, соответственно. Запись для файла `auto_master` будет выглядеть следующим образом:

   ```
   /remote  /tmp/mount.map
   ```

   Запись для файла `/tmp/mount.map` будет выглядеть следующим образом:

   ```
   dir1 -rw serve1:/local/dir1
   dir2 -rw serve1:/local/dir2
   ```
3. Убедитесь, что расширение ядра **AutoFS** загружено и демон **automountd** работает. Это можно сделать двумя способами:
   a. С помощью команды **automount**: Введите команду `/usr/bin/automount -v`.
   b. С помощью **SRC**: Введите команду `lssrc -s automountd`. Если подсистема **automountd** не запущена, введите команду `startsrc -s automountd`.

      Примечание: Запустив демон **automountd** с помощью команды **startsrc**, вы проигнорируете все сделанные ранее изменения в файле `auto_master`.
4. Для остановки демона **automount** вызовите команду `stopsrc -s automountd`.

   Если по какой-либо причине демон **automountd** был запущен без **SRC**, введите:

   ```
   kill automountd_PID
   ```

   где *automountd_PID* - это ИД процесса демона **automountd**. (ИД процесса демона **automountd** можно узнать с помощью команды **ps -e**.) Команда **kill** отправляет демону **automountd** сигнал SIGTERM.

Установка предопределенных монтирований NFS

Для создания предопределенных точек монтирования NFS выполните одну из следующих процедур.

Примечание: При настройке предопределенных каталогов, монтируемых во время загрузки, укажите опции **bg** (фоновый режим) и **intr** (разрешено прерывание) в файле `/etc/filesystems`. Процессы монтирования каталогов, работающие в интерактивном режиме, могут вызвать зависание клиента при загрузке, если в этот момент не работает сеть или сервер. Если клиент не может установить соединение с сетью или сервером, то необходимо перезапустить его в режиме обслуживания и исправить соответствующие параметры смонтированных каталогов.

- Для того чтобы задать предопределенные каталоги с помощью SMIT, выполните следующие действия:
  1. Введите:

     ```
     smit mknfsmnt
     ```
  2. В появившемся окне задайте значения для всех предопределенных каталогов. Заполните все обязательные поля (такие поля слева помечены звездочкой). Измените значения в остальных полях или оставьте значения по умолчанию. При данном способе установки в файл `/etc/filesystems` добавляется запись для необходимого каталога, а потом выполняется попытка монтирования.

- Для того чтобы задать в файле /etc/filesystems каталоги NFS, монтируемые по умолчанию, выполните следующие действия:
  1. Откройте файл /etc/filesystems в текстовом редакторе.
  2. Добавьте в файл записи обо всех удаленных файловых системах, которые должны монтироваться при запуске системы. Например:

     ```
     /home/jdoe:
     dev = /home/jdoe
     mount = false
     vfs = nfs
     nodename = mach2
     options = ro,soft
     type = nfs_mount
     ```

     Эта группа операторов означает, что система должна смонтировать удаленный каталог /home/jdoe поверх одноименной локальной точки монтирования. К файловой системе будет разрешен доступ только для чтения (ro). Поскольку она также имеет атрибут soft, то в случае, если сервер не отвечает, возникнет ошибка. Поскольку в параметре *type* указано значение nfs_mount, система пытается смонтировать файловую систему /home/jdoe (и все остальные файловые системы, указанные в группе операторов type = nfs_mount) при вызове команды **mount -t nfs_mount**.

     Ниже приведена последовательность операторов для монтирования файловой системы /usr/games во время запуска системы. Если монтирование будет завершено неудачно, то система будет повторять попытки в фоновом режиме.

     ```
     /usr/games:
     dev = /usr/games
     mount = true
     vfs = nfs
     nodename = gameserver
     options = ro,soft,bg
     type = nfs_mount
     ```

     Для команд, имеющих отношение к монтированию каталогов NFS, обязательно нужно задать следующие параметры:

Элемент	Описание
dev=*имя-файловой-системы*	Указывает путь к монтируемой удаленной файловой системе.
mount=[true|false]	Значение true указывает, что файловая система NFS монтируется при запуске системы. Значение false указывает, что файловую систему NFS не нужно монтировать при запуске системы.
nodename=*имя-хоста*	Указывает имя хоста, на котором расположена удаленная файловая система.
vfs=nfs	Указывает, что монтируемая виртуальная файловая система является файловой системой NFS.

Следующие параметры задавать в командах монтирования каталогов NFS не обязательно:

Элемент	Описание
type=*тип*	Указывает, что файловая система монтируется в составе группы монтирования *тип*. Этот параметр относится к команде **mount -t**, которая выполняет одновременное монтирование заданных файловых систем.
options=*опции*	Задает одну или несколько из перечисленных ниже *опций*: **biods=***N* Указывает максимальное число используемых демонов **biod**. Значение по умолчанию для NFS версии 2 - семь, а для NFS версий 3 и 4 - четыре. **bg** Означает, что в случае неудачного завершения первой попытки монтирования, вторая попытка будет повторена в фоновом режиме.

Элемент	Описание	
	fg	Означает, что в случае неудачного завершения первой попытки монтирования, вторая попытка будет повторена в интерактивном режиме.
	noacl	Отключает для данного монтируемого каталога поддержку журнализированной файловой системой NFS Списка управления доступом (ACL). При работе с двумя разными системами NFS обеспечивает поддержку списков управления доступом. Если же при монтировании файловой системы включить опцию `noacl`, то NFS не будет поддерживать списки ACL. Таким образом, опция `noacl` приводит к тому же результату, что и монтирование с сервера NFS, который не поддерживает списки ACL. За дополнительной информацией об ACL обратитесь к разделу “Поддержка списков управления доступом NFS” на стр. 515.
	retry=*n*	Задает количество попыток монтирования.
	rsize=*n*	Задает размер буфера чтения в байтах.
	wsize=*n*	Задает размер буфера записи в байтах.
	timeo=*n*	Задает значение тайм-аута NFS в десятых долях секунды. Этот параметр рекомендуется задавать при работе с сильно загруженным сервером, время ответа которого довольно велико.
	retrans=*n*	Здесь *n* - это время повторных передач NFS.
	port=*n*	Здесь *n* - номер порта сервера.
	soft	В том случае, если сервер не отвечает, возвращает сообщение об ошибке.
	hard	В том случае, если сервер не отвечает, продолжает повторять запросы. **Примечание:** Параметр монтирования `hard` может привести к зависанию процесса во время ожидания ответа от сервера. Для того чтобы этот процесс можно было прервать и завершить вручную, рекомендуется задавать для монтирования каталога переменную `intr`.
	intr	Разрешает прерывание сильного (hard) монтирования с клавиатуры.
	ro	Устанавливает значение переменной "только для чтения".

Элемент	Описание
	rw Устанавливает значение "чтение/запись". Это значение рекомендуется указывать только с режимом монтирования `hard`, так как монтирование `soft` с разрешением на чтение и запись может привести к конфликту между приложениями. Информация о возможных неполадках при слабом и сильном монтировании приведена в разделе “Устранение неполадок NFS” на стр. 559. **secure** Указывает, что для транзакций NFS необходимо использовать более защищенный протокол. **sec** Опция **sec** задает список разновидностей защиты для монтирования NFS. Доступны следующие разновидности: **des**, **unix**, **sys**, **krb5**, **krb5i** и **krb5p**. Это относится только к AIX 5.3 или более поздней версии. **actimeo=***n* Увеличивает время записи на диск содержимого буферов на *n* секунд (как для обычных файлов, так и для каталогов). **Примечание:** Атрибуты файлов сохраняются в кэше атрибутов на компьютере-клиенте. Файловым атрибутам присваивается время их удаления. Если один из файлов был изменен до времени сброса, то время сброса для него увеличивается на период времени, прошедший с момента последнего изменения (исходя из того, что недавно измененные файлы и в дальнейшем будут изменяться чаще). Существуют значения минимального и максимального увеличения времени сброса как для стандартных файлов, так и для каталогов. **vers** Указывает версию NFS. Значение по умолчанию - последняя версия протокола NFS, применяемая в соединении между клиентом и сервером и доступная в обеих системах. Если сервер NFS не поддерживает NFS версии 3, монтирование NFS будет использовать NFS версии 2. Опция **vers** позволяет выбрать версию NFS. По умолчанию монтирование NFS не будет использовать NFS версии 4, если только это не указано явно. **acregmin=***n* Сохраняет кэшированные атрибуты в течение минимум *n* секунд после изменения файла. **acregmax=***n* Указывает, что после изменения файла атрибуты могут храниться в кэше не более *n* секунд. **acdirmin=***n* Сохраняет кэшированные атрибуты в течение по крайней мере *n* секунд после обновления каталога. **acdirmax=***n* Указывает, что после изменения каталога атрибуты могут храниться в кэше не более *n* секунд.

Элемент	Описание

	cio Указывает, какая файловая система должна быть смонтирована для параллельных программ чтения и записи. Ввод-вывод в файлы этой файловой системы происходит таким образом, как будто они открыты с помощью **O_CIO**, указанной в вызове функции **open()**. Если эта опция задана, разрешен доступ к файлом только с помощью CIO. Применение кэшированного ввода-вывода в файловой системе, смонтированной с указанием опции **cio**, невозможно. Это означает, что команды преобразования (например, **mmap()** и **shmat()**) не выполняются с ошибкой **EINVAL** для всех файлов в файловой системе, смонтированной с указанием опции **cio**. Одним из побочных эффектов является то, что невозможно выполнять двоичные команды в файловой системе, смонтированной с указанием опции **cio**, поскольку загрузчик может использовать **mmap()**.
	dio Указывает, что ввод-вывод в файловой системе происходит таким образом, как будто все файлы этой системы открыты с помощью **O_DIRECT**, указанной в вызове функции **open()**.
	Примечание: Указание опций **-odio** или **-ocio** может увеличить производительность при выполнении определенных задач, однако пользователи должны быть предупреждены о том, что использование этих опций делает невозможным кэширование файлов в этих файловых системах. Поскольку упреждающее чтение в этих файловых системах отключено, быстродействие может снизиться при выполнении большого числа последовательных операций чтения.
	maxpout=*n*
	Указывает уровень выгрузки страниц для файлов данной файловой системы, на котором должны быть приостановлены нити. Если задано значение **maxpout**, следует также указать значение **minpout**. Значение данного параметра должно быть положительным и больше значения **minpout**. Значение по умолчанию для **maxpout** - уровень ядра.
	minpout=*n*
	Указывает уровень выгрузки страниц для файлов данной файловой системы, на котором следует считывать нити. Если задано значение **minpout**, следует также указать значение **maxpout**. Значение этого параметра должно быть неотрицательным. Значение по умолчанию для **minpout** - уровень ядра.
	rbr Возможность быстрого освобождения при чтении. При обнаружении последовательного чтения файла в этой файловой системе страницы физической памяти, которые используются файлом, освобождаются сразу после копирования во внутренние буферы.
	Примечание: По умолчанию ядро присваивает перечисленным ниже опциям следующие значения:
	`fg`
	`retry=10000`
	`rsize=8192`
	`wsize=8192`
	`timeo=7`
	`retrans=5`
	`port=NFS_PORT`
	`hard`
	`secure=off`
	`acregmin=3`
	`acregmax=60`
	`acdirmin=30`
	`acdirmax=60`

3. Удалите записи о каталогах, которые не нужно автоматически монтировать во время запуска системы.
4. Сохраните и закройте файл.
5. Вызовите команду **mount -a**, для того чтобы смонтировать все каталоги, указанные в файле `/etc/filesystems`.

Размонтирование смонтированной вручную или автоматически файловой системы

Данная процедура предназначена для того чтобы размонтировать каталог NFS, смонтированный вручную или автоматически.

Для того чтобы размонтировать каталог NFS, смонтированный вручную или автоматически, введите следующую команду:

```
p mount /каталог/для/размонтирования
```

Удаление предопределенных монтирований NFS

Для удаления предопределенных точек монтирования NFS выполните одну из следующих процедур.

- Для того чтобы удалить предопределенный каталог NFS с помощью SMIT, выполните следующие действия:
  1. Введите:
     ```
     smit rmnfsmnt
     ```
- Для того чтобы удалить из файла `/etc/filesystems` предопределенный каталог, монтируемый NFS, выполните следующие действия:
  1. Введите команду: `umount /каталог/для/размонтирования`.
  2. Откройте файл `/etc/filesystems` в любом текстовом редакторе.
  3. Найдите запись о каталоге, который был только что размонтирован, и удалите ее.
  4. Сохраните и закройте файл.

PC-NFS

PC-NFS - это программа, позволяющая монтировать на персональном компьютере каталоги, экспортированные с сервера NFS (Сетевой файловой системы).

Кроме того, сервер NFS может выполнять преобразование сетевых адресов и имен хостов. При условии, что на сервере NFS запущена программа-демон **rpc.pcnfsd**, он может предоставлять персональным компьютерам службы идентификации и буферизации печати.

Демон **rpc.pcnfsd** рекомендуется настроить на следующих системах:

- Системах, выполняющих проверку прав доступа пользователей
- Системах, выполняющих буферизацию печати
- На всех серверах Службы информации о сети (NIS) (как главных, так и подчиненных).

Примечание: Поскольку сети NIS обычно настроены таким образом, что PC-NFS может выбрать любой сервер NIS в качестве сервера по умолчанию, то необходимо, чтобы демон **rpc.pcnfsd** был запущен на всех серверах. Если вы считаете, что запускать данный демон на всех серверах NIS нецелесообразно, или хотите выделить определенный сервер для обработки запросов от клиентов, то добавьте в файл `autoexec.bat` каждого персонального компьютера команду **net pcnfsd**. После этого персональный компьютер будет вынужден обращаться только к указанному серверу NIS.

Информация, связанная с данной:

Службы информации о сети (NIS)

Служба идентификации PC-NFS

По умолчанию PC-NFS регистрируется на серверах NFS как пользователь `nobody`. При этом владельцем всех пользовательских файлов PC будет считаться пользователь `nobody`, и, следовательно, различать различных пользователей персонального компьютера будет невозможно.

Демон **rpc.pcnfsd** позволяет контролировать использование системных ресурсов и целостность защиты путем идентификации пользователей и предоставления им разных прав доступа.

Если демон **rpc.pcnfsd** запущен, то пользователь PC-NFS может вызвать команду **net name** и войти в систему PC-NFS так же, как он входит в операционную систему. Проверку имени пользователя и пароля выполняет демон **rpc.pcnfsd**. Подобная процедура идентификации не повышает защищенности сервера, но упрощает управление доступом к файлам NFS.

Служба буферизации печати PC-NFS

Служба буферизации печати демона **rpc.pcnfsd** позволяет персональным компьютерам с PC-NFS печатать на принтерах, которые не подключены к ним непосредственно.

PC-NFS перенаправляет файлы для принтеров персональных компьютеров в файл печати на сервере NFS. Этот файл помещается в каталог буферизации сервера NFS. Затем демон **rpc.pcnfsd** обращается к печатающему устройству сервера. (Каталог буферизации должен находиться в экспортированной файловой системе, чтобы клиенты PC-NFS могли монтировать их.) В запросе на печать файла, который программа PC-NFS отправляет демону **rpc.pcnfsd**, содержится следующая информация:

- Имя печатаемого файла
- ИД пользователя на данном клиенте
- Имя принтера, на котором должен быть напечатан файл.

Настройка демона rpc.pcnfsd

Для повышения производительности настройте демон **rpc.pcnfsd** следующим образом.

Для настройки демона **rpc.pcnfsd** выполните следующие действия:

1. Установите программу PC-NFS на персональном компьютере.
2. Выберите каталог буферизации на сервере NFS. По умолчанию в качестве каталога буферизации применяется `/var/tmp`. В этом каталоге должно быть как минимум 100 Кб свободного пространства.
3. Экспортируйте каталог буферизации. Во избежание неполадок в сети, устанавливать ограничения на доступ к экспортированному каталогу не рекомендуется. Дополнительная информация приведена в разделе “Экспорт файловой системы NFS” на стр. 540.
4. Запустите демон **rpc.pcnfsd**, выполнив инструкции из раздела “Запуск демона rpc.pcnfsd”.
5. Убедитесь в том, что демон **rpc.pcnfsd** доступен. Для этого выполните инструкции из раздела “Проверка доступности демона rpc.pcnfsd” на стр. 554.

Примечание: В результате выполнения запросов на перенаправление печати в каталогах буферизации PC-NFS остаются списки файлов нулевого размера. Рекомендуется время от времени удалять эти списки из каталогов буферизации.

Запуск демона rpc.pcnfsd

Для запуска демона **rpc.pcnfsd** с применением каталога буферизации по умолчанию выполните следующие действия.

1. Откройте файл `/etc/inetd.conf` в текстовом редакторе и удалите символы комментария из следующей записи:

   ```
   pcnfsd sunrpc_udp udp wait root /usr/sbin/rpc.pcnfsd pcnfsd 150001 1
   ```

2. Сохраните файл и закройте редактор.

Для запуска демона **rpc.pcnfsd** с применением каталога буферизации, отличного от заданного по умолчанию, выполните следующие действия:

1. Откройте файл `/etc/rc.nfs` в текстовом редакторе и добавьте в него следующую строку:

   ```
   if [ -f /usr/sbin/rpc.pcnfsd ] ; then
   /usr/sbin/rpc.pcnfsd -s каталог-буферизации ;
   echo ' rpc.pcnfsd\c'
   fi
   ```

   где *каталог-буферизации* - это полное имя каталога буферизации.

2. Сохраните файл и закройте редактор.
3. Откройте файл /etc/inetd.conf в текстовом редакторе и добавьте символ комментария (#) в начало следующей записи:

   ```
   #pcnfsd sunrpc_udp udp wait root /usr/sbin/rpc.pcnfsd pcnfsd 150001 1
   ```

   Теперь демон **rpc.pcnfsd**, запускаемый демоном **inetd**, не будет применять каталог буферизации по умолчанию.
4. Запустите программу буферизации **rpc.pcnfsd**, введя в командной строке следующую команду:

   ```
   /usr/sbin/rpc.pcnfsd -s каталог
   ```

   где *каталог-буферизации* - это полное имя каталога буферизации.

Дополнительная информация об обновлении базы данных конфигурации **inetd** приведена в разделе “Настройка демона inetd” на стр. 360.

Примечание: Каталог, используемый демоном **rpc.pcnfsd** по умолчанию, нельзя изменить в файле inetd.conf.

Проверка доступности демона rpc.pcnfsd

Выполните данную процедуру чтобы определить, доступен ли демон **rpc.pcnfsd**.

Для того чтобы убедиться, что демон **rpc.pcnfsd** доступен, введите следующую команду:

```
rpcinfo -u хост 150001
```

где *хост* - имя хоста, на котором выполняется настройка **rpc.pcnfsd**, а 15001 - номер RPC демона **rpc.pcnfsd**. После ввода команды должно появиться сообщение о том, что программа готова к работе.

Схемы автоматического монтирования LDAP

Подсистему автоматического монтирования можно настроить таким образом, чтобы она получала схемы с сервера LDAP.

Для настройки схем автоматического монтирования в LDAP добавьте в файл /etc/irs.conf следующую строку:

```
automount nis_ldap
```

Для управления схемами автоматического монтирования в LDAP следует создать соответствующие файлы LDIF. Файлы локальной схемы автоматического монтирования можно преобразовать в формат LDIF с помощью команды **nistoldif**. Допустим, например, что применяется сервер LDAP с именем ldapserver, тогда его базовый суффикс - dc=suffix, а файл /etc/auto_home содержит следующие строки:

```
user1   server1:/home/user1
user2   server1:/home/user2
user3   server1:/home/user3
```

Для создания файла LDIF /etc/auto_home и его добавления на сервер LDAP введите следующую команду:

```
nistoldif -d dc=suffix -sa -f /etc/auto_home > /tmp/auto_home.ldif
ldapadd -D cn=admin -w passwd -h ldapserver -f /tmp/auto_home.ldif
```

Для изменения или удаления существующих записей автоматического монтирования на сервере LDAP файлы LDIF необходимо создавать вручную. Например, если домашний каталог пользователя user2 находится на сервере server2, то необходимо создать следующий файл LDIF:

```
# cat /tmp/ch_user2.ldif
dn: automountKey=user2,automountMapName=auto_home,dc=suffix
changetype: modify
replace: automountInformation
automountInformation: server2:/home/user2
```

После создания приведенного выше файла LDIF запустите команду:

```
ldapmodify -D cn=admin -w passwd -h ldapserver -f
/tmp/ch_user2.ldif
```

Следует также создать файл LDIF для удаления пользователя. Например, для удаления пользователя `user3`, создайте следующий файл LDIF:

```
# cat /tmp/rm_user3.ldif
dn: automountKey=user3,automountMapName=auto_home,dc=suffix
changetype: delete
```

После создания приведенного выше файла LDIF запустите команду:

```
ldapmodify -D cn=admin -w passwd -h ldapserver -f
/tmp/rm_user3.ldif
```

WebNFS

Операционная система может выполнять функции сервера NFS для WebNFS.

Разработанный Oracle стандарт WebNFS представляет собой простое расширение протокола NFS, упрощающее доступ к серверам и клиентам через брандмауэры Internet.

Веб-браузер с поддержкой WebNFS может использовать адреса Internet (URL) NFS для непосредственной работы с данными на сервере. Ниже приведен пример URL NFS:

`nfs://www.`*`ваш-сайт`*`.com/`

Для обеспечения доступа к данным WebNFS взаимодействует с другими стандартными протоколами WWW.

WebNFS также использует преимущества масштабируемости серверов NFS.

Диспетчер сетевой блокировки

Диспетчер сетевой блокировки работает совместно с Сетевой файловой системой (NFS), поддерживая управляющий файл и блокировку записей сетевых баз данных, принятую в System V.

Диспетчер сетевой блокировки (**rpc.lockd**) и монитор состояния сети (**rpc.statd**) - это демоны сетевых служб. Демон **rpc.statd** - это процесс пользовательского уровня, а демон **rpc.lockd** реализован как набор нитей ядра (по аналогии с сервером NFS). Оба демона необходимы ядру для предоставления основных сетевых служб.

Примечание:

1. NFS не поддерживает обязательную и принудительную блокировку.
2. Диспетчер сетевой блокировки доступен только в NFS версии 2 и 3.

Архитектура диспетчера сетевой блокировки

Диспетчер сетевой блокировки сочетает в себе свойства как сервера, так и клиента.

Функции клиента выполняют обработку запросов от приложений и их передачу диспетчеру сетевой блокировки на сервере. Сервер отвечает за прием от клиентов запросов на блокировку и генерирует вызовы для блокировки. В частности, сервер отвечает на запросы клиентов о блокировке.

В отличие от системы NFS, не сохраняющей данные о состоянии, Диспетчер сетевой блокировки неявно хранит информацию о состоянии. Иначе говоря, Диспетчер сетевой блокировки должен знать, установлена ли клиентом блокировка. В мониторе состояния сети **rpc.statd** применен простой протокол, позволяющий Диспетчеру сетевой блокировки отслеживать состояние остальных компьютеров сети. Обладая подробной информацией о состоянии, Диспетчер сетевой блокировки может поддерживать согласованное состояние в среде NFS, которая сама по себе таким свойством не обладает.

Процесс блокировки сетевых файлов

Для того чтобы заблокировать локальный файл, приложение отправляет запрос ядру с помощью функций **lockf** , **fcntl** и **flock**.

Затем ядро обрабатывает запрос на блокировку. В том случае, если приложение на клиенте NFS запрашивает блокировку для удаленного файла, Диспетчер сетевой блокировки генерирует вызов удаленной процедуры (RPC) и отправляет его на сервер.

Когда клиент получает начальный запрос на удаленную блокировку, он с помощью демона **rpc.statd** регистрирует этот запрос на сервере. То же самое верно и для диспетчера сетевой блокировки на сервере. При получении начального запроса от клиента сервер зарегистрирует его на клиенте с помощью локального монитора состояния сети.

Процесс восстановления после сбоя

Каждый демон **rpc.statd** уведомляет демоны **rpc.statd** других систем о своей работе. Когда демон **rpc.statd** получает извещение о том, что на другом компьютере произошел сбой или выполняется процедура восстановления после сбоя, он уведомляет об этом демон **rpc.lockd**.

В случае выхода сервера из строя, клиенты с заблокированными файлами должны иметь возможность восстановить блокировку. В случае выхода из строя клиента, сервер должен сохранять блокировку файлов до восстановления клиента. Кроме того, в целях сохранения общей целостности системы NFS, сам процесс аварийного восстановления должен происходить без вмешательства приложений.

Процедура аварийного восстановления проста. При получении извещения о сбое клиентской системы сервер освобождает все блокировки, установленные этим клиентом, предполагая, что в случае необходимости приложение клиентской системы еще раз отправит запрос на блокировку. После того, как клиенту становится известно о сбое и восстановлении сервера, он заново передает запросы на блокировку для всех ранее заблокированных файлов. Вновь переданные серверу данные необходимы для восстановления за период отсрочки его состояния блокировки. (Период отсрочки, по умолчанию длящийся 45 секунд, - это время, в течение которого сервер ожидает от клиентов повторных запросов на блокировку.)

Программа-демон **rpc.statd** хранит имена хостов в файлах `/var/statmon/sm` и `/var/statmon/sm.bak`, позволяя отслеживать те хосты, которые нужно уведомлять во время восстановления клиента.

Запуск диспетчера сетевой блокировки

По умолчанию сценарий `/etc/rc.nfs` запускает демоны **rpc.lockd** и **rpc.statd** вместе с остальными демонами NFS.

Если служба NFS уже запущена, проверьте, работают ли демоны **rpc.lockd** и **rpc.statd**, выполнив инструкции из раздела “Просмотр текущего состояния демонов NFS” на стр. 525. Оба демона должны находиться в состоянии *активен*. Если демоны **rpc.lockd** и **rpc.statd** не активны, выполните следующие действия:

1. Откройте файл `/etc/rc.nfs` в любом текстовом редакторе.
2. Найдите следующие строки:

```
if [ -x /usr/sbin/rpc.statd ]; then
             startsrc -s rpc.statd
fi
if [ -x /usr/sbin/rpc.lockd ]; then
             startsrc -s rpc.lockd
fi
```

3. Если в начале каких-либо из этих строк есть знак #, удалите его. Затем сохраните и закройте файл. После этого запустите демоны **rpc.statd** и **rpc.lockd**, следуя инструкциям из раздела “Запуск демона NFS” на стр. 525.

   Примечание: Соблюдайте указанную последовательность. Первым всегда нужно запускать демон **statd**.

4. Если служба NFS запущена, и записи в файле /etc/rc.nfs не содержат ошибок, то остановите и снова перезапустите демоны **rpc.statd** и **rpc.lockd** согласно инструкциям из разделов “Остановка демонов NFS” на стр. 525 и “Запуск демона NFS” на стр. 525.

   Примечание: Соблюдайте указанную последовательность. Первым всегда нужно запускать демон **statd**.

Если вам не удалось запустить демоны **rpc.statd** и **rpc.lockd**, обратитесь к разделу “Устранение неполадок диспетчера сетевой блокировки”.

Устранение неполадок диспетчера сетевой блокировки

Некоторые неполадки диспетчера сетевой блокировки можно устранить с помощью следующих советов.

Если в системе клиента получено следующее сообщение:

```
clnttcp_create: RPC: Ошибка удаленной системы - Соединение отклонено
rpc.statd:нет связи с statd на {сервер}
```

то локальная система пытается информировать другую систему о процессе восстановления. При перезагрузке системы или перезапуске демонов **rpc.lockd** и **rpc.statd** имена систем перемещаются из файла /var/statmon/sm в файл /var/statmon/sm.bak. Демон **rpc.statd** пытается проинформировать все системы, указанные в файле /var/statmon/sm.bak, о необходимости восстановления.

Если демону **rpc.statd** удается подключиться к компьютеру, запись об этом компьютере удаляется из файла /var/statmon/sm.bak. Если демону **rpc.statd** не удается сразу подключиться к компьютеру, он повторяет попытки через фиксированный интервал времени. В каждом случае, если компьютер не отвечает, отправляется приведенное выше сообщение. В том случае, если необходима целостность блокировки, демон будет продолжать попытки; однако это может оказать неблагоприятное воздействие на время выполнения операций блокировки. Действия будут зависеть от состояния целевого компьютера (компьютер совсем не отвечает или время от времени переходит в рабочее состояние). Для того, чтобы это сообщение больше не появлялось, выполните следующие действия:

1. Убедитесь, что на сервере запущены демоны **statd** и **lockd**. Для этого выполните инструкции из раздела “Просмотр текущего состояния демонов NFS” на стр. 525. (Эти демоны должны быть *активны*.)
2. Если демоны **rpc.statd** и **rpc.lockd** не запущены, запустите их, выполнив инструкции из раздела “Запуск демона NFS” на стр. 525.

   Примечание: Соблюдайте указанную последовательность. Первым всегда нужно запускать демон **statd**.

   Помните, что после перезапуска демонов действует период отсрочки. Во время этого периода программы-демоны **lockd** получают повторные запросы на блокировку от клиентов, файлы которых ранее были блокированы сервером. Поэтому запрос на блокировку может быть выполнен лишь через некоторое время после запуска демонов.

Существует и другой способ избежать отправки сообщения:

1. Завершите работу демонов **rpc.statd** и **rpc.lockd** на клиенте, выполнив инструкции из раздела “Остановка демонов NFS” на стр. 525.
2. В системе клиента удалите запись о целевой системе из файла /var/statmon/sm.bak с помощью следующей команды:

   ```
   rm /var/statmon/sm.bak/ имя-целевой-системы
   ```

   В результате целевому компьютеру не будут поступать сообщения о необходимости восстановления блокировок. Такой способ может применяться только в том случае, если вы точно знаете, что на целевом компьютере не выполняются приложения, устанавливающие сетевые блокировки.
3. Запустите демоны **rpc.statd** и **rpc.lockd** на клиенте, выполнив инструкции из раздела “Запуск демона NFS” на стр. 525.

Если заблокировать файл для клиента не удается, выполните следующие действия:

1. Проверьте соединение между клиентом и сервером с помощью команды **ping**. Если оба компьютера включены, и сеть работает нормально, убедитесь, что в файле `/var/statmon/hosts` обоих компьютеров указаны имена соответствующих хостов. Для того чтобы клиент и сервер узнавали друг друга, необходимо, чтобы имена хостов в их записях точно совпадали. Если преобразование адресов хостов выполняет сервер имен, то убедитесь в том, что его данные в точности соответствуют данным файла `/var/statmon/hosts`.
2. Убедитесь, что на клиенте и сервере запущены демоны **rpc.lockd** и **rpc.statd**. Для этого выполните инструкции из раздела “Просмотр текущего состояния демонов NFS” на стр. 525. Оба демона должны находиться в состоянии *активен*.
3. Если демоны **rpc.statd** и **rpc.lockd** не активны, запустите их, выполнив инструкции из раздела “Запуск демона NFS” на стр. 525.
4. Если же демоны активны, рекомендуется перезапустить их как на сервере, так и на клиенте. Для этого завершите все приложения, запрашивающие блокировку.
5. Затем завершите работу демонов **rpc.statd** и **rpc.lockd** на клиенте и сервере, выполнив инструкции из раздела “Остановка демонов NFS” на стр. 525.
6. После этого перезапустите демоны **rpc.statd** и **rpc.lockd** на сервере, а затем на клиенте, выполнив инструкции из раздела “Запуск демона NFS” на стр. 525.

   Примечание: Соблюдайте указанную последовательность. Первым всегда нужно запускать демон **statd**.

Если после выполнения описанной процедуры ошибка не будет исправлена, запустите демон **lockd** в режиме отладки. Для этого выполните следующие действия:

1. Завершите работу демонов **rpc.statd** и **rpc.lockd** на клиенте и сервере, выполнив инструкции из раздела “Остановка демонов NFS” на стр. 525.
2. Запустите демон **rpc.statd** на клиенте и сервере, выполнив инструкции из раздела “Запуск демона NFS” на стр. 525.
3. Запустите демон **rpc.lockd** на клиенте и сервере, введя следующую команду:

   ```
   /usr/sbin/rpc.lockd -d1
   ```

   При вызове данной команды с флагом **-d1** демон **lockd** заносит в системный протокол диагностические сообщения. Сначала вы увидите серию сообщений, относящихся к операциям периода отсрочки. Подождите, пока они не закончатся. После того как период отсрочки закончится и на сервере, и на клиентах, запустите приложение, в котором возникли ошибки блокировки, и проверьте, передается ли запрос на блокировку от клиента серверу и обратно.

Можно ограничить диапазон IP-портов, используемых клиентом NFS для соединения с сервером NFS, добавив переменную **NFS_PORT_RANGE** в файл `/var/statmon/environment`.

Диапазон портов NFS

С помощью переменной среды **NFS_PORT_RANGE** можно ограничить исходный порт сетевых вызовов сервера клиентом.

Для использования этой переменной ее следует добавить в файл `/etc/environment`. Эта переменная среды должна быть задана следующим образом:

```
NFS_PORT_RANGE=udp[4000-5000]:tcp[7000-8000]
```

В этом примере исходный порт у отправляемых клиентом пакетов UDP будет в диапазоне от 4000 - 5000, а у соединений TCP - в диапазоне от 7000 - 8000. Для того чтобы избежать неполадок, связанных с повторным использованием портов, номера портов из этого диапазона не следует использовать в качестве фиксированных номеров портов в демонах сетевой файловой системы (NFS) из файла `/etc/services`.

Защита NFS

Информация о защите NFS содержится в нескольких источниках.

Сведения о защите DES приведена в главе Network File System security книги *Защита*. Сведения о защите Kerberos приведены в разделе “Настройка сети для RPCSEC-GSS” на стр. 541.

Устранение неполадок NFS

Как и при работе с другими сетевыми службами, в Сетевой файловой системе (NFS) могут возникнуть неполадки. Для их устранения необходимо: определить стратегию поиска неполадки NFS, проанализировать сообщения об ошибках, относящиеся к NFS и, наконец, определить требуемые действия по исправлению.

На этапе поиска неполадки NFS рекомендуется определить, на каком из трех возможных объектов возникла ошибка: на сервере, на клиенте или в самой сети.

Примечание: Сведения о неполадках блокировки файлов приведены в разделе “Устранение неполадок диспетчера сетевой блокировки” на стр. 557.

Неполадки при работе с сильно и слабо смонтированными файлами

В случае неполадок сервера и сети сбои программ, работающих с сильно и слабыми смонтированными удаленными файлами, проявляются по-разному.

Если сервер не отвечает на запрос о сильном монтировании файла, NFS выдает следующее сообщение:

```
Сервер имя-хоста NFS не отвечает, запрос будет повторен
```

Сильно смонтированные файловые системы приводят к зависанию программ до момента ответа сервера, поскольку клиент продолжает повторять запрос на монтирование до тех пор, пока он не будет принят. При сильном монтировании в команде **mount** рекомендуется указывать флаг **-bg**. В этом случае при сбое сервера клиент будет повторять запросы на монтирование в фоновом режиме.

В случае, если сервер не отвечает на запрос о слабом монтировании файла, NFS выдает следующее сообщение:

```
Тайм-аут соединения
```

Слабо смонтированные удаленные файловые системы после нескольких неудачных запросов прекращают попытки и возвращают сообщение об ошибке. К сожалению, многие программы не проверяют состояние после выполнения операций с файловой системой, поэтому при обращении к слабо смонтированным файлам вы не видите этих сообщений об ошибках. Однако на консоли будет показано сообщение об ошибке сервера NFS.

Определение неполадок NFS

Если вы столкнулись с неполадкой NFS, выполните следующие действия.

Если на клиенте NFS возникла ошибка, выполните следующие действия:

1. Проверьте правильность работы сетевых соединений.
2. Убедитесь, что на клиенте запущены демоны **inetd**, **portmap** и **biod**. Для этого выполните инструкции из раздела “Просмотр текущего состояния демонов NFS” на стр. 525.
3. Убедитесь в том, что для монтируемой файловой системы правильно задана точка монтирования. Дополнительная информация приведена в разделе “Настройка клиента NFS” на стр. 538.
4. Убедитесь, что сервер включен и работает. Для этого в командной строке клиента введите следующую команду:

   ```
   /usr/bin/rpcinfo  -p  имя-сервера
   ```

   Если сервер работает, то будет показан список используемых программ, версий, протоколов и номеров портов:

   ```
   программа  верс  прот   порт
   100000        2      tcp       111    portmapper
   100000        2      udp       111    portmapper
   ```

```
100005        1       udp       1025    mountd
100001        1       udp       1030    rstatd
100001        2       udp       1030    rstatd
100001        3       udp       1030    rstatd
100002        1       udp       1036    rusersd
100002        2       udp       1036    rusersd
100008        1       udp       1040    walld
100012        1       udp       1043    sprayd
100005        1       tcp        694    mountd
100003        2       udp       2049    nfs
100024        1       udp        713    status
100024        1       tcp        715    status
100021        1       tcp        716    nlockmgr
100021        1       udp        718    nlockmgr
100021        3       tcp        721    nlockmgr
100021        3       udp        723    nlockmgr
100020        1       udp        726    llockmgr
100020        1       tcp        728    llockmgr
100021        2       tcp        731    nlockmgr
```

Если подобный вывод не будет получен, войдите в систему сервера с консоли и проверьте состояние демона **inetd**, следуя инструкциям из раздела “Просмотр текущего состояния демонов NFS” на стр. 525.

5. Убедитесь в том, что демоны **mountd**, **portmap** и **nfsd** запущены на сервере NFS. Для этого в командной строке клиента введите следующую команду:

   ```
   /usr/bin/rpcinfo  -u  имя-сервера mount
   /usr/bin/rpcinfo  -u  имя-сервера portmap
   /usr/bin/rpcinfo  -u  имя-сервера nfs
   ```

   Если программы-демоны на сервере запущены, появятся такие сообщения:

   ```
   программа 100005  версии 1  готова к работе и ожидает
   программа 100000  версии 2  готова к работе и ожидает
   программа 100003  версии 2  готова к работе и ожидает
   ```

   Номера программ соответствуют командам, показанным в предыдущем примере. Если подобный вывод не будет получен, войдите в систему сервера с консоли и проверьте состояние демонов, выполнив инструкции из раздела “Просмотр текущего состояния демонов NFS” на стр. 525.

6. Убедитесь, что имя файловой системы, которую планирует смонтировать клиент, указано в файле сервера `/etc/exports`, а сама файловая система экспортирована. Для этого введите следующую команду:

   ```
   showmount -e имя-сервера
   ```

   Будет показан список файловых систем, экспортированных указанным *сервером*.

7. В случае NFS версии 4 убедитесь, что домен NFSv4 правильно настроен.
8. В случае NFS версии 4 убедитесь, что демон **nfsrgyd** запущен.
9. В случае использования расширенной защиты обратитесь к разделу “Определение неполадок RPCSEC-GSS” на стр. 566.

Ошибки асинхронной записи

Когда приложение отправляет запрос на запись в файл смонтированной файловой системы NFS, демон **biod** планирует асинхронную обработку операции записи.

Если в момент записи данных на диск на сервере NFS возникает ошибка, то клиенту NFS возвращается сообщение об ошибке, а демон **biod** сохраняет данные о возникшей ошибке во внутренних структурах данных NFS. Сохраненное сообщение об ошибке снова возвращается приложению при попытке вызвать функцию **fsync** или **close**. При подобных ошибках приложение не получит уведомления об ошибке записи до тех пор, пока не закроет файл. Типичный пример - ситуация, когда в файловой системе на сервере отсутствует свободное дисковое пространство, из-за чего клиенты не могут выполнить запись в файл.

Сообщение об ошибке nfs_server

Если буфер передачи слишком мал, появится сообщение об ошибке.

Недостаточное количество буферов передачи в сети может привести к следующей ошибке:

```
nfs_server:  bad  sendreply
```

Для увеличения числа буферов передачи выполните команду SMIT `smit commodev`. Затем выберите тип адаптера и увеличьте число буферов передачи.

Сообщения об ошибках команды mount

При монтировании файловой системы могут возникнуть сбои нескольких типов. Примеры сообщений об ошибках монтирования приведены ниже.

mount: ... уже смонтирован

Файловая система, которую вы хотите смонтировать уже была смонтирована ранее.

mount: ... не найден в /etc/filesystems

Указано неверное имя файловой системы или каталога.

Если в команде **mount** задано только имя каталога или только имя файловой системы, то за недостающими данными команда обращается к файлу `/etc/filesystems`, в котором находит запись для указанного имени файловой системы или каталога. Если команда **mount** обнаруживает следующую запись:

```
/dancer.src:
          dev=/usr/src
          nodename  =  d61server
          type  =  nfs
          mount  =  false
```

то она может выполнить монтирование так, как если бы все необходимые данные были заданы ей в командной строке:

```
/usr/sbin/mount -n dancer -o rw,hard /usr/src /dancer.src
```

... нет в базе данных хостов

Если в сети не применяется Служба информации о сети, то это сообщение означает, что имя хоста, указанное в команде **mount**, не найдено в файле `/etc/hosts`. Если в сети применяется NIS, это сообщение означает, что служба NIS не обнаружила имя хоста в базе данных `/etc/hosts`, или демон NIS **ypbind** не активен в локальной системе. Если файл `/etc/resolv.conf` существует, и за преобразование имен хостов отвечает сервер имен, то, скорее всего, в базе данных **named** содержится ошибка. См. раздел “Определение имени хоста на сервере NFS” на стр. 565.

Проверьте правильность написания и формат команды **mount**. Если команда была введена правильно, в сети отсутствует NIS, и подобное сообщение получено только для данного хоста, то проверьте соответствующую запись в файле `/etc/hosts`.

Если в сети применяется служба NIS, убедитесь, что в системе запущен демон **ypbind**, введя следующую команду:

```
ps  -ef
```

Если демон **ypbind** запущен, он должен быть показан в списке. С помощью команды **rlogin** попробуйте войти в удаленную систему, или попробуйте скопировать какой-либо файл на удаленный компьютер с помощью команды **rcp**. Если и эти попытки не увенчаются успехом, то это значит, что скорее всего демон **ypbind** завершил работу или завис.

Если это сообщение появляется только при работе с данным хостом, проверьте правильность записи `/etc/hosts` на сервере NIS.

mount: ... сервер не отвечает: ошибка отображения порта - тайм-аут RPC

Либо сервер, с которого выполняется монтирование, не работает, либо его программа отображения портов завершила работу или зависла. Перезапустите демоны **inetd**, **portmap** и **ypbind**, перезагрузив сервер.

Если сервер работает, но вы не можете подключиться к нему с помощью команды **rlogin**, проверьте правильность работы сетевого соединения, подключившись к какому-нибудь другому компьютеру. Кроме того, проверьте сетевое соединение сервера.

mount: ... сервер не отвечает: программа не зарегистрирована
Это означает, что команде **mount** удалось получить номера портов сервера, однако программа-демон монтирования NFS **rpc.mountd** не была зарегистрирована.

mount: доступ запрещен ...
В списке экспорта файловой системы, которую вы хотите смонтировать, отсутствует имя вашего компьютера.

Для того чтобы получить список файловых систем, экспортированных сервером, введите следующую команду:

```
showmount -e имя-хоста
```

Если требуемая файловая система отсутствует в списке, либо в списке пользователей этой файловой системы отсутствует имя вашего компьютера или имя группы, войдите в систему сервера и исправьте в файле `/etc/exports` запись об этой файловой системе. Если имя файловой системы указано в файле `/etc/exports`, но отсутствует в списке, который выводит на экран команда **showmount**, то в работе демона **mountd** возникла ошибка. Вероятно, демон либо не смог проанализировать данную строку в файле, ему не удалось найти каталог, или данный каталог не был смонтирован на этом компьютере. Если файл `/etc/exports` содержит правильную запись, и в сети применяется служба NIS, убедитесь, что на сервере активен демон **ypbind**. Возможно, его работа была завершена или он завис.

mount: ...: Доступ запрещен
Это сообщение указывает на ошибки одного из этапов идентификации на сервере. Например, имя вашего компьютера отсутствует в списке экспорта, либо сервер не идентифицировал демон **ypbind** вашей системы или предоставленный пользовательский профайл.

Проверьте правильность записей файла `/etc/exports` и убедитесь, что демон **ypbind** активен. Вы можете изменить имя хоста с помощью команды **hostname** и еще раз вызвать команду **mount**.

mount: ...: Не каталог
Путь к удаленному или локальному каталогу указан неверно (каталог отсутствует). Проверьте правильность введенной команды и попробуйте запустить ее в обоих каталогах.

mount: ...: У вас нет прав доступа
Для того чтобы запускать на своем компьютере команду **mount**, вы должны иметь права доступа root или быть членом группы system. Это связано с тем, что выполнение данной команды изменяет файловые системы всех пользователей этого компьютера. Монтирование и размонтирование каталогов NFS имеют право выполнять только пользователи root и члены системной группы.

Информация, связанная с данной:

Службы информации о сети (NIS)

Причины медленного доступа для NFS

Медленный доступ к удаленным файлам может быть связан с перегрузкой демона или плохой линией **tty**.

Сетевые соединения:

Для сбора информации о сетевых соединениях можно использовать команду **nfsstat**.

С помощью команды **nfsstat** можно узнать, отбрасываются ли некоторые пакеты. С помощью команд **nfsstat -c** и **nfsstat -s** определите, не выполняет ли сервер или клиент повторную передачу больших блоков. Из-за потери пакетов и перегрузки серверов всегда есть вероятность повторной передачи. Если более 5 % пакетов передаются повторно, то сетевые параметры настроены не оптимальным образом.

Для уменьшения вероятности повторной передачи следует изменить параметры очереди передачи адаптера связи. Изменить эти параметры можно с помощью меню программы SMIT. Дополнительная информация приведена в разделе Доступные интерфейсы управления системой в *Управление операционной системой и устройствами*.

Ниже приведены рекомендуемые значения для серверов NFS.

Примечание:

1. Если случаи повторной передачи будут продолжаться, установите для клиентов NFS приведенные выше значения.
2. Размер MTU должен быть одинаковым для всех узлов сети.

Таблица 91. Размер максимального блока передачи (MTU) и очереди передачи адаптера связи

Адаптер	MTU	Очередь передачи
Token-Ring		
4 МБ	1500 3900	50 40 (Увеличьте в случае тайм-аута команды **nfsstat**.)
16 МБ	1500 8500	40 (Увеличьте в случае тайм-аута команды **nfsstat**.) 40 (Увеличьте в случае тайм-аута команды **nfsstat**.)
Ethernet	1500	40 (Увеличьте в случае тайм-аута команды **nfsstat**.)

Чем больше размер MTU для каждого соединения Token-Ring, тем меньше нагрузка на процессор, и тем лучше выполняются операции чтения/записи.

Настройка размеров MTU:

Размер MTU можно задать с помощью команды быстрого доступа SMIT `smit chif`.

Выберите соответствующий адаптер и введите в поле Максимальный размер пакета IP значение размера MTU.

Для установки размера MTU можно также использовать команду **ifconfig** (ее *необходимо* использовать, если вы хотите задать значение размера MTU 8500). Команда **ifconfig** имеет следующий формат:

`ifconfig tr`*`n имя-узла`* `up mtu` *`размер-MTU`*

где `tr`*`n`* - имя адаптера, например, `tr0`.

Размер MTU можно изменить, воспользовавшись командой **ifconfig** в сочетании со SMIT.

1. Добавьте в файл `/etc/rc.bsdnet` команду **ifconfig** для соединений Token-Ring, как это показано в приведенном выше примере.
2. Введите команду быстрого доступа `smit setbootup_option`. Укажите в параметре **Тип BSD** значение **да**.

Размеры очереди передачи:

Размер очереди передачи адаптера связи устанавливаются с помощью SMIT.

Введите команду быстрого доступа `smit chgtok`, выберите соответствующий адаптер и введите значение размера очереди в поле Передача.

Зависание программ:

При зависании программ во время выполнения операций с файлами сервер NFS может перестать работать.

В этом случае появится следующее сообщение об ошибке:

```
Сервер NFS имя-хоста не отвечает, запрос будет повторен
```

Сервер NFS *имя-хоста* не работает. Подобное сообщение означает, что возникла ошибка либо на сервере NFS, либо в сети, либо на сервере NIS.

Если ваш компьютер полностью завис, проверьте серверы, с которых были смонтированы файловые системы. Если один или несколько из этих серверов не работают, то никаких действий предпринимать не нужно. Программы автоматически возобновят работу вместе с сервером. Файлы повреждены не будут.

Выход из строя сервера, с которого было произведено слабое монтирование, не повлияет на работу других программ. Если при обращении к слабо смонтированной файловой системе возникает тайм-аут, программа возвращает сообщение об ошибке с кодом `errno`. Однако доступ к другим файловым системам сохраняется.

Если все серверы работают нормально, выясните, не возникли ли неполадки при работе с этими же серверами у других клиентов. Если неполадки возникли и у других клиентов, то это указывает на ошибки в работе демонов **nfsd** на сервере. В этом случае войдите в систему сервера, вызовите команду **ps** и выясните, работает ли демон **nfsd** (потребляет ли он ресурсы CPU). Если демон **nfsd** не работает, перезапустите его. Если неполадку устранить не удалось, перезапустите сервер.

Если остальные системы включены и нормально работают, проверьте соединения вашего компьютера с сервером и с сетью в целом.

Схемы разрешений и идентификации:

Иногда после успешного монтирования каталогов возникают ошибки при чтении, записи или создании удаленных файлов и каталогов. Как правило, подобные сложности возникают из-за неполадок, связанных с правами доступа и идентификацией.

Причины неполадок, связанных с правами доступа и идентификацией, могут отличаться, в зависимости от того, используется ли система NIS, и выполняется ли защищенное монтирование.

Если применяется незащищенное монтирование и NIS не используется, то устранить ошибки достаточно просто. В этом случае идентификаторы пользователей (UID) и групп (GID) преобразуются с помощью файла сервера `/etc/passwd` и файла клиента `/etc/group`. Для того чтобы пользователь мог работать на клиенте и на сервере под именем `B`, в файлах `/etc/passwd` клиента и сервера для него должен быть задан одинаковый UID. Ниже показано, каким образом это может привести к неполадкам:

```
UID пользователя B в системе клиента
foo равен 200.
UID пользователя B в системе сервера bar равен 250.
UID пользователя G в системе сервера bar равен 200.
```

Каталог `/home/bar` смонтирован с сервера `bar` на клиенте `foo`. Если пользователь `B` изменит файлы удаленной файловой системы `/home/bar` на клиенте `foo`, то при сохранении этих файлов возникнет конфликт.

Сервер `bar` считает, что эти файлы принадлежат `пользователюG`, поскольку в системе `bar` идентификатор `200` соответствует пользователю `G`. Если пользователь `B` напрямую подключится к серверу `bar` с помощью команды **rlogin**, файлы, только что созданные им при работе через удаленную смонтированную файловую систему, могут оказаться недоступными. Однако пользователю `G` эти файлы будут доступны, поскольку права доступа предоставляются не по имени пользователя, а по его UID.

Для того чтобы исправить эту ошибку, следует заново задать правильные UID на обоих компьютерах. Присвойте пользователю B номер UID 200 на сервере bar или 250 на клиенте foo. Для файлов, принадлежащих пользователю B, необходимо вызвать команду **chown**, для того чтобы был изменен связанный с ними идентификатор пользователя.

Поскольку поддерживать согласованность UID и GID во всех системах сети вручную сложно, часто для выполнения всех необходимых преобразований применяется NIS.

Определение имени хоста на сервере NFS:

При получении запроса на монтирование сервер NFS определяет имя запрашивающего клиента. Сервер преобразует IP-адрес (адрес протокола Internet) запрашивающего клиента в имя хоста.

После определения имени хоста сервер обращается к списку экспорта запрашиваемого каталога и проверяет, разрешен ли данному хосту доступ к каталогу. Если запись для этого клиента имеется, и имя хоста указано правильно, это означает, что первый этап идентификации пройден успешно.

Если серверу не удается выполнить преобразование IP-адреса в имя хоста, то сервер отклоняет запрос на монтирование. Для выполнения запроса на монтирование серверу необходимо преобразовать IP-адрес клиента в соответствующее имя. Даже если доступ к экспортированному каталогу разрешен для всех клиентов, то для того, чтобы разрешить его монтирование, сервер должен выполнить обратное преобразование имени.

Также необходимо, чтобы сервер мог правильно выяснить имя клиента. Например, если в файле /etc/exports есть следующая запись:

```
/tmp      -access=silly:funny
```

В файле /etc/hosts ей соответствуют следующие записи:

```
150.102.23.21        silly.domain.name.com
150.102.23.52        funny.domain.name.com
```

Обратите внимание, что имена не совсем совпадают. Имена хостов silly и funny, найденные сервером в списке доступа к экспортированному каталогу, не совпадают с именами, преобразованными из IP-адресов хостов. Подобные трудности обычно возникают в случае, если преобразование имен выполняет демон **named**. Большинство баз данных демонов **named** содержат псевдонимы полных имен хостов домена, чтобы при обращении к хостам не требовалось вводить их полные имена. И хотя записи для преобразования имени хоста в IP-адрес существуют для всех псевдонимов, обратное преобразование не всегда можно выполнить. База данных обратного преобразования имен (IP-адреса в имя хоста) состоит из записей, содержащих IP-адрес и полное имя домена каждого хоста, но не его псевдоним. Иногда в записях списка экспортированных каталогов указывают сокращенные псевдонимы, что вызывает трудности при монтировании.

Ограничение числа групп в структуре NFS:

NFS версии 2 или 3 не позволяет пользователю быть членом более 16 групп. В противном случае возникают неполадки.

Группы задаются командой **groups**. Пользователь, входящий в 17 или более групп, не сможет получить доступ к файлам, принадлежащим семнадцатой (или большей по счету) группы. Для того чтобы пользователь смог получить к ним доступ, нумерацию групп придется изменить.

Выше описано поведение по умолчанию. Дополнительная информация приведена в описании параметра **maxgroups** команды **mount**.

Серверы NFS с ранними версиями NFS:

Клиент NFS версии 3 не может монтировать на сервере NFS версии 4.

В случае, если каталог NFS монтируется клиентом с с сервера версии 3 (или более ранней) и пользователь входит более чем в 8 групп, могут возникать ошибки. Некоторые серверы вообще не справляются с подобной задачей и отклоняют запрос на монтирование. Для того чтобы обойти это ограничение, нужно изменить число групп пользователя так, чтобы оно было меньше 8, а затем повторить попытку монтирования. Обычно при подобных неполадках появляется следующее сообщение об ошибке:

```
RPC:    Ошибка идентификации; why=Неправильное одноразовое разрешение клиента
```

Определение неполадок RPCSEC-GSS:

При возникновении неполадок RPCSEC-GSS обратите внимание на следующие решения.

- С помощью команды **klist**, запущенной на клиенте, убедитесь в том, что у вас имеются действительные идентификационные данные.
- Убедитесь в том, что время клиента, сервера и KDC синхронизировано. Для проверки синхронизации всей области Kerberos воспользуйтесь NTP или его аналогом.
- Убедитесь, что на сервере есть действительный файл `keytab` и субъект хоста. Если выполнить следующую команду не удастся, это означает, что сервер работать не будет:

  ```
  kinit -kt 'tail -n 1 /etc/nfs/hostkey' 'head -n 1 /etc/nfs/hostkey'
  ```
- Убедитесь с помощью следующей команды в том, что демон **gssd** запущен и откликается на запросы клиента и сервера:

  ```
  rpcinfo -u localhost 400234
  ```

  Если демон **gssd** не откликается, произойдет ошибка RPCSEC-GSS; это можно исправить, остановив и вновь запустив демон **gssd**.
- Если вы получаете ошибки записи из-за нарушения целостности или секретности, убедитесь, что вы используете модуль ядра. Целостность и секретность не поддерживаются без модуля ядра. (Модуль ядра - это модуль ядра Kerberos, `/usr/lib/drivers/nfs.ext`. Он устанавливается вместе с набором файлов `modcrypt.base` из пакета расширения.)
- В случае, если отдельные пользователи будут сталкиваться с запретами на доступ к данным, к которым они должны иметь доступ, проверьте правильность синхронизации соответствующих субъектов в KDC с учетной записью пользователя AIX.
- Активизируйте системный протокол. Большинство ошибок RPCSEC-GSS будут записаны в протокол. Код ошибки состоит из двух частей: первая часть - это код ошибки GSS (подробное описание содержится в RFC 2744), а вторая часть - это код ошибки Kerberos.

  Примечание: Активизация системного протокола может снизить производительность системы, поэтому после устранения неполадки следует отключить системный протокол.
  Ниже приведены некоторые распространенные коды ошибок и способы их устранения:

  KRB5_CC_NOTFOUND
  : Действительные идентификационные данные Kerberos не обнаружены. Это можно исправить с помощью команды **kinit**.

  KRB5_KDC_UNREACH
  : KDC недоступен. Убедитесь в том, что KDC запущен и в соединении между клиентом или сервером и KDC нет сетевых неполадок.

  KRB5_KT_NOTFOUND
  : Запись `keytab` для субъекта сервера не обнаружена. С помощью команды **nfshostkey -l** убедитесь, что вы используете правильный субъект (это должно быть `nfs/`*<полное имя домена>*) и файл `keytab`. С помощью команды **klist -ke** проверьте файл `keytab` на сервере на наличие соответствующей записи.

KRB5KRB_AP_ERR_TKT_NYV
Скорее всего, неполадка синхронизации.

KRB5KRB_AP_WRONG_PRINC и KRB5KDC_ERR_S_PRINCIPAL_UNKNOWN
Эти сообщения об ошибках указывают на то, что используемый клиентом субъект не совпадает с субъектом хоста сервера.

KRB5KRB_AP_WRONG_PRINC
Указывает, что клиент успешно преобразовал имя хоста сервера в существующий субъект в форме `nfs/<полное имя домена>`, но субъект хоста сервера не соответствует этому субъекту.

KRB5KDC_ERR_S_PRINCIPAL_UNKNOWN
Указывает, что клиент не смог преобразовать имя хоста сервера в существующий субъект. С помощью команды **nfshostkey -l** проверьте, есть ли у сервера правильный субъект. Если есть, возможно, таблица преобразований имен хостов нуждается в обновлении; подробные сведения см. в описании команды **nfshostmap** в *Справочник по командам, том 4*.

Определение неполадок EIM:

При устранении неполадок EIM обратите внимание на следующие советы.

При возникновении неполадок EIM обратите внимание на следующее:

- Если с помощью команды **nfsrgyd** или **chnfsim** не удалось соединиться с сервером EIM LDAP, то убедитесь, что процесс **ibmslapd** запущен на сервере EIM LDAP, с помощью следующей команды:
  ```
  ps -ef | grep ibmslapd
  ```
  Если процесс **ibmslapd** не запущен, введите следующую команду для его активации:
  ```
  /usr/sbin/ibmslapd
  ```
- Если с помощью команды **nfsrgyd** или **chnfsim** удалось соединиться с сервером EIM LDAP, но не удалось выполнить операции преобразования идентификаторов, то убедитесь, что процесс **ibmslapd** запущен не в режиме настройки. Это может случиться, если база данных **ldapdb2** не активна в момент запуска сервера **ibmslapd**. Выполните следующие действия:
  1. Войдите в систему сервера EIM LDAP как пользователь root.
  2. Просмотрите содержимое файла `/var/ldap/ibmslapd.log`. Выясните, когда в последний раз запускался процесс **ibmslapd**. Также выясните, не был ли сервер запущен в режиме настройки, так как он не может соединиться с базой данных **ldapdb2**.

  Если серверу не удалось подключиться к базе данных **ldapdb2**, то следует перезапустить базу данных. Для запуска базы данных **ldapdb2** выполните следующие действия:
  1. Войдите в систему сервера EIM LDAP как пользователь root.
  2. Введите следующую команду для проверки состояния процесса **ibmslapd**:
     ```
     ps -ef | grep ibmslapd
     ```
     Если процесс активен, отключите его с помощью следующей команды:
     ```
     kill ibmslapd
     pid
     ```
     где *pid* - это ИД процесса, выданный командой **ps -ef**.
  3. После отключения процесса **ibmslapd** запустите базу данных **ldapdb2**:
     a. Войдите в систему сервера EIM LDAP как пользователь **ldapdb2**.
     b. Введите `db2start`.
  4. После запуска базы данных **ldapdb2** активизируйте процесс **ibmslapd**:
     a. Войдите в систему сервера EIM LDAP как пользователь root.
     b. Введите `ibmslapd`.

Неполадки, возникающие если расширение ядра NFS не загружено.:

Некоторые команды NFS выполняются неправильно, если не загружено расширение ядра NFS. К числу таких команд относятся команды **nfsstat**, **exportfs**, **mountd**, **nfsd** и **biod**.

При установке службы NFS в системе расширение ядра помещается в файл `/usr/lib/drivers/nfs.ext`. Затем, после настройки системы, этот файл загружается в качестве расширения ядра NFS. Файл `/etc/rc.net` загружается специальным сценарием. Этот сценарий выполняет множество других функций, в частности загрузку расширения ядра NFS. Важно учесть, что расширение ядра **TCP/IP (Протокола управления передачей/протокола Internet)** и файл **nfs_kdes_null.ext** необходимо загружать перед расширением ядра NFS.

Примечание: Для загрузки расширения NFS в ядро во время начальной загрузки системы применяется команда **gfsinstall**. Эту команду можно выполнять в процессе загрузки системы несколько раз. Команда **gfsinstall** по умолчанию указана в обоих файлах `/etc/rc.net` и `/etc/rc.nfs`. Отменять какой-либо из этих вызовов не нужно.

Неполадки, возникающие в случае, если поддержка Kerberos не установлена:

Если поддержка kerberos не установлена, запустить демон **gssd** не удастся.

Убедитесь, что наборы файлов `krb5.client.rte` и `modcrypt.base` установлены. Если хотя бы один из них не установлен, демон **gssd** запустить не удастся.

Что необходимо проверить, если демон реестра не запускается:

Запустить демон **nfsrgyd** невозможно, если домен NFS версии 4 не настроен.

Информация о настройке домена NFS версии 4 приведена в разделе “Файл /etc/nfs/local_domain” на стр. 522.

Файлы NFS

Здесь приведены сведения о файлах NFS и их описания.

Элемент	Описание
`bootparams`	Содержит список компьютеров, с которых могут загружаться бездисковые клиенты.
`exports`	Содержит список каталогов, которые могут быть экспортированы для клиентов NFS.
`filesystems`	Содержит список всех файловых систем, которые монтируются автоматически при перезагрузке системы.
`hostkey`	Указывает субъекты хоста Kerberos и расположение файла `keytab`.
`local_domain`	В этом файле содержится локальный домен NFS системы.
`networks`	Содержит данные о сетях в составе сети Internet.
`pcnfsd.conf`	Содержит параметры настройки программы-демона **rpc.pcnfsd**.
`princmap`	Преобразует имена хостов в субъекты Kerberos, когда субъект не является полным именем домена сервера.
`realm.map`	Этот файл используется демоном реестра NFS для преобразования входящих субъектов Kerberos.
`rpc`	Содержит базу данных по программам протокола RPC (Вызов удаленных процедур).
`security_default`	Содержит параметры по умолчанию для защиты NFS.
`xtab`	Содержит список экспортированных в настоящее время каталогов.

Команды NFS

Здесь приведены сведения о командах NFS и их описания.

Элемент	Описание
chnfs	Запускает заданное количество программ-демонов **biod** и **nfsd**.
chnfsdom	Изменяет локальный домен NFS.
chnfsim	Изменяет преобразование внешних идентификаторов NFS.
chnfssec	Изменяет тип защиты по умолчанию, используемый клиентом NFS
chnfsrtd	Изменяет преобразования области в домен в локальной NFS.
mknfs	Команда настройки NFS и запуска демонов NFS.
nfso	Задает сетевые параметры NFS.
automount	Автоматически монтирует файловую систему NFS.
chnfsexp	Изменяет атрибуты экспортированного каталога NFS.
chnfsmnt	Изменяет атрибуты смонтированного каталога NFS.
exportfs	Экспортирует (и отменяет экспорт) каталогов для клиентов NFS.
lsnfsexp	Показывает параметры экспортированных каталогов NFS.
lsnfsmnt	Показывает характеристики смонтированных файловых систем NFS.
mknfsexp	Экспортирует каталог NFS.
mknfsmnt	Монтирует каталог NFS.
nfshostkey	Настраивает ключ хоста для сервера NFS.
nfs4cl	Показывает информацию о файловых системах, к которым обращается клиент с помощью NFS версии 4.
nfs4smctl	Управляет аннулированием состояния NFS версии 4.
rmnfs	Завершает работу программ-демонов NFS.
rmnfsexp	Удаляет экспортированные каталоги NFS из списка экспорта сервера.
rmnfsmnt	Удаляет смонтированные файловые системы NFS из списка смонтированных каталогов клиента.

Демоны NFS

Здесь приведены сведения о демонах NFS и их описания.

Демоны блокировки

Элемент	Описание
lockd	Обрабатывает запросы на блокировку с помощью пакета RPC.
statd	Обеспечивает функции восстановления после сбоев службы блокировки NFS.

Демоны и утилиты сетевой службы

Элемент	Описание
biod	Передает запросы на чтение/запись от клиента серверу.
mountd	Отвечает на запросы клиентов о монтировании файловых систем.
nfsrgyd	Выполняет преобразование между субъектами защиты, строками идентификаторов NFS версии 4 и их соответствующими числовыми ИД. Кроме того, предлагается преобразование информации идентификаторов из внешних доменов NFS версии 4.
nfsd	Запускает демоны, обрабатывающие запросы клиента на выполнение операций над файловыми системами.
nfsstat	Выдает информацию о том, может ли указанная система принимать запросы.
on	Запускает команды на удаленных компьютерах.
pcnfsd	Обрабатывает запросы от клиентов PC-NFS.
portmap	Связывает номера программ RPC с номерами портов Internet.
rexd	Принимает запросы на запуск программ на удаленных компьютерах.
rpcgen	Создает код на языке C для реализации протокола RPC.
rpcinfo	Создает отчет о состоянии серверов RPC.
rstatd	Показывает статистические данные о производительности ядра.
rup	Показывает данные о состоянии удаленного хоста в локальной сети.
rusers	Создает список пользователей, подключенных к удаленным компьютерам.
rusersd	Отвечает на запросы команды **rusers**.
rwall	Отправляет сообщения всем пользователям в сети.
rwalld	Обрабатывает запросы команды **rwall**.
showmount	Показывает список всех клиентов, на которых смонтированы удаленные файловые системы.
spray	Отправляет на хост указанное количество пакетов.
sprayd	Принимает пакеты, переданные командой **spray**.

Утилиты и демоны сетевой защиты

Элемент	Описание
chkey	Изменяет ключ шифрования пользователя.
gssd	Обеспечивает доступ NFS к службам защиты, предоставляемым Службами сетевой идентификации.
keyenvoy	Выполняет функции посредника между пользовательскими процессами и сервером ключей.
keylogin	Расшифровывает и сохраняет секретный ключ пользователя.
keyserv	Хранит общие и частные ключи.
mkkeyserv	Запускает демон **keyserv** и удаляет метки комментариев из соответствующих записей в файле `/etc/rc.nfs`.
newkey	Создает новый ключ в файле `publickey`.
rmkeyserv	Завершает работу демона **keyserv** и ставит метки комментариев в соответствующих записях в файле `/etc/rc.nfs`.
ypupdated	Обновляет информацию в базах данных Службы информации о сети (NIS).

Более подробная информация о средствах защиты NFS приведена в разделе Защита сетевой файловой системы (NFS) in *Защита*.

Поддержка бездисковых клиентов Sun

Элемент	Описание
bootparamd	Предоставляет бездисковым клиентам информацию, которая требуется для их загрузки.

Функции NFS

Здесь описаны функции NFS.

Элемент	Описание
cbc_crypt , **des_setparity** или **ecb_crypt**	Реализует процедуры Стандарта шифрования данных (DES).

Файловая система протокола SMB

Файловая система протокола SMB (SMBFS) позволяет обращаться к общим каталогам на серверах SMB как к локальным файловым системам в AIX.

В такой файловой системе пользователь может создавать, удалять, считывать и записывать файлы, а также изменять время обращения к файлам и каталогам. Владельца и режим доступа к файлам и каталогам изменить нельзя.

SMBFS позволяет работать с файлами на сервере SMB. В роли сервера SMB может выступать сервер, на котором работает Samba либо сервер или рабочая станция Windows XP, Windows NT или Windows 2000. Все эти серверы позволяют экспортировать каталог как общий ресурс. Этот общий каталог можно смонтировать в системе AIX с помощью SMBFS.

Установка SMBFS

Для того чтобы установить SMBFS в системе AIX, установите пакет `bos.cifs_fs`.

После установки пакета `bos.cifs_fs` будет создано устройство `nsmb0`. Это устройство позволяет с помощью команды **mount** создавать соединения между сервером SMB и клиентом.

Монтирование SMBFS

Файловую систему протоколов SMB (SMBFS) можно смонтировать одним из двух способов.

Это можно сделать с помощью команды AIX **mount**. Например:

```
mount -v cifs -n pezman/user1/pass1 -o uid=201,fmode=750 /home /mnt
```

Дополнительные сведения о команде **mount**, включая описание флагов, приведены в описании команды **mount** в книге *Справочник по командам, том 3*.

Можно указать параметры монтирования с помощью флага
`-o`

. Опции в командной строке следует разделять только запятой, а не запятой с пробелом. Предусмотрены следующие опции для файловой системы:

Элемент	Описание
`fmode`	Устанавливает восьмеричный режим для файла. Значение по умолчанию - 755.
`uid`	Присваивает UID файлам во время монтирования. Значение по умолчанию - **root**.
`gid`	Присваивает GID файлам во время монтирования. Значение по умолчанию - **system**.
`wrkgrp`	Рабочая группа, к которой принадлежит сервер SMB.
`op`	Устанавливает значение 1 при использовании доверительной блокировки. Устанавливает значение 0, если доверительная блокировка не используется.
`opfs`	Имя кэша, используемого файловой системой для хранения файлов кэша блокировки.
`opsz`	Размер отдельных файлов кэша, используемых для доверительной блокировки.
`opfssz`	Размер кэша файловой системы для доверительной блокировки.

Кроме того, смонтировать файловую систему можно с помощью инструмента SMIT `smit cifs_fs`, который запускает команду **mount**, собрав всю необходимую информацию.

Для монтирования файловой системы SMBFS необходимо ввести имя пользователя и пароль для идентификации на сервере. Эти имя пользователя и пароль будут применяться при выполнении на сервере всех необходимый операций с файлами. Поле **Пароль** в меню SMIT не помечено как обязательное. Если пароль в поле не указан, будет выполнен поиск в файле `cifscred` на предмет наличия идентификационных данных пользователя или сервера, совпадающих с предоставленными. При обнаружении совпадения будет использоваться сохраненный пароль из файла `cifscred`, иначе пользователю будет предложено ввести пароль с помощью стандартного приглашения AIX. Таким образом пользователь может ввести пароль, не раскрывая его.

Примечание: Длина пароля, применяемого для монтирования SMBFS, не должна превышать 14 символов; в пароле допустимы специальные символы.

При активации команды файловой системы (например, команды чтения) для файла в точке монтирования SMBFS на сервер отправляется запрос на чтение файла. Вместе с запросом отправляются имя пользователя и пароль, по которым сервер определяет наличие у пользователя прав на чтение данного файла на сервере. Таким образом, решение о предоставления доступа к файлу остается за сервером.

Однако опция `fmode` команды **mount** позволяет пользователю root в системе клиента управлять доступом к файлам на сервере до того, как запрос будет отправлен на сервер. Если опция `fmode` не задана, то по умолчанию применяется значение `755`. В следующей таблице продемонстрировано действие опции `fmode` для запроса на запись:

Таблица 92. Пять примеров, в которых пользователям был предоставлен или запрещен доступ, в зависимости от предоставленных прав доступа.

Номер примера	Пользователь, идентифицированный сервером	Пользователь клиента, желающий получить доступ для записи	Владелец, группа и режим монтирования	Владелец, группа и режим на сервере	Доступ предоставлен
Пример 1	user1	user2	user1, staff rwxr-xr-x	user1, staff rwxrwxr-x	нет
Пример 2	user1	root	user1, staff rwxr-xr-x	user2, staff rwxr-xr-x	нет
Пример 3	user1	user1	user1, staff rwxr-xr-x	user2, staff rwxrwxr-x	да
Пример 4	user1	user1	user, staff rwxr-xr-x	root, system rwx------	нет
Пример 5	user1	user1	user1, staff rwxr-xr-x	root, system rwxrwxrwx	да

В первом примере доступ запрещен, так как владелец, группа и режим в точке монтирования на клиенте не предоставили доступ пользователю `user2`.

Во втором примере доступ был запрещен. Хотя пользователь `root` и обладает доступом ко всем объектам в системе клиента, идентифицированный сервером пользователь `user1` не имеет прав доступа к файлу на сервере.

В третьем примере доступ был предоставлен, так как пользователь `user1` был владельцем точки монтирования, а пользователь `user1` из группы `staff` на сервере обладал правами доступа к файлу на этом сервере.

В четвертом примере доступ был запрещен, так как, хотя пользователь `user1` и являлся владельцем точки монтирования, файл на сервере принадлежал пользователю `root`, и у других пользователей и групп прав доступа к этому файлу не было.

В пятом примере доступ был предоставлен, так как пользователь `user1` был владельцем точки монтирования, и пользователь `user1` на сервере получил доступ к файлу через другие права доступа.

Примечания:

1. В смонтированной файловой системе операция копирования одного файла в другой выполняется успешно для файла, размер которого равен 4 ГБ + 4096 байт или менее. Для файлов, размеры которых превышают указанный, выдается предупреждающее сообщение, и в целевую точку копируются 4 ГБ + 4096 байт исходного файла.
2. В смонтированной файловой системе в именах файлов недопустимы следующие символы: : обратная косая черта {\}, косая черта {/}, двоеточие {:}, звездочка {*}, знак вопроса {?}, меньше {< }, больше {>}, вертикальная черта { | }.

Сохраненные пароли

SMBFS может сохранять идентификационные данные `server/user/password` в файле `/etc/cifs_fs/cifscred` для автоматического извлечения паролей при монтировании SMBFS.

Идентификационные данные можно добавлять, изменять и удалять из этого файла с помощью команд **mkcifscred**, **chcifscred** и **rmcifscred** (находящихся в файле `/usr/sbin`). Добавляемые в этот файл пароли шифруются. При попытке выполнения монтирования без указания пароля, происходит поиск соответствующих идентификационных данных в файле `cifscred`. При обнаружении совпадения будет использоваться сохраненный пароль из файла `cifscred`, иначе пользователю будет предложено ввести пароль с помощью стандартного приглашения AIX.

Поддержка сохранения паролей обладает следующими ограничениями:

- Для надлежащего функционирования извлечения сохраненных паролей следует иметь согласованное соглашение об именах сервера. Например, если идентификационные данные добавляются с помощью IP-адреса, а не имени хоста или полного имени домена (FQDN), пароли будут извлекаться только при монтировании по IP-адресу.
- Идентификация с незашифрованными паролями при использовании метода извлечения сохраненных паролей не поддерживается. Если серверу требуется применение незашифрованных паролей, произойдет сбой идентификации.

Поддержка /etc/filesystems

SMBFS поддерживает `/etc/filesystems` для автоматического монтирования при запуске системы.

Поддержка `/etc/filesystems` также обеспечивает доступ при монтировании к сохраненным данным сервера, пользователя, паролю и опций. С помощью команд **mkcifsmnt**, **chcifsmnt**, **rmcifsmnt** и **lscifsmnt** (расположенных в каталоге `/usr/sbin`) можно добавить, изменить, удалить и просмотреть разделы cifs в `/etc/filesystems`. Идентификационные данные следует хранить в файле `cifscred`.

Устранение неполадок SMBFS

Выполните данные действия при обнаружении неполадок в SMBFS.

Если команда **mount** или команда быстрого доступа `smit cifs_fs` возвратит сообщение об ошибке, выполните следующие действия:

1. Проверьте правильность имени пользователя и пароля. Имя пользователя и пароль необходимы для предоставления доступа к общему каталогу на сервере.
2. Проверьте правильность имени сервера. Если имя сервера указано верно, то укажите полное имя хоста в случае, если сервер и клиент находятся в разных подсетях. Кроме того, можно задать IP-адрес сервера.
3. Убедитесь в том, что команда `lsdev -L|grep nsmb` возвращает имя устройства. Если устройство `nsmb` недоступно, то клиент AIX не может установить соединение с сервером SMB.
4. Проверьте правильность указанного имени общего каталога. Если каталог на сервере не существует или недоступен для заданных имени пользователя и пароля, то сервер SMB не выполнит запрос на установление соединения.
5. Соберите системные данные трассировки для SMBFS, используя ИД события 525.
6. Убедитесь, что сервер настроен для приема NTLM, LM и незашифрованных паролей. Это единственные типы шифрования паролей, поддерживаемые в SMBFS.
7. При необходимости выполнения идентификации в домене, имя домена должно быть указано опцией **wrkgrp**. Без этой опции идентификация обрабатывается сервером локально.

Асинхронная связь

AIX предлагает следующие категории драйверов асинхронных устройств, также называемых драйверами терминальных устройств:

- Драйверы последовательных портов системного планара
- Драйверы последовательных портов, подключенных через адаптер
- Драйверы псевдотерминалов

Драйверы из первой категории относятся к адаптерам PCI. К ним относятся быть 2-, 8- и 128-портовые адаптеры.

Во второй категории 8- и 128-портовые адаптеры PCI называются интеллектуальными адаптерами, так как в них встроен процессор Intel 8086, позволяющий не загружать центральный процессор обработкой символов. Эти адаптеры контролируются посредством опрашивающего устройства с периодом 20 мс, а не аппаратными прерываниями, и их быстродействие отвечает требованиям большинства последовательных устройств и приложений. По мере добавления устройств в систему нагрузка на нее возрастает лишь незначительно, поэтому эти адаптеры могут поддерживать большое число последовательных устройств, намного больше, чем адаптеры, использующие аппаратные прерывания. Кроме того, благодаря использованию этими адаптерами запатентованного средства повышения производительности программного обеспечения они могут отправлять и принимать большие объемы данных быстрее и эффективнее, чем встроенные системные порты, если данные перемещаются крупными блоками. Дополнительная информация приведена в описании wantio в файле `/usr/include/sys/pse/README.pse`.

Примечание: Интегрированные POWER5 *системные порты* схожи с последовательными портами за исключением того, что системные порты доступны только особым поддерживаемым функциям. Дополнительная информация приведена в разделе “Функциональные различия между системными и последовательными портами” на стр. 581.

Однако некоторые устройства или приложения ожидают или требуют очень низкой латентности при обработке одиночных символов, поэтому при подключении их к интеллектуальным адаптерам вы можете столкнуться с неполадками синхронизации. Латентность символов, или эхоповтор символов, можно определить как время, которое занимает прием одного символа последовательным портом, передача этого символа приложению и эхоповтор этого символа на том же последовательном порте.

Из-за использования высокоприоритетных прерываний (INTCLASS0) порты, управляемые прерываниями, обеспечивают латентность в диапазоне от 0,10 до 0,2 мс в простаивающей системе. 8-портовые адаптеры PCI обеспечивают латентность в среднем от 10 до 12 мс, которая может изменяться в ту или иную сторону на 10 мс из-за опрашивающего устройства с периодом 20 мс. 128-портовые адаптеры PCI имеют такое же опрашивающее устройства с периодом 20 мс, связанное через опрашивающий канал связи с узлами удаленного доступа (RAN). в которых драйвер опроса управляет последовательными портами. Среднее значение латентности этих портов составляет 30 мс, но иногда может превышать 60 мс.

Значения латентности 8- и 128-портовых адаптеров PCI можно настроить для специальных приложений с помощью параметра "задержка события" (EDELAY). Для максимальной быстроты реагирования при получении одиночного символа следует уменьшить значение параметра EDELAY. Это снижает до минимума время, необходимое для передачи одного символа из последовательного порта приложению, однако может стать причиной снижения пропускной способности и общей производительности системы при одновременном получении большого количества символов.

2-портовый адаптер PCI EIA-32 является асинхронным адаптером связи на базе Exar 17D152 Universal PC Dual UART. 2-портовый адаптер поддерживает два разъема DB-9 и обеспечивает подключение к асинхронным устройствам EIA-32, таким как модемы и терминалы.

На платформе IBM eServer p5 внутренние системные порты доступные AIX отсутствуют. Хотя интерфейс виртуального терминала расширен для обеспечения поддержки физических серийных портов, находящихся на FSP, с помощью гипервизора, этот интерфейс поддерживает только определенный набор серийных устройств. Поэтому его не следует применять для замены общего физического серийного порта. Поведение 2-портового адаптера похоже на работу встроенного системного порта. Драйвер адаптера управляется прерываниями, поддерживает программируемые переходы и получает уровни переключения FIFO. Он является адаптером PCI; поэтому драйвер устройства поддерживает EEH, оперативную замену и очереди VPD. 2-портовый адаптер не поддерживает функции встроенного системного порта, при работе с которыми применяется виртуальный интерфейс, например, в процессе загрузки, установки и поддержки KDB.

Псевдотерминальные устройства применяются при доступе к системе через сеть с использованием команд **rlogin** или **telnet**, либо при доступе к системе с использованием оконной системы на графическом мониторе. Драйвер псевдотерминала предоставляет средства запуска устаревших текстовых приложений, например текстового редактора **vi**, через средства связи, отличные от последовательных. Важно помнить, что псевдотерминальные драйверы не симметричны. Подчиненный конец предоставляет совместимый со стандартом POSIX интерфейс для приложений более ранних версий. Главный конец контролируется демоном **rlogin** или **telnet** или X-windows, которые должны предоставить драйверу псевдотерминала эмуляцию последовательного терминального устройства. AIX может поддерживать очень большое число псевдотерминалов.

Быстродействия линий не-POSIX

Интерфейс последовательных устройств, указанных в стандарте POSIX и последующих стандартах UNIX, например X/OPEN, основывается на структуре данных **termios**, указанной в `/usr/include/termios.h`. К сожалению, эту структуру данных нельзя использовать для указания быстродействия линий свыше 38400 бит/с. В настоящее время большинство последовательных устройств поддерживают скорость до 230000 бит/с. Для использования этих высоких скоростей в AIX следует указать требуемую скорость при настройке порта с помощью SMIT. Порт можно настроить, если аппаратное обеспечение последовательного порта (UART) поддерживает указанное быстродействие линии.

Установите атрибут ioctls с помощью структуры **termio** или **termios**, чтобы установить быстродействие линии равным 50 бит/с. Порт будет использовать быстродействие линии не-POSIX до внесения изменений, поэтому приложениям, использующим установленный атрибут ioctls со структурами **termio** и **termios**, не следует изменять флаги CBAUD, если только они не собираются изменять быстродействие линии. Если последовательный порт (UART) не поддерживает требуемую скорость, настроить порт не удастся и будет выдано сообщение об ошибке.

Примечание: Для линий не-POSIX 8- и 128-портовые адаптеры PCI поддерживают быстродействие только 115200 и 230000 бит/с. 128-портовый адаптер PCI имеет дополнительное ограничение пропускной способности в 2,5 Мбит/с (с 8-жильным кабелем), что соответствует 11 устройствам, работающим со скоростью 230000 бит/с. Это ограничение относится к линии, соединяющей адаптер с RAN, поэтому один адаптер может быть использован 22 такими устройствами.

Асинхронные адаптеры

Продукты асинхронной связи отличаются низкой стоимостью, поддержкой многопользовательских терминалов и устройствами связи средней и высокой производительности.

AIX позволяет работать с системными ресурсами и приложениями нескольким пользователям одновременно. Каждый пользователь должен быть подключен с помощью терминального сеанса. Это соединение может быть как локальным, так и удаленным - через последовательный порт.

В каждом системном блоке есть по крайней мере один последовательный порт, а в некоторых блоках число таких портов может доходить до трех. Эти порты могут поддерживать асинхронную связь и подключение устройств.

Асинхронные порты позволяют подключать асинхронные периферийные устройства, отвечающие стандартам EIA 232, EIA 422 или RS-423, например:

- Асинхронные модемы
- Сканеры штриховых кодов
- Графические и текстовые принтеры
- Клавиатуру и дисплей
- Персональные компьютеры
- Графопостроители и принтеры
- Терминалы торговых точек

- Датчики и контролирующие устройства
- Текстовые сканеры
- Часы

Адаптеры асинхронной связи

С помощью адаптеров, использующих шины Peripheral Component Interconnect (PCI), в систему можно добавить расширенные асинхронные функции.

На выбор типа асинхронного соединения влияют несколько факторов. В следующей таблице приведена общая информация об этих продуктах.

Таблица 93. Продукты асинхронной связи

Асинхронное соединение	На базе процессоров POWER	На базе Itanium	Тип шины	Код продукта или тип системы (модель)	Максимальная скорость передачи данных на порт (Кбит/с)	Особые функции
Стандартный последовательный порт	**X**	**X**	Системный планар	н/д	Изменяема в зависимости от скорости передачи в бодах универсального асинхронного приемопере- датчика (UART).	Стандартная функция
232 RAN	**X**	**X**		8130	57,6	Удаленная поддержка
Расширенный 232 RAN	**X**	**X**		8137	230	Удаленная поддержка
16-портовый RAN EIA 422	**X**	**X**		8138	230	Удаленная поддержка
128-портовый контроллер	**X**			8128	230	Эффектив- ность, большее число устройств
128-портовый контроллер	**X**			2933	230	Эффектив- ность, большее число устройств
128-портовый контроллер	**X**	**X**	PCI	2944	230	Эффектив- ность, большее число устройств

Примечание: Номер монтируемого в стойке RAN FC - 8136.

Примечание: Максимальная скорость передачи данных через порт ограничена пропускной способностью линии (1,2 Мбит/с для стандартного RAN или 2,4 Мбит/с для расширенного RAN).

Первый элемент в этой таблице представляет подключенные через планар последовательные порты, поставляемые в стандартной комплектации любого системного блока. Следующие элементы обозначают адаптеры. 128-портовая асинхронная подсистема содержит подключенные к ней удаленные асинхронные узлы (RAN).

Подключаемые к планару асинхронные порты

Большинство системных блоков имеют два встроенных (стандартных) асинхронных последовательных порта EIA 232. Асинхронные последовательные устройства EIA 232 можно подключать напрямую к стандартным последовательным портам с помощью стандартных последовательных кабелей с 9-штырьковыми D-образными разъемами.

У некоторых многопроцессорных систем может быть третий последовательный порт для связи с удаленными сервисным центром.

Примечание: Системы с Itanium имеют один или два встроенных последовательных порта. Базовые рабочие станции содержат один порт, тогда как серверы начального уровня - два порта.

Подключаемые к адаптеру асинхронные порты

Каждому адаптеру требуется разъем шины, использовать его можно только в системах, поддерживающих требуемый тип шины.

128-портовые, 8-портовые адаптеры ISA и 8-портовые адаптеры PCI являются интеллектуальными адаптерами, существенно снижающими нагрузку на основной системный процессор.

EIA 232 является наиболее распространенным стандартом связи, но поддерживается и стандарт EIA 422A (применяемый при более длинном кабеле). Реализация EIA 422A не предусматривает возможность определения состояния устройства или сигналы управления модемом RS 232.

Примечание: Платформы с Itanium поддерживают только 8- и 128-портовые адаптеры PCI.

Неподключенные асинхронные порты

128-портовый адаптер для шины Micro Channel, ISA или PCI позволяет подключить от одного до восьми удаленных асинхронных узлов (RAN).

У каждого RAN есть 16 асинхронных портов для подключения устройств и собственный блок питания. К каждому из двух соединений 128-портовой карты адаптера можно подключить до 4 RAN в гирляндной цепи. RAN могут поддерживать до 16 устройств EIA 232 или EIA 422. 128-портовый контроллер является интеллектуальным адаптером, который может увеличить число возможных асинхронных сеансов при заданном уровне занятости центрального процессора.

Ниже перечислены дополнительные параметры 128-портового оборудования:

- Устройства RAN могут быть расположены на удалении 300 метров от системного процессора за счет использования 8-контактного изолированного кабеля и полностью сохранять при этом свою функциональность.
- Расстояние может быть увеличено до 1200 метров за счет снижения скорости передачи данных между RAN и системным процессором.
- RAN можно подключать к процессору системы и при помощи асинхронных модемов EIA 232 и EIA 422. В гирляндной цепи из четырех RAN может быть только одна пара модемов.
- Производительность системы можно повысить, если выгрузить обработку символов терминала из процессора системы.

Замечания по выбору продукта

Подходящий асинхронный адаптер зачастую зависит от конкретной ситуации.

Следующие вопросы помогут вам выбрать продукт.

Расширяемость

Сколько асинхронных портов вам необходимо?

Сколько портов потребуется вам в будущем?

Топология

Устройства будут находиться в другом здании или в удаленном расположении?

Где будет находиться системный/сетевой администратор?

Есть ли там кластер HACMP?

Какое соединение потребуется или уже существует?

Быстродействие

Насколько интенсивно загружают центральный процессор ваши приложения?

Какие типы устройств будут подключаться?

Какая общая асинхронная пропускная способность требуется для всех устройств?

Таблица 94. Требования к пропускной способности для устройств

Низкие требования	Средние требования	Высокие требования
ASCII-терминалы (AIX), терминалы торговых точек, асинхронные модемы	Принтеры, низкоскоростные факс-модемы, сканеры штриховых кодов	Последовательные X-терминалы, высокоскоростные факс-модемы, высокоскоростные принтеры, приложения передачи файлов

Требования к интерфейсу устройств

Какой асинхронный интерфейс вам необходим (например, EIA 232, EIA 422A, EIA 423)?

Устройствам или приложениям требуется полнофункциональный интерфейс EIA 232?

Защита

Требуется ли системная защита ядра (SAK)? (только для портов, подключенных через планар)

В следующей таблице приведена подробная информация о продуктах.

Таблица 95. Параметры подключаемого асинхронно продукта

Параметр	Встроенные посл. порты	2-портовый PCI	8-портовый PCI	128-портовый PCI с RAN
Число асинхронных портов адаптера	н/д	61 см	8	128
Максимальное число адаптеров	н/д	не ограничено	20	20
Максимальное число асинхронных портов	2 или 3	61 см	160	2560
Число асинхронных портов RAN	н/д	н/д	н/д	16
Максимальное число RAN	н/д	н/д	н/д	160
Максимальная скорость передачи данных (Кбит/с)	Изменяема в зависимости от скорости передачи в бодах UART.	230	230	230
Способ подключения	Планар	direct	direct	узел
Поддерживаемые асинхронные электрические интерфейсы	EIA 232	EIA 232	EIA 232 EIA 422A	EIA 232 EIA 422
Стандартный разъем	DB9	DB9	DB25M	RJ-45 (10-штырьковый или 8-штырьковый)
Возможность использования кабеля DB25	н/д	н/д	н/д	RJ-45-DB25
Возможность монтирования в стойке	н/д	н/д	н/д	да
Блок питания	н/д	н/д	н/д	внешний
Поддерживаемые сигналы (EIA 232)	TxD RxD RTS CTS DTR DSR DCD RI	TxD RxD RTS CTS DTR DSR DCD RI	TxD RxD RTS CTS DTR DSR DCD RI	TxD RxD RTS CTS DTR DSR DCD RI

Применение асинхронного адаптера

Составить представление о каждом продукте можно по перечню его основных достоинств. Ниже перечислены адаптеры и их спецификации, чтобы вы могли сделать выбор исходя из потребностей для определенного сценария.

Элемент	Описание
Шина PCI 2-портового адаптера EIA 232	• Есть разъем PCI. • До двух портов на адаптер. • Требуются все порты EIA 232. • Асинхронная скорость до 230 Кбит/с.
Шина PCI 8-портового адаптера EIA 232/EIA 422	• Есть разъем PCI. • Необходимо менее 8 портов, при этом расширение не требуется или почти не требуется. • Необходимы все порты EIA 232, EIA 422 или сочетание портов EIA 232 или EIA 422. • Снимает с центрального процессора процессы обработки ввода-вывода терминала и символьные прерывания. • Асинхронная скорость до 230 Кбит/с. • Максимальное быстродействие для высокоскоростных (33,6 Кбит/с) модемов со сжатием данных.
128-портовый адаптер (PCI)	• Есть разъем Micro Channel, ISA или PCI. (Дополнительная информация о Micro Channel и ISA приведена в “128-портовый адаптер (Micro Channel, ISA)” на стр. 666) • 16 портов с возможностью расширения до 128 портов без использования дополнительных разъемов. • Максимальное удаление терминала от системы - 90 м, максимальная скорость обмена данными - 230 Кбит/с. • Расположение терминалов: на малом или большом расстоянии в том же здании или удаленное. • Высокая пропускная способность асинхронной связи, низкая нагрузка на процессор. • Возможность подключения принтера к терминалу. • Возможность подключения к удаленным системам через оптоволоконные или синхронные модемы.

Пользовательские сценарии для асинхронных адаптеров

Пользовательские сценарии, приведенные здесь, были проверены с помощью 8-портового PCI и 128-портового асинхронного контроллера.

Элемент	Описание
Офис агентства недвижимости	• Наивысшие приоритеты - простота и стоимость. • Операционная система и сервер. • От шести до десяти устройств связаны с базой данных через сервер. • Один разъем доступен для асинхронной связи. • Устройства удалены от сервера на расстояние не свыше 61 метра. **Вариант:** 8-портовый PCI.
Торговая точка	• Наивысший приоритет - стоимость одного рабочего места. • Операционная система и сервер. • 20 или более ASCII-терминалов: например, кассовых аппаратов. • Один разъем доступен для асинхронной связи. • В будущем планируется установка дополнительных терминалов. **Вариант:** 128-портовый асинхронный контроллер с двумя RAN. Будущее расширение будут обеспечивать дополнительные RAN.

Замечания о топологиях

Асинхронные адаптеры позволяют создавать множество различных конфигураций. Важную роль при этом играет топология, т.е. расстояния между устройствами.

Максимальная длина кабеля адаптеров, подключенных через планар или напрямую, обычно составляет расстояние между портом и асинхронным устройством, работающим на максимальной указанной скорости. Расстояние для 128-портового адаптера измеряется от карты адаптера до подключенного к нему по гирляндной цепи узла RAN. При использовании 128-портового адаптера можно увеличить допустимую длину кабелей практически до бесконечности за счет подключения узлов RAN к адаптеру через синхронные модемы EIA 422.

Правильное подключение кабелей играет чрезвычайно важную роль и уникально в каждой конкретной среде.

Последовательная связь

Здесь содержится информация о стандартах асинхронной связи, аппаратном обеспечении, терминологии и концепциях.

Последовательные порты служат для физического подключения асинхронных устройств к компьютеру. Они расположены на задней панели системного блока и являются интегрированными или могут использовать многопортовые адаптеры, например 2-, 8-, 16 и 128-портовые асинхронные адаптеры.

Примечание: Интегрированные системные порты POWER5 не являются полнофункциональными последовательными портами общего предназначения. Дополнительная информация приведена в разделе “Функциональные различия между системными и последовательными портами” на стр. 581.

Для того, чтобы понять, как функционирует последовательный порт, необходимо сначала изучить параллельную связь. Стандартный параллельный порт использует восемь контактов (жил, проводов) для одновременной передачи битов данных, составляющих один символ. На следующем рисунке показана параллельная передача символа a.

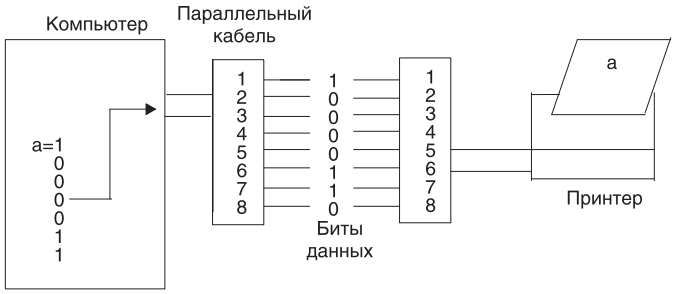


Рисунок 29. Параллельный порт связи

Последовательным портам необходим только один контакт, или провод, для пересылки такого же символа устройству. Это достигает за счет преобразования данных из параллельной формы (в которой их отправляет компьютер) в последовательную, в которой биты расположены один за другим. Затем данные передаются устройству с незначащим битом (т.н. нуль-битом) в начале. После того, как данные получены удаленным устройством, они вновь преобразуются в параллельную форму. На следующем рисунке показана последовательная передача символа a.

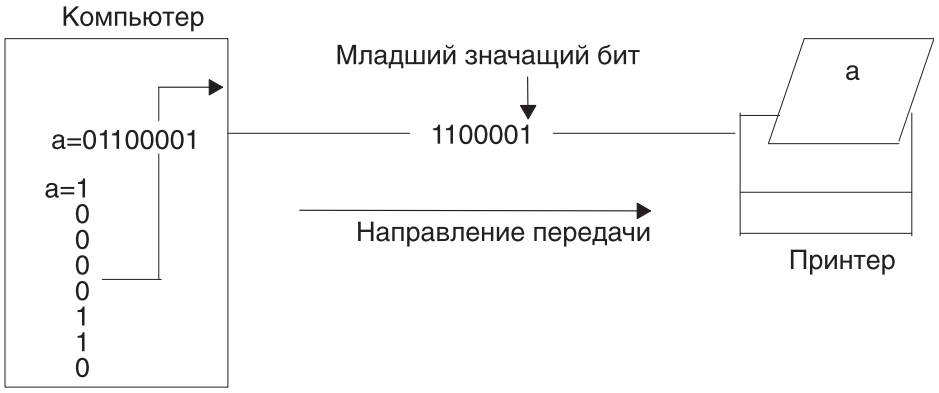


Рисунок 30. Последовательный порт связи

Последовательная передача одного символа довольно проста, однако при передаче большого числа символов могут возникнуть затруднения, как показано на следующем рисунке. Принимающая система не знает, где заканчивается один символ и начинается другой. Для решения этой проблемы концы соединения должны быть синхронизированы.

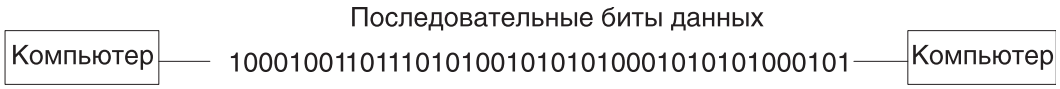


Рисунок 31. Последовательная передача данных

Функциональные различия между системными и последовательными портами

Интегрированные POWER5 *системные порты* схожи с последовательными портами за исключением того, что системные порты доступны только особым поддерживаемым функциям.

При подключении консоли аппаратного обеспечения (HMC) к порту HMC системные порты отключены. Можно использовать либо порты HMC, либо системные порты, но не те и другие одновременно.

Даже если HMC е подключена, встроенные системные порты ограничены функциями консоли подключенного через последовательный порт терминала. Они функционируют должным образом только с модемами для вызова сервисного центра, асинхронными терминалами и определенными блоками UPS. Для подключения других последовательных устройств (включая межсистемные соединения HACMP) необходим адаптер последовательного порта к разъему PCI.

Синхронизация

Синхронизация - это процесс согласования по времени последовательной передачи данных, необходимый для правильной идентификации отправляемых данных.

Два наиболее распространенных режима синхронизации - это синхронный и асинхронный.

Синхронная передача:

В *синхронном* режиме блоки данных пересылаются непрерывно и согласованно.

Такие соединения используются, когда необходимо очень быстро переслать большой объем данных. Высокая скорость синхронной связи достигается за счет передачи данных крупными блоками, а не отдельными символами.

Блоки данных группируются и располагаются с определенным интервалом и предваряются специальными символами, называемыми `syn` или символами синхронного простоя. См. рисунок.

Рисунок 32. Синхронная передача

После получения символов `syn` удаленное устройство декодирует их и использует для синхронизации соединения. После синхронизации соединения начинается передача данных.

Это тип соединения можно сравнить с передачей большого текстового документа. Перед передачей документа по синхронному соединению он разбивается на блоки, состоящие из предложений или абзацев. Затем эти блоки передаются по линии связи удаленной системе. В других режимах передачи текст преобразуется в длинные строки букв (или символов), составляющие слова и абзацы. Эти символы поочередно передаются по линии связи и вновь объединяются в слова удаленной системой.

Синхронизация соединения обеспечивается устройствами, находящимися на линии связи. Все устройства при синхронном соединении должны быть синхронизированы одинаково.

Ниже приведен список основных особенностей синхронной связи:

- Между передаваемыми символами нет пропусков.
- Синхронизацию обеспечивают модемы или другие устройства на обоих концах соединения.
- Передаваемым данным предшествуют специальные символы `syn`.
- Символы `syn` используются для синхронизации и размещаются между блоками данных.

Асинхронная передача:

В *асинхронном* соединении в передаваемые данные добавляются старт-биты и стоп-биты, указывающие начало и конец каждого символа.

На следующем рисунке приведен пример асинхронной передаче.

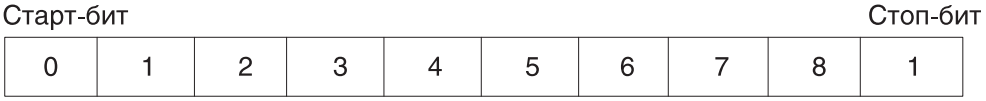


Рисунок 33. Асинхронная передача

Эти биты обеспечивают синхронизацию соединения, указывая начало и конец отправки или приема символа. Таким образом, синхронизация каждого символа начинается со старт-бита и заканчивается стоп-битом.

Когда между сеансами передачи символов появляются паузы, говорят, что асинхронное соединение переходит в состояние метки. Метка - это двоичная единица (или отрицательное напряжение), которая отправляется во время бездействия соединения, как показано на следующем рисунке.

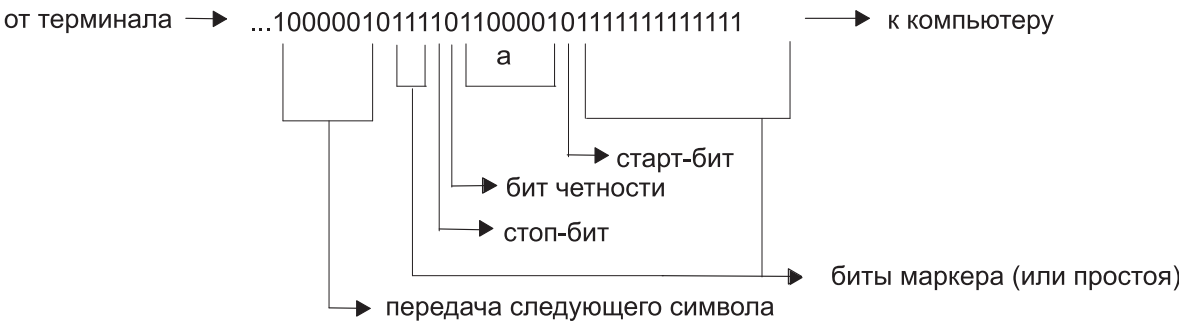


Рисунок 34. Биты метки (простой) в потоке данных

Когда состояние метки прерывается положительным напряжением (двоичным нулем), принимающая система узнает, что сейчас начнется передача данных. Поэтому старт-бит, предшествующий символу данных, всегда является битом пробела (двоичный нуль), а стоп-бит, указывающий на конец символа, всегда является битом метки (двоичная единица).

Ниже приведен список основных особенностей асинхронной связи:

- Перед каждым символом находится старт-бит, а после символа - один или несколько стоп-битов.
- Между символами могут быть промежутки.

Параметры последовательной связи

К используемой для последовательной связи параметрам относятся число бит на символ, бит/с, четность, начальный бит, конечный бит и бит метки.

Число битов в символе:

Число битов в символе (bpc) указывает, сколько битов используется для представления одного символа данных в последовательном соединении.

Это число не учитывает общее количество битов контроля четности, старт-битов и стоп-битов, включенных в символ. Значение bpc может быть равно 7 или 8.

Если bpc равно 7, то можно пересылать только первые 128 символов (0-127) из стандартной таблицы ASCII. Каждый из этих символов представлен 7 битами данных. Для пересылки символов из расширенной таблицы ASCII (128-255) нужно использовать 8 битов. Символы из расширенной таблицы могут быть представлены только восемью битами.

Бит в секунду, бит/с:

Описание статистики бит в секунду.

Число битов в секунду (bps, бит/с) - это число битов данных (двоичных единиц и нулей), передаваемых по линии связи за одну секунду.

Скорость передачи в бодах:

Скорость передачи в бодах - это число изменений состояния сигнала последовательной связи за одну секунду; состоянием сигнала может быть напряжение, частота или фаза.

Если сигнал меняет состояние один раз для каждого бита данных, то один бит/с равен одному боду. Например, модем с быстродействием 300 бод меняет свое состояние 300 раз в секунду.

Биты контроля четности:

Бит контроля четности, в отличие от старт-бита и стоп-бита, является необязательным параметром. Он используется в последовательной связи, чтобы определить, правильно ли передаваемый символ был получен удаленным устройством.

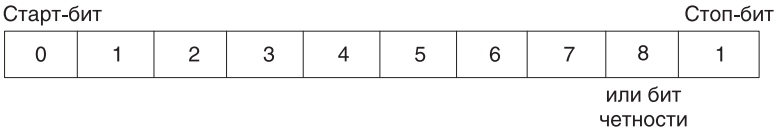


Рисунок 35. Контроль четности

Бит контроля четности может принимать одно из пяти значений:

Элемент	Описание
Нет	Указывает, что локальная система не должна создавать бит контроля четности для передаваемых символов. Это также означает, что локальная система не проверяет наличие этого бита в данных, полученных от удаленной системы.
Четное	Указывает, что общее количество двоичных единиц в каждом символе должно быть четным. Если это не так, бит контроля четности принимает значение 1, чтобы число двоичных единиц стало четным. Например, если при включенном контроле по четности передается буква "a" (в двоичном формате 1100001), то отправляющая система добавляет бит контроля четности со значением 1, чтобы сделать число единиц четным. При отправке буквы "A" (в двоичном формате 1000001) бит контроля четности примет значение 0, сохранив четное число двоичных единиц.
Нечетное	Действует аналогично опции контроля по четности, но число двоичных единиц должно быть нечетным.
Пробел	Указывает, что бит контроля четности всегда будет двоичным нулем. Контроль по пробелу также называют заполняющим битом, поскольку он использовался для пополнения 7-битовых данных, передаваемых на устройство, распознающее только 8-битовые данные. Такие устройства принимали бит контроля по пробелу за дополнительный восьмой бит передаваемого символа.
Метка	Действует так же, как и бит контроля по пробелу, но бит контроля четности всегда будет двоичной единицей. Бит метки используется только для заполнения.

Старт-бит, стоп-бит и бит метки:

Старт- и стоп-биты используются в асинхронной связи как средства синхронизации передаваемых символов.

Без этих битов отправляющая и принимающая системы не смогли бы определить, где заканчивается один символ и начинается другой.

Еще одним битом, используемым для разделения символов во время передачи, является бит метки (или простоя) RS. Этот бит, двоичная 1, передается, когда линия связи простаивает и символы не отправляются и не принимаются.

Когда система принимает старт-бит (двоичный нуль), она понимает, что за ним последует фиксированное число символьных бит (определяемое параметром **бит на символ**) и даже бит четности (определяемый параметром **четность**). Затем система принимает стоп-бит (двоичную единицу). В следующем примере существует бит **четности**, а число **бит на символ** равно 7.

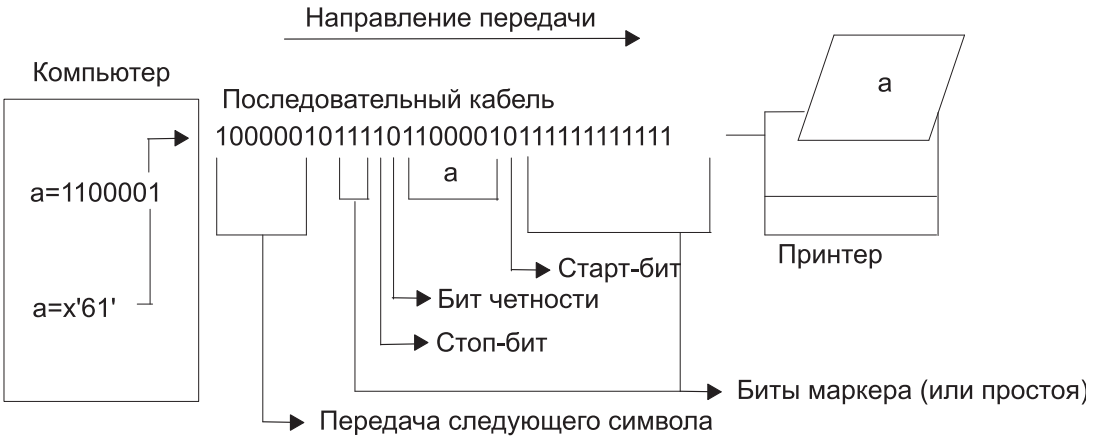


Рисунок 36. Старт-бит, стоп-бит и бит метки

Стандарт EIA 232D

Стандарт EIA 232D был разработан в 1969 году с целью определить параметры соединения между компьютером и модемом.

Само название стандарта является сокращением, означающим:

Утвержденный Ассоциацией электронной промышленности (EIA) стандарт, идентификационный номер 232, издание D.

EIA 232D задает параметры физического и электрического соединения между двумя устройствами. Каждому контакту или проводу, участвующему в последовательной связи, присваивается имя или аббревиатура, например:

Таблица 96. Соединения EIA 232D

Сигнал	Тип оборудования	Символ	Контакт
Передать данные	DCE	TxD	61 см
Принять данные	DTE	RxD	3
Готовность к отправке	DCE	RTS	4
Готовность к приему	DTE	CTS	5
Сигнал готовности к отправке данных	DTE	DSR	6
Земля сигнала		SG	7
Определить несущую частоту	DTE	CD	8
Сигнал готовности терминала	DCE	DTR	20
Индикатор звонка	DTE	RI	22

Согласно стандарту EIA 232D, устройства, использующие контакт 2 (TxD) для вывода, например компьютеры и терминалы, называются "терминальное оборудование" (DTE). Устройства, использующие контакт 2 (TxD) для ввода, например модемы, называются "оборудование передачи данных" (DCE).

Стандарт EIA 232D также определяет вид разъемов. Устройства DTE обычно имеют штепсельные разъемы, а устройства DCE - розеточные. Однако производители не всегда соблюдают этот стандарт, поэтому пользователям следует ознакомиться с описанием устройств перед их подключением.

Способы асинхронной связи

Здесь приведено описание двух типов асинхронной связи - односторонняя и двусторонняя (включающая в себя полудуплекс и полный дуплекс).

Симплексное, или одностороннее, соединение является простейшим типом соединения двух устройств. Этот способ связи позволяет передавать данные только в одном направлении и требует подключения двух линий, например TxD (или RxD) и SG.

Кроме этого, есть два типа двусторонней связи: полудуплексная и дуплексная. Полудуплексное соединение позволяет передавать данные в двух направлениях, но не одновременно. Полудуплексную связь можно сравнить с первыми моделями телефонов, когда связь была двусторонняя, но говорить можно было только по очереди.

В полностью дуплексном режиме передача данных может происходить в обе стороны одновременно. Дуплексную связь можно сравнить с современной телефонной связью, когда партнеры могут разговаривать одновременно.

Управление потоком

Для ограничения объема данных, передаваемых системой на последовательное устройство, последнему необходимо применять то или иное управление потоком.

Последовательные устройства, такие как принтеры и модемы, не могут обрабатывать данные так же быстро и эффективно, как компьютеры, к которым они подключены.

Термин *управление потоком* используется для описания способа, с помощью которого последовательное устройство контролирует объем поступающих на него данных.

Аппаратное управление потоком RTS/CTS:

Готовность к передаче/готовность к приему (RTS/CTS) иногда также называется окном подтверждения или квитированием, а не управлением потоком.

Термин "квитирование аппаратного обеспечения" объясняется тем, что ранее для управления передачей данных применялись кабели и напряжение. В отличие от XON/XOFF, который отправляет управляющие символы в потоке данных, RTS/CTS использует положительное и отрицательное напряжение вместе с выделенными контактами и проводами в кабелях устройств.

Положительное напряжение означает разрешение на передачу данных, тогда как отрицательное напряжение означает, что передачу данных необходимо приостановить.

Аппаратное управление потоком DTR/DSR:

Сигнал готовности терминала - еще одна форма аппаратного управления потоком данных - обычно генерируется устройствами, например принтерами, с целью сообщить, что они готовы к обмену информацией с системой. Этот сигнал используется вместе с Сигналом готовности к отправке данных (DSR), генерируемым системой, для управления потоком данных.

Положительное напряжение означает разрешение на передачу данных, тогда как отрицательное напряжение означает, что передачу данных необходимо приостановить.

Программное управление потоком XON/XOFF:

Управление потоком Передатчик включен/передатчик выключен (XON/XOFF) отправляет символы управления передачей данных вместе с потоком данных (TxD и RxD). По этой причине его называют программным управлением потоком.

Когда данные отправляются на модем, они помещаются в буфер. Непосредственно перед заполнением буфера модем отправит символ XOFF системе, и система прекратит передачу данных. Когда буфер модема почти опустеет и будет готов к приему новых данных, модем отправит символ XON системе, и та продолжит отправку данных.

Настройка порта для аппаратного квитирования RTS/CTS:

Если подключенные к серверу модемы работают на скорости от 9600 бод, рекомендуется использовать аппаратное квитирование RTS/CTS вместо управления потоком XON/XOFF.

Это позволит избежать перегруженности буфера в системах с ограниченными ресурсами. RTS не является значением по умолчанию для портов терминала и должен быть установлен системным администратором.

`Предварительные требования`

Для поддержки RTS/CTS следует использовать по крайней мере пятижильный кабель.

Для того чтобы включить RTS/CTS для порта, выполните следующие действия:

1. Введите команду быстрого доступа `smit tty`.
2. Выберите пункт **Изменить / Показать параметры терминала**.
3. Выберите терминал, для которого необходимо включить RTS/CTS.
4. В поле Управление потоком выберите **rts**.
5. Выберите **Выполнить**.
6. Завершите работу SMIT.

Терминал

Терминал - это символьное устройство, обеспечивающее посимвольный ввод и вывод данных.

Связью между терминалами и программами, работающими с ними, управляет интерфейс терминала. Ниже приведены примеры устройств tty:

- Модемы
- Терминалы ASCII
- Системная консоль (LFT)
- **aixterm** для AIXwindows

Устройства tty (терминалы) можно добавлять, удалять, просматривать и изменять с помощью инструмента SMIT или специальных команд, как и любые другие устройства системы.

Значения TERM для различных терминалов и окон

Информация о функциях терминала хранится в базе данных `terminfo`.

Значение переменной среды **TERM** задает конкретное описание терминала в базе данных `terminfo`. Это описание содержит всю информацию, необходимую программе для связи с текущим устройством tty.

Таблица 97. Значения TERM для различных терминалов

Дисплей/Терминал	Значение
Терминал 3161 ASCII	ibm3161
Терминал 3163 ASCII	ibm3161
DEC VT100 (терминал)	vt100
DECVT220	vt220
Дисплейная станция 3151 ASCII с кассетой или дисплейная станция 3161 ASCII с кассетой	ibm3161-C
Дисплейная станция 3162 ASCII	ibm3161
Дисплейная станция 3162 ASCII с кассетой	ibm3162
Дисплей 6091	lft
AIXwindows	aixterm

Дополнительная информация о записях базы данных terminfo приведена в описании формата файла terminfo в книге *Справочник по файлам*. Информация о преобразовании записей termcap в terminfo приведена в описании команды **captoinfo** в*Справочник по командам, том 1*. (В файле termcap содержатся описания терминалов для устаревших версий системы Berkeley).

Параметры терминала

Протокол передачи данных предоставляет аппаратно-независимый пользовательский интерфейс для связи между компьютером и асинхронным устройством.

Например, пользователь может ввести несколько символов в определенной последовательности для удаления строки или отмены текущего процесса. Последовательность символов, как и другие параметры терминала, например, быстродействие линии связи, могут быть заданы пользователем с помощью команды **chdev**, Инструмента управления системой (SMIT) или команды **stty**.

Требования к атрибутам подключенных терминалов

Для правильного взаимодействия хоста с подключенным терминалом должны быть выполнены следующие условия:

- Должен быть подключен кабель
- Параметры связи хоста и подключенного терминала (быстродействие линии, размер символов, четность, стоп-биты и интерфейс) должны быть согласованы между собой.

Управление терминалом

В этом разделе описаны задачи по управлению терминалами и связанные с ними команды быстрого доступа SMIT.

Таблица 98. Задачи управления терминалами

Процедура	Команды быстрого доступа SMIT	Команда или файл
Просмотреть список определенных терминалов	smit lsdtty	**lsdev -C -c tty -H**
Добавить терминал	smit mktty	**mkdev -t tty**[1,2]
Изменить порт терминала [3]	smit movtty	**chdev -l** *имя* **-p** *имя-родительского-устройства* **-w** *расположение-соединения*[2,4]
Изменить/Показать параметры терминала	smit chtty	**lsattr -l***имя* **-E** (для просмотра); **chdev -l** *имя* (для изменения)[4,5]
Удалить терминал [3]	smit rmtty	**rmdev -l** *имя*
Настроить терминал (сделать доступным для использования)	smit mktty	**mkdev -l** *имя*

Примечание:

1. Для дополнительной настройки нового терминала можно указывать и другие флаги. Например, для того чтобы определить и настроить терминал RS-232, подключенный к порту 0 8-портового асинхронного адаптера sa3, установить атрибут скорость равным 19200, а для остальных атрибутов установить значения, указанные в файле foo, введите:

   ```
   mkdev -t tty -s rs232 -p sa3 -w 0 -a speed=19200 -f foo
   ```

2. Команды **mkdev** и **chdev** поддерживают опции, которые нельзя задать с помощью SMIT.
3. Перед выполнением задачи отключите терминал. Дополнительная информация приведена в описании команды **pdisable** в *Справочник по командам, том 4*.
4. С помощью флагов измените характеристики терминала из командной строки.
5. Скорость передачи данных Posix можно выбрать в списке или ввести в текстовом поле. Если выбранная скорость передачи данных не поддерживается модемом, появится сообщение об ошибке.

Ниже приведен список *атрибутов*, которые можно указывать с флагом **-a** *Атрибут***=***Значение* для изменения различных параметров терминалов из командной строки. Например, для настройки параметра терминала

Скорость передачи в бодах укажите `-a speed=Значение`.

Таблица 99. Атрибуты TTY

Параметр	Имя атрибута
Разрешить LOGIN	login
Скорость передачи в БОДАХ	speed
ЧЕТНОСТЬ	parity
Число БИТ на символ	bpc
Число СТОП-БИТ	stops
Время ожидания перед переходом к следующему значению	timeout
Квитирование XON-XOFF	xon
Тип ТЕРМИНАЛА	term
Способ УПРАВЛЕНИЯ ПОТОКОМ	flow_disp
АЛГОРИТМ ОТКРЫТИЯ	open_disp
Атрибуты STTY времени ВЫПОЛНЕНИЯ	runmodes
Атрибуты STTY для входа в систему	logmodes
ЗАПУСТИТЬ администратор оболочки	shell
Имя РЕГИСТРИРУЮЩЕЙ ПРОГРАММЫ	logger
СОСТОЯНИЕ устройства во время ЗАГРУЗКИ	autoconfig
Объем буфера ПЕРЕДАЧИ	tbc
Объем буфера ПРИЕМА	rtrig
Модули ПОТОКОВ, передаваемые во время открытия	modules
Файл таблицы преобразования ПРИНИМАЕМЫХ ДАННЫХ	imap
Файл таблицы преобразования ОТПРАВЛЯЕМЫХ ДАННЫХ	omap
Файл таблицы преобразования КОДИРОВАННОГО НАБОРА СИМВОЛОВ	csmap
Символ ПРЕРЫВАНИЯ	intr
Символ ЗАВЕРШЕНИЯ РАБОТЫ	quit
Символ СТИРАНИЯ	erase
Символ ЗАВЕРШЕНИЯ ПРОЦЕССА	kill
Символ КОНЦА ФАЙЛА	eof
Символ КОНЦА СТРОКИ	eol
Дополнительный символ КОНЦА СТРОКИ	eol2
Символ ОТЛОЖИТЬ ПРИОСТАНОВЛЕННЫЙ ПРОЦЕСС	dsusp
Символ ПРИОСТАНОВИТЬ ПРОЦЕСС	susp
Символ СЛЕДУЮЩИЙ ЛИТЕРАЛ	lnext
Символ ЗАПУСКА	start
Символ ЗАВЕРШЕНИЯ РАБОТЫ	stop
Символ УДАЛЕНИЯ СЛОВА	werase
Символ ПОВТОРНЫЙ ВЫВОД СТРОКИ	reprint
Символ ОТКАЗА	discard

Устранение неполадок терминала

Существует несколько распространенных приемов устранения неполадок терминалов.

К ним относятся ошибки `Слишком частое создание процессов`, зависание портов терминала, а также сообщения протокола ошибок, команд и отчетов об ошибках.

Ошибки слишком частого создания процессов:

Система регистрирует число процессов **getty**, созданных для каждого терминала за определенный промежуток времени. Если за это время было создано более пяти процессов **getty**, то на консоль выводится сообщение о `слишком частом создании процессов`, и система отключает порт.

Терминал остается отключенным в течение примерно 19 минут, либо пока системный администратор заново не активизирует порт. Через 19 минут система автоматически включит порт, после чего будет создан новый процесс **getty**.

Возможными причинами возникновения данной ошибки могут являться следующие моменты:

- Неправильная конфигурация модема
- Порт настроен и включен, но к нему не подключен кабель или устройство
- Поврежден кабель или разорвано соединение
- Шум в линии связи
- Файлы `/etc/environment` и `/etc/inittab` повреждены или содержат неправильную информацию
- Повреждена конфигурация терминала
- Неисправно аппаратное обеспечение

Используйте для устранения неполадки одну из следующих процедур, подходящую для вашей ситуации.

- Неправильная конфигурация модема:

  Убедитесь, что для модема *не* установлен высокий уровень сигнала обнаружения несущей частоты.

  Примечание: Следующее относится только к Hayes-совместимым модемам

  1. Подключитесь к модему и проверьте активный профайл.
  2. Для модема должен быть задан уровень обнаружения несущей частоты **&C1**, а не **&C0** (высокий). Для установки и изменения атрибута несущей частоты примените команды модема AT:

     ```
     AT&C1
     AT&W
     ```

     Примечание:

     a. Обратитесь к разделу “Отправка команд AT с помощью команды cu” на стр. 602

     b. Дополнительная информация приведена в технической документации по модему.
- Отключите терминал, удалите определение терминала или подключите устройство к порту:
  – Для отключения определения терминала введите команду **chdev**, как показано в следующем примере:

    ```
    chdev -l
    имя-терминала -a Login=disable
    ```

    Если вы ввели эту команду, терминал *не* будет активизирован после перезапуска системы.
  – Для удаления определения терминала:
    1. Отключите порт терминала командой **pdisable**:

       ```
       pdisable имя-терминала
       ```
    2. Удалите определение терминала из системы. Дополнительная информация приведена в разделе “Управление терминалом” на стр. 588.
- Проверьте, не поврежден ли кабель, и не нарушено ли соединение:
  1. Проверьте исправность кабелей. Замените поврежденные разъемы.
  2. Убедитесь, что применяется кабель фирмы IBM с кодом 6323741 или аналогичный ему. Замените испорченные участки кабеля или кабель другого стандарта.
- Устраните шум на линии связи:
  1. Проверьте длину и сопротивление кабеля.

2. Убедитесь, что на длинных кабелях установлены ферритовые кольца.
3. Проверьте правильность прокладки кабелей: рядом с ними не должны находиться лампы дневного света и электродвигатели.

- Проверьте, не повреждены ли файлы `/etc/environment` и `/etc/inittab`:
  1. Сравните эти файлы с сохранившимися копиями (если они есть).
  2. Создайте резервную копию этих файлов и внесите в них соответствующие изменения.
  3. Удалите из файла `/etc/environment` все строки, *кроме* следующих:
     – пустые строки
     – комментарии
     – строки вида *переменная=значение*
  4. Проверьте строки файла `/etc/inittab`, содержащие записи о терминалах. Если для терминала указано значение off, то, скорее всего, порт терминала в данный момент не применяется. Если порт не применяется, удалите описание tty или подключите устройство к порту.
- Удалите поврежденную конфигурацию терминала:
  1. Удалите определение терминала. Дополнительная информация приведена в разделе “Управление терминалом” на стр. 588.
  2. Если вы хотите напечатать определение терминала перед его удалением из системы, нажмите клавишу Image (F8 или Esc+8). При этом текущее изображение на экране будет скопировано в файл `smit.log`, расположенный в каталоге `$HOME`.
  3. Ознакомьтесь с определением терминала. За дополнительной информацией обратитесь к инструкциям по добавлению терминала, приведенным в разделе “Управление терминалом” на стр. 588.
- Найдите неисправное аппаратное обеспечение:
  1. Запустите средства диагностики с помощью команды **diag**.
  2. При обнаружении аппаратных неполадок выполните соответствующие действия по их устранению.

Сведения о протоколе ошибок и протоколе терминала:

Ниже приведены команды и файлы протоколов, относящиеся к терминалам.

Команда: **errclear**

Удаляет записи из протокола ошибок. Для удаления всего протокола служит команда `errclear 0`. Кроме того, с помощью этой команды можно удалить записи с определенными идентификаторами ошибок, записи определенного класса или типа.

Команда: **errpt**

Эта команда создает отчет об ошибке, обращаясь к записям из системного протокола ошибок. Обычно эта команда вызывается в формате `errpt -a | pg` для создания подробного отчета, начинающегося с наиболее часто встречающихся ошибок.

Файл: `/var/adm/ras/errlog`

В этом файле хранятся сообщения об ошибках и сбоях, обнаруженных системой. Со временем файл `errlog` может стать очень большим. Если не выполнять периодическую очистку файла, то через некоторое время он будет занимать очень много места на жестком диске. Для очистки файла вызовите описанную выше команду **errclear**.

Файл: `/usr/include/sys/errids.h`

В файле `errids.h` приведен список идентификаторами ошибок и соответствующих им меток.

Ниже приведены распространенные сообщения об ошибках, относящихся к терминалам:

Таблица 100. Сообщения об ошибках TTY

Сообщение	Описание	Комментарии
Создан дамп ядра	Аварийное завершение программы	Сообщение об этой ошибке заносится в протокол при аварийном завершении программы, которое привело к созданию дампа ядра. Это может произойти из-за неправильного выхода из приложения, выключения системы во время работы с приложениями или из-за блокировки пользовательского терминала и остановки приложения.
Включен Errlog	Включен Errdaemon	Это сообщение заносится в протокол демоном **error** при создании протокола ошибок. При отключении системы создание протокола ошибок будет прервано автоматически.
Блок Lion неисправен	Прервано соединение с 64-портовым концентратором	Эта ошибка заносится в протокол драйвером 64-портового концентратора после прерывания соединения с концентратором. Обратите внимание на дату и время возникновения ошибки - возможно, ошибка вызвана неверной работой с программой. Если вы получили несколько подобных сообщений, то это значит, что возникла неполадка 64-портового адаптера или аппаратного обеспечения, связанного с этим адаптером.
Переполнение буфера Lion	Переполнение буфера: 64-портовый концентратор	Сообщение об этой ошибке появляется после переполнения аппаратного буфера в 64-портовом концентраторе. Если устройство и кабель позволяют добавить запрос на передачу квитирования (RTS) на порт и устройство, добавьте его. Также постарайтесь снизить скорость передачи в бодах.
Lion Chunknumc	Сбой счетчика буфера: 64-портовый контроллер	Сообщение об этой ошибке появляется в том случае, если число символов в участке памяти не соответствует действительным значениям в буфере. Это сообщение также может появляться из-за неполадок аппаратного обеспечения. В этом случае запустите программу диагностики устройств.
Lion Hrdwre	Невозможно получить доступ к памяти на 64-портовом контролере.	Эта ошибка заносится в протокол драйвером 64-портового концентратора, если ему не удается получить доступ к памяти 64-портового контроллера.
Lion Mem ADAP	Невозможно выделить память: структура ADAP	Эта ошибка заносится в протокол драйвером 64-портового концентратора после сбоя в процедуре **malloc** для структуры adap.
Lion Mem List	Невозможно выделить память: Список TTYP_T	Сообщение об этой ошибке заносится в протокол драйвером 64-портового концентратора, если ему не удалось выделить память для структуры списка *ttyp_t* с помощью функции **malloc**.
Lion Pin ADAP	Ошибка обращения к памяти: структура ADAP	Эта ошибка заносится в протокол драйвером 64-портового концентратора после сбоя в процедуре **pin** для структуры adap.

Таблица 100. Сообщения об ошибках TTY (продолжение)

Сообщение	Описание	Комментарии
SRC	Ошибка программы	Эта ошибка заносится в протокол демоном Системного контроллера ресурсов (SRC) в случае обнаружения аварийных условий. Аварийные условия разделяются на три категории: ошибки в подсистемах, ошибки связи и другие ошибки.
Lion Unkchunk	Неизвестный код ошибки 64-портового концентратора	Код ошибки: Число символов в полученном блоке данных.
TTY Badinput	Поврежден кабель или разорвано соединение	Система не успевает обрабатывать данные ввода, поступающие через этот порт, и часть данных удаляется. Как правило, ошибки ввода бывают вызваны получением за короткий промежуток времени нескольких повторяющихся сигналов RS-232 с разными значениями состояния, когда обработчик прерываний использует слишком много процессорного времени. Ошибки сигналов обычно возникают в результате отключения или выхода из строя одного из разъемов; из-за поврежденного, незаземленного, неизолированного кабеля или помех в линии связи.
Переполнение в TTY	Переполнение получателя при вводе	В большинстве портов TTY применяется буфер FIFO ввода размером в 16 символов, а значение по умолчанию задает прерывание при получении каждых 14 символов. Это сообщение об ошибке выводится в случае, если обработчик прерываний драйвера очистил буфер ввода FIFO и данные были утеряны. Возможные способы исправления зависят от применяемого аппаратного обеспечения: • 8-портовые и 128-портовые адаптеры Проверьте правильность настройки функции управления потоком. Если она настроена правильно, запустите диагностику и замените неисправное аппаратное обеспечение. • Внутренние порты Если неполадка возникла в системе со свободными ресурсами, то перенесите нагрузку на другой порт. Если неполадка будет устранена, обновите встроенное программное обеспечение системы. • Универсальные способы исправления – Уменьшите значение параметра "порог активации RECEIVE" для данного порта с 3 до 2 или 1. – Уменьшите скорость передачи линии связи для этого порта. – Проверьте другие устройства и процессы, чтобы попытаться сократить время, затрачиваемое системой при выключенных прерываниях.

Таблица 100. Сообщения об ошибках TTY (продолжение)

Сообщение	Описание	Комментарии
TTY TTYHOG	Переполнение TTYHOG	Эта ошибка, как правило, бывает вызвана несоответствием способов управления потоком, применяемых передающей и принимающей сторонами. Драйвер терминала несколько раз попытался приостановить передачу на удаленном устройстве, но ввод не остановился, в результате чего были удалены данные. Убедитесь, что в передающей и принимающей системах применяется один и тот же способ управления потоком.
TTY Parerr	Ошибка четности/формирования кадра при вводе	Это сообщение означает, что обнаружены ошибки контроля четности в поступающих данных для асинхронных портов, работающих по принципу символ-за-символом. Эта ошибка обычно бывает вызвана несовпадением параметров управления линией (контроль четности, скорость линии, размер символа, количество стоп-битов) в передающей и принимающей системах. Для эффективного обмена данными в обеих системах должны быть заданы одинаковые параметры управления линией.
TTY Prog PTR	Внутренняя ошибка драйвера	Эта ошибка заносится в протокол драйвером tty, если указатель *t_hptr* нулевой.

Очистка зависшего порта терминала:

В этом примере очистки порта предполагается, что завис порт терминала `tty0`.

Для выполнения описанной ниже процедуры необходимо обладать правами доступа пользователя root.

1. Введите следующую команду, чтобы узнать, обрабатывает ли терминал какие-либо процессы:

   ```
   ps -lt tty0
   ```

   Вывод команды должен выглядеть следующим образом:

   ```
        F S UID   PID  PPID   C PRI NI ADDR    SZ    WCHAN    TTY  TIME CMD
   240001 S 202 22566  3608   0  60 20 781a   444 70201e44   tty0  0:00 ksh
   ```

   В данном случае ИД процесса (PID) равен 22566. Введите следующую команду, чтобы удалить этот процесс:

   ```
   kill 22566
   ```

   Убедитесь в успешном удалении процесса с помощью команды `ps -lt tty0`. Если процесс не был удален, введите команду kill с флагом `-9`, как это показано ниже.

   Примечание: Не указывайте флаг -9 для удаления процесса slattach. В результате удаления процесса slattach с флагом `-9` в файле `/etc/locks` может остаться блокировка. Удалите этот файл блокировок, чтобы выполнить очистку памяти после выполнения команды slattach.

   ```
   kill -9 22566
   ```

2. Введите следующую команду, чтобы определить наличие процессов, применяющих данный терминал:

   ```
   ps -ef | grep tty0
   ```

   Примечание: Если вывод команды **ps -ef | grep tty** выглядит примерно следующим образом:

   ```
   root 19050      1    0    Mar 06       -  0:00 /usr/sbin/getty /dev/tty
   ```

где между датой (Mar 06) и временем (0:00) указан символ "-", то к терминалу подключен неправильный кабель. Это состояние указывает, что процесс входа в систему (getty) пытается открыть терминал, а процесс открытия терминала завис, так как сигнал обнаружения несущей RS-232 (DCD) не был подтвержден. Эту неполадку можно устранить, подключив правильный нуль-модемный адаптер. Если процессу getty удастся открыть порт, то вместо символа "-" будет указан номер терминала. Дополнительная информация о кабелях приведена в разделе “Подключение модема с помощью кабеля” на стр. 601.

Примечание: Следующая команда позволяет отключить процесс входа в систему на терминале tty0.

```
pdisable tty0
```

Если процесс был очищен, но терминал все еще не отвечает, то перейдите к следующему шагу.

3. Введите следующую команду:

   ```
   fuser -k /dev/tty0
   ```

   Она удаляет все процессы, работающие с данным портом, и показывает соответствующие PID. Если терминал все еще не отвечает, перейдите к следующему шагу.

4. С помощью команды **strreset** удалите исходящие данные из порта, который завис из-за невозможности доставить данные в результате разрыва соединения с удаленной системой.

   Примечание: Если команда **strreset** устранит неполадку, значит этот порт неправильно подключен или настроен, так как при потере соединения с удаленной системой удаление данных из буфера должно происходить автоматически.

   Сначала необходимо узнать основной и дополнительный номера устройства терминала с помощью следующей команды:

   ```
   ls -al /dev/tty0
   ```

   Вывод команды будет выглядеть приблизительно следующим образом:

   ```
   crw-rw-rw-    1 root     system    18,   0 Nov  7  06:19 /dev/tty0
   ```

   Это означает, что у терминала tty0 основной номер устройства - 18 и дополнительный номер - 0. Укажите эти номера в команде **strreset**:

   ```
   /usr/sbin/strreset -M 18 -m 0
   ```

   Если терминал все еще не отвечает, перейдите к следующему шагу.

5. Отключите и снова подключите кабель к неотвечающему порту. AIX определяет наличие подключенного к порту устройства с помощью сигнала обнаружения несущей (DCD). Прерывание сигнала DCD путем отключения и подключения кабеля обычно позволяет сбросить зависшие процессы.

   Для того чтобы найти порт, к которому подключен терминал, введите следующую команду:

   ```
   lsdev -Cl tty0
   ```

   Вывод команды должен выглядеть следующим образом:

   ```
   tty0    доступен  00-00-S1-00  Асинхронный терминал
   ```

   В третьем столбце приведенного выше вывода команды указан код расположения терминала. В данном примере S1 указывает, что последовательный порт настроен для внутреннего последовательного порта 1. Дополнительная информация по интерпретации кодов приведена в разделе Коды расположения устройств в *Управление операционной системой и устройствами.*

   Если терминал все еще не отвечает, перейдите к следующему шагу.

6. Сбросьте порт с помощью команды **stty-cxma**. Введите следующую команду:

   ```
   /usr/lbin/tty/stty-cxma flush tty0
   ```

Эта команда предназначена для терминалов, подключенных к портам 8-портовых и 128-портовых адаптеров. Однако в некоторых случаях с помощью этой команды можно сбрасывать и другие порты терминалов.

Если терминал все еще не отвечает, перейдите к следующему шагу.

7. На клавиатуре зависшего терминала, удерживая клавишу `Ctrl`, нажмите клавишу `Q`. При этом будет передан символ **Xon**, и приостановленный вывод возобновится.

   Если терминал все еще не отвечает, перейдите к следующему шагу.

8. Иногда программы открывают порт терминала, изменяют некоторые атрибуты и закрывают порт, не восстанавливая прежних значений атрибутов. Для исправления этой ситуации отключите терминал, переведя его в состояние DEFINED, и снова включите его, воспользовавшись следующими командами:

   ```
   rmdev -l tty0
   ```

   Эта команда отключает терминал, не удаляя информацию о нем из базы данных.

   Следующая команда снова включает терминал:

   ```
   mkdev -l tty0
   ```

   Если неполадку на терминале устранить не удастся, то попробуйте подключить устройство к другому порту и настроить терминал на этом порте, пока не представится возможность перезагрузить систему. Если после перезагрузки неполадка не будет устранена, то, скорее всего, она вызвана неисправностью аппаратного обеспечения. Введите следующую команду, чтобы просмотреть отчет об ошибках аппаратного обеспечения порта:

   ```
   errpt -a | pg
   ```

Некоторые из указанных выше команд не будут выполняться и выведут сообщения об ошибке, указав, что устройство занято. Это вызвано процессом, работающим на терминале. Если ни один из описанных выше способов устранения неполадки не позволил возобновить работу терминала, то перезагрузите систему AIX и сбросьте ядро, чтобы удалить этот процесс.

Модемы

Модемы - это устройства, позволяющие устанавливать соединения по обычным телефонным линиям. К основным сведениям о модемах относятся стандарты и общие принципы настройки модемов, а также советы по работе с наиболее распространенными типами модемов.

Модем - это устройство, которое позволяет устанавливать соединение между компьютерами по обычной телефонной линии. Современная система телефонных линий не может передавать сигналы путем изменения напряжения, что необходимо для прямой цифровой связи. Поэтому для передачи по телефонной линии модем преобразует цифровую информацию в звуковые сигналы, а при получении этих сигналов преобразует их обратно в цифровые данные. Для работы с модемами, как правило, применяется программа Основные сетевые утилиты (BNU) или другие версии Программы копирования UNIX-UNIX (UUCP). При использовании высокоскоростных модемов (14400 бит/с и более) для передачи данных по Протоколу управления передачей/протоколу Internet (TCP/IP) применяется Протокол подключения к Internet по последовательной линии (SLIP).

Для описания скорости передачи данных модема (быстродействия модема) помимо термина бит/с употребляется термин *бод*. Бод - это единица измерения частоты модуляции. В модемах устаревших моделей одним изменением сигнала кодировался только один бит, поэтому скорость передачи данных в бодах для модема была равна скорости передачи данных в бит/с. Тем не менее, даже модемы с более высокой скоростью передачи данных по-прежнему работают на частоте 2400 (или даже 1200) бод, но кодируют два или более бит за период изменения сигнала. Скорость передачи данных в бит/с равна произведению числа бит данных, передаваемых за сигнал, на значение бод (например, 2400 бод x 6 бит за период изменения сигнала = 14400 бит в секунду). Современные модемы поддерживают различные скорости передачи данных (например, 28800, 14400, 9600, 7800, 4800 и 2400 бит/с).

Стандарты связи

Ранее были разработаны стандарты для модемов со скоростью передачи 300, 1200 и 2400 бит/с. Со временем технологии производства модемов с высоким быстродействием усовершенствовались, и современные производители предлагают модемы, не совместимые с модемами других фирм. На сегодняшний день стандарты высокопроизводительных линий связи определяются организацией ITU-TSS (бывший Консультативный комитет ООН по международной телефонной и телеграфной связи).

Модемы даже с самым высоким быстродействием работают гораздо медленнее других средств компьютерной связи. Модем с высоким быстродействием передает данные со скоростью 28800 бит/с, в то время как по соединению Ethernet данные передаются со скоростью 10000000 бит/с. Для повышения скорости передачи данных модемы с высоким быстродействием обычно поддерживают один или несколько алгоритмов сжатия данных. С помощью этих алгоритмов можно увеличить производительность модема до 57600 бит/с (при скорости передачи 14400 бит/с) или 115200 бит/с (при скорости передачи данных 28800 бит/с). Учтите, что применение алгоритмов сжатия напрямую зависит от передаваемых данных. Если данные уже были предварительно сжаты (например, с помощью команды **compress**), то сжатие передаваемых данных не увеличит скорость передачи, а скорее наоборот - замедлит ее. Применение технологии сжатия данных может значительно повысить скорость передачи данных по соединению DTE/DCE между компьютером и модемом. Например, скорость передачи данных модемом V.32*bis* со сжатием данных V.42*bis* (скорость передачи по телефонной линии) - 14400 бит/с. Алгоритм сжатия V.42*bis* позволяет повысить фактическую скорость передачи данных до 57600 бит/с. Для наибольшей эффективности при применении сжатия следует выбрать скорость передачи данных соединения между компьютером и модемом равной 57600 бит/с.

Все стандарты для линий связи с высокой скоростью передачи данных, в том числе алгоритмы сжатия данных, определяются организацией ITU-TSS. Стандарты ITU-TSS обычно называются V.*nn*, где *nn* - некоторый номер. Кроме того, применяется стандарт Microcom Networking Protocol (MNP). Стандарт MNP существует в версиях 1-9 (называемых классами). Этот высокопроизводительный, высокоскоростной протокол был разработан достаточно давно и фактически являлся основным стандартом до появления стандартов ITU-TSS.

Полудуплексный и дуплексный режим связи:

При рассмотрении стандартов связи важно понимать разницу между полудуплексным и дуплексным режимами.

При *полудуплексной* (HDX) передаче пакет данных отправляется одной системой и принимается другой. Следующий пакет данных нельзя отправить, пока принимающая система не подтвердит получение предыдущего пакета.

При полностью *дуплексной* (FDX) передаче обе системы, и отправляющая и принимающая, могут передавать данные одновременно. Это означает, что модем может принимать один пакет данных и одновременно подтверждать получение другого.

Стандарты связи ITU-TSS:

Ниже приведен список основных стандартов связи, определенных ITU-TSS.

Учтите, что этот список не полный. Полный список приведен на веб-сайте организации ITU-TSS.

Элемент	Описание
V.29	Стандарт ITU-TSS для полудуплексных соединений со скоростью передачи данных 9600 бит/с.
V.32	Стандарт ITU-TSS для дуплексных соединений со скоростью передачи данных 9600 бит/с.
V.32*bis*	Стандарт ITU-TSS для соединений со скоростью передачи данных 14400. V.32*bis* - это дальнейшее развитие стандарта V.32.
V.34	Стандарт ITU-TSS для соединений со скоростью передачи данных 33600 бит/с. Заметим, что в этом стандарте данные передаются со скоростью 33600 бит/с благодаря кодированию нескольких бит, а не сжатию данных, применяемому в MNP класса 9. Этот стандарт ранее назывался V.*fast*.
V.42	Процедуры исправления ошибок ITU-TSS для DCE при передаче данных между асинхронным и синхронным оборудованием.
V.42*bis*	Измененный стандарт ITU-TSS для сжатия данных.

Microcom Networking Protocol:

Еще одним фактическим стандартом является**Microcom Networking Protocol (MNP)**, разработанный компанией Microcom, Inc.

Стандарт **MNP** существует в версиях 1-9 (называемых классами). Этот высокопроизводительный, высокоскоростной протокол применялся до появления стандартов ITU-TSS. В **MNP** ошибки в пакетах передаваемых данных обнаруживаются удаленным модемом, который затем отправляет запрос на повторную передачу ошибочного пакета. Возможность распознать и быстро исправить ошибки данных делает **MNP** одним из наиболее распространенных на сегодня протоколов.

В следующей таблице приведены стандарты связи **MNP**.

Элемент	Описание
MNP класса 1	Асинхронный полудуплексный побайтовый способ передачи данных позволяет реализовать лишь 70% возможностей линии. В современных модемах этот стандарт применяется редко.
MNP класса 2	Дуплексный аналог **MNP класса 1** также редко применяется в современных модемах.
MNP класса 3	Синхронный, побайтовый дуплексный способ передачи данных. Реализует около 108% физических возможностей линии. Эффективность, превышающая 100%, достигается за счет удаления необходимых для асинхронного соединения старт- и стоп-битов. DTE/DCE между модемом и системой по-прежнему асинхронное.
MNP класса 4	Дальнейшее развитие **MNP класса 3**, включающее механизм изменения размера пакета (настраиваемая сборка пакета) и средства удаления лишних управляющих данных (оптимизация этапа обработки данных). Модемы **MNP класа 4** обеспечивают производительность на уровне примерно 120%.
MNP класса 5	Обеспечивает сжатие данных, как и Класс 4. Модем **MNP класа 5** обеспечивает производительность на уровне примерно 200%.
MNP класса 6	Позволяет реализовать в одном модеме несколько несовместимых способов модуляции (универсальное согласование линии). Это обеспечивает для модемов **MNP класса 6** значительное увеличение скорости передачи данных. В **MNP класса 6** включена статистическая дуплексная схема, которая динамически изменяет полудуплексную модуляцию для эмуляции дуплексной связи. Поддерживаются все функции **MNP класса 5**.
MNP класса 7	Обеспечивает улучшенное сжатие данных. При объединении с классом 4 позволяет добиться эффективности около 300%.
MNP класса 8	Неприменим.
MNP класса 9	Для увеличения скорости передачи данных до 28 800 бит/с применяется улучшенное сжатие вместе с технологией V.32.

Замечания по настройке модемов

Требования к интерфейсу модема для обычного пользователя могут различаться.

Конфигурация модема, подключенного к этой операционной системе, отличается от конфигураций модемов персональных компьютеров (PC) или рабочих станций.

Поддерживаемые модемы:

К этой операционной системе можно подключить любой модем, совместимый с EIA 232 и способный возвращать результаты в ответ на команду.

Обработка обнаружения несущей для данных.:

Сервер использует сигнал обнаружения несущей (DCD) для наблюдения за состоянием модема.

Если сигнал DCD порта модема "высокий", сервер полагает, что модем в настоящее время используется. По этой причине, важно знать, какие обстоятельства могут повлиять на переход сигнала в "высокое" состояние. Сигнал DCD может стать высоким по следующим причинам:

- Использования значения **clocal** в атрибутах stty для поля времени выполнения в панели SMIT **Конфигурация трминала**.
- **Включение** опции Игнорировать определение несущей в панели SMIT **Конфигурация терминала** для терминалов, подключенных к 128-портовому адаптеру.
- Модем принудительно повышает сигнал DCD из-за команд AT или переключателей.
- Порт терминала уже используется другим приложением.

  Примечание: Когда модем подключается к другому модему, он повышает CD. Большинство значений по умолчанию модема всегда устанавливают "высокое" состояние этого сигнала, даже если модем простаивает. Уровень сигнала CD не должен повышаться принудительно.

Быстродействие Data Terminating Equipment или Data Circuit-Terminating Equipment:

Понятия Data Terminating Equipment (DTE) и Data Communication Equipment (DCE) используются для описания двух разных групп аппаратного обеспечения.

Термин DTE в основном применяется для устройств, показывающих пользовательскую информацию. К ним также относятся устройства, предназначенные для хранения или создания пользовательских данных. В частности, к DTE относятся все системные блоки, терминалы и принтеры.

В категорию DCE входят все устройства, которые используются для доступа к системам по телекоммуникационным линиям. Наиболее распространенные устройства DCE - это модемы и мультиплексоры.

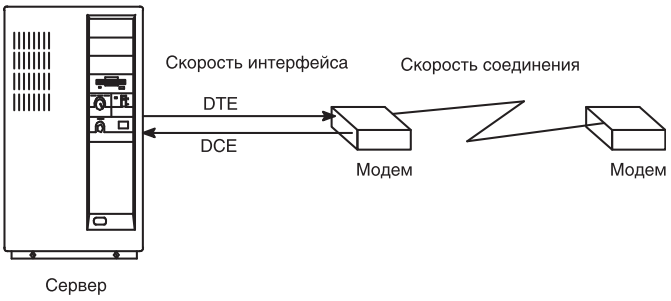


Рисунок 37. Замечания по быстродействию модемов

При использовании последовательного модемного соединения в этой операционной системе, как показано на приведенном выше рисунке, следует принять во внимание три основных фактора:

- Быстродействие интерфейса DTE (от сервера к модему). Это скорость передачи информации от сервера к модему.
- Быстродействие интерфейса DCE (от модема к серверу), также называемое "быстродействие интерфейса последовательного порта". Это скорость передачи информации от модема к серверу.
- Быстродействие соединения (от модема к модему). Это скорость передачи информации от одного модема к другому.

Большинство современных высокоскоростных модемов допускают, чтобы быстродействие интерфейса DCE отличалось от быстродействия соединения. Это позволяет зафиксировать быстродействие DTE, в то время как быстродействие соединения между модемами может изменяться в ту или другую сторону по необходимости.

Современные высокоскоростные модемы могут хранить предназначенные для отправки данные в буфере и отправлять их, когда система будет готова к приему. Они также могут хранить в буфере данные, предназначенные для отправки другому модему, и отправлять их, когда модем будет готов к приему. Этот тип передачи данных требует, чтобы модем и сервер использовали *управление потоком*.

Управляющие сигналы модема:

Модемы часто используются для вызова удаленных систем и приема звонков. Поэтому важно запрограммировать модем так, чтобы он устанавливал соединение на максимальной возможной скорости и сбрасывался после завершения соединения.

Сервер выключит Сигнал готовности терминала (DTR), дав команду модему завершить соединение. Большинство модемов могут сбрасываться после получения сигнала выключения DTR.

Примечание: Можно настроить терминал таким образом, чтобы он не сбрасывал DTR. Для этого нужно отключить параметр **hupcl** в атрибутах выполнения stty.

Для того чтобы возможности соединения модема и сервера использовались в полной мере, соединение по кабелю должно соответствовать следующим требованиям:

- Оно должно отвечать техническим требованиям.
- Оно должно быть правильно изолировано.
- Оно должно обеспечивать передачу следующих сигналов: RxD, TxD, RTS, CTS, SG, DCD и DTR.

  Примечание: 16-портовый асинхронный адаптер не поддерживает сигналы RTS и CTS. По этой причине, с данным адаптером нельзя использовать аппаратное управление потоком RTS/CTS.

При передаче двоичных данных с помощью модема и этого адаптера следует использовать протокол передачи файлов, который обнаруживает неверные данные и повторно отправляет потерянные данные (например, Xmodem, zmodem, Kermit или UUCP).

Ниже описаны сигналы, используемые сервером:

Сигнал	Описание
FG	Корпус. Контакт 1 по спецификации EIA 232D экранированных кабелей. При правильном использовании сигнал подключен к контакту 1 только на одной стороне кабеля и подключен к металлической оболочке кабеля.
TxD	Передать данные. Контакт 2 по спецификации EIA 232D. Данные передаются по этому сигналу. Контролируется сервером.
RxD	Принять данные. Контакт 3 по спецификации EIA 232D. Данные принимаются по этому сигналу, отправляемому модемом. Контролируется модемом.
RTS	Готовность к отправке. Контакт 4 по спецификации EIA 232D. Используется при управлении потоком RTS/CTS. Уровень сигнала повышается, когда система готова к отправке данных, и падает, когда модем должен прекратить отправку данных.
CTS	Готовность к приему. Контакт 5 по спецификации EIA 232D. Используется при управлении потоком RTS/CTS. Уровень сигнала повысится, когда модем будет готов к отправке или приему данных. Уровень сигнала упадет, когда модем запросит сервер прекратить отправку данных. Контролируется модемом.
DSR	Сигнал готовности к отправке данных. Контакт 6 по спецификации EIA 232D. Сообщает серверу о том, что модем готов к работе. Контролируется модемом.
SG	Земля сигнала. Контакт 7 по спецификации EIA 232D. Этот сигнал указывает напряжение для других сигналов.
DCD	Обнаружение несущей для данных. Контакт 8 по спецификации EIA 232D. Этот сигнал сообщает серверу, что модем установил соединение с другим модемом. Когда уровень этого сигнала повышается, выполняемые на сервере программы смогут использовать порт. Контролируется модемом.

Сигнал	Описание
DTR	Сигнал готовности терминала. Контакт 20 по спецификации EIA 232D. Указывает модему, что сервер активен и готов к приему соединения. Уровень сигнала падает, когда сервер требует от модема прервать соединение с другим модемом. Уровень сигнала повышается при открытии порта. Контролируется сервером.
RI	Индикатор звонка. Контакт 22 по спецификации EIA 232D. Этот сигнал сообщает серверу, что модем принимает звонок. Этот сигнал используется довольно редко и в обычных ситуациях не требуется. Контролируется модемом.

Подключение кабеля модема

В следующей таблице приведена информация о кабелях для подключения модема через последовательные контроллеры.

Адаптер/Контроллер	Код(ы) изделий IBM
Встроенный последовательный порт (S1 или S2)	00G0943*, 6326741
2-портовый контроллер	00G0943*, 6326741
8-портовый контроллер	6323741
128-портовый контроллер	43G0935, 6323741

Код изделия IBM	Описание	Длина
00G0943*	Перемычка последовательного порта (pigtail)	10 см
6323741	Асинхронный кабель	10
43G0935	Кабель с переходником с разъема RJ-45 на DB25	61 см

*Это изделие не требуется для некоторых типов систем.

Настройка терминала в операционной системе

С помощью инструмента управления системой (SMIT) определите порт для подключения терминала.

Большинство полей предназначены для устройства общего типа. Единственное поле, значение которого влияет на работу модема, - поле включить вход в систему, в котором можно указать следующие значения:

Элемент	Описание
Выключить	Для порта не запускаются процессы getty. Этот параметр применяется для портов модемов, обрабатывающих только исходящие звонки.
Включить	Для порта запускается процесс getty. Этот параметр предназначен для настройки модемов, обрабатывающих только входящие звонки.
Общий	Для порта запускается процесс getty, который позволяет программам работать с данным портом в любом режиме, не включая и не выключая параметр. Этот параметр предназначен для двунаправленного применения порта.
Задержка	Процесс getty запускается для порта в двухстороннем режиме, но уведомление не отправляется, пока процесс getty не получит от пользователя сигнал с клавиатуры.

Поля, относящиеся к асинхронному адаптеру со 128 портами:

Элемент	Описание
Принудительная несущая или игнорировать обнаружение несущей	отключить*
Выполнять предварительную обработку в адаптере	отключить

Примечание: Параметр, отмеченный звездочкой (*) выключается при использовании 10-штырькового разъема RJ-45. Для 8-штырькового разъема RJ-45 этот параметр следует включить.

Подключение модема с помощью кабеля

Первое действие, которое необходимо выполнить при установке модема - это подключение кабелей.

Ниже описаны типы кабелей и их кодовые номера.

6323741

Кабель EIA-232 применяется для подключения ко всем асинхронным устройствам, а также в сочетании с другими кабелями и переходниками.

59F3740

Переходник с 10- и 25-штырьковыми разъемами применяется для подключения асинхронного кабеля 6323741 к встроенным последовательным портам S1 и S2, как показано на следующем рисунке.

На рисунке показан переходник с 10-штырькового разъема на 25-штырьковый.

Рисунок 38. Переходник с 10- на 25-штырьковый разъем

Примеры кабельных соединений:

1. Для подключения модема к встроенному последовательному порту S1 применяются следующие кабели:

   На рисунке показан кабель с разъемом 59F3740 со стороны последовательного порта и разъемом

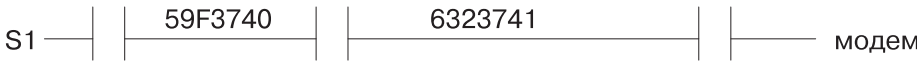


Рисунок 39. Кабель для подключения модема к последовательному порту

   6323741 со стороны модема.

2. Для подключения модема к интерфейсному кабелю 8-портового асинхронного адаптера (EIA-232) применяются следующие кабели:

   На рисунке показан 8-портовый интерфейс, подключенный к модему с помощью кабеля 6323741.

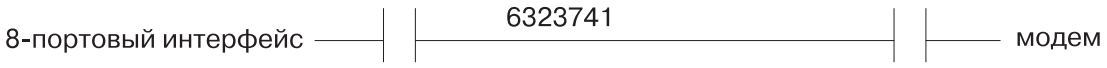


Рисунок 40. Подключение 8-портового интерфейса к модему

Добавление терминала для модема

Воспользуйтесь данной информацией при добавлении терминала для модема.

Система должна быть включена, а модем - выключен. Выполните команду быстрого доступа SMIT `smit mktty`.

Настройка модема

Для настройки модема воспользуйтесь одним из описанных здесь способов.

Если в системе установлен компонент Основные сетевые утилиты (BNU), то обратитесь к разделу “Отправка команд AT с помощью команды cu”. Если этот компонент не установлен, обратитесь к разделу “Отправка команд AT с помощью программы на C” на стр. 603. Инструкции по установке BNU приведены в разделе “Основные сетевые утилиты” на стр. 438.

Отправка команд AT с помощью команды cu:

Если в системе установлены Основные сетевые утилиты (BNU), то для настройки модема можно воспользоваться командой **cu**.

Команды и параметры, описанные в этом разделе, предназначены для настройки Hayes-совместимого модема с основными параметрами, необходимыми для подключения к последовательным портам сервера.

1. Добавьте следующую строку в файл **/usr/lib/uucp/Devices**. Если в файле эта строка уже есть, то ее добавлять не нужно. (Замените # на номер порта).

   ```
   Direct tty# - Any direct
   ```

2. Проверьте, выключен ли терминал, с помощью следующей команды:
   ```
   pdisable tty#
   ```
3. Введите следующую команду:
   ```
   cu -ml tty#
   ```

   Должно быть выведено сообщение `Подключен`.
4. С помощью следующей команды убедитесь, что модем принимает команды:
   ```
   AT
   ```

   Модем ответит `OK`. Если этого не произойдет, обратитесь к “Устранение неполадок модема” на стр. 605.

   Дополнительная информация о командах **AT** и их описания приведены в разделе “Команды AT” на стр. 607.
5. В зависимости от выбранной опции getty, введите одну из следующих команд. Замените *n* на устройство терминала.
   - `penable tty`*n*
   - `pshare tty`*n*
   - `pdelay tty`*n*
   - `pdisplay tty`*n*

Теперь на модеме настроено большинство команд, необходимых для выполнения функций обмена данными системы через последовательные интерфейсы. При возникновении ошибки вызовите команду **cu -dl** для диагностики соединения.

Отправка команд AT с помощью программы на C:

Если настроить модем с помощью команды **cu** не удалось, или в системе не установлен BNU, то запустите следующую программу на C.

Создайте файл с именем `motalk.c`, содержащий следующий код. Сохраните файл. Откомпилируйте его и запустите согласно инструкциям, приведенным в комментариях.

```
/*************************************************************/
/*  MoTalk - Программа на "C" для настройки модема.                */
/*           Эта программа приведена в качестве примера    */
/*           и не поддерживается фирмой IBM.                         */
/*             Для компиляции:  cc -o motalk motalk.c      */
/*                   Вызов:  motalk /dev/tty? [скорость]      */
/*************************************************************/
#include <errno.h>
#include<stdio.h>
#include <signal.h>
#include <fcntl.h>
#include <termio.h>
FILE *fdr, *fdw;
int fd;
struct termio term_save, stdin_save;
void Exit(int sig)
{
   if (fdr) fclose(fdr);
   if (fdw) fclose(fdw);
   ioctl(fd, TCSETA, &term_save);
   close(fd);
   ioctl(fileno(stdin), TCSETA, &stdin_save);
   exit(sig);
}
main(int argc, char *argv[])
{
   char *b, buffer[80];
   int baud=0, num;
```

```
struct termio term, tstdin;
if (argc < 2 || !strcmp(argv[1], "-?"))
{
   fprintf(stderr, "Вызов: motalk /dev/tty? [скорость]\n");
   exit(1);
}
if ((fd = open(argv[1], O_RDWR | O_NDELAY)) < 0)
{
   perror(argv[1]);
   exit(errno);
}
if (argc > 2)
{
   switch(atoi(argv[2]))
   {
      case   300: baud = B300;
                  break;
      case  1200: baud = B1200;
                  break;
      case  2400: baud = B2400;
                  break;
      case  4800: baud = B4800;
                  break;
      case  9600: baud = B9600;
                  break;
      case 19200: baud = B19200;
                  break;
      case 38400: baud = B38400;
                  break;
      default:    baud = 0;
                  fprintf(stderr, "%s: скорость %s не
                   поддерживается\n", argv[0],argv[2]);
                  exit(1);
      }
   }
/* Сохранить состояние stdin и tty; отслеживание некоторых сигналов */
ioctl(fd, TCGETA, &term_save);
ioctl(fileno(stdin), TCGETA, &stdin_save);
signal(SIGHUP, Exit);
signal(SIGINT, Exit);
signal(SIGQUIT, Exit);
signal(SIGTERM, Exit);
/*  Перевести stdin в линейный режим, выключить эхо */
ioctl(fileno(stdin), TCGETA, &tstdin);
tstdin.c_iflag = 0;
tstdin.c_lflag &= ~(ICANON | ECHO);
tstdin.c_cc[VMIN] = 0;
tstdin.c_cc[VTIME] = 0;
ioctl(fileno(stdin), TCSETA, &tstdin);
/*  Задать состояние tty */
ioctl(fd, TCGETA, &term);
term.c_cflag |= CLOCAL|HUPCL;
if (baud > 0)
{
   term.c_cflag &= ~CBAUD;
   term.c_cflag |= baud;
}
term.c_lflag &= ~(ICANON | ECHO); /* линейный режим */
term.c_iflag &= ~ICRNL; /* исключить пустые строки */
term.c_cc[VMIN] = 0;
term.c_cc[VTIME] = 10;
ioctl(fd, TCSETA, &term);
fcntl(fd, F_SETFL, fcntl(fd, F_GETFL, 0) & ~O_NDELAY);
/*  Открыть tty для чтения и записи */
if ((fdr = fopen(argv[1], "r")) == NULL )
{
   perror(argv[1]);
```

```
      exit(errno);
   }
   if ((fdw = fopen(argv[1], "w")) == NULL )
   {
      perror(argv[1]);
      exit(errno);
   }
   /*  Связь с модемом */
   puts("Готов... ^C для выхода");
   while (1)
   {
      if ((num = read(fileno(stdin), buffer, 80)) > 0)
         write(fileno(fdw), buffer, num);
      if ((num = read(fileno(fdr), buffer, 80)) > 0)
         write(fileno(stdout), buffer, num);
      Exit (0);
   }
}
```

Применение модемов Hayes и совместимых с Hayes

Эта процедура применяется для настройки модемов Hayes или совместимых с Hayes модемов.

1. При необходимости измените параметры терминала команды SMIT `smit chtty`. Например, вам может потребоваться изменить значение поля Разрешить вход в систему на **Общий** или **Разрешить**.
2. Добавьте в файл `/usr/lib/uucp/Systems` следующую строку:

   ```
   hayes Nvr HAYESPROG 2400
   ```

3. Добавьте следующие строки в файл `/usr/lib/uucp/Devices`:

   ```
   # Только для программирования модемов hayes:
   HAYESPROG tty0 - 2400 HayesProgrm2400
   #обычная запись ACU:
   ACU tty0 - Любой hayes
   ```

4. Добавьте следующие строки в файл `/usr/lib/uucp/Dialers`:

   ```
   # Эта запись служит только для программирования модема:
   # Следующие 5 строки следует ввести в виде одной длинной строки:
   HayesProgrm2400 =,-, "" \d\dAT\r\c OK AT&F\r\c OK ATM1\r\c OK
   AT&D3\r\c OK AT&K3&C1\r\c OK ATL0E0Q2\r\c OK ATS0=1\r\c OK AT&W\r\c
   OK
   hayes     =,-,     "" \dAT\r\c OK ATDT\T\d\r\c CONNECT
   ```

5. Для программирования модема введите команду `cu -d hayes`. Это команда применяет **cu** для настройки модема. Если соединение с другой системой не установлено, команда не будет выполнена. Если на экране будет показано `sendthem AT&W`, а затем `OK got it`, значит модем настроен успешно.

   Если вы не передаете двоичные файлы и не используете BNU, то не вызывайте команду **&K3** и выберите XON для управления потоком. Как правило, более эффективным оказывается аппаратное управление потоком, а не квитирование XON-XOFF. Для выбора этого способа управления потоком укажите значения и записи `Dialers`, описанные на следующем шаге.
6. После настройки модема вы можете разрешить драйверу устройства применять аппаратное управление потоком. Измените режим управления потоком на RTS с помощью SMIT (`smit chtty`). Узнайте с помощью руководства по модему, поддерживает ли модем аппаратное управление потоком.

Устранение неполадок модема

При обнаружении неполадок модема имейте в виду следующее.

- Некоторые модемы учитывают регистр символов. Вводите команды **AT** прописными буквами.
- В нормальном режиме работы рекомендуется, чтобы модем сбрасывался при отсутствии сигнала DTR (параметр &D3). Однако при первоначальной настройке модема рекомендуется отключить сброс модема при отсутствии сигнала DTR (параметр &D2). Если модем сбросится, то все запрограммированные параметры, не сохраненные в памяти модема, будут потеряны.

Отключение этой функции также позволяет защитить изменения в случае, если задан параметр &C1. В результате изменения состояния обнаружения несущей на некоторых модемах переключается линия обнаружения несущей, и команда **cu** разрывает соединение. Завершив настройку модема, целесообразно включить опцию &D3.

- Хотя в этом наборе разделов приведены стандартные для большинства Hayes-совместимых модемов команды, они не обязательно являются стандартными для вашего модема. Перед выполнением этой процедуры сверьте указанные команды с документацией по модему.

Удобным способом отладки любой неполадки модема является отключение модема и подключение ASCII-терминала (через переходник или нуль-модем) к тому же порту и через тот же кабель, который использует модем. Настройте терминал с теми же значениями быстродействия линии, числа битов на символ и контроля четности, что используются модемом. Если порт разрешает вход в систему, на терминале появится приглашение на вход в систему. Если на экране терминала появилось приглашение на вход в систему, значит, неполадки возникают из-за конфигурации модема.

Следующие советы помогают изолировать неполадки, связанные с соединениями по модему:

Таблица 101. Неполадки, связанные с соединением модема, и способы их устранения

Неполадка	Исправление
Слишком частое возобновление программы	Выполнение программы **getty** возобновляется с помощью **init**.
Сообщения на консоли	Если команде **init** приходится возобновлять выполнение какой-либо программы более пяти раз за 225 секунд, то она выдает сообщение на консоли и прекращает возобновление программы на некоторое время. Для устранения этой неполадки следует выяснить, почему прерывается работа программы **getty**. Это может происходить по нескольким причинам: • Неверные параметры модема, обычно объясняющиеся повышением CD на модемах или кабельных соединениях, а также включением опции "эхо" или "ответ команды". (CD также может быть повышен из-за добавления clocal в режиме выполнения и/или режиме входа в систему в конфигурацию порта, либо принудительно повышен на 128 порте.) • Выключение сигнала CD. Процесс **getty** будет завершаться каждый раз, когда сигнал CD переключается из состояния "включено" в "выключено". (Это может происходить по ряду причин. Убедитесь в том, что кабель правильно изолирован. Причиной также может быть несколько операций входа и выхода из системы, выполняемых за короткий промежуток.)
После соединения с модемом не появляется приглашение на вход в систему	Убедитесь, что для порта запущен процесс **getty**. Если это так, убедитесь в том, что сигнал обнаружения несущей соединения с модемом повышается после того, как модем соединился с удаленной системой. Если CD подается правильно, то убедитесь, что модем соединяется с правильным портом. Если вы по-прежнему не видите приглашения на вход в систему, то с помощью переходника подключите терминал вместо модема и проверьте, появляется ли приглашение на вход в систему. Если вы по-прежнему не видите приглашения, попробуйте включить эхоповтор введенных символов на терминале, чтобы убедиться в правильном подключении кабелей и нормальном функционировании аппаратного обеспечения.
После подключения удаленного модема соединение сразу же прерывается	Убедитесь, что модем подключается к серверу с той же скоростью передачи данных, которая установлена на сервере. Попробуйте установить другую скорость передачи на терминале, либо запрограммируйте модем так, чтобы он фиксировал скорость DTE в соответствии с быстродействием порта терминала. Убедитесь, что модем или порт не повышают сигнал обнаружения несущей и порт не используется другим процессом.
Вместо приглашения на вход в систему появляются посторонние символы	Это объясняется различиями в параметрах протоколов. Убедитесь, что модем и порт терминала применяют одинаковые параметры контроля четности, скорости передачи, управления потоком данных и длины символов.
Иногда после успешного сеанса работы вход в систему становится невозможен	Возможная причина - модем не сбрасывается после завершения соединения. Найдите в документации модема инструкции по сбросу модема после завершения соединения.
Буфер приема переполняется	Буфер чипа UART переполняется. Уменьшите значение триггера приема в SMIT для нужного терминала. Этот способ годится только для стандартных, 8- или 16-портовых адаптеров. Убедитесь в том, что модем и порт терминала применяют одинаковое управление потоком.
Ошибки ttyhog	Модем и терминал не смогли согласовать способ управления потоком, либо управление потоком не применяется вовсе.

Вопросник для обращения в службу поддержки для устранения неполадок модема:

Перед тем, как обратиться в службу технической поддержки, соберите следующую основную информацию:

К этим сведениям относятся:
- Версия операционной системы. Как давно вы работаете с этой версией операционной системы?
- Работал ли модем ранее?
- Тип вашего модема. Тип модема на другом конце линии связи.
- Тип адаптера, к которому подключен модем.
- Номер порта, к которому подключен модем.
- Номер терминала, к которому подключен модем.
- Тип кабеля.
- Настройка программы login (share, delay, enable).
- Удается ли вашему модему установить соединение с другими модемами?
- Удается ли другим модемам подключиться к вашему модему?
- Значения следующих параметров SMIT, конфигурации модема или порта.
  - XON/XOFF?
  - RTS/CTS?
  - Скорость в бит/с.
- Не забудьте указать в описании неполадки следующую информацию:
  - Происходит ли периодическое блокирование порта?
  - Удается ли отправить сигнал? Удается ли получить сигнал?
  - Встречаются ли какие-нибудь другие ошибки?
- Эти ошибки на консоли? Какие это ошибки?
- Занесены ли эти ошибки в отчет об ошибках? (**errpt** или **errpt -a**)
- Какая команда применяется для набора номера?
- Какие программы установлены в вашей системе?

Команды AT:

Набор команд Hayes Smartmodem включен в набор команд AT, применяемых модемами распространенных моделей.

Данная информация взята из документа *Quick Reference Card* для модема Hayes Smartmodem 2400, выпущенного компанией Hayes Microcomputer Products, Inc. Уточните список команд AT в документации модема.

Элемент	Описание
AT	Префикс команды - ставится в начале командной строки.
<CR>	Символ возврата каретки (новая строка) - ставится в конце командной строки.
A	Отмена текущей команды, возврат в командный режим.
A/	Повтор предыдущей команды. Перед этой командой не требуется указывать **AT**, а после нее не нужно указывать **<CR>**/.
B0	Выбор стандарта CCITT V.22 для соединений со скоростью передачи данных 1200 бит/с.
B1	Выбор стандарта Bell 212A для соединений со скоростью передачи данных 1200 бит/с.
D	Включение режима вызова, набор номера и попытка установить соединение. После D обычно указывается T, если применяется тоновый набор, или P, если набор - импульсный.
DS=*n*	Набирает номер из ячейки *n*
E0	Отключить отображение символов в командном режиме.
E1	Включить отображение символов в командном режиме.
H0	Разрыв соединения (вешает трубку).
H1	Поднимает трубку в модеме.

Элемент	Описание
I0	Выдает идентификационный код продукта.
I1	Вычисляет контрольную сумму ПЗУ производителя и показывает ее на экране.
I2	Вычисляет контрольную сумму ПЗУ производителя; в результате показывает значение OK или ERROR.
L0	Выключает динамик модема.
L1	Маленькая громкость динамика модема.
L2	Обычная громкость динамика модема.
L3	Большая громкость динамика модема.
M0	Выключает динамик модема.
M1	Отключает динамик модема после обнаружения несущего сигнала.
M2	Динамик модема включен всегда.
M3	Динамик модема включен до тех пор, пока не обнаружен несущий сигнал, но не во время набора номера.
O0	Диалоговый режим.
O1	Входит в диалоговый режим и инициирует тест линии.
Q0	Модем возвращает коды завершения.
Q1	Модем не возвращает коды завершения.
Sr	Переводит указатель на регистр r.
Sr=*n*	Присваивает регистру r значение *n*.
V0	Показывает коды завершения в числовом формате.
V1	Показывает коды завершения в подробном формате (словами).
X0	Активизирует функции, представленные кодами завершения 0-4.
X1	Активизирует функции, представленные кодами завершения 0-5, 10.
X2	Активизирует функции, представленные кодами завершения 0-6, 10.
X3	Активизирует функции, представленные кодами завершения 0-5, 7, 10.
X4	Активизирует функции, представленные кодами завершения 0-7, 10.
Y0	Отменить отключение при длительном разрыве связи.
Y1	Разрешить отключение при длительном разрыве связи.
Z	Выполнить сброс модема
&C0	Считать, что несущая частота всегда присутствует.
&C1	Проверить наличие несущей частоты.
&D0	Игнорировать сигнал DTR.
&D1	Включать командный режим при переходе сигнала DTR из включенного состояния в выключенное.
&D2	Вешает трубку в модеме и включает командный режим при переходе сигнала DTR из включенного состояния в выключенное.
&D3	Выполняет сброс модема при переходе сигнала DTR из включенного состояния в выключенное.
&F	Восстанавливает в модеме первоначальные параметры конфигурации.
&G0	Отключение защитного тона.
&G1	Защитный тон с частотой 500 Гц.
&G2	Защитный тон с частотой 1800 Гц.
&J0	Телефонной разъем RJ-11/RJ41/RJ45S.
&J1	Телефонной разъем RJ-11/RJ-13.
&P0	Импульсный набор с отношением импульс/пауза, равным 39/61.
&P1	Импульсный набор с отношением импульс/пауза, равным 33/67.
&Q0	Работа в асинхронном режиме.
&Qn	Работа в синхронном режиме *n*
&R0	Обнаружение CTS согласно RTS.
&R1	Игнорировать RTS; всегда считать, что CTS присутствует.
&S0	Считать, что сигнал DSR присутствует.
&S1	Проверить наличие сигнала DSR.
&T0	Завершить выполняемый тест.
&T1	Включить внутреннюю аналоговую циклическую проверку.
&T3	Включить цифровую циклическую проверку.
&T4	Предоставляет запрос на удаленный канал передачи данных (RDL) от удаленного модема.
&T5	Отклоняет запрос на RDL от удаленного модема.
&T6	Включить удаленную циклическую проверку.
&T7	Включить удаленную циклическую проверку с самотестированием.
&T8	Включить внутреннюю аналоговую циклическую проверку с самотестированием.
&V	Просмотреть активную конфигурацию, пользовательский профайл и сохраненные значения.
&Wn	Сохранить параметры активной конфигурации как пользовательский профайл *n*.

Элемент	Описание
&X0	Синхронизация передачи обеспечивается модемом.
&X1	Синхронизация передачи обеспечивается терминалом.
&X2	Синхронизация передачи обеспечивается полученным несущим сигналом.
&Yn	Восстановить пользовательский профайл *n*.
&Zn=*x*	Сохранить телефонный номер *x* в ячейке *n*.

Обзор регистров S:

В следующей таблице приведены регистры S, их диапазоны и описания.

Таблица 102. Описания регистров S

Регистр	Диапазон	Описание
S0	0-255	Выберите число звонков до ответа модема.
S1	0-255	Прошедшее число звонков (с увеличением на единицу после каждого звонка).
S2	0-127	Определяет символ escape-последовательности (ASCII).
S3	0-127	Определяет символ возврата каретки (ASCII).
S4	0-127	Определяет символ перевода строки (ASCII).
S5	0-32, 127	Определяет символ Backspace (ASCII).
S6	2-255	Задает время в секундах до начала набора номера "вслепую".
S7	1-55	Задает время ожидания сигнала в секундах.
S8	0-255	Выберите продолжительность паузы по запятой, в секундах.
S9	1-255	Время ответа при обнаружении несущей частоты, в десятых долях секунды.
S10	1-255	Пауза между потерей несущего сигнала и моментом, когда модем вешает трубку (в десятых долях секунды).
S11	50-255	Продолжительность/ интервал тонов в миллисекундах.
S12	50-255	Время для управляющей последовательности, в десятых долях секунды.
S13	—	Зарезервировано.
S14	—	Зарезервировано.
S15	—	Зарезервировано.
S16	—	Зарезервировано. Функции этого регистра управляются командами **&T**.
S17	—	Зарезервировано.
S18	0-255	Продолжительность теста в секундах.
S19	—	Зарезервировано.
S20	—	Зарезервировано.
S21	—	Зарезервировано.
S22	—	Зарезервировано.
S23	—	Зарезервировано.
S24	—	Зарезервировано.
S25	0-255	Время обнаружения изменения DTR, в десятых долях секунды.

Таблица 102. Описания регистров S (продолжение)

Регистр	Диапазон	Описание
S26	0-255	Задержка RTS/CTS, в десятых долях секунды.
S27	—	Зарезервировано.

Коды завершения для асинхронных адаптеров:

Коды завершения, возвращаемые асинхронными адаптерами, включая номера, слова и описания, как показано в следующей таблице.

Таблица 103. Коды завершения асинхронного адаптера

Номер	Слово	Описание
0	OK	Команда выполнена.
1	CONNECT	Установлено соединение со скоростью передачи данных 0-300 бит/с.
61 см	RING	Обнаружен сигнал вызова.
3	NO CARRIER	Несущий сигнал потерян или не обнаружен.
4	ERROR	Недопустимая команда, контрольная сумма, ошибка в командной строке, либо слишком длинная командная строка.
5	CONNECT 1200	Установлено соединение со скоростью передачи данных 1200 бит/с.
6	NO DIALTONE	Нет непрерывного гудка на линии.
7	BUSY	Линия занята.
8	NO ANSWER	При попытке установить соединение удаленная система не отвечает на вызов.
9	CONNECT 2400	Установлено соединение со скоростью передачи данных 2400 бит/с.

Опции набора номера:

В следующей таблице приведены опции набора номера и их описания.

Элемент	Описание
0-9 # * A-D	Цифры и символы для набора номера.
P	Импульсный набор номера.
T	Тоновый набор номера.
,	Задержка перед обработкой следующего символа.
!	Сигнал отбоя.
@	Ожидание тишины.
W	Ожидание сигнала.
;	Возврат в командный режим после набора номера.
R	Инверсный режим.
S=*n*	Набрать номер из ячейки *n*.

Помощь в настройке модема:

При возникновении связанных с модемом неполадок вы можете обратиться за помощью к следующим источникам.

- С настройкой модема вам могут помочь в местном сервисном представительстве.
- Служба поддержки предлагает много вариантов предоставления помощи, включая выезд к клиенту и поддержку по телефону. Обратитесь за помощью в ближайшее сервисное представительство.

- Часто забывают, что за помощью можно обратиться и к производителю модема. Большинство производителей предлагают определенные услуги по поддержке на своих веб-сайтах.

Записи файла /usr/lib/uucp/Dialers.samples:

Приведенные в этом разделе примеры предоставляются без каких-либо гарантий, "как есть". Они будут работать с указанными моделями, но могут не отвечать вашим конкретным потребностям.

Возможно, вам потребуется внести в них некоторые изменения. Более подробное описание параметров модема содержится в документации по нему.

Для того чтобы вы могли программировать модем с помощью параметров, в файле /usr/lib/uucp/Systems должна быть примерно следующая запись:

```
hayes Nvr HayesPRGM Any
```

В файле /usr/lib/uucp/Devices должна быть примерно следующая запись:

```
HayesPRGM tty0 - 2400 HayesProgrm2400
```

После добавления этих двух записей можно программировать модем с помощью команды **cu**:

```
cu -d hayes
```

```
# COMPONENT_NAME: cmduucp
#
#
# (C) COPYRIGHT International Business Machines Corp. 1994
#   Лицензионные материалы - собственность IBM
#   US Government Users Restricted Rights - Use, duplication or
#   disclosure restricted by GSA ADP Schedule Contract with IBM
#   Corp.
#############################################################
# Модем Motorola UDS                                        #
#
# Используйте команду udsmodemPROGRAM для программирования модема.
# Для портов должно быть задано rts/cts.
# Используйте номеронабиратель uds или hayes.
#
# "udsmodemPROGRAM" нужно записать в одной строке без пробелов
#
#############################################################
udsmodemPROGRAM =,-, "" \dAT&FQ2\r\c OK
ATE0Y0&C1&D2&S1%B5%E0*LC\r\c OKAT&K3&W\r\c OK

uds =,-, "" \dAT\r\c OK\r ATDT\T\d\r\c CONNECT

#############################################################
#
# IBM 7855 Модель 10
# Используйте команду IBMProgrm для программирования модема.
# Она задает управление потоком rts/cts, выключает
# xon/xoff и задает быстродействие DTE 19200 бит/с.
# Модем установит соединение с сервером с нужным
# быстродействием и управлением потоком.
# Для портов должно быть задано rts/cts.
#
# "IBMProgrm" нужно записать в одной строке без пробелов
#
#############################################################
IBMProgrm =,-, "" \dATQ0\r\c OK AT&F\r\c OK ATM1\r\c OK
AT&D3\r\c OK AT&C1\R2\Q2\M14\r\c OK AT&B8N1L0E0\A0\r\c OK
ATS0=1\r\c OK ATQ1&W0&Y0\r\c ""

#####################################################
# Следующий пример иллюстрирует вызов с обычным
```

```
# устройством ACU 7855. Необходимо включить коды
# результата (Q0), так как они были отключены при
# создании этого файла. (Это предотвращает вход в
# систему в верхнем регистре в ходе дозвона.)
# Также добавлен лишний символ "\" перед "\N", так
# как программы BNU удаляют этот символ перед "N".
#####################################################
ibm =,-, "" \dATQ0\r\c OK ATDT\T\d\r\c CONNECT

# IBM 7855 ECL (Без сжатия)
ibmecl =,-, "" \dAT\\N3%C0Q0\r\c OK ATDT\T\d\r\c CONNECT

# IBM 7855 ECLC (Со сжатием)
ibmeclc =,-, "" \dAT\\N3%C1Q0\r\c OK ATDT\T\d\r\c CONNECT

# IBM 7855 ECLC Сжатие с 256-байтовым размером блока
ibmeclc256 =,-, "" \dAT\\N3%C1Q0\A3\r\c OK ATDT\T\d\r\c CONNECT

# IBM 7855 Без сжатия 1200 бит/с
ibm_ne12 =,-, "" \dATQ0\\N0&A2%C0\r\c OK ATDT\T\d\r\c CONNECT

# IBM 7855 Без сжатия 2400 бит/с
ibm_ne24 =,-, "" \dATQ0\\N0&A3%C0\r\c OK ATDT\T\d\r\c CONNECT

# IBM 7855 Без сжатия 9600 бит/с
ibm_ne96 =,-, "" \dATQ0\\N0&A6%C0\r\c OK ATDT\T\d\r\c CONNECT

# IBM 7855 Без сжатия 19200 бит/с
ibm_ne192 =,-, "" \dATQ0\\N0%C0\r\c OK ATDT\T\d\r\c CONNECT

# IBM 7855 Без сжатия 12000 бит/с
ibm_ne120 =,-, "" \dATQ0\\N3%C0&AL8\r\c OK ATDT\T\d\r\c CONNECT

# IBM 7855 Без сжатия 1200 бит/с (бесшумный набор номера)
ibmq12 =,-, "" \dATQ0\r\c OK AT&A2M0DT\T\d\r\c CONNECT

# IBM 7855 Без сжатия 2400 бит/с (бесшумный набор номера)
ibmq24 =,-, "" \dATQ0\r\c OK AT&A3M0DT\T\d\r\c CONNECT

# IBM 7855 Без сжатия 9600 бит/с (бесшумный набор номера)
ibmq96 =,-, "" \dATQ0\r\c OK AT&A6M0DT\T\d\r\c CONNECT

# IBM 7855 Без сжатия 19200 бит/с (бесшумный набор номера)
ibmq192 =,-, "" \dATQ0\r\c OK ATM0DT\T\d\r\c CONNECT

#############################################################
#
# Модем Intel 9600EX
# Используйте команду IntelProgram для программирования модема.
# Она задает управление потоком rts/cts, выключает
# xon/xoff. Для портов должно быть задано rts/cts.
# Для портов должно быть задано rts/cts. (Используйте номеронабиратель hayes)
#
# "IntelProgram" нужно записать в одной строке без пробелов
#
#############################################################
#IntelProgram =,-, "" \d\dAT\r\c OK AT&F\r\c OK AT&S1M1\r\c OK
AT&D3\r\c OKAT&C1\r\c OK ATL0E0Y0&Y0\X1\r\c OK ATS0=1\r\c OK
AT&W\r\c OK

#############################################################
# Модем Practical Peripherals 1440FXMT
# Используйте команду PracPerProgram144 для программирования модема.
# Она задает управление потоком rts/cts, выключает
# xon/xoff. Для портов должно быть задано rts/cts. (Используйте номеронабиратель hayes)
# Быстродействие DTE будет зафиксировано равным
# быстродействию соединения при программировании модема. (Рекомендовано: 38400 бод)
```

```
#
# "PracPerProgram144" нужно записать в одной строке без пробелов
#
#############################################################
PracPerProgram144 =,-, "" \d\dAT\r\c OK AT&F\r\c OK ATM1\r\c OK
AT&D3\r\c OKAT&C1&K3\r\c OK ATQ2E1&Q9\r\c OK ATS0=1S9=20\r\c OK
AT&W\r\c OK

#############################################################
# Модем Practical Peripherals 9600 бит/с
# Используйте команду PracPerProgram9600 для программирования модема.
# Она задает управление потоком rts/cts, выключает
# xon/xoff. Для портов должно быть задано rts/cts. (Используйте номеронабиратель hayes)
#
# "PracPerProgram144" нужно записать в одной строке без пробелов
#
#############################################################
PracPerProgram9600 =,-, "" \d\dAT\r\c OK AT&F\r\c OK ATM1\r\c OK
AT&D3\r\c OKAT&C1&K3\r\c OK ATL0E0\r\c OK ATS0=1S9=20\r\c OK
AT&W\r\c OK

#############################################################
# Модем Practical Peripherals 2400 бит/с
# Используйте команду PracPerProgram для программирования модема.
#
# "PracPerProgram2400" нужно записать в одной строке без пробелов
#
#############################################################
PracPerProgram2400 =,-, "" \d\dAT\r\c OK AT&F\r\c OK ATM1\r\c OK
AT&D3\r\c OKAT&C1\r\c OK ATL0E0\r\c OK ATS0=1S9=20\r\c OK AT&W\r\c OK

#############################################################
# Модем Hayes 2400 бит/с
# Используйте команду HayesProgrm2400 для программирования модема.
# (Используйте номеронабиратель hayes)
#
# "HayesProgrm2400" нужно записать в одной строке без пробелов
#
#############################################################
HayesProgrm2400 =,-, "" \d\dAT\r\c OK AT&F\r\c OK ATM1\r\c OK
AT&D3\r\c OKAT&C1\r\c OK ATL0E0\r\c OK AT S0=1\r\c OK AT&W\r\c OK

#############################################################
# Telebit t2000 Trailblazer Plus
# Используйте команду TelebitProgrm для программирования модема.
# Она задает управление потоком rts/cts, выключает
# xon/xoff и устанавливает быстродействие DTE по
# умолчанию 19200 бит/с.
# Для портов должно быть задано rts/cts.
# Модем будет отправлять сигналы PEP в конце, так
# как они могут вызвать ошибку в других модемах.
#
# "TelebitProgram" нужно записать в одной строке без пробелов
#
#############################################################
TelebitProgram  =,-,    "" \dAT&F\r\c OK
ats2=255s7=60s11=50s41=2s45=255s51=254s52=2s54=3s58=2s64=1s66=1\r\c OK
ATs69=1s92=1s96=0s105=0s110=1s111=30s130=3s131=1F1M0Q6TV1W0X3Y0\r\c OK
ATE0&W\r\c OK
# Записи номеронабирателя Telebit T2000:
# Для принудительного соединения PEP:
tbfast =,-, "" \dATs50=255s7=60\r\c OK\r ATDT\T\r\c
CONNECT-\d\c-CONNECT

#   соединение 2400 бит/с:

#tb2400 =,-, "" \dATs50=3\r\c OK\r ATDT\T\r\c CONNECT
```

```
#  2400 MNP:
tb24mnp =,-, "" \dAT\r\c OK ATS0=0S95=2S50=3S41=0\r\c OK
ATDT\T\r\c CONNECT

# соединение на скорости 1200 бит/с:#tb1200 =,-, "" \dATs50=2\r\c OK\r
ATDT\T\r\c CONNECT

# 1200 MNP:
tb12mnp =,-, "" \dAT\r\c OK ATS0=0S95=2S50=2S41=0\r\c OK
ATDT\T\r\c CONNECT

############################################################
# Telebit WorldBlazer
# WORLDBLAZERProgram устанавливает быстродействие DTE
# 38400, но если соединение DTE позволяет, его можно
# повысить. Сигналы PEP отправляются в конце,
# чтобы не вызвать ошибку в других модемах. Это
# отключает xon/xoff и включает управление потоком # RTS/CTS. Порт должен быть заблокирован на 38400 с
# этими параметрами, RTS/CTS следует включить.
#
# "WORLDBLAZERProgram" нужно записать в одной строке без пробелов
#
############################################################
WORLDBLAZERProgram =,-, "" \dAT\r\c AT AT&F3M0\r\c AT
ATs51=253s92=1\r\c ATAT&W\r\c AT

#####################################################
# Номеронабиратели ACU с различным быстродействием
# для WorldBlazer - каждый задает определенное
# или меньшее быстродействие модема. WBlazer примет
# максимальное быстродействие удаленного модема.
# Для применения PEP с другими моделями Telebit
# используйте WBlazer38400 или WBlazer19200.
#####################################################
# WBlazer =,-, "" \dAT\r\c OK ATDT\T\d\r\c CONNECT
WBlazer38400 =,-, "" \dATs50=255\r\c OK ATDT\T\d\r\c CONNECT
WBlazer19200 =,-, "" \dATs50=255\r\c OK ATDT\T\d\r\c CONNECT
# Для соединения WBlazer14400 V.42bis.
WBlazer14400 =,-, "" \dATs50=7\r\c OK ATDT\T\d\r\c CONNECT

# Для соединения V.32:
WBlazer9600 =,-, "" \dATs50=6\r\c OK ATDT\T\d\r\c CONNECT

# Для соединения V.22:
WBlazer2400 =,-, "" \dATs50=3\r\c OK ATDT\T\d\r\c CONNECT

# Для соединения 1200 бит/с:
WBlazer1200 =,-, "" \dATs50=2\r\c OK ATDT\T\d\r\c CONNECT
```

Замечания о подключении 128-портового модема:

Эта операционная система не требует DSR в приложениях, управляющих модемом, а так как почти все современные модемы предусматривают возможность автоматического ответа, наличие индикатора входящего звонка обычно не требуется.

10-штырьковые разъемы RJ-45 являются не самым распространенным стандартом кабелей и не так широко представлены в продаже. Подсистема терминала этой системы предлагает опцию ALTPIN, которая заменяет логические функции DSR (Сигнал готовности к отправке данных) на DCD (Обнаружение несущей данных) для порта. При включении ALTPIN сигнал DCD становится доступным на контакте 1 8-штырькового разъема RJ-45 (эквивалент контакта 2 10-штырькового разъема).

Если вы хотите приспособить 8-жильный модемный кабель для 128-портового RAN, воспользуйтесь 8-штырьковым разъемом RJ-45, руководствуясь следующей таблицей:

Таблица 104. Подключение 128-портового модема

Элемент	Описание	Модем
SYSTEM END CONNECTOR 8-штырьковый RJ-45	**DEVICE ENDRI**	22
1	DSR	6
61 см	RTS	4
3 (шасси)	GND	SHELL
4	TxD	61 см
5	TxD	3
6 (сигнал)	GND	7
7	CTS	5
8	DTR	20
	CD	8

Примечание: Физические расположения DSR и CD можно поменять местами, если включен параметр ALTPIN, с помощью команды stty-cmxa.

В приведенной ниже таблице представлено асинхронное соединение между системным блоком и подключенным модемом. Здесь сигнал отправляется из локальной системы в удаленную систему.

Таблица 105. Асинхронное соединение

Устройство	Сигнал	Вкл./Выкл.	Значение
Компьютер	DTR	+	Модем, ты готов подключиться к другой системе?
Модем	DSR	+	Да, я готов. Набирай номер.
Модем	DCD	+	Удаленная система ответила на звонок.
Компьютер	RTS	+	OK, теперь можно отправлять данные?
Модем	CTS	+	Конечно, начинай.
Компьютер	TxD		Происходит отправка данных модему.
Модем	RxD		Я получил данные.
Модем	CTS	–	Пока не присылай мне новых данных, я отправляю их...
Модем	CTS	+	OK, я готов к приему новых данных, пришли их мне.
Эти операции по передаче данных могут повторяться до... Компьютер	DTR	–	Готово! Теперь повесь трубку.
Модем	DCD	–	OK.
Здесь приведен пример обмена сигналами между RS/6000 и модемом, ожидающим принять звонок от удаленной системы. Компьютер	DTR	+	Я готов и включил порт для входящего звонка.
Модем	DSR	+	Я также готов, но просто жду звонка.
Кто-то звонит! Модем	DCD	+	Кто-то позвонил, и мы установили соединение.
Модем	CTS	+	Я получил данные от другой системы, можно их передать тебе?
Компьютер	RTS	+	Я готов к приему. Пересылай мне данные.

Таблица 105. Асинхронное соединение (продолжение)

Устройство	Сигнал	Вкл./Выкл.	Значение
Модем	RxD		Хорошо, пересылаю.
Модем продолжает отправлять данные до тех пор, пока... Компьютер	RTS	–	Подожди! Мой буфер заполнился, пока не присылай мне данных.
Компьютер	RTS	+	Теперь все в порядке. Пришли мне еще данных.
Модем	DCD	–	Звонок завершен.
Компьютер	DTR	–	Хорошо, теперь повесь трубку.

Опции терминала stty-cxma

stty-cxma - это утилита, предназначенная для настройки и просмотра параметров терминала для 2-портовых, 8-портовых и 128-портовых адаптеров PCI. Она находится в каталоге `/usr/lbin/tty`.

Формат команды следующий:

```
stty-cxma [-a] [параметры] [имя-терминала]
```

Если ввести команду **stty-cxma** без параметров, то она показывает все специальные параметры драйвера, сигналы модема и все стандартные параметры, которые команда **stty(1)** выводит для терминала, соответствующего стандартному потоку ввода. Опции команды позволяют изменять параметры управления потоком, задавать параметры печати через терминал, вводить команды модема и просматривать параметры терминала. Нераспознанные опции передаются для распознавания в команду **stty(1)**. В команде могут быть указаны следующие опции:

-a Показывает значения уникальных параметров адаптера, а также все стандартные параметры терминала, возвращаемые командой **stty -a**.

имя-терминала
Задает и показывает параметры для указанного терминала вместо стандартного ввода. В этом формате можно указывать путь к терминалу с префиксом **/dev/** или имя терминала, начинающееся с символов tty. Эту опцию можно указать для линии управления модемом при отсутствии несущей.

Следующие опции задают операции для немедленного выполнения:

break Отправляет 250 мс прерывающее сообщение в линию терминала.

flush Указывает немедленный сброс (удаление) ввода и вывода терминала.

flushin Сбрасывает только ввод терминала.

flushout
Сбрасывает только вывод терминала.

Следующие опции задают действия, сбрасываемые при закрытии устройства. В следующем сеансе работы с устройством будут применяться значения по умолчанию.

stopout
Останавливает вывод, как если бы был получен символ XOFF.

startout
Возобновляет остановленный вывод, как если бы был получен символ XON.

stopin Включает управление потоком для остановки ввода.

startin Освобождает управление потоком для возобновления остановленного ввода.

[-]dtr [drop]
Включает линию управления модемом DTR, если не выбрано аппаратное управление потоком DTR.

[-]rts [drop]
Включает линию управления модемом RTS, если не выбрано аппаратное управление потоком RTS.

Следующие опции действуют до перезагрузки системы или изменения их значений.

[-]fastcook
Выполняет обработку подготовленного вывода на карте с процессором, чтобы снизить использование центрального процессора системы и повысить производительность в режиме прямого ввода.

[-]fastbaud
Вносит изменения в таблицы скорости передачи в бодах; для поддерживаемых устройств значение 50 бод заменяется на 57 600 бод, 75 - на 76 800, 110 - на 115 200, а 200 - на 230 000.

[-]rtspace
Включает и выключает аппаратную функцию управления входящим потоком RTS, чтобы приостановить передачу данных удаленным устройством.

[-]ctspace
Включает и выключает аппаратную функцию управления исходящим потоком CTS, чтобы локальное устройство приостанавливало передачу данных в отсутствие сигнала CTS.

[-]dsrpace
Включает и выключает аппаратную функцию управления исходящим потоком DSR, чтобы локальное устройство приостанавливало передачу данных в отсутствие сигнала DSR.

[-]dcdpace
Включает и выключает аппаратную функцию управления исходящим потоком DCD, чтобы локальное устройство приостанавливало передачу данных в отсутствие сигнала DCD.

[-]dtrpace
Включает и выключает аппаратную функцию управления входящим потоком DTR, чтобы приостановить передачу данных удаленным устройством.

[-]forcedcd
Выключает (включает повторно) функцию обнаружения несущей, чтобы терминал можно было открыть и работать с ним даже в отсутствие несущей.

[-]altpin
Связывает соединения разъема RJ-45 со значениями 10-штырькового (по умолчанию) или 8-штырькового разъема. Если этот параметр **включен**, то DSR и DCD меняются местами, чтобы сигнал DCD был доступен при использовании 8-штырькового разъема RJ-45 вместо 10-штырькового. (Значение по умолчанию - **выключен**.)

Возможные значения:

включен (задает значения 8-штырькового разъема)

выключен (задает значения 10-штырькового разъема)

startc *c*
Задает символ управления потоком XON. Символ может быть представлен десятичным, восьмеричным или шестнадцатеричным числом. Восьмеричные значения начинаются с нуля, а шестнадцатеричные - с комбинации символов 0x. Например, стандартный символ XON CTRL-Q можно указать как число 17 (десятичное), 021 (восьмеричное) или 0x11 (шестнадцатеричное).

stopc *c* Задает символ управления потоком XOFF. Символ может быть представлен десятичным, восьмеричным или шестнадцатеричным числом (формат восьмеричных и шестнадцатеричных значений указан в описании опции **startc**).

astartc *c*

Задает вспомогательный символ управления потоком XON. Символ может быть представлен десятичным, восьмеричным или шестнадцатеричным числом (формат восьмеричных и шестнадцатеричных значений указан в описании опции **startc**).

astopc *c*

Задает вспомогательный символ управления потоком XOFF. Символ может быть представлен десятичным, восьмеричным или шестнадцатеричным числом (формат восьмеричных и шестнадцатеричных значений указан в описании опции **startc**).

[-]aixon

Включает вспомогательную функцию управления потоком для применения двух уникальных символов для XON и XOFF. Если указаны оба символа XOFF, то передача возобновляется только после получения обоих символов XON.

[-]2200flow

Для порта применяется управление потоком в формате 2200. Терминалы 2200 поддерживают подключение принтера и применяют четыре символа управления потоком: XON терминала (0xF8), XON принтера (0xF9), XOFF терминала (0xFA) и XOFF принтера (0xFB).

[-]2200print

Определяет способ интерпретации этих символов управления потоком. Если задана опция 2200print, то для терминала и принтера, подключенного к терминалу, следует использовать независимые функции управления потоком. В противном случае эти функции будут логически связаны. При получении любого из символов XOFF весь вывод приостанавливается, пока не будет получен соответствующий символ XON.

maxcps *n*

Задает максимальную скорость передачи символов на принтер (в символах в секунду - cps). Рекомендуется выбирать это значение несколько меньшим, чем средняя скорость печати данного принтера. Если установить слишком маленькое значение, то снизится скорость печати. Если задано слишком большое значение, то принтер применяет функцию управления потоком, в результате чего сокращается время ввода пользователя. Значение по умолчанию - 100 cps.

maxchar *n*

Задает максимальный размер (в символах) очереди вывода, создаваемой драйвером принтера, подключенного к терминалу. Уменьшение этого значения приводит к увеличению нагрузки в системе; чем больше это значение, тем медленнее система будет реагировать на нажатие клавиш во время работы принтера. Значение по умолчанию - 50 символов.

bufsize*n*

Задает приблизительный размер буфера ввода принтера, подключенного к терминалу, которым будет руководствоваться драйвер принтера. По истечении периода простоя драйвер передаст на подключенный к терминалу принтер заданное количество символов, перед тем как восстановить скорость maxcps. Значение по умолчанию - 100 символов.

onstr*s* Задает escape-последовательность терминала для включения печати на принтере, подключенном к терминалу. Эти строки могут состоять из стандартных печатаемых и непечатаемых символов ASCII. Для управляющих символов (непечатаемых) следует указывать восьмеричные значения в виде обратной косой черты и трех цифр. Например Escape-символ, восьмеричное 33, задается как \033. Если печать на принтере, подключенном к терминалу включается строкой <Esc>[5i (стандарт ANSI), то она будет задана в следующем формат: \033[5i.

offstr*s* Задает escape-последовательность терминала для выключения печати на принтере, подключенном к терминалу. Формат строк указан в описании опции **onstr** *s*.

term*t* Задает в качестве строк включения и выключения печати на принтере, подключенном к терминалу, значения из внутренней таблицы по умолчанию. Внутренние значения по умолчанию применяются для следующих терминалов: adm31, ansi, dg200, dg210, hz1500, mc5, microterm, multiterm, pcterm, tvi, vp-a2, vp-60, vt52, vt100, vt220, wyse30, wyse50, wyse60 и wyse75. Если тип терминала отсутствует во внутренней таблице по умолчанию, то команда ditty считывает запись terminfo, указывающую тип

терминала, и присваивает строкам включения и выключения печати на принтере, подключенном к терминалу, значения, заданные атрибутами mc5/mc4 в записях terminfo.

Подсистема асинхронного канала связи, PPP

Подсистема **двухточечного протокола (PPP) для асинхронного канала связи** представляет собой альтернативу SLIP.

Протокол **PPP** обеспечивает стандартный способ передачи многопротокольных дейтаграмм по двухточечным линиями связи. **PPP** включает три уровня:

1. Протокол формирования дейтаграмм различных протоколов. **PPP** поддерживает протоколы TCP/IP сетевого уровня.
2. **Протокол управления каналом передачи данных (LCP)**, предназначенный для установления, настройки и проверки соединения уровня канала передачи данных. Этот уровень реализован в виде потоковых расширений ядра.
3. Семейство **протоколов управления сетью (NCP)** для выбора и настройки различных протоколов сетевого уровня. Для согласования соединений TCP/IP протокол **PPP** использует **Протокол управления IP (IPCP/IPv6CP)**.

Эта реализация **PPP** соответствует следующим RFC:

- RFC 1661, *The Point-to-Point Protocol, LCP*
- RFC 1332, *The PPP Internet Protocol Control Protocol (IPCP)*
- RFC 1662, *PPP in HDLC-like Framing*
- RFC 1334, *PPP Authentication Protocols*
- RFC 1990, *PPP Multilink*
- RFC 2472, *IP версии 6 через PPP*

PPP состоит из двух компонентов: клиента и сервера. Данная операционная система может выполнять функцию как клиента, так и сервера. Различие позволяет упростить настройку. Серверы **PPP** пытаются выделить пул IP/IPv6CP-адресов для устанавливаемых соединений. Между устройствами передачи данных есть определенная взаимосвязь. Данная реализация **PPP** нарушает эту взаимосвязь. Все соединения серверов **PPP** выделяются с помощью первого доступного устройства. Это упрощает отделение протокола **PPP** от конкретного устройства передачи данных. Процесс подключения должен указывать в запросе необходимый тип линии связи.

Процессы пользовательского уровня PPP

В данной операционной системе **двухточечный протокол для асинхронного канала связи** использует три процесса пользовательского уровня.

1. Управляющий демон (**pppcontrold**), который запускается пользователем root с помощью Контроллера системных ресурсов (**startsrc -s pppcontrold**). Назначение этой программы-демона заключается в загрузке и настройке всех расширений ядра, связанных с данной подсистемой. Она продолжает работу до тех пор, пока операционной системе необходимы функции **PPP**.
2. Процесс подключения (**pppattachd**), который связывает поток TTY с экземпляром протокола **управления каналом передачи данных**, **протокола управления сетью** и протокола дейтаграмм. Для каждого активного соединения **PPP** создается свой экземпляр процесса **pppattachd**. Все пользователи процесса подключения должны входить в группу **uucp**, а в переменной среды **PATH** этих пользователей должен быть указан каталог `/usr/sbin`.
3. Процесс набора номера **pppdial**, который устанавливает соединение с удаленной системой. Для установления соединения при наборе номера используется процесс **pppattachd**. Назначение этого процесса - взаимодействие с асинхронным устройством перед согласованием параметров соединения **PPP**. Алгоритм этого взаимодействия аналогичен алгоритму обмена сообщениями по протоколу UUCP. Процесс набора номера устанавливает соединение с удаленной системой. Фактическое установление сеанса связи выходит за рамки протокола **PPP**.

Настройка асинхронного канала связи, PPP

Для настройки **двухточечного протокола для асинхронного канала связи** можно воспользоваться программой SMIT.

Задачи, которые необходимо будет выполнить при настройке системы, перечислены в следующей таблице. Для выполнения этих задач у вас должны быть права доступа root.

Во время первоначальной настройки системы необходимо выполнить следующие задачи:

- Добавить конфигурацию линии связи.
- Создать интерфейс сервера (если вы настраиваете компьютер в качестве сервера PPP).
- Добавить интерфейс запросов (если вы хотите, чтобы система поддерживала соединения, устанавливаемые по запросу).
- Настроить пользователей и пароли для протоколов PAP или CHAP (если вы хотите применять идентификацию в соединениях **PPP**).
- Для того чтобы изменения вступили в силу, запустить (или перезапустить) **PPP**.

Таблица 106. Задачи настройки PPP асинхронного канала связи

Процедура	**Команды быстрого доступа SMIT**
Создать конфигурацию управления каналом связи	`smit ppplcp`
Добавить конфигурацию линии связи.	`smit addlcp`
Просмотреть или изменить конфигурацию линии связи	`smit chglcp`
Удалить конфигурацию канала связи [1]	`smit rmlcp`
Создать IP-интерфейсы PPP	`smit pppip`
Добавить интерфейсы сервера	`smit addpppserver`
Показать/Изменить интерфейс сервера	`smit listserver`
Удалить интерфейс сервера[1]	`smit rmlistserver`
Добавить интерфейс запросов	`smit addpppdemand`
Показать/Изменить интерфейс запросов	`smit listdemand`
Удалить интерфейс запросов[1]	`smit rmlistdemand`
Управлять пользователями/паролями PAP	`smit ppppap`
Добавить пользователя PAP	`smit addpapuser`
Показать/Изменить пользователя PAP	`smit listpapuser`
Удалить пользователя PAP	`smit rmpapuser`
Управлять пользователями/паролями CHAP	`smit pppchap`
Добавить пользователя CHAP	`smit addchapuser`
Показать/Изменить пользователя CHAP	`smit listchapuser`
Удалить пользователя CHAP	`smit rmchapuser`
Запустить **PPP**[2]	`smit startppp`
Завершить **PPP**[3]	`smit stopppp`
Интерфейсы **PPP** IPv6	`smit pppipv6`
Добавить интерфейс сервера **PPP** IPv6	`smit addpppv6server`
Показать или изменить интерфейс **PPP** IPv6.	`smit listv6server`
Удалить интерфейс **PPP** IPv6.	`smit rmlistv6server`
Добавить интерфейс клиента **PPP** IPv6.	`smit addpppv6client`
Показать или изменить интерфейс клиента **PPP** IPv6.	`smit listpppv6client`
Удалить интерфейс клиента **PPP** IPv6.	`smit rmlistpppv6client`
Добавить интерфейс запроса **PPP** IPv6	`smit addpppv6demand`
Показать или изменить интерфейс запроса **PPP** IPv6.	`smit listpppv6demand`
Удалить интерфейс запросов **PPP** IPv6.	`smit rmlistpppv6demand`

Таблица 106. Задачи настройки PPP асинхронного канала связи (продолжение)

Процедура	Команды быстрого доступа SMIT
Интерфейсы **PPP** IP и IPv6	`smit pppipv4_6`
Добавить интерфейс сервера **PPP** IP/IPv6	`smit addpppv4_6server`
Показать или изменить интерфейс **PPP** IP/IPv6.	`smit listv4_6server`
Удалить интерфейс **PPP** IP/IPv6.	`smit rmlistv4_6server`
Добавить интерфейс клиента **PPP** IP/IPv6.	`smit addpppv4_6client`
Показать или изменить интерфейс клиента **PPP** IP/IPv6.	`smit listpppv4_6client`
Удалить интерфейс клиента **PPP** IP/IPv6.	`smit rmlistpppv4_6client`
Добавить интерфейс запросов **PPP** IP/IPv6	`smit addpppv4_6demand`
Показать или изменить интерфейс запросов **PPP** IP/IPv6.	`smit listpppv4_6demand`
Удалить интерфейс запросов **PPP** IP/IPv6.	`smit rmlistpppv4_6demand`

Примечание:

1. В результате выполнения этой процедуры существующая информация уничтожается.
2. **PPP** можно запустить и другим способом, вызвав команду **startsrc -s pppcontrold**. Интерфейс SMIT позволяет задать опцию автоматического запуска **PPP** при загрузке системы.
3. Работу **PPP** можно завершить и другим способом, вызвав команду **stopsrc -s pppcontrold**. Интерфейс SMIT позволяет отключить опцию автоматического запуска **PPP** при загрузке системы.

Включение PPP SNMP

PPP может взаимодействовать с программой-демоном TCP/IP SNMP. PPP передает ей информацию о конфигурации канала связи PPP, а также сведения об активных интерфейсах **протокола управления каналом передачи данных (LCP)**.

При правильной настройке протокола TCP/IP SNMP и программ управления SNMP можно выполнять следующие операции:

- Получать информацию о конфигурации канала связи **PPP** (размер максимального блока приема, асинхронное преобразование символов и т.п.)
- Задавать информацию о конфигурации канала связи **PPP**
- Получать информацию об интерфейсе LCP для активных каналов связи LCP
- Изменять состояние активных соединений LCP, управлять которыми можно с помощью соответствующего объекта **ifAdminStatus** MIB.

В MIB **PPP** поддерживаются не все объекты, определенные в RFC1471. В подсистеме **PPP** применяется только таблица **pppLink**; разделы **pppLqr** и **pppTests** не поддерживаются. **pppLink** поддерживается со следующими ограничениями:

- Объект **pppLinkConfigMagicNumber** нельзя изменять. В **PPP** согласование сигнатур выполняется всегда. Отключить эту функцию нельзя.
- Объект **pppLinkConfigFcsSize** нельзя изменять. В данной операционной системе **PPP** поддерживает только FCS, равный 16.

По умолчанию поддержка SNMP для **PPP** отключена. Для того чтобы включить эту поддержку, выполните описанную ниже процедуру. Для выполнения этих задач необходимы права доступа root.

Примечание: При выполнении данной процедуры предполагается, что линия связи **PPP** уже настроена. Если это не так, то перед включением поддержки SNMP для **PPP** выполните процедуру, описанную в разделе “Настройка асинхронного канала связи, PPP” на стр. 620.

1. Запустите SMIT и перейдите в меню Изменить/показать конфигурацию канала связи:
   ```
   smit chglcp
   ```

2. Укажите в поле Включить агент SNMP для **PPP** значение Да.
3. Сохраните изменения и выйдите из SMIT.

Поддержка SNMP для **PPP** будет включена только после перезапуска **PPP**.

- Если протокол **PPP** уже активен:
  1. Завершите работу **PPP** с помощью команды `smit stopppp` (обратитесь к таблице из раздела “Настройка асинхронного канала связи, PPP” на стр. 620).
  2. С помощью следующей команды периодически проверяйте, завершена ли работа подсистемы:
     ```
     lssrc -s pppcontrold
     ```
     Время, необходимое для полного отключения подсистемы, зависит от числа каналов связи, заданных в конфигурации **PPP**. После того как работа подсистемы будет полностью завершена, команда выдаст сообщение о том, что подсистема `отключена`.
  3. Запустите **PPP** с помощью команды `smit startppp` (обратитесь к таблице из раздела “Настройка асинхронного канала связи, PPP” на стр. 620).
- Если протокол **PPP** не активен, запустите его с помощью команды `smit startppp` (обратитесь к таблице из раздела “Настройка асинхронного канала связи, PPP” на стр. 620).

Протокол SLIP

Протокол подключения к Internet по последовательной линии (SLIP) применяется протоколом TCP/IP при подключении через последовательное соединение.

Обычно он используется на выделенных последовательных линиях и в соединениях по телефонной линии на скоростях от 1200 бит/с до 19,2 Кбит/с и выше.

Примечание: Для того чтобы использовать скорость подключения 38400 и выше, укажите скорость 50 в файле `/etc/uucp/Devices` для нужного терминала, затем измените параметры SMIT для этого терминала, выбрав требуемую скорость подключения.

Пример: для запуска команды **cu** на терминале tty0 со скоростью передачи 115200 бод выполните следующие действия:

1. Убедитесь, что аппаратное обеспечение поддерживает данную скорость передачи.
2. Отредактируйте файл `/etc/uucp/Devices`, добавив туда следующую строку:
   ```
   Direct tty0 - 50 direct
   ```
3. Введите команду быстрого доступа `smit chtty`.
4. Выберите терминал tty0.
5. Измените скорость передачи в бодах на 115200.
6. Завершите работу SMIT.

Настройка SLIP

Настройку параметров **SLIP** рекомендуется выполнять в два этапа.

Этот двухшаговый подход позволяет выполнять настройку аппаратного обеспечения и машинно-зависимой конфигурации отдельно от настройки программного обеспечения SLIP и работы с синтаксисом команд.

1. Для успешного входа в удаленную систему воспользуйтесь ATE или утилитой **cu**. Это позволит убедиться в исправности и готовности к работе физической линии.

   Важно проверить работоспособность всех модемов, используемых при работе **SLIP**, так как именно они чаще всего становятся причиной неполадок на этапе установки.
2. Установив соединение с удаленной системой с помощью ATE или команды **cu** и убедившись в отсутствии ошибок, вы можете приступать к настройке параметров **SLIP**.

Замечания о модемах SLIP

При настройке модемов для **SLIP** важно, чтобы следующие изменения были сделаны на обоих концах соединения.

Локальный и удаленный модемы должны быть настроены абсолютно одинаково.

1. Модем должен определять наличие DTR.

   Если локальный модем игнорирует DTR, то он не сможет дать отбой. Он сможет закрыть соединение или дать сигнал отбоя, только когда определит потерю несущей с другого конца соединения. Это означает, что прерывание соединения может произойти только по инициативе другой стороны. Команды AT &D2 или &D3 подходят для большинства совместимых с Hayes модемов.

2. Модем не должен принудительно включать, игнорировать или симулировать функцию обнаружения несущей (DCD).

   DCD должна соответствовать реальному состоянию. Это означает, что несущая должна присутствовать после установления соединения с другой стороной (удаленным модемом) по коммутируемой телефонной линии. Это также относится и к выделенной линии. &C1 является рекомендуемым параметром для большинства совместимых с Hayes модемов.

3. Модем не должен принудительно включать, игнорировать или симулировать функцию готовности к приему (CTS).

   CTS должна следовать за функцией готовности к передаче (RTS). Если функция CTS включается принудительно, то модем не сможет открыть порт при отправке в порт команды **getty** или при добавлении в порт протокола управления потоком RTS.

4. Следует отключить коды автоматического запроса повторов (ARQ) в параметрах модема, если во время попыток дозвона **slattach** возникают неполадки.

   Если все попытки установить модемное соединение с помощью дозвона **slattach** оказываются неудачными, то следует проверить конфигурацию модема и отключить коды ARQ, если они включены. У большинства совместимых с Hayes модемов за это отвечает параметр &A0.

   Отключение кодов завершения ARQ не влияет на соединения с контролем ошибок и не предотвращает выдачу модемом стандартных сообщений CONNECT (если коды завершения включены), как требуется для строк набора номера **slattach**.

5. ECL (Проверка ошибок соединения) является важным параметром.

   Эту функцию могут использовать либо оба модема, либо ни один из них. Обычно модемы договариваются о ее использовании во время соединения. Если функция ECL включена, то связь по физической телефонной линии должна быть достаточно качественной, чтобы можно было исправить ошибку данных до того, как наступит тайм-аут во время ожидания пакета подтверждения для последнего пакета данных, отправленного по соединению **SLIP**.

6. Сжатие данных в соединении.

   В соединении можно применять сжатие данных, если оно полностью поддерживается обоими модемами. **SLIP** не выполняет никакое сжатие данных. При использовании сжатия данных настоятельно рекомендуется применять однотипные модемы, тогда сжатие будет осуществляться одинаковым способом и с одной и той же скоростью.

РУчное программирование модемов с помощью команды cu

С помощью следующих действий можно вручную запрограммировать модемы, подключенные к системному блоку.

- В системе должна быть установлена программа копирования UNIX-UNIX (UUCP). Проверить, установлена ли программа, можно с помощью команды **lslpp -f | grep bos.net.UUCP**.
- Модем должен быть подключен к системе и включен.
- Для изменения соответствующих файлов пользователю необходимы права доступа root.

1. Добавьте следующую строку в файл `/etc/uucp/Devices`, если это еще не сделано (вместо # укажите номер порта).

   ```
   Direct tty# - Any direct
   ```

Примечание: Строки файла Devices, начинающиеся с символа # в крайнем левом столбце, являются комментариями.

2. Сохраните и закройте файл.
3. Введите следующую команду в командной строке:

   ```
   cu -ml tty#
   ```

4. На экране появится сообщение о том, что модем подключен и готов к работе.
5. Введите AT и нажмите Enter. В ответ вы получите сигнал модема OK. Если вы не получили ответ от модема или введенные символы не появились на экране, сделайте следующее:
   - Проверьте провода подключения модема.
   - Убедитесь, что питание модема включено.
   - Посмотрите на индикаторы на передней панели модема в момент нажатия клавиши Enter. Если индикаторы приема данных (RD) и отправки данных (SD) мигают, значит модем и система обмениваются информацией и неполадки могут быть вызваны текущими параметрами модема. Если индикаторы не мигают, значит, причина неполадок кроется в соединении модема с системой.
   - Введите следующие команды и проверьте, изменяется ли при этом состояние модема:

     ```
     ATE1 <enter>
     ATQ0 <enter>
     ```

     ATE1 включает режим эхоповтора, в котором все введенные символы отображаются на экране. ATQ0 включает режим показа кодов завершения.
6. Запрограммируйте модем, используя параметры, приведенные в предыдущем разделе, "Замечания по настройке модема". Приведенный ниже пример демонстрирует, как программировать и сохранять основные параметры для совместимых с Hayes модемов. Введите следующую команду:

   ```
   AT&F    <enter>
   AT&D2   <enter>
   ATS0=1  <enter>
   ATS9=12 <enter>
   AT&C1   <enter>
   AT&W    <enter>
   ~.      <enter>
   ```

   Параметр &F служит для сброса параметров модема на значения по умолчанию, &D2 устанавливает DTR, S0 и S9 устанавливают значения регистра, &C1 устанавливает несущую, а &W сохраняет параметры в модеме. Тильда и точка закрывают соединение.

Автоматическая настройка модема

Пользователи могут настроить модем вручную или воспользоваться командой **cu** и связанными с ней файлами для автоматической настройки модема.

- В системе должна быть установлена программа UUCP. Проверить, установлена ли программа, можно с помощью команды **lslpp -f | grep bos.net.UUCP**.
- Модем должен быть подключен к системе и включен.
- Текст команды AT уже должен существовать (например, at&f&c1&d3). Прежде чем пытаться выполнить автоматическую настройку, следует попробовать настроить модем вручную с помощью команды **cu**.
- Для изменения соответствующих файлов пользователю необходимы права доступа root.

Следующий пример демонстрирует, как можно автоматически настроить модем Telebit T3000, подключенный к tty0.

1. Внесите необходимые изменения в файл /etc/uucp/Systems.
2. Добавьте следующую строку в конец файла. Запись должна начинаться с крайнего левого столбца файла.

   ```
   telebit Nvr TELEPROG 19200
   ```

3. Сохраните и закройте файл.
4. Внесите необходимые изменения в файл /etc/uucp/Devices.

5. Добавьте следующую строку в конец файла. Запись должна начинаться с крайнего левого столбца файла.

   ```
   TELEPROG tty0 - 19200 TelebitProgram
   ```

6. Сохраните и закройте файл.
7. Внесите необходимые изменения в файл `/etc/uucp/Dialers`.
8. Добавьте следующую строку в конец файла. Записи должны начинаться с крайнего левого столбца файла.

   Примечание: Следующие четыре строки следует ввести в виде одной длинной строки:

   ```
   TelebitProgram  =,-,    "" \dAT&F\r\c OK
   ats0=1s2=255s7=60s11=50s41=2s45=255s51=252s63=1s58=2s64=1\r\c OK
   ATs69=2s105=0s111=30s255=0M0&C1Q2&D3&Q0&R3&S1&T5\r\c OK
   ATE0X12&W\r\c OK
   ```

9. Сохраните и закройте файл.
10. Для запуска процесса автоматической настройки введите следующую команду:

    ```
    cu -d telebit
    ```

    Команда не будет выполнена, так как вы не подключены к системе. Посмотрите результаты отладки команды и убедитесь, что команда `ATE0X12&W` отправлена модему и получен отклик OK. Если это так, значит, модем успешно запрограммирован.

Из-за неправильных значений, указанных в файле `Dialers`, или из-за существующей конфигурации модема могут возникнуть неполадки. В таком случае попробуйте запрограммировать модем вручную и поочередно ввести строки набора номера (указанные в пункте 8).

Настройка SLIP через модем

Для настройки модемного соединения **SLIP** между двумя системами выполните описанную ниже процедуру, в которой для настройки применяется Инструмент управления системой (SMIT) и командная строка.

Ниже приводится пример настройки соединения между хостами с именами bronze и gold.

1. Подключите модемы к системам bronze и gold.
2. Для того чтобы создать терминал на хосте bronze с помощью программы SMIT выполните следующие действия:
   a. Введите:

      ```
      smit maktty
      ```

   b. В качестве типа создаваемого терминала выберите **rs232**.
   c. Выберите один из доступных последовательных портов, например, `sa0` (системный последовательный порт 1).
   d. Выберите в списке номер порта для данного терминала.
   e. Укажите скорость передачи данных, с которой работает ваш модем.
   f. Отключите опцию Разрешить вход в систему.
   g. Завершите работу SMIT.
3. Создайте терминал на хосте gold.

   Для этого выполните ту же процедуру, что и для хоста bronze (шаг 2), но укажите в поле Разрешить вход в систему значение **разрешить**.

   Далее предполагается, что обоим терминалам в хостах bronze и gold присвоены номера tty1.
4. Проверьте физическое соединение с помощью программы ATE.
   a. На хосте bronze введите следующую команду:

      ```
      ate
      ```

b. В меню "Главное меню (соединение не установлено)" выберите команду **Изменить**. Для параметра Скорость укажите значение, равное скорости передачи в бодах вашего модема, а для параметра Устройство - значение tty1.

c. В меню "Главное меню (соединение не установлено)" выберите команду **Соединить**. В появившемся приглашении ATE введите номер телефона хоста gold и нажмите Enter.

d. На экране должно появиться приглашение на вход в систему хоста gold. Войдите в систему.

e. Выйдите из системы хоста gold, нажмите Ctrl-V для возврата в главное меню, затем T, чтобы прервать соединение, и Q, чтобы завершить работу программы ATE.

Примечание: Если приглашение для входа в систему удаленного хоста не появилось, то перейдите к шагу 1 и убедитесь, что все параметры конфигурации были заданы правильно. Продолжать эту процедуру следует только после того, как вам удастся войти в систему хоста gold.

5. Конфигурация терминала для работы с ATE отличается от конфигурации, которая применяется для соединений SLIP, поэтому необходимо внести следующие изменения:

   a. На хосте bronze введите следующую команду:

   ```
   smit chgtty
   ```

   b. На хосте gold введите следующую команду:

   ```
   smit chgtty-pdisable tty1
   ```

   c. Выберите **tty1**, а затем опцию **Изменить/просмотреть программу терминала**.

   d. Отключите опцию Разрешить LOGIN, а затем завершите работу SMIT.

6. В файл `/usr/lib/uucp/Devices` на хостах bronze и gold добавьте следующую строку:

   ```
   Direct tty1 - 9600 direct
   ```

   где вместо `9600` нужно указать скорость вашего модема.

7. Создайте сетевой интерфейс **SLIP** на хосте bronze.

   a. Введите:

   ```
   smit mkinet1sl
   ```

   b. В параметре Порт TTY для сетевого интерфейса SLIP введите значение **tty1**.

   c. Укажите IP-адрес, например, 130.130.130.1.

   d. Укажите Целевой адрес (хоста gold), например, 130.130.130.2.

   e. Укажите скорость передачи данных вашего модема.

   f. Укажите команду набора номера, например:

   - "" AT OK ATDT555-1234 CONNECT ""
   - Эта команда означает, что будет применяться устройство **tty1**, работающее с быстродействием 9600 бод. Отправить модему сигнал AT. В ответ вы получите сигнал модема OK. Наберите номер телефона `555-1234`. Модем отправит сигнал `CONNECT`. До и после символов `""` должны быть указаны пробелы.

   g. Завершите работу SMIT.

8. Создайте сетевой интерфейс **SLIP** на хосте gold. Для этого выполните ту же процедуру, что и для хоста bronze (шаг 5), указав соответствующие значения для параметров IP-адрес и Целевой адрес.

9. В файлы **/etc/hosts** хостов bronze и gold добавьте следующие записи:

   ```
   130.130.130.1   bronze
   130.130.130.2   gold
   ```

   Присвоенные имена должны быть уникальными. Другими словами, если интерфейсу Token-Ring хоста bronze уже присвоено имя `bronze`, то для интерфейса **SLIP** укажите другое имя, например, `bronze_slip`.

   Примечание: Для того чтобы упростить работу с командой **slattach**, воспользуйтесь сценарием `/usr/sbin/slipcall`.

10. Проверьте соединение **SLIP**.

    a. На хосте bronze введите следующую команду:

```
ping gold
```

b. На хосте gold введите следующую команду:

```
ping bronze
```

Если обе проверки завершились успешно, то соединение **SLIP** установлено. Если нет, то перейдите к шагу 5 и проверьте правильность настройки хостов bronze и gold.

Настройка SLIP через нуль-модемный кабель

Для настройки соединения **SLIP** между двумя системами, связанными с помощью нуль-модемного кабеля, выполните описанную ниже процедуру, в которой для настройки применяется как Инструмент управления системой (SMIT), так и командная строка.

Ниже приведен пример настройки соединения между хостами bronze и gold.

1. Установите физическое соединение между хостами bronze и gold с помощью нуль-модемного кабеля. Вам потребуются следующие кабели. (Кабели перечислены в том порядке, в котором они будут подключаться от хоста bronze к хосту gold.)
   a. Кабель B (код изделия 00G0943). Кабель для последовательного порта; с каждой системой поставляется два кабеля такого типа (за исключением моделей 220, 340 и 350, для которых он не нужен).
   b. Кабель D (код изделия 6323741, специальный код 2936). Асинхронный кабель EIA-232/V.24.
   c. Кабель E (код изделия 59F2861, специальный код 2937). Переходник принтер/терминал EIA-232 (нуль-модемный кабель).
   d. Адаптер для соединения кабелей (с обеих сторон адаптера должны быть гнезда для подключения кабеля).
2. Создайте терминал на хосте bronze.
   a. Введите:

      ```
      smit maktty
      ```

   b. В качестве типа создаваемого терминала выберите **rs232**.
   c. Выберите один из доступных последовательных портов, например, **sa0** (системный последовательный порт 1).
   d. Выберите в списке номер порта для данного терминала.
   e. Присвойте параметру Скорость передачи данных значение 19200. (Позже это значение будет изменено на 38400, но сейчас укажите значение 19200).
   f. Отключите опцию Разрешить вход в систему, а затем завершите работу SMIT.
3. Создайте терминал на хосте gold. Для этого выполните ту же процедуру, что и для хоста bronze (шаг 2), но укажите в поле Разрешить вход в систему значение **разрешить**.

   Примечание: Далее предполагается, что обоим терминалам в хостах bronze и gold присвоены номера tty1.
4. Проверьте физическое соединение с помощью программы ATE.
   a. На хосте bronze введите следующую команду:

      ```
      ate
      ```

   b. В меню "Главное меню (соединение не установлено)" выберите команду **Изменить**. Для параметра Скорость установите значение 19200, а для параметра Устройство - tty1.
   c. В меню "Главное меню (соединение не установлено)" выберите команду **Соединить**. Когда появится приглашение ATE для ввода телефонного номера, нажмите Enter. Вы получите следующее сообщение:

      ```
      ate: 0828-010 Команда установила соединение через порт tty1
      ```

   d. Нажмите Enter. Должно появится приглашение для входа в систему хоста gold. Войдите в систему.

e. Выйдите из системы хоста gold, нажмите Ctrl-V для возврата в главное меню, затем T, чтобы прервать (завершить) соединение, и Q, чтобы завершить работу программы ATE.

Примечание: Если приглашение для входа в систему удаленного хоста не появилось, то перейдите к шагу 1 и убедитесь, что все параметры конфигурации были заданы правильно. Продолжать эту процедуру следует только после того, как вам удастся войти в систему хоста gold.

5. Конфигурация терминала для работы с ATE отличается от конфигурации, которая применяется для соединений **SLIP**, поэтому необходимо внести следующие изменения:
   a. На хосте bronze введите следующую команду:
      ```
      smit chgtty
      ```
   b. Выберите **tty1**. Установите параметр Скорость передачи в БОДАХ равным `38400`, а затем завершите работу SMIT.
   c. На хосте gold введите следующую команду:
      ```
      pdisable tty1
      ```
   d. На хосте gold введите следующую команду:
      ```
      smit chgtty
      ```
   e. Выберите **tty1**. Отключите опцию Разрешить LOGIN, установите параметр Скорость передачи в БОДАХ равным `38400`, а затем завершите работу SMIT.
6. В файл `/usr/lib/uucp/Devices` на хостах bronze и gold добавьте следующую строку:
   ```
   Direct tty1 - 38400 direct
   ```
7. Создайте сетевой интерфейс **SLIP** на хосте **bronze**.
   a. Введите:
      ```
      smit mkinet1sl
      ```
   b. В параметре Порт TTY для сетевого интерфейса SLIP введите значение **tty1**.
   c. Укажите IP-адрес, например, 130.130.130.1.
   d. Укажите Целевой адрес (адрес хоста gold) - например, `130.130.130.2`. Затем нажмите OK или Enter.
   e.
8. Создайте сетевой интерфейс **SLIP** на хосте gold. Для этого выполните ту же процедуру, что и для хоста bronze (шаг 5), указав соответствующие значения для параметров IP-адрес и Целевой адрес.
9. В файлы `/etc/hosts` хостов bronze и gold добавьте следующие записи:
   ```
   130.130.130.1   bronze
   130.130.130.2   gold
   ```
   Присвоенные имена должны быть уникальными. Другими словами, если интерфейсу Token-Ring хоста bronze уже присвоено имя `bronze`, то для интерфейса **SLIP** укажите другое имя, например, `bronze_slip`.
10. Запустите **SLIP** на хостах bronze и gold. Введите:
    ```
    slattach tty1
    ```
11. Проверьте соединение **SLIP**.
    a. На хосте bronze введите следующую команду:
       ```
       ping gold
       ```
    b. На хосте gold введите следующую команду:
       ```
       ping bronze
       ```

Если обе проверки завершились успешно, то соединение **SLIP** установлено. Если нет, то перейдите к шагу 5 и проверьте правильность настройки хостов bronze и gold.

Завершение соединения SLIP

Для того чтобы разорвать соединение **SLIP**, выполните следующие действия:

1. Введите:
   ```
   ps -ef | grep slatt
   ```

Запомните номера процессов, связанных с командой **slattach**.

2. Для каждого процесса выполните команду:

   ```
   kill номер-процесса
   ```

   Не указывайте флаг **-9** в команде **kill**.

   Если процесс **slattach** будет завершен с флагом **-9**, в файле etc/locks может остаться запись о блокировке SLIP. Удалите этот файл блокировок, чтобы выполнить очистку памяти после выполнения команды **slattach**.

Для временной деактивации соединения **SLIP**, выполните следующие действия в локальной и удаленной системах:

1. Введите:

   ```
   ifconfig sl# down
   ```

2. Просмотрите список текущих процессов **slattach** с помощью команды:

   ```
   ps -ef | grep slat
   ```

   Вывод этой команды выглядит примерно следующим образом:

   ```
   root  1269  1    0   Jun 25  ... slattach
   ```

3. Убейте процесс **slattach**, используя ИД процесса. Например, для того чтобы убить приведенный выше процесс **slattach**, введите:

   ```
   kill 1269
   ```

   где 1269 - это ИД процесса **slattach**. *Не* удаляйте процесс **slattach** с помощью флага **-9** команды **kill**.

   Теперь соединение **SLIP** выключено.

Активация соединения SLIP

С помощью следующих действий активируйте соединение **SLIP**, которое было временно отключено.

Запустите эти команды в локальной и удаленной системах.

1. Введите:

   ```
   ifconfig sl# up
   ```

2. Еще раз выполните команду **slattach**.

Удаление интерфейса SLIP

С помощью приведенных ниже инструкций можно удалить интерфейс **SLIP**.

После выполнения этих инструкций интерфейс sl# и связанный с ним процесс **slattach** будут удалены. Все внесенные в файл `/etc/hosts` записи сохранятся, и их необходимо будет удалить вручную.

1. Для удаления интерфейса **SLIP** и связанного с ним процесса **slattach** используйте команду быстрого доступа `smit rminet`, открывающую окно **Доступные сетевые интерфейсы**.
2. Выберите подходящую запись в окне **Доступные сетевые интерфейсы** и выберите **Выполнить**.

   Примечание: Все внесенные в файл `/etc/hosts` записи сохранятся, и их необходимо будет удалить вручную.

Устранение неполадок SLIP

С помощью этих команд можно устранить неполадки **SLIP**.

Каждая команда приведена с примерами ее использования для устранения неполадок **SLIP**.

Помимо того для справки приведен список распространенных неполадок и сообщений об ошибках.

Команда netstat:

Команда **netstat** работает вместе с командой **ifconfig** и служит для отображения состояния сетевого интерфейса TCP/IP.

Например, с помощью команды **netstat -in** и флага **-i** можно посмотреть информацию о сетевых интерфейсах, тогда как флаг **-n** служит для печати IP-адресов вместо имен хостов. С помощью этой команды можно проверить интерфейсы SLIP, адреса и имена хостов. Ниже приведено описание вывода команды **netstat -in**.

Запрограммируйте модем, используя параметры, приведенные в разделе, “Замечания о модемах SLIP” на стр. 623. Приведенный ниже пример демонстрирует, как программировать и сохранять основные параметры для совместимых с Hayes модемов. Введите следующую команду:

```
Имя  MTU   Сеть    Адрес             Ipkts     Ierrs Opkts Oerrs  Col
lo0   1536  <Link>                       2462       0    2462   0     0
lo0   1536  127        localhost.austi     2462       0    2462   0     0
tr0   1492  <Link>                       1914560    0   21000   0     0
tr0   1492  129.35.16  glad.austin.ibm     1914560    0   21000   0     0
sl0   552   1.1.1.0    1.1.1.1             48035      0   54963   0     0
sl1*  552   140.252.1  140.252.1.5         48035      0   54963   0     0
```

Обратите внимание на * рядом с интерфейсом sl1. Это означает, что сетевой интерфейс не работает или недоступен. Это можно исправить с помощью команды **ifconfig sl1 up**, при условии, что это допустимый интерфейс **SLIP**.

Команда **netstat** показывает статистику приема и отправки пакетов, а также информацию об ошибках приема и отправки, которая может быть полезна при устранении неполадок соединений **SLIP**.

Например, пользователь вводит команду **ping** для проверки связи с удаленным хостом по соединению **SLIP**, и процесс **ping** кажется зависшим. Затем пользователь быстро запускает команду **netstat -in** из другой командной оболочки и замечает, что показатель Opkts увеличивается, но пакеты Ipkts от удаленного хоста не поступают. Это означает, что удаленная система не возвращает или не принимает информацию. Необходимо запустить ту же команду **netstat** на удаленной системе, чтобы убедиться в получении пакетов **ping** или увеличении счетчика ошибок.

Преобразование имен хостов в IP-адреса связано с функцией преобразования имен и поэтому необходимо для правильной работы **SLIP**. Для обнаружения неполадок, вызванных именами хостов, псевдонимами или маршрутизацией, используйте команду **netstat -rn**. Из файла **/etc/hosts** будет возвращено только базовое имя хоста. Если система обслуживается по серверу имен (то есть `/etc/resolv.conf` существует), тогда сервер имен выдаст полное имя домена в ответ на эту команду.

Команда ifconfig:

Команда **ifconfig** - это инструмент настройки сетевого интерфейса. Она позволяет динамически создавать или удалять структуру сетевого интерфейса из памяти ядра.

Эта команда получает данные из командной строки и на основе заданных параметров создает структуру памяти. При отладке команда **ifconfig** позволяет выяснить состояние интерфейса связи.

Примечание: При перезагрузке системы все изменения, внесенные в атрибуты интерфейса с помощью команды **ifconfig**, будут потеряны.

Например, для просмотра текущего состояния интерфейса sl1:

1. Введите команду **netstat -i** и изучите вывод, выбрав требуемый интерфейс sl#. Например, sl0, sl1, sl2 и т.д.
2. Введите команду **ifconfig sl#** и просмотрите значения следующих ключевых полей вывода:

Элемент	Описание
POINTTOPOINT	Этот параметр должен всегда присутствовать в работающем соединении SLIP. Если это не так, соединение может быть прервано или отключено. Попробуйте изменить его состояние с помощью команд **ifconfig sl# up** и **ifconfig sl#**.
UP	Указывает, что сетевой интерфейс sl# активирован и должен работать.
RUNNING	Указывает на удачное выполнение команды **slattach**. Это означает, что связь установлена, номер набран, удаленная система ответила и вернула сигнал состояния обнаружения несущей CARRIER DETECT. При переходе в состояние CD к этому параметру добавляется бит выполнения.

Команды pdisable и lsdev:

Все порты терминалов, используемые для соединений **SLIP**, должны быть отключены или недоступны.

Для того чтобы убедиться в том, что порт tty1 отключен, войдите в систему с правами доступа пользователя root и введите одну из следующих команд:

- `lsattr -El tty1  -a login`

  С помощью этой команды можно просмотреть постоянное состояние порта терминала, как оно записано в системном Администраторе системных данных (ODM). Если вывод команды отличается от `login disable`, то с помощью SMIT измените значение поля LOGIN на **disable**.
- `pdisable | grep tty1`

  С помощью этой команды, запущенной без параметров, можно просмотреть все порты терминала, находящиеся в отключенном состоянии. В этом примере команда **pdisable** используется вместе с командой **grep**, чтобы отсечь ненужную информацию вывода. Если tty1 не указан в выводе команды, значит, порт не отключен.

Команда ps:

Команда **ps** показывает информацию об активных процессах в стандартном выводе.

С помощью этой команды можно проверить наличие или отсутствие процессов **slattach**, используемых для того, чтобы присвоить линию терминала сетевым интерфейсам.

Если команда **netstat -in** показала, что интерфейс не работает, следует запустить команду **ps -ef | grep slat**, чтобы проверить, запущен ли процесс **slattach** на связанном порте терминала. Учтите, что в случае напрямую подключенного интерфейса **SLIP** прерванные соединения восстанавливаются автоматически, без вмешательства пользователя. В случае интерфейса **SLIP**, подключенного через модем, прерванные соединения следует восстанавливать вручную. Если вы ввели строку набора номера в командной строке **slattach**, следует повторить ввод команды и строки набора номера для восстановления прерванного соединения.

Команда ping и индикаторы модема:

Команда **ping** и индикаторы модема служат для отладки связи **SLIP**.

Команда проверки связи ping отправляет пробный пакет запроса и затем принимает ответный пакет. Эти действия полезны в случае, если системному администратору видны индикаторы модема.

Например, локальная система создает пакет запроса и отправляет его удаленной системе. На локальном модеме в это время горит индикатор Отправка данных (SD). Это означает, что локальный TCP/IP, **slattach** и терминал смогли собрать информацию и отправить ее через модем удаленной системе.

Когда удаленный модем получает пакет, начинает мигать индикатор приема данных, но не индикатор SD. Это означает, что удаленная система не смогла отправить (вернуть) пробный запрос в локальную систему. В результате команда **ping** в локальной системе зависает, и для ее завершения требуется нажать Ctrl-C.

Наиболее вероятная причина возникновения неполадки - использование управления потоком XON/XOFF на одном или обоих модемах; возможны также конфликты маршрутизации или адресации в системах.

Распространенные неполадки SLIP и сообщения об ошибках:

Здесь приведены распространенные неполадки **SLIP** и сообщения об ошибках, возможные причины их возникновения и рекомендуемые действия пользователя.

Сообщение: 0821-296 Невозможно установить описания строк для `/dev/tty#` равным slip.ioctl(TXSETLD). В системный вызов передан недопустимый параметр.

Возможные причины: Обычно этот тип неполадок возникает при запуске процесса **slattach** и связанных с ним команд при неверной конфигурации SLIP. Скорее всего, неполадка вызвана несоответствием номеров терминала и интерфейса sl. Это также объясняет, почему система выдала сообщение о том, что команда ifconfig не была выполнена перед **slattach**.

Кроме того, эта неполадка может возникнуть, когда процессы **slattach** завершаются неверно или когда пользователь пытается перенести соединение **SLIP** на другой порт терминала, но забывает перенастроить интерфейс sl# в соответствии с этим терминалом. Проверьте процессы **slattach**, которые все еще могут выполняться (например, `ps -ef | grep slat`).

Действие: Терминал для SLIP - это **/dev/tty24**, а пользователь создал интерфейс sl0. Это недопустимо. Пользователь должен создать интерфейс sl24, в соответствии с номером терминала (tty24 и sl24). Если неполадки не исчезли, следует закрыть интерфейс sl (см. "Закрытие интерфейса SLIP") и изменить конфигурацию соединения с помощью следующих команд:

```
lsdev  -Cc if  -s SL
lsattr -El sl0
```

Сообщение:

в настоящее время сеть недоступна

маршрут к удаленному хосту недоступен

Возможная причина: Эти неполадки чаще всего возникают, когда пользователь пытается проверить связь с хостом через неправильно установленное соединение **SLIP**. Скорее всего, неполадка вызвана тем, что один или оба порта терминалов, связанные с интерфейсом sl#, находятся во включенном состоянии. Также не исключена возможность конфликта адресации или маршрутизации между хостами.

Действия:

- Удалите интерфейс sl# с помощью команды быстрого доступа `smit rminet`. Это следует сделать и в локальной, и в удаленной системе.
- Выполните следующие действия на обоих хостах соединения SLIP:
  1. Введите `pdisable | grep tty#`.
  2. Если устройство терминала не перечислено в списке вывода предыдущей команды, терминал не отключен. Отключите терминал посредством SMIT или командной строки. Отключив терминалы, с помощью SMIT заново создайте интерфейсы **SLIP** в обеих системах. Если неполадка не исчезла, проверьте сетевые адреса и маршруты, если они есть. С помощью команды **netstat -ir** просмотрите информацию об адресах, маршрутах и интерфейсах.

Неполадка: Когда удаленная система звонит локальной, локальный модем устанавливает соединение, но не выполняет вход в систему.

Возможные причины: если два модема соединяются и начинают квитирование или обмен информацией для соединения, но затем соединение прерывается, то неполадки могут быть вызваны кодами завершения. Также

причиной неполадки может быть неправильная строка набора номера **slattach**. Если два модема после звонка не начинают квитирование, возможно, у модема не включена функция автоматического ответа.

Действия:

1. Проверьте соединение модема с помощью команды **cu**. Модем удаленной системы должен разрешать пользователю войти в систему. Во время входа в систему на экране не должно быть никаких помех и посторонних символов, в противном случае это может указывать на высокий уровень шума в телефонной линии, что может быть причиной неполадки. Во время входа в систему на экране *не* должно появляться сразу несколько приглашений на вход в систему. Противное может указывать на высокий уровень шума в телефонной линии или неверные параметры модема.
2. Проверьте конфигурацию модема и попробуйте отключить коды ARQ, если они включены. У большинства совместимых с Hayes модемов за это отвечает параметр &A0. Отключение кодов завершения ARQ не влияет на соединения с контролем ошибок и не предотвращает выдачу модемом стандартных сообщений CONNECT (если коды завершения включены), как требуется для строк набора номера **slattach**.

Неполадки: Пользователь не может выполнить команду **ping** в модемном соединении **SLIP**. Команда **ping** зависает или выдает сообщения об ошибке.

Возможные причины:

1. Модемы и/или порты терминала настроены для использования управления потоком XON/XOFF.
2. На удаленном хосте прерван процесс **slattach** или модемное соединение прервалось.
3. Присвоенные хостам **SLIP** адреса неверны.

Действия:

1. Проверьте конфигурацию локального и удаленного модемов. Они должны быть настроены на использование аппаратного управления потоком RTS/CTS или вообще не использовать управление потоком. Отправьте пробные пакеты из обеих систем. Проверьте связь между systemA и systemB.
2. Убедитесь в том, что процесс **slattach** выполняется и в локальной, и в удаленной системе. Воспользуйтесь командой `ps -ef |grep slat`. Убедитесь, что интерфейс sl# по-прежнему активен. Воспользуйтесь командой `ifconfig sl#`.
3. Убедитесь, что между адресами **SLIP** и адресами, связанными с другими сетевыми интерфейсами (если они есть), нет конфликтов. Воспользуйтесь командой **netstat -ir**. Если адрес или адресный класс вызывают подозрение, упростите адресную схему, применяемую в конфигурации **SLIP**, например, укажите 1.1.1.1 для локального хоста и 1.1.1.2 для удаленного хоста.

Вопросник SLIP

В этой справочной таблице можно записать данные о конфигурациях **SLIP**.

Собранная информация может быть отправлена в сервисное представительство в случае, если вам понадобится дополнительная помощь в настройке **SLIP**.

1. Эта конфигурация **SLIP** работала ранее? (Да/Нет) ____
2. Какие типы системы? (например: UNIX/PC, DOS/PC и т.д.)

   Локальная система: _____ Удаленная система: _____

   Если хост не является системой IBM UNIX, укажите тип программного обеспечения, используемого для установки соединения **SLIP**.

   _____
3. Какие версии операционной системы IBM UNIX установлены в каждом системном блоке? Запустите команду `/bin/oslevel`. Если выполнить команду не удалось, воспользуйтесь следующим способом:

   ```
   lslpp -h bos.rte
   ```

   найдите версию системы в строке *active commit*.

Локальная система: _____ Удаленная система: _____

4. Перечислите все доступные интерфейсы в обеих системах (например, sl0, sl1). Для этого используйте команду: `lsdev -Cc if`

   Локальная система: _____ Удаленная система: _____

   _____ _____

   _____ _____

   _____ _____

   Номер интерфейса **SLIP** должен совпадать с номером терминала. Например, `/dev/tty53` следует использовать с sl53.

5. **SLIP** настраивается с помощью SMIT или команд? Конфигурация SLIP, созданная с помощью команд, является временной и не сохраняется после перезагрузки.

   _____

6. **SLIP** настраивается для соединения по модему или по прямой последовательной линии?

   _____

7. В случае использования модемов укажите производителя и модели модемов локальной и удаленной систем.

	Тип	Скорость	Кабель IBM передачи	Если не кабель IBM, (Да/Нет)	укажите тип.
Локальный:	_____	_____	___	_____	
Удаленный:	_____	_____	___	_____	

8. Если используются модемы, каков тип телефонной несущей? (выделенная линия или коммутируемая)

   _____

9. С каким аппаратным обеспечением используется соединение **SLIP**?

   128-портовый адаптер (с 16-портовым RAN: ___

   2-портовый адаптер: ___

   8-портовый адаптер: ___

   Встроенные последовательные порты S1 или S2: ___

10. Можно ли отправить пробный пакет из локальной системы в удаленную с помощью команды ping?

    (Да/Нет) _____ (в локальной системе введите <адрес удаленной системы>)

11. Можно ли отправить пробный пакет из удаленной системы в локальную с помощью команды ping?

    (Да/Нет) _____ (в удаленной системе введите ping <адрес локальной системы>)

12. Отключены ли порты терминала в локальной и удаленной системах?

    (Да/Нет) _____

    Воспользуйтесь командой `pdisable | grep tty#`. В выводе этой команды указываются только номера отключенных терминалов.

13. Выдаются ли какие-либо сообщения об ошибках? Если да, перечислите их ниже:

    _____

    _____

    _____

    _____

    _____

    _____

    _____

    _____

    _____

    _____

Эмуляция асинхронного терминала

Программа эмуляции асинхронного терминала (ATE) позволяет терминалам операционной системы эмулировать терминал, благодаря чему пользователи могут подключаться к большинству других систем, поддерживающих асинхронные терминалы.

ATE представляет удаленной системе терминал в виде системного дисплея или терминала DEC VT100. Опция VT100 позволяет пользователю подключаться к системам, которые не поддерживают его терминал, но поддерживают терминалы VT100.

Для установления связи между системой пользователя и удаленной системой программа ATE применяет как прямые соединения (по кабелю), так и соединения через модем, как показано на рисунке.

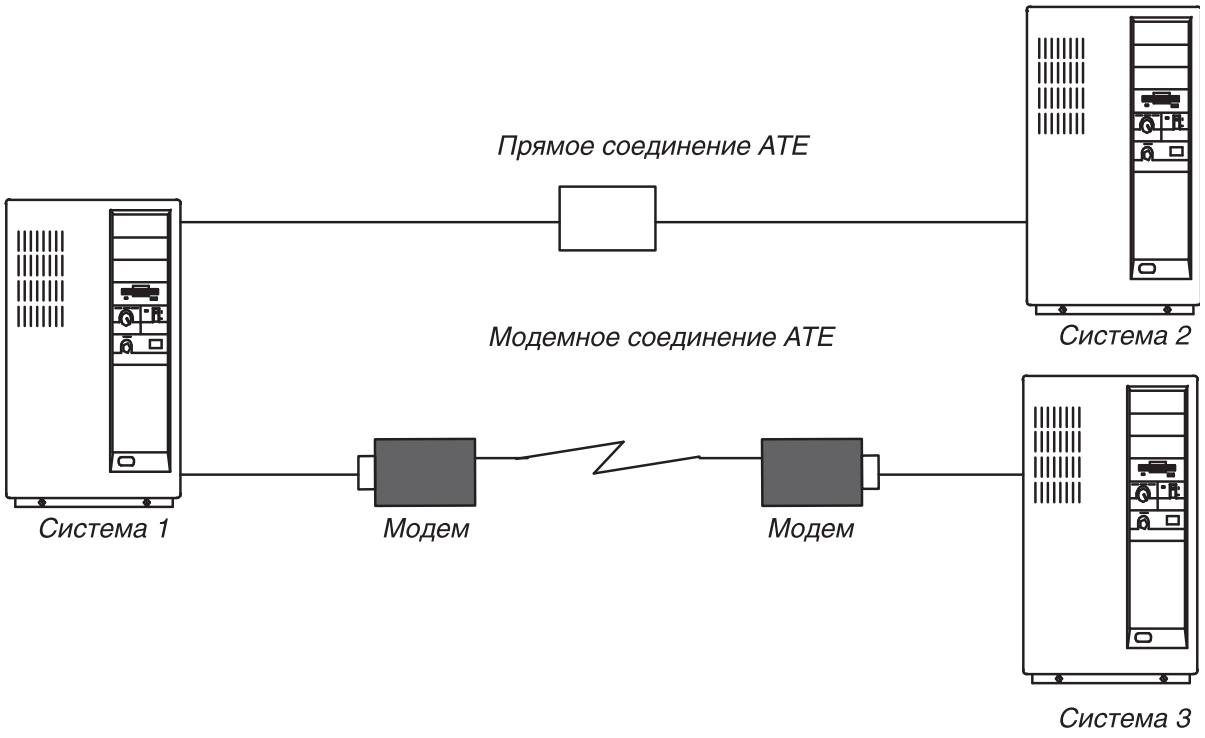


Рисунок 41. Типы соединения ATE

В зависимости от типа соединения, пользователь может настроить ATE для работы с удаленной системой, которая может находиться как в соседней комнате, так и в другой стране. При прямом соединении пользователь должен знать номер порта системы. При соединении через модем пользователь должен знать номер порта своей системы и телефонный номер удаленной системы. Кроме того, у пользователя должны быть имя пользователя и пароль для входа в удаленную систему.

ATE позволяет пользователю выполнять команды в удаленной системе, отправлять и получать файлы, а также проверять целостность данных в передаваемых файлах с помощью протокола **xmodem**. Кроме того, пользователь может собирать данные, поступающие из удаленной системы, и сохранять их в файле.

> **Примечание:** Для работы с ATE пользователь должен входить в группу UUCP. Добавление отдельных пользователей в группы выполняется пользователем с правами доступа root с помощью Инструмента управления системой (SMIT).

Настройка ATE

Перед запуском ATE системный администратор должен установить необходимое программное обеспечение, если это еще не сделано, и настроить порты и соединения терминалов.

- ATE - это дополнительный программный продукт. Все файлы, необходимые для работы ATE, содержатся в программном продукте **bos.net.ate** на установочном носителе. С помощью следующих команд проверьте, установлена ли программа ATE в вашей системе:

  ```
  lslpp -h | more   <return>
  /bos.net.ate      <return>
  ```

  Если ATE отсутствует в вашей системе, установите образ **bos.net.ate** с установочного носителя (магнитной ленты, дискеты или сетевого сервера).
- Если ATE установлена в вашей системе, то можно посмотреть список файлов, связанных с этой программой, с помощью следующих команд:

  ```
  lslpp -f | more   <return>
  /bos.net.ate      <return>
  ```
- Для настройки номера порта устройства связи пользователь должен войти в систему под именем root.

ATE поддерживает как прямые соединения (по кабелю), так и соединения по модему. Локальные соединения RS-232C устанавливаются для компьютеров, расстояние между которыми не превышает 15 метров (50 футов), а соединения RS-422A устанавливаются для компьютеров, находящихся на расстоянии не больше 1200 метров (4000 футов).

Перед вызовом удаленной системы с помощью ATE убедитесь, что терминал удаленной системы готов к приему вызовов.

Для подготовки к запуску ATE выполните следующие действия:

1. Установите карту асинхронного адаптера в нужный разъем системного блока, если в системе отсутствует встроенный последовательный порт.
2. Подключите кабель RS-232C или RS-422A к гнезду карты адаптера или встроенному последовательному порту.
3. Создайте терминал и выберите для него порт связи с помощью команды быстрого доступа `smit mkdev`.
4. Выберите тип терминала для эмуляции ATE и внесите необходимые изменения в параметры среды. Чаще всего необходимо задать быстродействие линии, параметры контроля четности, число бит на символ, тип линии (локальная или удаленная). Если вы планируете применять Поддержку национального языка (NLS), то укажите значения BPC 8 и no parity (отсутствие контроля четности).
5. Выберите порт для устройства. Для того чтобы порт применялся только для обработки исходящих вызовов ATE, введите команду **pdisable**. Например, для настройки порта tty1 введите:

   ```
   pdisable tty1
   ```

   Для того чтобы через порт принимались все входящие вызовы, введите команду **penable**. Например, если для приема вызовов должен применяться порт tty2, введите:

   ```
   penable tty2
   ```
6. Убедитесь, что устройство было определено в удаленной системе. После определения программу ATE нужно настроить в соответствии с параметрами устройства удаленной системы. Измените значения по умолчанию с помощью команд alter и modify или путем редактирования файла значений по умолчанию `ate.def`. Значения по умолчанию для телефонного соединения можно переопределить в соответствующей записи файла списка телефонных номеров.

Главные меню ATE

ATE отображает меню исходя из введенных подкоманд.

После запуска ATE с помощью команды **ate** появляется главное меню, которые позволяет выполнять следующие операции:

- Временное изменение характеристик ATE (**modify** и **alter**).
- Подключение к другой системе (**directory** и **connect**).
- Просмотр справки (**help**).

- Выполнение команд операционной системы (**perform**)
- Завершение работы с ATE (**quit**).

В зависимости от подкоманды, выбранной в главном меню, появится соответствующее подменю:

Таблица 107. Подменю ATE

Выбранная подкоманда	**Показанное меню ATE**
Подкоманда **modify**	Редактировать меню (дополнительные сведения приведены в описании команды **ate** в *Справочник по командам, том 1*)
Подкоманда **alter**	Изменить меню (дополнительные сведения приведены в описании команды **ate** в *Справочник по командам, том 1*)
Подкоманды **connect** и **directory** для подключения к удаленной системе	главное меню (соединение установлено)
Подкоманда **directory**	Телефонный справочник (список телефонных номеров)

При наличии установленного соединения в главном меню будут перечислены подкоманды для выполнения следующих действий:

- Отправка файлов в удаленную систему и получение файлов из удаленной системы (**send** и **receive**).
- Прерывание текущей операции, выполняемой в удаленной системе (**break**).
- Завершение соединения с удаленной системой (**terminate**).

Кроме того, в данном меню доступны подкоманды **modify**, **alter**, **help**, **perform** и **quit**, которые выполняют те же действия, что и аналогичные подкоманды главного меню при отсутствии соединения.

Существует ряд функций, которые можно выполнять с помощью управляющих клавиш ATE. Эти клавиши называются CAPTURE_KEY, MAINMENU_KEY и PREVIOUS_KEY. Описание управляющих клавиш приведено в разделе “Управляющие клавиши ATE” на стр. 638. При установке ATE управляющим клавишам присваиваются значения по умолчанию, но вы можете изменить соответствующие комбинации клавиш, указанные в файле `ate.def`.

Главное меню ATE (соединение не установлено):

Воспользуйтесь командой **ate**, чтобы открыть главное меню ATE при отсутствии соединения.

После установления соединения выберите подкоманду ATE **connect** чтобы открыть главное меню для установленного соединения.

В главном меню ATE при не установленном соединении можно выполнять следующие подкоманды. Для запуска подкоманды введите в командном приглашении меню первую букву этой подкоманды. Например, для запуска подкоманды **directory** введите букву **d**.

Элемент	**Описание**
alter	Позволяет временно изменить характеристики передачи данных, например, скорость передачи.
connect	Устанавливает соединение.
directory	Показывает телефонный справочник.
help	Предназначена для просмотра справочной информации.
modify	Позволяет временно изменить локальные параметры, например, файл для хранения полученных данных.
perform	Предназначена для выполнения команд операционной системы при работе с ATE.
quit	Завершает работу программы ATE.

Примечание: При отсутствии соединения в главном меню ATE из всех управляющих клавиш (CAPTURE_KEY, MAINMENU_KEY и PREVIOUS_KEY) доступна только PREVIOUS_KEY.

Главное меню ATE (соединение установлено):

В главном меню ATE при не установленном соединении выберите подкоманду **connect**, чтобы открыть главное меню, соответствующее установленному соединению.

Вы также можете нажать клавишу MAINMENU_KEY при установленном соединении с удаленной системой.

В главном меню ATE при установленном соединении можно выполнять следующие подкоманды. Определения этих подкоманд приведены в описании команды **ate** в *Справочник по командам, том 1*. Для запуска подкоманды введите в командном приглашении меню первую букву этой подкоманды. Например, для запуска команды `alter` введите букву **a**.

Элемент	Описание
alter	Позволяет временно изменить характеристики передачи данных, например, скорость передачи.
break	Прерывает текущую операцию, выполняемую в удаленной системе.
help	Предназначена для просмотра справочной информации.
modify	Позволяет временно изменить локальные параметры, например, файл для хранения принимаемых данных.
perform	Предназначена для выполнения команд операционной системы при работе с ATE.
quit	Завершает работу программы ATE.
receive	Предназначена для получения файлов из удаленной системы.
send	Предназначена для отправки файлов в удаленную систему.
terminate	Завершает соединение ATE.

При наличии соединения в главном меню ATE доступны все три управляющие клавиши.

Управляющие клавиши ATE

Ниже приведено описание управляющих клавиш, применяемых при работе с ATE. Для изменения комбинаций клавиш необходимо отредактировать файл `ate.def`.

Элемент	Описание
CAPTURE_KEY	Начинает или заканчивает сохранение данных, отображаемых на экране в течение сеанса эмуляции. По умолчанию для вызова функции CAPTURE_KEY применяется комбинация клавиш Ctrl-B. Функция CAPTURE_KEY работает как переключатель. При первом нажатии управляющей комбинации клавиш начинается сохранение данных. При повторном нажатии этой комбинации клавиши сохранение данных заканчивается. Данные сохраняются в файле, имя которого указано в файле `ate.def`. По умолчанию для хранения поступающих данных применяется файл `$HOME/capture`. Для временной замены этого файла введите подкоманду **modify**. Для того чтобы другой файл применялся во всех сеансах ATE, необходимо отредактировать файл значений по умолчанию для ATE. См. раздел “Редактирование файла ATE по умолчанию” на стр. 647. Если выполняется передача файлов или соединение не установлено, управляющие клавиши CAPTURE_KEY недоступны. Если соединение еще не установлено, то при нажатии управляющих клавиш CAPTURE_KEY следующая команда выполнена не будет и появится сообщение об ошибке.
PREVIOUS_KEY	Предназначена для возврата к предыдущему меню. Кроме того, функция PREVIOUS_KEY применяется для завершения передачи файлов. По умолчанию для PREVIOUS_KEY применяется комбинация клавиш Ctrl-R. Функция PREVIOUS_KEY доступна в главном меню как при наличии, так и при отсутствии соединения.

Элемент	Описание
MAINMENU_KEY	При нажатии этой клавиши появится главное меню, соответствующее установленному соединению, и пользователь сможет выполнить подкоманду ATE. По умолчанию для вызова функции MAINMENU_KEY применяется комбинация клавиш Ctrl-V. Нажав эту комбинацию клавиш, вы можете перейти к главному меню, когда соединение уже установлено. Если соединение еще не установлено, то при нажатии управляющих клавиш MAINMENU_KEY следующая команды выполнена не будет и появится сообщение об ошибке. Для того чтобы изменить параметры управляющих клавиш или имя файла поступающих данных для всех сеансов, необходимо отредактировать файл ATE по умолчанию. См. См. раздел “Редактирование файла ATE по умолчанию” на стр. 647.

Настройка ATE

При первом запуске программа ATE создает файл значений по умолчанию `ate.def` в текущем каталоге. Файл `ate.def` применяется для настройки различных характеристик ATE.

Например, можно изменить имя файла, содержащего телефонные номера, тип протоколов передачи данных, применяемых для отправления и приема файлов из удаленной системы, и скорость передачи данных через модем в бодах. Дополнительная информация о файле `ate.def` приведена в разделе “Редактирование файла ATE по умолчанию” на стр. 647.

Кроме того, в параметры ATE можно вносить временные изменения с помощью команд **modify** и **alter**. С помощью этих команд можно изменять все значения по умолчанию, применяемые в ATE, кроме комбинаций управляющих клавиш (которые можно изменить только путем редактирования файла значений по умолчанию) и имени файла со списком телефонных номеров (которое можно изменить с помощью команды **directory** или путем редактирования файла значений по умолчанию). Изменения, внесенные с помощью команды **modify**, **alter** или **directory**, действуют только в текущем сеансе ATE. При следующем запуске ATE будут применены параметры из файла значений по умолчанию.

При использовании ATE с подключением через модем можно создать список, содержащий до 20 телефонных номеров. Команда **directory** выдает список телефонных номеров в виде меню, что позволяет выбрать удаленную систему для вызова. Дополнительная информация приведена в разделе “Настройка каталога со списком телефонных номеров ATE” на стр. 643.

Список телефонных номеров позволяет избежать поиска номера удаленной системы для ее вызова. Кроме того, в списке телефонных номеров можно указать определенные параметры передачи данных. Это может быть полезно, если некоторые соединения используют параметры, отличающиеся от значений ATE по умолчанию.

Пользователь может создать свой личный телефонный справочник, а системный администратор может создать телефонный справочник, доступный для всех пользователей. Применяемый телефонный справочник указывается в файле значений по умолчанию ATE. Дополнительная информация приведена в разделе “Настройка каталога со списком телефонных номеров ATE” на стр. 643.

Файл конфигурации ate.def:

Файл `ate.def` содержит значения по умолчанию для асинхронных соединений и передачи файлов.

Он создается в текущем каталоге во время первого запуска ATE. Файл `ate.def` содержит следующие значения по умолчанию для программы ATE:

- Параметры передачи данных
- Параметры локальной системы
- Имя файла со списком телефонных номеров
- Список управляющих клавиш

При первом запуске программы ATE из любого каталога она создает файл `ate.def` в этом каталоге.

```
LENGTH          8
STOP            1
PARITY          0
RATE            1200
DEVICE          tty0
INITIAL         ATDT
FINAL
WAIT            0
ATTEMPTS        0
TRANSFER        p
CHARACTER       0
NAME            kapture
LINEFEEDS       0
ECHO            0
VT100           0
WRITE           0
XON/XOFF        1
DIRECTORY       /usr/lib/dir
CAPTURE_KEY     002
MAINMENU_KEY    026
PREVIOUS_KEY    022
```

Для того чтобы изменить и сохранить значения этих параметров, отредактируйте с помощью любого текстового редактора ASCII файл `ate.def`. Временно изменить значения этих параметров можно с помощью команд ATE **alter** и **modify**, которые доступны из главного меню ATE.

Имена параметров в файле `ate.def` следует вводить прописными буквами. Их следует указывать в точности так, как они написаны в файле значений по умолчанию. Каждый параметр должен занимать отдельную строку. Если параметр задан неверно, ATE выдаст сообщение об ошибке. Однако программа продолжит работу, используя значение по умолчанию для этого параметра. Ниже перечислены параметры файла `ate.def`:

LENGTH

Указывает число битов в символе данных. Длина должна совпадать со значением, ожидаемым удаленной системой.

```
Варианты:  7 или 8
По умолчанию:  8
```

STOP Указывает число стоп-битов, добавляемых к символу для обозначения его конца при передаче данных. Это число должно совпадать с числом стоп-битов, ожидаемым удаленной системой.

```
Варианты:  1 или 2
По умолчанию:  1
```

PARITY

Проверяет, успешно ли был передан или получен символ от удаленной системы. Значение должно совпадать со значением контроля четности в удаленной системе.

Например, если пользователь выбрал контроль по четности, то в случае, когда в символе нечетное число единичных битов, бит контроля четности включается, чтобы сделать общее число единичных битов четным.

```
Варианты: 0 (нет), 1 (контроль по нечетности) или
2 (контроль по четности)
По умолчанию:  0.
```

RATE Указывает скорость передачи в бодах, или число битов, передаваемых за одну секунду (бит/с). Это значение должно совпадать с быстродействием модема и с аналогичным значением удаленной системы.

```
Варианты:  50,75,110,134,150,300,600,1200,1800,2400,4800,9600,19200
По умолчанию:  1200
```

Устройство

Указывает имя асинхронного порта для соединения с удаленной системой.

```
Варианты: локально созданные имена портов.
По умолчанию: tty0.
```

INITIAL

Указывает префикс - строку, добавляемую перед телефонным номером при автоматическом дозвоне через модем. Для правильного использования команд модема ознакомьтесь с документацией по модему.

```
Варианты: ATDT, ATDP или другие команды, в зависимости от типа модема.
По умолчанию:  ATDT.
```

FINAL

Указывает суффикс - строку, добавляемую после телефонного номера при автоматическом дозвоне через модем. Для правильного использования команд модема ознакомьтесь с документацией по модему.

```
Варианты: пустой суффикс или допустимый суффикс
модема.
По умолчанию: нет значения по умолчанию.
```

WAIT Указывает интервал ожидания между попытками дозвона. Ожидание начинается при прерывании попытки дозвона или ее завершении по тайм-ауту. Если параметр ATTEMPTS равен 0, то повторные попытки дозвона не выполняются.

```
Варианты: 0 (нет) или положительное целое число, задающее время ожидания в
секундах.
По умолчанию:  0
```

ATTEMPTS

Указывает максимальное число попыток повторного набора номера программой ATE. Если параметр ATTEMPTS равен 0, то повторные попытки дозвона не выполняются.

```
Варианты: 0 (нет) или положительное целое число,
задающее число попыток повторного набора номера.
По умолчанию:  0
```

TRANSFER

Указывает тип асинхронного протокола для передачи файлов по соединению.

p (окно подтверждения)

Этот протокол передачи файлов управляет передачей данных, ожидая заданный символ или делая заданную паузу между передачей строк. Это помогает предотвратить потерю данных, если передаваемые блоки слишком велики или передаются слишком быстро и система не успевает их обрабатывать.

x (xmodem)

8-разрядный протокол передачи файлов, позволяющий обнаружить ошибки передачи данных и повторить передачу.

```
Варианты: p (окно подтверждения), x (xmodem)
По умолчанию: p.
```

CHARACTER

Указывает тип используемого протокола подтверждения. Это сигнал для передачи строки. Выберите один символ.

Когда команда **send** встречает символ конца строки при передаче данных, она ожидает символ подтверждения, прежде чем отправить следующую строку.

Когда команда **receive** готова к приему данных, она отправляет символ подтверждения и ждет 30 секунд. При обнаружении символа возврата каретки команда **receive** вновь отправляет символ подтверждения. Выполнение команды **receive** завершается, если в течение 30 секунд она не получает данных.

Варианты: любой символ
По умолчанию: 0

Interval

Системное время ожидания между передачей строк, в секундах. Это значение должно быть целым числом. Значение по умолчанию равно 0, что соответствует нулевому времени ожидания.

По умолчанию: 0.

NAME

Имя файла для принимаемых данных.

Варианты: любое допустимое имя файла длиной менее 40 символов.
По умолчанию: kapture

LINEFEEDS

Добавляет символ перевода строки после каждого символа возврата каретки в принимаемом потоке данных.

Варианты: 1 (вкл.) или 0 (выкл.).
По умолчанию: 0.

ECHO Отображает введенные данные на экране отправителя. Если на удаленном компьютере включен эхоповтор, то все отправленные символы возвращаются обратно и отображаются на экране. При включенном параметре ECHO каждый символ отображается дважды: сначала при вводе, а затем еще раз, когда он возвращается по соединению. Когда параметр ECHO выключен, каждый символ отображается только один раз, когда он возвращается по соединению.

Варианты: 1 (вкл.) или 0 (выкл.).
По умолчанию: 0.

VT100 Локальная консоль эмулирует терминал DEC VT100, что позволяет применять код DEC VT100 при работе с удаленной системой. При отключенном параметре VT100 локальная консоль функционирует как рабочая станция.

Варианты: 1 (вкл.) или 0 (выкл.).
По умолчанию: 0.

WRITE

Захватывает входящие данные и направляет их в файл приема, указанный в параметре NAME, а также на дисплей. Комбинации символов возврата каретки и перевода строки перед записью в файл приема преобразуются в символы перевода строки. Если файл уже создан, данные добавляются в его конец.

При работе с соединением можно включать и выключать режим приема с помощью клавиши приема CAPTURE_KEY (обычно это комбинация клавиш Ctrl-B).

Варианты: 1 (вкл.) или 0 (выкл.).
По умолчанию: 0.

XON/XOFF

Контролирует передачу данных через порт следующим образом:

- При получении сигнала XOFF передача прекращается.
- При получении сигнала XON передача возобновляется.
- Сигнал XOFF отправляется, когда буфер приема почти заполнен.
- Сигнал XON отправляется, когда буфер приема освобождается.

  Варианты: 1 (вкл.) или 0 (выкл.).
  По умолчанию: 1.

DIRECTORY

Указывает имя файла со списком телефонных номеров.

По умолчанию: /usr/lib/dir.

CAPTURE_KEY

Задает комбинацию клавиш для переключения режима приема. Нажатие CAPTURE_KEY (обычно

это комбинация клавиш Ctrl-B) включает или выключает режим приема (сохранения) данных, отображаемых на экране во время активного соединения.

```
Варианты: любой управляющий символ ASCII.
По умолчанию: символ с восьмеричным кодом ASCII 002 (STX).
```

MAINMENU_KEY

Задает комбинацию клавиш для вызова главного меню соединения, чтобы пользователь мог запускать команды во время активного соединения. MAINMENU_KEY (обычно это комбинация клавиш Ctrl-V) работает только при наличии соединения.

```
Варианты: любой управляющий символ ASCII.
По умолчанию: символ с восьмеричным кодом ASCII 026 (SYN).
```

PREVIOUS_KEY

Задает комбинацию клавиш для показа предыдущего содержимого экрана в любой момент работы программы. Показанная на экране информация будет зависеть от того, когда пользователь нажимает PREVIOUS_KEY (обычно это комбинация клавиш Ctrl-R).

```
Варианты: любой управляющий символ ASCII.
По умолчанию: символ с восьмеричным кодом ASCII 022 (DC2).  Управляющий символ
ASCII привязан к сигналу прерывания.
```

Настройка каталога со списком телефонных номеров ATE

В файле со списком телефонных номеров ATE содержатся телефонные номера, которые программа ATE использует для установки удаленных соединений с помощью модема.

Для настройки каталога с номерами телефонов для доступа к удаленным системам ATE, требуется выполнить следующие предварительные требования:

- В системе должна быть установлена программа эмуляции асинхронного терминала (ATE)
- Для настройки системного телефонного справочника у пользователя должны быть права на запись в файл `/usr/lib/dir`.

Пользователи могут присвоить любое имя этому файлу и разместить его в любом каталоге, доступном для чтения и записи. Редактировать файл можно с помощью любого текстового редактора ASCII. По умолчанию список телефонных номеров содержится в файле `/usr/lib/dir`, как показано ниже:

Примечание: В приведенном ниже примере некоторые записи ATE разбиты на отдельные строки для удобства чтения. В настоящем файле со списком телефонных номеров все элементы записи расположены в одной неразрывной строке.

```
#   COMPONENT_NAME: BOS dir
#
#   FUNCTIONS:
#
#   ORIGINS: 27
#
#
#   (C) COPYRIGHT International Business Machines Corp. 1985, 1989
#   Лицензионные материалы - собственность IBM
#
#   US Government Users Restricted Rights - Use, duplication or
#   disclosure restricted by GSA ADP Schedule Contract with IBM Corp.
#
#   dir - пример каталога со списком телефонных номеров
#
#
# Micom   9,555-9400 1200 7 1 2 0 0
# R20     9,555-9491 1200 7 1 2 0 0
# QT      9,555-8455 1200 7 1 2 0 0
# Dallas1 9,555-7051 1200 8 1 0 0 0
```

Список телефонных номеров ATE можно просмотреть с помощью подкоманды **directory**, доступной в **главном меню при отсутствии соединений**. На экране будет показан список телефонных номеров в том виде, в котором он выдается при работе с программой ATE.

Можно создать несколько списков телефонных номеров. Для переключения ATE на другой список номеров измените файл `ate.def` в текущем каталоге.

> **Примечание:** Файл списка телефонных номеров может содержать до 20 строк (по одной записи в каждой строке). ATE игнорирует все дополнительные записи.

Файл со списком телефонных номеров можно сравнить с телефонной книгой, содержащей телефонные номера удаленных систем, с которыми связывается программа ATE. Ниже приведен формат записи в списке телефонных номеров:

```
Имя Телефон Скорость Длина Стоп-бит Четность Эхо Перевод_строки
```

Эти поля должны быть разделены по крайней мере одним пробелом. Для удобства их можно разделить несколькими пробелами. Ниже приведено описание каждого поля:

Имя Идентифицирует телефонный номер. Длина имени ограничена 20 символами. Вместо пробелов в имени следует указывать знак подчеркивания "_", например: data_bank.

Телефон
Номер телефона для подключения к удаленной системе. Длина записи ограничена 40 символами. Список допустимых цифр и символов приведен в документации по модему. Например, если для выхода на внешнюю линию требуется набрать 9, поставьте 9, (девятка и запятая) перед номером: 9,1112222.

Несмотря на то, что телефонный номер может содержать до 40 символов, команда просмотра списка выдает только первые 26 символов.

Скорость
Скорость передачи в битах в секунду (бит/с). Указывает число символов, передаваемых за одну секунду. Выберите скорость передачи, соответствующую применяемой линии связи. Допустимы следующие значения скорости:

50, 75, 110, 134, 150, 300, 600, 1200, 1800, 2400, 4800, 9600 или 19200.

В случае скоростей передачи данных, отличных от POSIX, выбор скорости 50 приведет к тому, что ATE будет использовать скорость, указанную посредством SMIT для этого устройства.

Длина Число бит, образующих символ. Значением этого поля может быть 7 или 8.

Стоп-бит
Стоп-бит указывает на конец символа. Значением этого поля может быть 1 или 2.

Четность
Указывает, следует ли проверять правильность передачи или приема символа из удаленной системы. Варианты: 0 (нет), 1 (контроль по нечетности) или 2 (контроль по четности).

Эхо Указывает, следует ли отображать вводимые символы на локальном дисплее. Значением поля может быть 0 (выкл.) или 1 (вкл.).

Перевод_строки
Ставит символ перевода строки в конце каждой строки данных, получаемой из удаленной системы. Символ перевода строки выполняет те же функции, что и символы возврата каретки и новой строки. Значением поля может быть 0 (выкл.) или 1 (вкл.).

Примечание: Если комбинации клавиш совпадают в различных приложениях, то может потребоваться их изменение. Например, в случае конфликта комбинаций клавиш в программе ATE и в текстовом редакторе следует изменить комбинации управляющих клавиш в ATE.

Примечание: Управляющий символ ASCII может быть в восьмеричном, десятичном или шестнадцатеричном формате, например:

восьмеричный
От 000 до 037. Должен начинаться с нуля.

десятичный
От 0 до 31.

шестнадцатеричный
от 0x00 до 0x1F. Должен начинаться с 0x. Символ X можно указывать в любом регистре.

Создайте файл `ate.def` с этими параметрами для изменения параметров эмуляции ATE. Например, для изменения RATE на 300 бит/с, DEVICE на tty3, режима передачи TRANSFER на x (протокол xmodem) и DIRECTORY на my.dir создайте файл `ate.def` со следующими записями в каталоге запуска программы ATE:

```
RATE          300
DEVICE        tty3
TRANSFER      x
DIRECTORY     my.dir
```

Программа будет использовать установленные значения после запуска из этого каталога.

1. Создайте файл телефонного справочника:
   a. Перейдите в каталог, где должен быть расположен файл телефонного справочника.
   b. Скопируйте файл шаблона `/usr/lib/dir`. Измените имя этого файла.
   c. Создайте записи телефонных номеров в формате согласно формату файла телефонного справочника.
   d. Сохраните этот файл.

   Примечание: Если новый телефонный справочник будет применяться по умолчанию для всей системы, то сохраните его в файле с именем `/usr/lib/dir`.
2. Если файлу телефонного справочника присвоено имя, отличное от значения по умолчанию (`/usr/lib/dir`), то измените файл `ate.def` , находящийся в каталоге, из которого запускается программа ATE. В параметре *DIRECTORY* файла `ate.def` нужно указать новое имя файла телефонного справочника. См. Обратитесь к разделу “Редактирование файла ATE по умолчанию” на стр. 647
3. Запустите ATE и просмотрите каталог с номерами телефонов с помощью команды **directory**.

Вызов с набором номера ATE

С помощью следующих действий можно установить соединение с удаленной системой, используя ATE и пользовательский файл со списком телефонных номеров `/usr/lib/dir`.

Перед тем, как устанавливать соединение через модем, убедитесь, что все предварительные требования выполнены.

- В системе должна быть установлена программа ATE.
- Модем должен быть подключен, настроен и готов к работе.
- Пользователь должен входить в рабочую группу UUCP (дополнительная информация содержится в разделе “Настройка ATE” на стр. 635).
- Файл со списком телефонных номеров `/usr/lib/dir` уже содержит всю необходимую информацию.
- В текущем рабочем каталоге пользователя (`pwd`) находится файл `ate.def` со всеми необходимыми изменениями.
- В SMIT полю входа в систему для порта `/dev/tty` должно быть присвоено значение Отключить, Общий или Задержка.

1. Введите следующую команду:
   ```
   ate
   ```
2. В главном меню введите `d` и нажмите Enter.

3. Введите имя нужного файла со списком телефонных номеров и нажмите Enter. Для работы с текущим файлом просто нажмите Enter.
4. Введите номер записи файла в столбце # для набора соответствующего телефонного номера.

Передача файла с помощью ATE

С помощью следующих действий можно передать файл из локальной системы в удаленную.

Прежде чем приступать к передаче файла с помощью ATE, убедитесь, что выполнены все предварительные требования, перечисленные ниже:

- Соединение с помощью программы **ATE** должно быть уже установлено.
- В локальной и удаленной системах должен быть установлен протокол передачи файлов Xmodem. В операционной системе протокол Xmodem расположен в каталоге `/usr/bin`.

1. После входа в удаленную систему выполните следующую команду **xmodem**:

   ```
   xmodem -r новый_файл
   ```

   где **r** - это флаг Xmodem для приема данных, а *новый_файл* - имя принимаемого файла. Это имя может не совпадать с именем передаваемого файла.
2. Нажмите Enter.
3. Появится следующее сообщение:

   ```
   ate: 0828-005 Система готова получить файл "новый_файл". Для прекращения работы xmodem нажмите Ctrl-X.
   ```

   Если это сообщение не показано, то, скорее всего, программа **xmodem** не установлена или ее каталог не указан в PATH.
4. Нажмите Ctrl-V для возврата в главное меню ATE соединения ATE CONNECTED MAIN MENU.
5. Нажмите S для отправки файла.
6. Появится следующее сообщение:

   ```
   Введите имя отправляемого файла и нажмите Enter. Для использования последнего
   файла () просто нажмите Enter.
   ```
7. Введите имя и полный путь к файлу, который следует передать.
8. Нажмите Enter.
9. ATE выдаст следующее сообщение и приступит к передаче файла:

   ```
   ate: 0828-024 Программа готова отправить файл "новый_файл". Вы получите еще одно сообщение
   по окончании передачи файла.
   ate: 0828-025 Система отправляет блок 1.
   ate: 0828-025 Система отправляет блок 2.
   ate: 0828-015 Передача файла завершена.
   ate: 0828-040 Нажмите Enter
   ```
10. Нажмите Enter по окончании передачи.

Получение файла с помощью ATE

С помощью этой процедуры можно получить файл от удаленного хоста.

Прежде чем приступать к приему файла с помощью ATE, убедитесь, что выполнены все предварительные требования, перечисленные ниже:

- Соединение с помощью программы ATE должно быть уже установлено.
- В локальной и удаленной системах должен быть установлен протокол передачи файлов Xmodem. В операционной системе протокол Xmodem расположен в каталоге `/usr/bin`.

1. После входа в удаленную систему выполните следующую команду **xmodem**:

   ```
   xmodem -s новый_файл
   ```

   где **s** - это флаг **xmodem** для отправки, а *новый_файл* - имя и полный путь к передаваемому файлу.
2. Нажмите Enter.
3. Появится следующее сообщение:

   ```
   ate: 0828-005 Система готова отправить файл "новый_файл". Для прекращения работы xmodem нажмите Ctrl-X.
   ```

Если это сообщение не показано, то, скорее всего, программа **xmodem** не установлена или ее каталог не указан в PATH.

4. Нажмите Ctrl-V для возврата в главное меню ATE соединения ATE CONNECTED MAIN MENU.
5. Нажмите R для приема файла.
6. Появится следующее сообщение:

```
Введите имя файла для приема входящих данных и нажмите
Enter.  Для использования последнего указанного файла () просто нажмите Enter.
```

7. Введите имя и полный путь к файлу, который следует передать.
8. Нажмите Enter.
9. ATE выдаст следующее сообщение и приступит к передаче файла:

```
ate: 0828-020 Программа готова к приему файла "новый_файл". Вы получите другое сообщение
по окончании передачи файла.
ate: 0828-028 Система принимает блок 1.
ate: 0828-028 Система принимает блок 2.
ate: 0828-040 Нажмите Enter
```

10. Нажмите Enter по окончании передачи.

Редактирование файла ATE по умолчанию

Для изменения файла ATE по умолчанию следует настроить программу ATE в системе.

Для изменения параметров настройки, указанных в файле `ate.def`, выполните следующие действия:

1. Откройте файл `ate.def` с помощью текстового редактора.
2. Измените значения нужных параметров. Остальные значения можно удалить или оставить прежними. Для всех удаленных параметров будут применяться значения по умолчанию.
3. Сохраните измененный файл `ate.def`.

Изменения, внесенные в файл `ate.def`, вступят в силу при следующем запуске ATE из каталога, содержащего файл `ate.def`.

Копию файла `ate.def` можно сохранить в любом каталоге, к которому у вас есть права доступа на чтение и запись. Например, если вы хотите создать несколько наборов значений по умолчанию для программы ATE, то разместите необходимое число копий файла `ate.def` с нужными наборами значений в разных подкаталогах каталога `$HOME`. Однако учтите, что большое число копий файла `ate.def` займет значительный объем памяти. В качестве альтернативы большинство параметров можно изменять временно с помощью подкоманд ATE **alter** и **modify**. Параметры для отдельного соединения можно указать в записи телефонного справочника. См. См. раздел “Настройка каталога со списком телефонных номеров ATE” на стр. 643.

Устранение неполадок ATE

При обнаружении следующих распространенных неполадок используйте приведенные варианты их решения.

Неполадка:
При отправке или приеме файлов команда **xmodem** зависает. Неполадка исчезает при нажатии Ctrl-X.

Вариант:
Откройте меню Alter и убедитесь, что используется протокол xmodem (или Способ передачи).

Неполадка:
При отправке или приеме файлов происходит прокрутка содержимого файла по экрану и выдается сообщение о том, что процесс передачи завершен, хотя это не так.

Вариант:
Откройте меню Alter и убедитесь, что используется протокол **xmodem** (или Способ передачи).

Неполадка:
При запуске ATE выдается следующее сообщение:

```
ate: 0828-008 Системе не удалось открыть порт /dev/tty0. Если имя порта задано неверно,
измените его с помощью меню Изменить. Либо выполните действия, указанные в
приведенном ниже системном сообщении.

Подключение: права доступа к файлу запрещают указанное действие.
ate: 0828-040 Нажмите Enter
```

Вариант:

Строка Подключение: в сообщении об ошибке сужает диапазон возможных причин неполадки. Убедитесь, что пользователь, запускающий ATE, входит в группу UUCP. Для этого пользователь может ввести *id* в командной строке; в появившемся списке должно быть указано `uucp`.

Неполадка:

При попытке установить соединение с помощью ATE появляется следующее сообщение:

```
ate: 0828-008 Системе не удалось открыть порт /dev/tty0. Если имя порта задано неверно,
измените его с помощью меню Изменить. Либо выполните действия, указанные в
приведенном ниже
системном сообщении.

Подключение: файл или каталог, указанный в полном имени, не существует.
ate: 0828-040 Нажмите Enter
```

Вариант:

Для использования с программой ATE был выбран недоступный или неправильный терминал. Изучите меню Alter в ATE.

Неполадка:

Передача файла проходит нормально, но размер полученного файла больше, чем отправленного.

Вариант:

Протокол xmodem заполняет файл во время передачи. Во избежание этого выполните сжатие файла с помощью команды **tar**, а затем отправьте его. Это также позволяет обойти ограничение протокола xmodem на одновременную передачу только одного файла. С помощью команды **tar** можно объединить несколько файлов в один образ tar и затем передать его с помощью xmodem.

Команды и подкоманды ATE

Список команд и подкоманд ATE с кратким описанием их принципа действия.

Дополнительная справочная информация приведена в разделе “Форматы файла ATE” на стр. 649.

Элемент	Описание
ate	Запускает программу ATE. Определение подкоманд приведено в описании команды **ate**:
	break Прерывает выполнение текущей операции в удаленной системе.
	connect Устанавливает соединение с удаленной системой.
	directory Предназначена для просмотра телефонного справочника ATE и выбора записи, позволяющей подключиться к удаленной системе.
	help Предназначена для просмотра справки по командам ATE.
	perform Предназначена для запуска команд операционной системы рабочей станции при работе с программой ATE.
	quit Завершает работу программы ATE.
	receive Предназначена для получения файлов из удаленной системы.
	send Предназначена для отправки файлов в удаленную систему.
	terminate Завершает соединение ATE с удаленной системой.

Кроме того, команда **xmodem** предназначена для передачи файлов с помощью протокола xmodem, поддерживающего обнаружение ошибок при передаче данных в асинхронном режиме.

Форматы файла ATE

К форматам файлов, применяемым программой Эмуляции асинхронного терминала (ATE), относится формат `ate.def` и формат телефонного справочника.

Элемент	Описание
`ate.def`	Содержит значения по умолчанию для соединений.
Каталог с телефонными номерами	Содержит телефонные номера и параметры отдельных соединений.

Дополнительная справочная информация приведена в разделе “Команды и подкоманды ATE” на стр. 648.

Утилита динамического выбора окна

Утилита динамического выбора окна, реализованная в виде команды **dscreen**, позволяет подключить один физический терминал одновременно к нескольким виртуальным сеансам (экранам).

В основном эта утилита предназначена для терминалов с памятью на две и более страниц (например, IBM 3151 модель 310 и 410 с дополнительной кассетой). Для таких терминалов переключение между виртуальными окнами означает переключение между страницами окна физического терминала, что позволяет сохранить, а затем восстановить содержимое каждого виртуального окна. Если терминал не поддерживает память на несколько страниц, то команда **dscreen** также применяется для переключения между виртуальными окнами, хотя при этом состояние содержимого окна не сохраняется.

Примечание: Для полной поддержки утилиты **dscreen** на терминале должна быть предусмотрена функция переключения между внутренними страницами окна и запоминания позиции курсора на каждой странице. Команда **dscreen** может применяться при работе как с программируемыми, так и непрограммируемыми терминалами. Однако в случае непрограммируемых терминалов состояние содержимого окна при переключении с одного окна на другое не сохраняется.

Файл сведений о конфигурации терминала dscreen

В файле конфигурации терминала **dscreen** (файле `dsinfo`) определяются различные комбинации клавиш, применяемые при работе с программой **dscreen**.

Например, это необходимо, если стандартные клавиши, определенные для команды **dscreen**, конфликтуют с управляющими клавишами работающего приложения.

Предполагается, что типы терминалов, задаваемые в файле `dsinfo`, поддерживают только одностраничную память. Если терминал поддерживает многостраничную память, то в файле `dsinfo` нужно указать функциональные клавиши для управления страничной памятью. Информацию о конкретных управляющих последовательностях вы можете найти в руководстве по работе с терминалом.

По умолчанию в качестве файла `dsinfo` применяется файл `/usr/lbin/tty/dsinfo`. Для того чтобы задать другой файл `dsinfo`, укажите флаг **-i**. Далее в этом разделе будет описываться работа с файлом по умолчанию. Однако вся приведенная ниже информация также относится и к любому другому файлу `dsinfo`.

Дополнительная информация о файле `dsinfo` приведена в разделе “Функциональные клавиши динамического выбора окна” на стр. 651.

Присваивание действий клавишам dscreen

При запуске команды **dscreen** появляется виртуальное окно. При нажатии некоторых клавиш клавиатуры сигнал не передается виртуальному окну, а перехватывается программой **dscreen**, которая выполняет некоторые действия.

Вот несколько примеров этих действий:

Элемент	Описание
Клавиша выбора (обратитесь к разделу “Клавиша выбора dscreen”)	Выбирает указанное окно.
Клавиша блокировки (обратитесь к разделу “Клавиши блокировки dscreen”)	Блокирует ввод и вывод.
Клавиша нового окна (обратитесь к разделу “Клавиша нового окна dscreen”)	Открывает новое окно.
Клавиша завершения (обратитесь к разделу “Клавиши завершения и выхода dscreen” на стр. 651)	Завершает работу **dscreen**.
Клавиша выхода (обратитесь к разделу “Клавиши завершения и выхода dscreen” на стр. 651)	Закрывает окно **dscreen**.
Клавиша предыдущего окна (обратитесь к разделу “Клавиша предыдущего окна dscreen” на стр. 651)	Переходит к предыдущему окну.
Клавиша списка (обратитесь к разделу “Клавиша списка dscreen” на стр. 651)	Показывает список функциональных клавиш **dscreen** и выполняемых ими действий.

Набор функциональных клавиш зависит от типа терминала и его описания, заданного в файле `/usr/lbin/tty/dsinfo`.

Клавиша выбора dscreen:

При создании нового виртуального окна ему присваивается клавиша выбора.

При нажатии клавиши выбора выполняются следующие действия:

- Переход от физического терминала к странице, связанной с данным виртуальным окном.
- Создание каналов ввода/вывода между физическим терминалом и виртуальным окном.

Допустимое число виртуальных окон ограничено числом клавиш выбора, определенных в файле `dsinfo`. Сеанс для каждого окна завершается одновременно с завершением работы процесса исходной оболочки. При этом связанная с окном клавиша выбора освобождается и может быть присвоена другому виртуальному окну. Работа утилиты **dscreen** завершается, если нет ни одного активного окна.

Клавиши блокировки dscreen:

Клавиши блокировки применяются для приостановки вывода, их действие аналогично действию клавиш Ctrl-S управления потоком IXON.

Эти клавиши предназначены для прозрачной настройки сеансов терминалов двух компьютеров, взаимодействующих с помощью терминала с двумя последовательными портами.

Клавиша нового окна dscreen:

При нажатии клавиши нового окна создается новое логическое окно, которому присваивается одна из клавиш выбора.

Для каждого нового окна требуется:

- Клавиша выбора, определенная в файле dsinfo
- Псевдотерминал **dscreen**
- Объем памяти, достаточный для хранения структур, которые применяются для отслеживания сеанса окна
- Процесс для запуска оболочки.

Если какой-либо из перечисленных выше ресурсов недоступен, новое окно не создается, и отправляется сообщение с указанием причины ошибки.

Клавиши завершения и выхода dscreen:

При нажатии клавиш завершения и выхода выполняется последовательность действий.

При нажатии клавиши завершения будут выполнены следующие действия:
- Всем сеансам, связанным с данным окном, отправляется сигнал **SIGHUP**.
- Выполняется очистка
- Выполняется процедура выхода с кодом возврата 0.

При нажатии клавиши выхода будут выполнены те же действия, но код возврата будет равен 1.

Клавиша предыдущего окна dscreen:

Клавиша предыдущего окна предназначена для перехода к окну, которое просматривалось последним.

Примечание:
1. Во время ввода информации в текущем окне переключаться между окнами нельзя; escape-последовательность может быть усечена, что приведет к переходу терминала в неопределенное состояние.
2. Некоторые терминалы сохраняют положение курсора в окне, но не сохраняют другие параметры состояния окна, например, режим вставки, инверсного изображения и т.д. Использовать такие режимы при переключении окон не рекомендуется.

Клавиша списка dscreen:

Клавиша списка предназначена для просмотра списка функциональных клавиш терминала и связанных с ними действий.

В списке будут показаны только те клавиши, которые распознаются программой **dscreen**. При создании нового окна командой **dscreen** появляется сообщение `Нажмите клавишу для получения справки`, где вместо слова `клавиша` указывается имя клавиши списка. Обратите внимание, что это сообщение появляется *только* в том случае, если клавиша списка определена.

Функциональные клавиши динамического выбора окна

Число клавиш выбора окна, задаваемое в записи определения терминала в файле `/usr/lbin/tty/dsinfo`, совпадает с числом страниц терминала. Если будет задано большее число клавиш выбора, то программа **dscreen** будет динамически выделять виртуальным окнам страницы физического окна.

При выборе виртуального окна, с которым не связана никакая страница физического окна, **dscreen** выделяет виртуальному окну страницу, которая использовалась позже всех. В зависимости от спецификаций, указанных в файле описания `/usr/lbin/tty/dsinfo`, при подключении физического окна к другому виртуальному окну могут выполняться некоторые действия, например, очистка экрана.

Файл dsinfo

Файл `dsinfo` представляет собой базу данных описаний терминалов, которая используется утилитой **dscreen**.

В файле содержится следующая информация:
- Список функциональных клавиш утилиты **dscreen** и выполняемых ими действий
- Число страниц памяти терминала
- Последовательности кодов, отправляемые или принимаемые при нажатии функциональных клавиш

Описания типов терминалов в файле `dsinfo` по умолчанию аналогичны следующим записям для терминала ASCII 3151:

```
# Для данной записи необходима кассета расширения (pn: 64F9314)
ibm3151|3151|IBM 3151,
dsks=\E!a^M|Shift-F1|,            # Выбор первого окна
dsks=\E!b^M|Shift-F2|,            # Выбор второго окна
dsks=\E!c^M|Shift-F3|,            # Выбор третьего окна
dsks=\E!d^M|Shift-F4|,            # Выбор четвертого окна
dskc=\E!e^M|Shift-F5|,            # Создание нового окна
dske=\E!f^M|Shift-F6|\E pA\EH\EJ, # Переход к окну 1 и завершение работы
dskl=\E!g^M|Shift-F7|,            # Просмотр списка функциональных клавиш
(справка)
dskp=\E!h^M|Shift-F8|,            # Переход к предыдущему окну
dskq=\E!i^M|Shift-F9|\E pA\EH\EJ, # Переход к окну 1 и выход из программы
dsp=\E pA|\EH\EJ,                 # Терминальная последовательность для окна 1
dsp=\E pB|\EH\EJ,                 # Терминальная последовательность для окна 2
dsp=\E pC|\EH\EJ,                 # Терминальная последовательность для окна 3
dsp=\E pD|\EH\EJ,                 # Терминальная последовательность для окна 4
dst=10,                           # Разрешить буферизацию в течение 1-секундного тайм-аута
```

Формат записей файла dsinfo:

Записи файла `dsinfo` состоят из полей, отделенных друг от друга запятыми.

В первом поле задается список альтернативных имен терминала, которые отделяются друг от друга символом конвейера (|). Текст, указанный после символа #, рассматривается как комментарий и игнорируется программой **dscreen**. В остальных полях содержится описание функций терминала, применяемое утилитой **dscreen**. В этих строках могут быть указаны следующие escape-последовательности:

Таблица 108. поля файла dsinfo

Escape-последовательность	Описание
\E,\e	escape-символ
\n,\l	символ перехода на новою строку (или переноса строки)
\r	возврат каретки
\t	символ табуляции
\b	символ возврата
\f	символ перевода страницы
\s	символ пробела
nnn	символ с восьмеричным значением *nnn*
^*x*	Ctrl-X для любого допустимого символа *x*.

Все остальные символы, перед которыми указана обратная косая черта, будут интерпретироваться как сами символы. Строки описания задаются в формате *тип=строка*, где *тип* - один из перечисленных ниже типов, а *строка* - строковое значение.

Записи в файле `dsinfo` обязательно должны отделяться друг от друга запятыми. Если в конце записи файла `dsinfo` запятая будет пропущена или усечена, то этот файл станет нечитаемым для утилиты **dscreen**, и на экран будет выдано сообщение об ошибке.

Типы строк disinfo:

Здесь приведены сведения о типах строк disinfo.

Существуют следующие типы строк:

Элемент **Описание**

dskx Описание клавиши. Имя типа должно содержать четыре символа, последний из которых (*x*) указывает действие, которое будет выполняться при нажатии клавиши. Существуют следующие типы клавиш:

Тип	Действие
dsks	Переключение между окнами
dskb	Блокировка ввода и вывода
dske	Завершение работы **dscreen**
dskq	Выход из программы **dscreen** (код возврата=1)
dskc	Создать новое окно
dskp	Перейти к предыдущему окну
dskl	Просмотреть список клавиш и связанных с ними действий

Все остальные типы клавиш (вида dsk*x*, где символ x отличен от s, b, e, q, p и l) не связаны ни с какими внутренними действиями программы **dscreen**, однако они будут показаны в списке клавиш и будут распознаваться этой программой. Тип dskn (n означает отсутствие операции) применяется в том случае, если программа **dscreen** не должна выполнять никаких внутренних действий.

Строковое значение каждой клавиши представляет собой три подстроки, разделенных символом конвейера (|). **Примечание:** Вместо символа | в подстроке нужно указать символы \|.

Первая подстрока - это последовательность символов, которую терминал отправляет при нажатии клавиши. Вторая подстрока - это название клавиши, которое указывается при просмотре списка клавиш. Третья подстрока - это последовательность символов, которая отправляется терминалу программой **dscreen** при нажатии клавиши перед выполнением действия, связанного с этой клавишей.

dsp Тип dsp предназначен для описания физического окна терминала. Для каждого физического окна терминала должна быть задана одна строка dsp. Строковое значение для каждого физического окна состоит из двух подстрок, разделенных символом конвейера (|).

Первая подстрока - это последовательность символов, которая предназначена для вывода на физическую страницу терминала.

Вторая подстрока отправляется терминалу при выводе на страницу новой информации. Часто она применяется для очистки содержимого окна. Эта подстрока отправляется, если выполнены следующие два условия:

1. Создается новый сеанс виртуального терминала.
2. Число виртуальных терминалов больше числа физических страниц. Если при выборе виртуального терминала программа **dscreen** выделяет ему физическое окно, уже связанное с другим виртуальным терминалом, то она отправляет эту последовательность символов, означающую, что содержимое окна не соответствует выводу виртуального терминала.

Примечание: Не рекомендуется, чтобы число применяемых виртуальных терминалов превышало число физических окон. Для того чтобы этого избежать, число клавиш выбора окна (dsks=) в записи dsinfo не должно превышать числа физических окон (dsp=).

dst A Строка типа dst предназначена для настройки тайм-аута ввода для **dscreen**. В качестве строкового значения указывается десятичное число. Значение тайм-аута задается в десятых долях секунды и не может превышать 255 (значение по умолчанию**=1** [или 0.1 секунды]).

Если программа **dscreen** распознала префикс входной последовательности символов, соответствующий функциональной клавише, то она ожидает получения всех символов последовательности. Если тайм-аут истек до того, как были получены все символы, то символы отправляются виртуальному окну, и программа **dscreen** не рассматривает эти символы как последовательность символов, соответствующую функциональной клавише.

Рекомендуется увеличить значение тайм-аута, если некоторые функциональные клавиши утилиты **dscreen** фактически представляют собой сочетание нескольких клавиш (например, выбор окна с помощь клавиш Ctrl-Z 1, Ctrl-Z 2, Ctrl-Z 3, создание окна с помощью клавиш Ctrl-Z N и т.д.).

Примеры dysinfo:

Следующие примеры `dysinfo` относятся к Wyse-60 для трех сеансов окон.

```
wy60|wyse60|wyse model 60,
dsks=^A^^M|Shift-F1|,
dsks=^Aa^M|Shift-F2|,
```

```
dsks=^Ab^M|Shift-F3|,
dskc=\200|Ctrl-F1|,
dske=\201|Ctrl-F2|\Ew0\E+,
dskl=\202|Ctrl-F3|,
dsp=\Ew0|\E+,
dsp=\Ew1|\E+,
dsp=\Ew2|\E+,
```

В этой записи задана следующая конфигурация:

- Сочетания клавиш Shift-F1, Shift-F2, Shift-F3 применяются для выбора окон 1, 2, 3.
- Сочетание клавиш Ctrl-F1 предназначено для создания нового окна.
- Сочетание клавиш Ctrl-F2 передает на экран последовательность символов `Esc w 0 Esc +` (переключиться в окно 0 и очистить экран), после чего завершает работу **dscreen**.
- Клавиши Ctrl-F3 предназначены для просмотра списка функциональных клавиш и связанных с ними действий.

Каждый раз, когда с новым окном связывается определенное физическое окно, терминалу отправляется последовательность `Esc +`, в результате чего экран очищается.

Это пример для Wyse-60 с тремя оконными сеансами, но одно из окон расположено на втором компьютере, который обменивается данными с терминалом через второй последовательный порт:

```
wy60-1|wyse60-1|wyse model 60 - первый последовательный порт
dsks=^A~^M|Shift-F1|,
dsks=^Aa^M|Shift-F2|,
dsks=^Ab^M|Shift-F3|\Ed#^Ab\r^T\Ee9,
dskc=\200|Ctrl-F1|,
dske=\201|Ctrl-F2|\Ed#\201^T\Ew0\E+,
dskl=\202|Ctrl-F3|,
dsp=\Ew0|\E+,dsp=\Ew1|\E+,
wy60-2|wyse60-2|wyse model 60 - второй последовательный порт
dsks=^A~^M|Shift-F1|\Ed#^A~\r^T\Ee8,
dsks=^Aa^M|Shift-F2|\Ed#^Aa\r^T\Ee8,
dsks=^Ab^M|Shift-F3|,
dskc=\200|Ctrl-F1|,
dske=\201|Ctrl-F2|\Ed#\201^T\Ew0\E+,
dskl=\202|Ctrl-F3|,
dsp=\Ew2|\E+,
```

Программа **dscreen** должна быть запущена на обоих компьютерах, причем на первом в качестве типа терминала должно быть указано значение wy60-1, а на втором - значение wy60-2 (с помощью опции **-t** команды **dscreen**). Первой будет просматриваться запись wy60-1.

Первые два описания функциональных клавиш не отличаются от исходной записи wy60. Однако для третьей клавиши задан тип dskb, то есть это клавиша блокировки ввода и вывода. При нажатии этой клавиши на терминал будет отправлена последовательность:

```
Esc d # Ctrl-A b CR Ctrl-T Esc e 9
```

После этого весь вывод блокируется, а программа **dscreen** просматривает ввод, выполняя поиск последовательности символов, соответствующей функциональной клавише. Все остальные данные игнорируются.

Последовательность `Esc d #` переводит терминал в режим прозрачной печати, в котором все символы передаются через второй последовательный порт, пока не встретится последовательность Ctrl-T.

Через второй последовательный порт отправляется последовательность символов `Ctrl-A b CR`, получив которую процесс **dscreen** на втором компьютере активизирует окно, связанное с клавишей Shift-F3.

Клавиши Ctrl-T предназначены для отмены режима прозрачной печати. Клавиши Esc 9 предназначены для изменения последовательного порта AIX, через который терминал обменивается данными.

После того как второй компьютер получает такую последовательность символов, он отправляет последовательность `Esc w 2` для перехода к третьему физическому окну, а затем возобновляет обычную работу.

В записи wy60-2 для клавиш Shift-F1 и Shift-F2 установлена следующая последовательность действий:

- Включить режим прямой печати
- Отправить на другой компьютер последовательность символов, соответствующую функциональной клавише
- Включить режим прямой печати
- Переключиться на другой последовательный порт

Клавиша завершения работы, Ctrl-F2, выполняет одинаковые действия на обоих компьютерах: с помощью механизма прозрачной печати отправляет на другой компьютер последовательность символов, соответствующих клавише завершения, активизирует окно 0, очищает экран, а затем завершает работу программы.

Среда общего интерфейса управления передачей данных

Общий интерфейс управления передачей данных (GDLC) - это шаблон определения интерфейса, который предоставляет пользователям общий набор команд для управления диспетчерами устройств DLC в составе операционной системы.

Для определения неполадок обратитесь к разделу Определение неполадок книги *Communications Programming Concepts*.

Общий интерфейс управления передачей данных (GDLC) - это шаблон определения интерфейса, который предоставляет пользователям общий набор команд для управления диспетчерами устройств DLC в составе операционной системы.

Интерфейс GDLC задает требования к определениям точек входа, а также функции и структуры данных для всех диспетчеров устройств DLC. С интерфейсом GDLC совместимы следующие типы DLC:

- 8023 (IEEE 802.3 для Ethernet)
- ETHER (Standard Ethernet)
- SDLC (Управление синхронным каналом передачи данных)
- TOKEN (Token-Ring)
- FDDI (Оптоволоконный интерфейс распределенных данных)

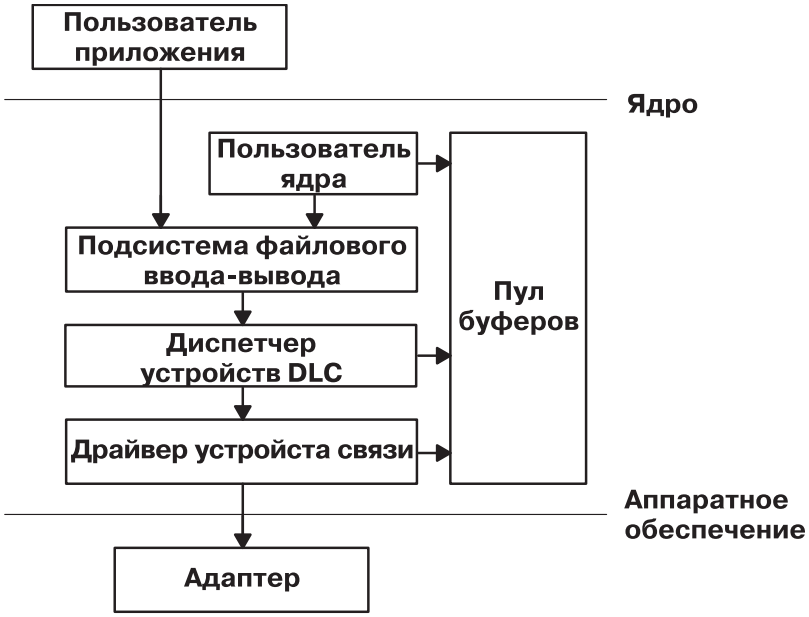
Диспетчеры устройств DLC поддерживают высокоуровневые протоколы и функции, лежащие вне области действия драйвера ядра. Тем не менее, для обеспечения максимальной производительности эти диспетчеры входят в состав ядра и отправляют запросы к адаптеру ввода-вывода через драйвер ядра. Пользователь DLC расположен над ядром или внутри него.

Примерами диспетчеров устройств DLC являются Управление синхронной передачей данных (SDLC) и Управление передачей данных IEEE 802.2. Каждый диспетчер устройства DLC взаимодействует с отдельным драйвером устройства или набором драйверов устройств. Например, SDLC взаимодействует с многопротокольным драйвером устройства для системного продукта и соответствующим адаптером.

Основная структура среды DLC показана на рисунке "Среда диспетчера устройств DLC". Пользователям уровня ядра доступны буферы связи (mbuf), и они могут добавлять точки входа с помощью служб ядра **fp**. Пользователям, находящимся выше уровня ядра, доступен стандартный интерфейс обращения к драйверам

ядра. Файловая система обращается к точкам входа **dd**. Для передачи данных требуется переместить их между пространством пользователя и пространством ядра.

На этом рисунке показано, каким образом пользователь приложения связан с адаптером (аппаратный



Среда диспетчера устройств DLC

Рисунок 42. Среда диспетчера устройств DLC

уровень). Промежуточными уровнями служат пользователь ядра, подсистема ввода-вывода файлов, диспетчер устройств DLC, драйвер устройств ввода-вывода и пул буферов. Эти объекты относятся к уровню ядра.

Набор компонентов среды диспетчера устройств DLC включает:

Элемент	Описание
Пользователь уровня приложений	Находится выше уровня ядра, например, приложение или способ доступа.
Пользователь уровня ядра	Находится на уровне ядра, например, как процесс или диспетчер устройства уровня ядра.
Файловая подсистема ввода-вывода	Преобразует подпрограммы дескриптора файла и указателя в записи таблицы точек входа для доступа к указателю на файл.
Пул буферов	Предоставляет подсистеме связи средства буферизации данных.
Драйвер устройства ввода-вывода	Управляет регистрами ввода-вывода аппаратного адаптера и прямого доступа к памяти (DMA), а также направляет принятые пакеты различным элементам DLC.
Адаптер	Предназначен для подключения средств связи.

Диспетчер устройства, созданный в соответствии со спецификациями GDLC, может выполняться в любой аппаратной конфигурации операционной системы при наличии драйвера устройства связи и целевого адаптера. Каждый диспетчер устройства поддерживает работу с несколькими пользователями на высшем уровне и с несколькими драйверами устройств и адаптерами на нижнем уровне. Как правило, пользователи параллельно используют один адаптер, или каждый пользователь работает с несколькими адаптерами. В зависимости от ограничений протокола, используются различные диспетчеры устройств DLC.

На рис. 43 на стр. 657 показана конфигурация с несколькими пользователями:

На этом рисунке показан уровень ядра, расположенный между пользователем приложения и адаптером.

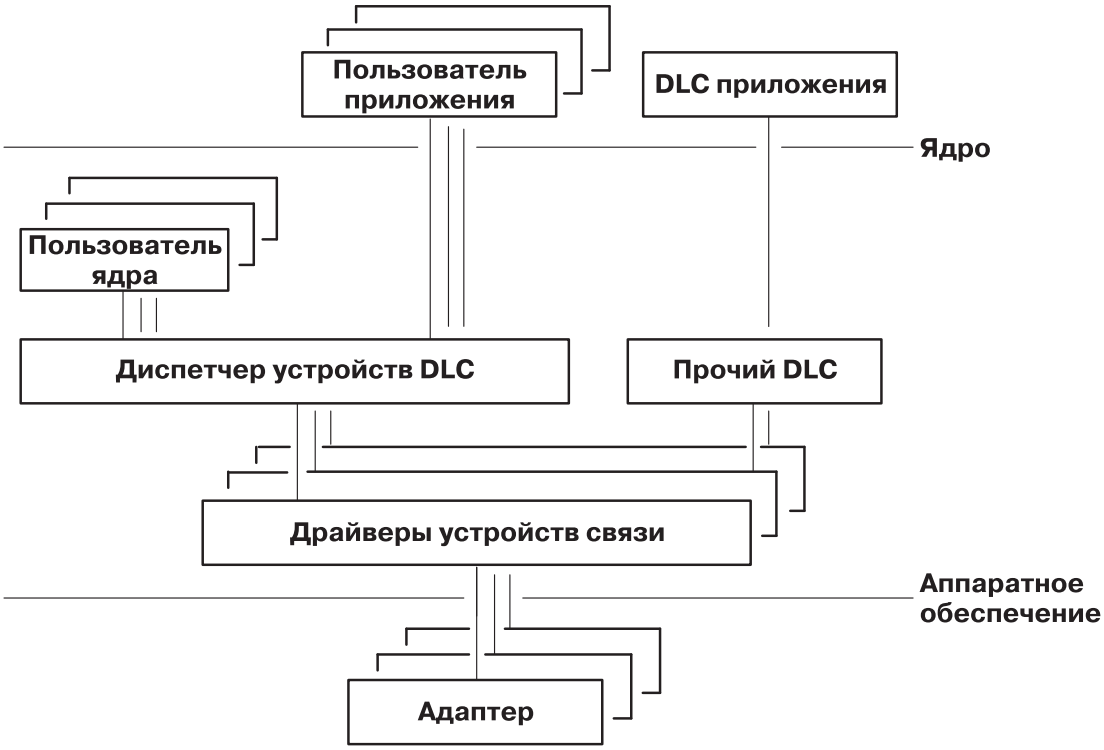


Рисунок 43. Конфигурация с несколькими пользователями и несколькими адаптерами

Нескольким пользователям соответствует несколько объектов.

Критерии GDLC

Критерии которым должен отвечать интерфейс GDLC.

- Гибкость и доступность как для пользователей уровня приложений, так и для пользователей уровня ядра.
- Поддержка нескольких пользователей и нескольких адаптеров, позволяющая протоколам использовать все преимущества работы с несколькими сеансами и несколькими портами.
- Поддержка всех служб, как с установлением соединения, так и без установления соединения.
- Прозрачная передача данных, не обусловленная работой диспетчера устройств DLC.

Интерфейс GDLC

Каждый диспетчер устройства DLC - это стандартная запись в каталоге `/dev`. Он работает в составе ядра как диспетчер мультиплексорного устройства для определенного протокола.

Для адаптера, не используемого DLC, каждая подпрограмма **open** диспетчера устройства DLC создает процесс уровня ядра. Кроме того, подпрограмма **open** вызывается и для обработчиков целевого адаптера. При необходимости вызовите подпрограмму **open** для нескольких портов адаптера DLC, связанных с одним протоколом. Все повторные вызовы подпрограммы **open** для порта не создают дополнительных процессов ядра, а подключаются к процессу, созданному при первом вызове **open**. Для каждого используемого порта существует один процесс уровня ядра.

Внутренняя структура диспетчера устройств DLC совпадает с базовой структурой программы работы с устройством за исключением того, что процесс уровня ядра заменяет программу обработки прерываний в асинхронных событиях. Функции чтения, записи, управления вводом-выводом и выбора работают так, как показано на рисунке "Стандартный диспетчер устройств ядра".

На этом рисунке показана внутренняя структура диспетчера устройств DLC. Диспетчер состоит из

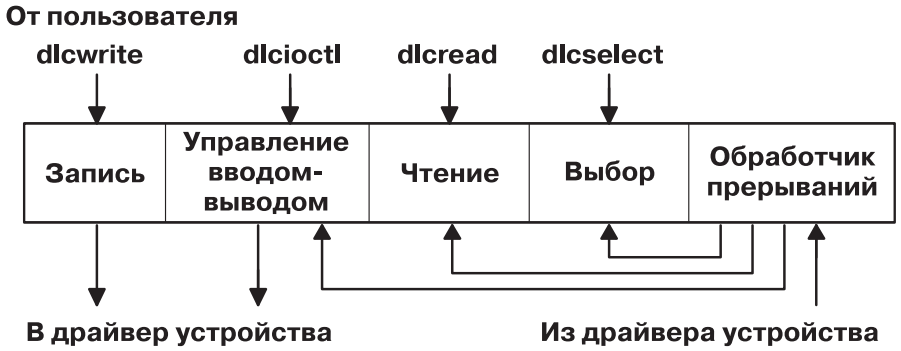


Рисунок 44. Стандартный диспетчер устройств ядра

обработчиков записи, управления вводом-выводом, чтения, выбора и прерываний. Информация, полученная от пользователя, проходит в диспетчере устройств несколько уровней, прежде чем она передается обработчику устройства.

Управление передачей данных GDLC

Элементы DLC можно устанавливать по отдельности или вместе. Для каждого установленного типа DLC диспетчер устройства DLC автоматически добавляется к ядру в состоянии "Доступен".

Проверить установку можно с помощью команды **lslpp**:

```
lslpp -h dlctype
```

где *dlctype* может принимать следующие значения:

Элемент	Описание
bos.dlc.8023	Управление передачей данных IEEE Ethernet (802.3)
bos.dlc.ether	Управление передачей данных - Стандартный Ethernet
bos.dlc.fddi	Управление передачей данных FDDI
bos.dlc.sdlc	Управление передачей данных SDLC
bos.dlc.token	Управление передачей данных Token-Ring

Сведения об установленном элементе DLC можно просмотреть с помощью Инструмента управления системой (SMIT) или из командной строки. Если система или порты связи сильно загружены, рекомендуется изменить атрибуты DLC для повышения производительности. Если обнаружено низкое быстродействие при приеме, и системный протокол ошибок содержит записи о переполнении очереди кольца при взаимодействии DLC и программы работы с устройством, увеличьте размер очереди приема DLC. Кроме того, если вы не планируете использовать элемент DLC в течение продолжительного периода времени, его следует удалить из ядра. Удаление DLC из ядра не приводит к его удалению из конфигурации системы, но освобождает ресурсы ядра для других задач на то время, пока DLC не нужен. Инструкции по выполнению этих задач приведены в разделе “Управление драйвером устройства DLC” на стр. 662.

Операции точки входа ioctl интерфейса GDLC

Интерфейс GDLC поддерживает следующие операции управления вводом-выводом (**ioctl**).

Элемент	Описание
DLC_ENABLE_SAP	Активизирует служебную точку доступа (SAP).
DLC_DISABLE_SAP	Деактивирует SAP.
DLC_START_LS	Запускает станцию связи для указанной SAP в режиме инициатора или приемника.
DLC_HALT_LS	Останавливает станцию связи.
DLC_TRACE	Отслеживает кратковременные и долговременные операции станции связи.
DLC_CONTACT	Устанавливает соединение с удаленной станцией для определенной локальной станции связи.
DLC_TEST	Проверяет соединение выбранной локальной станции связи с удаленной станцией.
DLC_ALTER	Изменяет параметры конфигурации станции связи.
DLC_QUERY_SAP	Запрашивает статистику по определенной SAP.
DLC_QUERY_LS	Запрашивает статистику по определенной станции связи.
DLC_ENTER_LBUSY	Переводит выбранную линию связи в режим local-busy.
DLC_EXIT_LBUSY	Выключает режим local-busy для выбранной станции связи.
DLC_ENTER_SHOLD	Переводит выбранную линию связи в режим short-hold.
DLC_EXIT_SHOLD	Выключает режим short-hold для выбранной станции связи.
DLC_GET_EXCEP	Возвращает пользователю уровня приложения асинхронное уведомление об исключительной ситуации. **Примечание:** Данная операция **ioctl** не применяется пользователями уровня ядра, так как все сообщения об исключительных ситуациях передаются пользователю уровня ядра через соответствующие программы обработки исключительных ситуаций.
DLC_ADD_GRP	Добавляет к порту групповой адрес приема или адрес приема многоцелевой рассылки.
DLC_DEL_GRP	Удаляет групповой адрес приема или адрес приема многоцелевой рассылки, связанный с портом.
DLC_ADD_FUNC_ADDR	Добавляет к порту групповой функциональный адрес приема или функциональный адрес приема многоцелевой рассылки.
DLC_DEL_FUNC_ADDR	Удаляет из порта групповой функциональный адрес приема или функциональный адрес приема многоцелевой рассылки.
IOCINFO	Возвращает структуру, описывающую диспетчер устройств GDLC. Дополнительная информация приведена в файле `/usr/include/sys/devinfo.h`.

Служебная точка доступа GDLC

Служебная точка доступа (SAP) определяет пользовательскую службу, предназначенную для отправки и приема конкретного класса данных.

При этом возможна раздельная пересылка данных различных классов соответствующим служебным обработчикам. Адреса исходной SAP и целевой SAP для DLC, параллельно поддерживающих несколько SAP, хранятся в заголовках пакетов. В DLC, поддерживающих единственную SAP, адресация SAP не используется. Тем не менее, принцип включения используется и для одной SAP. В общем случае, для каждого порта и каждого пользователя DLC существует включенная SAP.

Большинство адресов SAP соответствует стандартным объектам управления сетью IEEE или присваивается пользователям в соответствии с документом *Token-Ring Network Architecture Reference*. Ниже приведено несколько основных адресов SAP:

Элемент	Описание
Нулевая SAP (0x00)	Позволяет отвечать на запросы удаленных узлов, даже если ни одна SAP не была включена. Эта SAP поддерживает только службы без подтверждения связи и отвечает только на запросы обмена идентификаторами (XID) и тестирования блоков данных протокола связи (LPDU).
Управление путем SNA (0x04)	Представляет адрес SAP по умолчанию, который используется в узлах Системной сетевой архитектуры (SNA).
PC Network NETBIOS (0xF0)	Используется в средствах связи DLC, управляемых с помощью эмуляции Сетевой базовой системы ввода-вывода (NetBIOS).
SAP для распространения информации (0xFC)	Используется службами распространения информации об именах локальной сети (LAN).
Глобальная SAP (0xFF)	Определяет все активные SAP.

Станции связи GDLC

Станция связи (LS) определяет соединение между двумя узлами определенной пары SAP.

Это соединение может работать как служба без подтверждения связи (передача дейтаграмм) или как служба с подтверждением связи (передача полностью упорядоченных данных с восстановлением после ошибок). В общем случае, для каждого удаленного соединения запускается одна LS.

Режим Local-Busy GDLC

Если LS работает в режиме с подтверждением связи, то необходимо, чтобы удаленная станция прекратила отправку информационных пакетов из-за таких ошибок, как, например, нехватка ресурсов. Для перевода локальной станции в режим local-busy удаленной станции может быть отправлено соответствующее уведомление.

После освобождения ресурсов локальная станция уведомляет удаленную станцию о выходе из состояния local-busy и о том, что можно возобновить отправку информационных пакетов. В режиме local-busy приостанавливается работа только с упорядоченными информационными пакетами. На прохождение пакетов других типов этот режим не влияет.

Режим Short-Hold GDLC

Режим short-hold используется при работе с определенными типами сетей.

Режим short-hold используется в сетях передачи данных со следующими определенными параметрами:

- Быстрое установление соединения
- Система оплаты, при которой плата за установление соединения относительно невелика, по сравнению с оплатой времени соединения.

В режиме short-hold связь между двумя станциями устанавливается только при наличии данных для передачи. Если данные для отправки отсутствуют, то после заданного тайм-аута связь прерывается, и повторно устанавливается только при появлении данных для пересылки.

Проверка и трассировка соединений GDLC

Для проверки соединения между двумя станциями отправьте с локальной станции тестовый пакет. Если соединение исправно, этот пакет будет возвращен удаленной станцией.

Поддержка этой функции ограничивается возможностями протокола некоторых линий связи. Например, SDLC позволяет создавать тестовые пакеты только на главной системе или основной станции. Большинство других протоколов позволяют отправлять тестовые пакеты с любой станции, участвующей во взаимодействии.

Для трассировки канала связи, передаваемых данных и специальных событий (таких как активация станции, завершение ее работы и тайм-ауты) создайте канал трассировки общего назначения и настройте этот канал на всех станциях связи в качестве целевого канала для записи протоколов трассировки. Эта функция позволяет определить причину некоторых неполадок со связью. Поддерживаются как короткие, так и длинные записи трассировки.

Статистика GDLC

Пользователь GDLC может запросить статистику как по SAP, так и по LS.

Статистика по SAP состоит из сведений о текущем состоянии SAP и о программе работы с устройством. Статистика по LS состоит из сведений о текущем состоянии станции, а также показаний различных индикаторов надежности, доступности и возможности обслуживания, которые отслеживают работу станции с момента ее запуска.

Специальные службы ядра GDLC

Общий интерфейс управления передачей данных (GDLC) предоставляет пользователю уровня ядра специальные возможности.

Тем не менее, внутри ядра должна существовать надежная среда. Вместо того чтобы диспетчер устройств DLC копировал данные асинхронных событий в пользовательское пространство, пользователь уровня ядра должен задать указатели на специальные подпрограммы, называемые функциями-обработчиками. Функции-обработчики вызываются DLC во время выполнения. Этим обеспечивается максимальная производительность взаимодействия пользователя уровня ядра и слоев DLC. Каждый пользователь уровня ядра должен ограничить число функций-обработчиков минимальной длиной пути и использовать схему с буфером связи (mbuf).

Функция-обработчик не может обращаться непосредственно к другой точке входа DLC, так как при непосредственном обращении происходит блокировка, приводящая к полному останову. Единственным исключением из этого правила является ситуация, когда пользователь уровня ядра обращается к точке входа **dlcwritex** во время обслуживания одной из четырех функций приема данных. Обращение к точке входа **dlcwritex** дает возможность получить ответ немедленно, без дополнительного переключения заданий. Когда пользователь вызывает операцию записи, проверка идентификатора процесса в диспетчере устройств DLC требует специальной логики. Если это процесс DLC, и внутренняя очередь DLC переполнена, то в ответ на запрос о записи возвращается код ошибки (значение **EAGAIN**). При этом процесс DLC не переводится в состояние ожидания. Затем функция приема данных пользовательской подпрограммы может возвратить DLC специальное уведомление о повторении попытки приема данных буфера.

Предусмотрены следующие пользовательские функции-обработчики:

Элемент	Описание
Подпрограмма Данные дейтаграммы приняты	Вызывается при приеме пакета дейтаграмм для пользователя уровня ядра.
Подпрограмма Исключительная ситуация	Вызывается при возникновении асинхронного события для уведомления пользователя, например: SAP закрыта или Связь со станцией установлена.
Подпрограмма Принят информационный кадр	Вызывается при получении обычного пакета упорядоченных данных для пользователя уровня ядра.
Подпрограмма Приняты сетевые данные	Вызывается при приеме данных со специальной информацией о сети для пользователя уровня ядра.
Подпрограмма Приняты данные XID	Вызывается при приеме пакета идентификации обмена (XID) для пользователя уровня ядра.

Пользователь уровня ядра не может обратиться к точкам входа **dlcread** и **dlcselect** для DLC, так как обращение к асинхронным функциональным точкам входа производится непосредственно из диспетчера устройств DLC. Как правило, постановка этих событий в очередь производится пользовательской функцией-обработчиком. Но если пользователь уровня ядра не может обработать принятый пакет, то диспетчер устройств DLC может заблокировать буфер приема и перейти в один из следующих режимов класса user-busy:

Режим Занято с пользовательским сбросом (только для информационных кадров)

Если пользователь уровня ядра не может обработать принятый информационный кадр (например, из-за блокировки очереди), то возвращается код DLC_FUNC_BUSY, и DLC блокирует указатель на буфер и переходит в режим local-busy для прекращения передачи информационных кадров удаленной станцией. Для того чтобы сбросить режим local-busy и возобновить прием информационных кадров, пользователь уровня ядра должен вызвать функцию Выйти из режима local-busy. Можно прекратить передачу только обычных упорядоченных информационных кадров. Режим local-busy не оказывает влияния на данные XID, дейтаграммы и сетевые данные.

Режим Занято со сбросом по таймеру (для всех типов кадров)

Если пользователь уровня ядра не может обработать принятый пакет, и необходимо, чтобы DLC заблокировал буфер приема на короткий промежуток времени, а затем повторил вызов пользовательской функции приема, то DLC возвращается код DLC_FUNC_RETRY. Если принятый пакет - это упорядоченный информационный кадр, то на этот промежуток времени станция переходит в режим local-busy. В любом случае запускается таймер; по истечении времени работы таймера обращение к функциональной точке входа для приема данных повторяется.

Управление драйвером устройства DLC

Перед началом работы с драйвером DLC его необходимо добавить в систему.

Драйвер DLC автоматически добавляется в конфигурацию после установки, а также при каждом перезапуске системы (обратитесь к разделу “Управление передачей данных GDLC” на стр. 658). Если драйвер DLC был удален без последующей перезагрузки, его можно добавить повторно.

Таблица 109. Задачи управления драйверами DLC

Процедура	Команды быстрого доступа SMIT	Команда или файл
Добавить установленный элемент DLC	Выберите один (по имени драйвера устройства): `smit cmddlc_sdlc` `smit cmddlc_token` `smit cmddlc_qllc` `smit cmddlc_ether`[1] `smit cmddlc_fddi`, затем выберите **Добавить**	**mkdev**[2]
Изменить атрибуты DLC[3,4]	Выберите один (по имени драйвера устройства): `smit cmddlc_sdlc_ls` `smit cmddlc_token_ls` `smit cmddlc_qllc_ls` `smit cmddlc_ether_ls`[1] `smit cmddlc_fddi_ls`	**chdev**[2]
Начать отслеживающую трассировку локальной сети DLC[5]	`smit trace`	**trace -j** *nnn*, где *nnn* - это ИД соответствующей точки трассировки
Завершить отслеживающую трассировку локальной сети	`smit trcstop`	**trcstop**[2]
Создать отчет об отслеживающей трассировке локальной сети DLC	`smit trcrpt`	**trcrpt -d** *nnn*, где *nnn* - это ИД соответствующей точки трассировки для создания отчета
Показать текущую информацию DLC[3]	Выберите один (по имени драйвера устройства): `smit cmddlc_sdlc_ls` `smit cmddlc_token_ls` `smit cmddlc_qllc_ls` `smit cmddlc_ether_ls`[1] `smit cmddlc_fddi_ls`	**lsdev**[2] или **lsattr**[2]
Удалить DLC[3,6]	Выберите один (по имени драйвера устройства): `smit cmddlc_sdlc_rm` `smit cmddlc_token_rm` `smit cmddlc_qllc_rm` `smit cmddlc_ether_rm`[1] `smit cmddlc_fddi_rm`	**rmdev**[2]

Примечание:

1. Команды быстрого доступа к диспетчеру устройств Ethernet из SMIT позволяют работать с диспетчером устройств как стандартного Ethernet, так и Ethernet IEEE 802.3.
2. Сведения об опциях командной строки приведены в описаниях команд **mkdev**, **chdev**, **trace**, **trcstop**, **trcrpt**, **lsdev**, **lsattr** или **rmdev** в *Справочник по командам, том 4*.
3. Для составления списка, просмотра, изменения и удаления атрибутов драйвера DLC он должен быть установлен в системе и добавлен в конфигурацию (обратитесь к разделу “Управление передачей данных GDLC” на стр. 658). Изменять атрибуты можно только в том случае, если с DLC не связано никаких открытых соединений. Перед вызовом команды изменения необходимо запретить некоторым службам, например SNA, OSI и NetBIOS, использовать DLC.
4. Изменение размера очереди приема напрямую влияет на ресурсы системы. Изменять размер очереди приема следует только в том случае, если при работе DLC возникают неполадки с очередью приема, например пониженная производительность или переполнение при взаимодействии DLC и программы работы с устройством.
5. Отслеживающую трассировку следует применять с осторожностью, так как она напрямую влияет на производительность DLC.
6. Удалять DLC можно только в том случае, если с DLC не связано никаких открытых соединений. Перед вызовом команды удаления необходимо запретить некоторым службам, например SNA, OSI и NetBIOS, использовать DLC.

Справочник по средствам связи и сетевым адаптерам

Ниже приведены различные сценарии работы с адаптерами PCI и с асинхронными адаптерами.

Адаптеры PCI

Сведения об установке и настройке для адаптеров PCI.

Отдельные подразделы посвящены вопросам поддержки и настройки адаптеров PCI для глобальной сети (WAN) (“Драйвер 2-портового многопротокольного сетевого адаптера HDLC” и “Адаптер ARTIC960Hx PCI”).

Драйвер 2-портового многопротокольного сетевого адаптера HDLC

Драйвер двухпортового многопротокольного адаптера для сетей с высокоуровневым управлением передачей данных (HDLC) - это компонент подсистемы связи и ввода-вывода. Он обеспечивает поддержку HDLC с помощью двухпортового многопротокольного адаптера с быстродействием до 1,544 Мбит/с.

Для работы драйвера двухпортового многопротокольного адаптера в сетях HDLC используются следующие компоненты:

- Системная сетевая архитектура (SNA)
- Версия программного интерфейса GDLC Управление синхронной передачей данных (SDLC)
- Пользовательские приложения, совместимые с SDLC MPQP-API (Интерфейс прикладных программ для многопротокольных адаптеров с 4 портами)

Примечание: Для работы с этими компонентами требуется специальный файл `mpc`*n*, с помощью которого можно получить доступ к драйверу двухпортового многопротокольного адаптера HDLC через подсистему эмуляции драйвера SDLC COMIO. Эта подсистема должна быть установлена и настроена для каждого устройства в сети HDLC.

- Пользовательские приложения, совместимые с API общего интерфейса передачи данных (CDLI) HDLC

Драйвер двухпортового многопротокольного адаптера позволяет устанавливать соединение с удаленными хостами через этот адаптер напрямую по выделенной линии, либо через коммутируемую сеть. Драйвер адаптера может работать как шлюз между средой рабочей группы и средствами удаленной обработки данных.

Настройка 2-портового многопротокольного адаптера

Используйте эти сведения для настройки 2-портового многопротокольного адаптера.

Таблица 110. Задачи настройки двухпортового многопротокольного адаптера

Процедура	Команды быстрого доступа SMIT
Добавить драйвер устройства к адаптеру	`smit mkhdlcdpmpdd`
Изменить конфигурацию драйвера устройства для адаптера	`smit chhdlcdpmpdd`
Удалить драйвер устройства из адапетра	`smit rmhdlcdpmpdd`
Сделать доступным заданный драйвер устройства	`smit cfghdlcdpmpdd`
Добавить средство эмуляции SDLC COMIO для адаптера	`smit mksdlcsciedd`
Изменить конфигурацию средства эмуляции SDLC COMIO для адаптера	`smit chsdlcsciedd`
Удалить средство эмуляции SDLC COMIO для адаптера	`smit rmsdlcsciedd`
Сделать доступным заданное средство эмуляции SDLC COMIO	`smit cfgsdlcsciedd`

Адаптер ARTIC960Hx PCI

Средство эмуляции драйвера MPQP COMIO адаптера ARTIC960HX PCI - это компонент подсистемы связи и ввода-вывода. Он обеспечивает поддержку адаптера ARTIC960Hx PCI с быстродействием до 2 Мбит/с.

Необходимо использовать модемы, поддерживающие синхронизацию, так как возможна только внешняя синхронизация.

Работа с драйвером адаптера ARTIC960Hx PCI MPQP COMIO осуществляется с помощью следующих компонентов:

- Системная сетевая архитектура (SNA)
- Программный интерфейс Управление передачей данных по шаблону (GDLC)
- Пользовательские приложения, совместимые с API MPQP-API (многопротокольный четырехпортовый интерфейс прикладных программ), например, приложения SDLC и BiSync.

Для работы с этими компонентами требуется специальный файл `mpqx`, с помощью которого можно получить доступ к адаптеру ARTIC960Hx PCI через драйвер эмуляции MPQP COMIO. Этот драйвер должен быть установлен и настроен для каждого порта адаптера ARTIC960Hx PCI. Специальный файл `mpqx` находится в каталоге `/dev`.

Примечание: Символ *x* в имени `mpqx` задает экземпляр драйвера устройства, например, `mpq0`.

Драйвер эмуляции MPQP COMIO позволяет устанавливать соединение с удаленными хостами через адаптер ARTIC960Hx или напрямую по выделенной линии. Драйвер адаптера может работать как шлюз между средой рабочей группы и средствами удаленной обработки данных.

Настройка драйвера эмуляции MPQP COMIO для адаптера ARTIC960Hx PCI

Используйте эти сведения для настройки драйвера эмуляции MPQP COMIO для адаптера ARTIC960Hx PCI.

Таблица 111. Задачи для настройки драйвера эмуляции MPQP COMIO

Процедура	Команды быстрого доступа SMIT
Добавить драйвер устройства	`smit mktsdd`
Изменить конфигурацию драйвера эмуляции MPQP COMIO	`smit chtsdd`
Удалить драйвер устройства	`smit rmtsdd`
Настроить заданный драйвер устройства	`smit cfgtsdd`
Добавить порт	`smit mktsdports`
Изменить конфигурацию порта эмуляции MPQP COMIO	`smit chtsdports`
Удалить порт	`smit rmtsdports`
Настроить заданный порт	`smit cfgtsdports`
Трассировка драйвера эмуляции MPQP COMIO	`smit trace_link`

Асинхронные адаптеры

В этой таблице перечислены стандартные 8-портовые и 16-портовые асинхронные адаптеры.

В следующей таблице приведена общая информация об этих продуктах:

Таблица 112. Асинхронные адаптеры

Асинхронное соединение	Тип шины	Код продукта или тип системы (модель)	Максимальная скорость передачи данных через порт (Кбит/с)	Особые функции
8-портовый EIA 232	Micro Channel	2930	76,8	Распространенный стандарт
8-портовый EIA 422A	Micro Channel	2940	76,8	Большее расстояние
8-портовый MIL-STD 188	Micro Channel	2950	Изменяема в зависимости от скорости передачи в бодах UART.	MIL-STD 188-114 для цифрового интерфейса несбалансирован- ного напряжения

Таблица 112. Асинхронные адаптеры (продолжение)

Асинхронное соединение	Тип шины	Код продукта или тип системы (модель)	Максимальная скорость передачи данных через порт (Кбит/с)	Особые функции
8-портовый EIA 232	ISA	2931	115,2	Большая эффективность
8-портовый EIA 232	ISA	2932	115,2	Большая эффективность
8-портовый EIA 422	PCI	2943	230	Большая эффективность
16-портовый EIA 232	Micro Channel	2955	76,8	Ориентация на локальные соединения
16-портовый EIA 422A	Micro Channel	2957	76,8	Большее расстояние
-	ISA	2933	-	-
-	PCI	2944	-	-

В следующей таблице приведена подробная информация о продуктах.

Таблица 113. Параметры продуктов с асинхронным подключением

	Встроенные посл. порты	**8-портовый**		**16-порто- вый**	**128-портовый с RAN**	
		MC	**ISA**		**MC**	**ISA**
Число асинхронных портов адаптера	н/д	8	8	16	128	128
Максимальное число адаптеров	н/д	8	7	8	7	7
Максимальное число асинхронных портов	2 или 3	64	56	128	896	896
Число асинхронных портов RAN	н/д	н/д	н/д	н/д	16	16
Максимальное число RAN	н/д	н/д	н/д	н/д	56	56
Максимальная скорость передачи данных (Кбит/с)	Изменяема в зависимости от скорости передачи в бодах UART.	76,8	115,2	76,8	230	230
Способ подключения	встроенное	direct	direct	direct	узел	узел
Поддерживаемые асинхронные электрические интерфейсы	EIA 232	EIA 232 EIA 422A[4]MIL-STD[4]188-114[4]	EIA 232 EIA 422A	EIA 232 EIA 422A	EIA 232 EIA 422	EIA 232 EIA 422
Стандартный разъем	DB25M/ MODU	DB25M	DB25M	DB25M	RJ-45[2]	RJ-45[2]
Возможность использования кабеля DB25	н/д	н/д	н/д	н/д	RJ-45-DB25	RJ-45-DB25
Возможность монтирования в стойке	н/д	н/д	н/д	н/д	да	да
Блок питания	н/д	н/д	н/д	н/д	внешний	внешний
Поддерживаемые сигналы (EIA 232)	TxD RxD RTS CTS DTR DSR DCD RI	TxD RxD RTS RTS CTS DTR DSR DCD RI	TxD RxD RTS CTS DTR DSR DCD RI	TxD RxD RTS[3]-DTR -DCD -	TxD RxD RTS CTS DTR DSR DCD RI	TxD RxD RTS CTS DTR DSR DCD RI

Примечание:

1. Гнездо поддерживает 8-штырьковый RJ-45, 6-штырьковый RJ-11 или 4-штырьковый RJ-11 разъем с возможностью сокращения числа сигналов.
2. RTS повышена (+12 В) в соединении многопортового трансивера кабеля 16-портового интерфейса EIA 232 (FC 2996).
3. Только Micro Channel.

Составить представление о каждом продукте можно по перечню его основных достоинств. Ниже приведены рекомендации по использованию каждого продукта:

Понятия, связанные с данным:

“Терминал” на стр. 587
Терминал - это символьное устройство, обеспечивающее посимвольный ввод и вывод данных.

Информация, связанная с данной:

 Двухпортовый асинхронный адаптер PCI EIA-232: руководство по установке и использованию

8-портовый адаптер Micro Channel

В 8-портовом адаптере Micro Channel доступен разъем Micro Channel для асинхронного устройства ввода-вывода.

Другие функции включают в себя:

- Необходимо менее 8 портов, при этом расширение не требуется или почти не требуется.
- Все локальные терминалы удалены от системы на расстояние не свыше 61 метра.
- Требуются удаленные терминалы (поддержка через мультиплексор/модем OEM).
- Требуется невысокая пропускная способность (до 76,8 Кбит/с).

Шина ISA 8-портового адаптера EIA 232 или EIA 232/EIA 422

Шина ISA 8-портового адаптера EIA 232 или EIA 232/EIA 422 содержит разъем ISA.

Другие функции включают в себя:

- Необходимо менее 8 портов, при этом расширение не требуется или почти не требуется.
- Необходимы все порты EIA 232, EIA 422 или сочетание портов EIA 232 или EIA 422.
- Снимает с центрального процессора процессы обработки ввода-вывода терминала и символьные прерывания.
- Асинхронная скорость до 115,2 Кбит/с.
- Максимальное быстродействие для высокоскоростных (28,8 Кбит/с) модемов со сжатием данных.

16-портовый адаптер Micro Channel

В 16-портовом адаптере Micro Channel доступен разъем Micro Channel для асинхронного устройства ввода-вывода.

Другие функции включают в себя:

- 8 портов сейчас, менее 16 портов с незначительным расширением или вообще без него.
- Все локальные терминалы удалены от системы на расстояние не свыше 61 метра.
- Требуются удаленные терминалы (поддержка через мультиплексор/модем OEM).
- Устройствам не требуется использовать все сигналы EIA 232.
- Требуется невысокая пропускная способность (до 38,4 Кбит/с для асинхронных устройств).

128-портовый адаптер (Micro Channel, ISA)

К функциям 128-портового адаптера Micro Channel или ISA относятся 16 портов с возможностью расширения до 128 портов без использования дополнительных разъемов.

Другие функции включают в себя:

- Имеется разъем Micro Channel, ISA или PCI для асинхронного ввода-вывода. (Дополнительная информация о PCI приведена в “Замечания по выбору продукта” на стр. 577.)
- Максимальное удаление терминала от системы - 300 м при максимальной скорости обмена данными для адаптеров Micro Channel и ISA.
- Расположение терминалов: на малом или большом расстоянии в том же здании или удаленное.
- Высокая пропускная способность асинхронной связи, низкая нагрузка на процессор.
- Возможность подключения принтера к терминалу.
- Возможность подключения к удаленным системам через оптоволоконные или синхронные модемы.

Просмотр заданных 128-портовых асинхронных адаптеров Micro Channel с помощью SMIT

Для того чтобы просмотреть список всех определенных 128-портовых асинхронных адаптеров, как доступных, так и недоступных. выполните следующие действия:

1. Введите команду быстрого доступа `smit lsd128psync`. Система найдет информацию и покажет ее.
2. Закройте интерфейс SMIT.

8-портовый асинхронный адаптер ISA/PCI

8-портовый асинхронный адаптер ISA - это многоканальное, интеллектуальное устройство последовательной связи, предназначенное для компьютеров на базе С процессором POWER.

Адаптеры ISA имеют быстродействующую оперативную память (RAM) объемом 128 килобайт с возможностью буферизации данных. Асинхронные порты контролируются 32-разрядным процессором IDT 3041, работающим с частотой 16 Мгц, который поддерживает скорость передачи данных до 115 Кбит/с.

Процессор 3041 и двухпортовая оперативная память позволяют передать адаптеру значительную часть системных операций по обработке символов. Крупные блоки данных передаются напрямую адаптеру и затем посимвольно отправляются последовательными портами.

Двухпортовая оперативная память доступна для записи и чтения как адаптеру, так и системе. Компьютер распознает эту память как свою собственную и использует при работе с ней те же команды управления, что и для собственной внутренней памяти.

8-портовый ISA адаптер EIA 232 поддерживает только устройства EIA 232. Адаптер требует установки в системе пакета `devices.isa.cxia`.

8-портовый ISA адаптер EIA 232/422 поддерживает устройства EIA 232 и EIA 422. Оба типа устройств могут быть настроены в любом сочетании для каждого порта. Адаптер требует установки в системе пакета `devices.isa.pc8s`.

Указанным выше пакетам необходим пакет `devices.common.IBM.cx`.

Установка 8-портовых адаптеров:

Операционная система не может автоматически обнаружить адаптеры ISA, поэтому их нужно устанавливать вручную.

1. Для настройки 8-портовых асинхронных адаптеров IBM EIA 232/EIA 422 ISA воспользуйтесь командой быстрого доступа `smit mkdev_isa` для доступа к меню **Добавить адаптер ISA**.
2. Выберите **pcxr** (для 8-портового адаптера EIA 232) или **pc8s** (для 8-портового адаптера EIA 232/EIA 422), затем нажмите Enter.
3. Выберите нужный тип шины и нажмите Enter.
4. В поле Адрес шины ввода-вывода задайте адрес адаптера (устанавливается с помощью переключателей DIP на адаптере). Дополнительная информация о переключателях DIP содержится в книге *Руководство по*

установке 8-портового асинхронного адаптера ISA. Оставшаяся часть процедуры по настройке адаптера будет выполнена автоматически, когда появится сообщение `Доступный saX`.

5. По окончании процедуры нажмите **Выполнить**.

stty-cxma - это утилита, предназначенная для настройки и просмотра параметров терминала для 128-портового адаптера Micro Channel и 8- и 128 портовых адаптеров ISA. Она находится в каталоге `/usr/lbin/tty`. Формат команды следующий:

```
stty-cxma [-a] [параметры] [имя-терминала]
```

Если ввести команду **stty-cxma** без параметров, то она показывает все специальные параметры драйвера, сигналы модема и все стандартные параметры, которые команда **stty(1)** выводит для терминала, соответствующего стандартному потоку ввода. Опции команды позволяют изменять параметры управления потоком, задавать параметры печати через терминал, вводить команды модема и просматривать параметры терминала. Нераспознанные опции передаются для распознавания в команду **stty(1)**. Эти опции такие же, как и для адаптеров PCI. Дополнительная информация приведена в разделе “Опции терминала stty-cxma” на стр. 616.

Стандартные порты ввода-вывода

Большинство системных блоков имеют два встроенных (стандартных) асинхронных последовательных порта EIA 232.

Модель M20/M2A имеет один встроенный асинхронный последовательный порт, который можно настроить для поддержки двух последовательных устройств с помощью дополнительного многопортового кабеля. Асинхронные последовательные устройства EIA 232 можно подключать напрямую к стандартным последовательным портам с помощью стандартных последовательных кабелей с 9- или 25-штырьковыми D-образными разъемами.

> **Примечание:** Для платформ на базе Itanium асинхронные последовательные устройства EIA 232 можно подключать напрямую к стандартным последовательным портам с помощью стандартных последовательных кабелей с 9-штырьковыми D-образными разъемами.

Системы, поддерживающие многопроцессорность, имеют три последовательных порта.

Настройка асинхронного терминала EIA 232:

Эта процедура позволяет определить и настроить терминал, подключенный к стандартному последовательному порту или 8- или 16-портовому асинхронному адаптеру.

1. Откройте меню **Добавить очередь печати** с помощью команды `smit mktty`.
2. Выберите пункт **Добавить терминал**.
3. Выберите пункт **Асинхронный терминла tty rs232**.
4. Выберите стандартный адаптер ввода-вывода, 8-портовый адаптер или 16-портовый адаптер, как показано на экране. Если адаптеры не показаны или все они уже определены, проверьте конфигурацию и подключение проводов и запустите установку заново.
5. В показанных полях можно добавить или изменить атрибуты терминала.
6. По окончании процедуры нажмите **Выполнить**.

Настройка асинхронного принтера/графопостроителя EIA 232:

Эта процедура позволяет определить и настроить принтер/графопостроитель, подключенный к стандартному последовательному порту или 8- или 16-портовому асинхронному адаптеру.

1. Для подключения принтера/графопостроителя к асинхронному адаптеру откройте меню **Принтеры/Графопостроители** с помощью команды быстрого доступа `smit pdp`.
2. Выберите пункт **Добавить принтер/графопостроитель**.

3. Выберите тип принтера или графопостроителя в показанном на экране списке и нажмите Enter. В приведенном примере был сделан следующий выбор:

   ```
   osp Другой последовательный принтер
   ```

4. Выберите опцию **rs232**.
5. Выберите один из доступных 8-портовых контроллеров, показанных на экране. Если контроллеры не показаны или все они уже определены, проверьте конфигурацию и подключение проводов и запустите установку заново.
6. В показанных полях можно добавить или изменить атрибуты принтера/графопостроителя.
7. По окончании процедуры нажмите **Выполнить**.

8-портовые асинхронные адаптеры Micro Channel

Все асинхронные адаптеры данного семейства разработаны на основе одних и тех же принципов. Параметры отдельных адаптеров, тем не менее, определяются поддерживаемыми интерфейсами устройств.

Примечание: Информация следующего раздела неприменима к платформам на основе Itanium.

Это семейство состоит из трех адаптеров:

- 8-портовый асинхронный адаптер - EIA 232
- 8-портовый асинхронный адаптер - MIL-STD-188
- 8-портовый асинхронный адаптер - EIA 422A

Семейство 8-портовых адаптеров создано на основе микросхемы двойного универсального асинхронного приемопередатчика (DUART), обеспечивающего два канала последовательной связи.

В следующих разделах приведена подробная информация о 8-портовых адаптерах.

8-портовый асинхронный адаптер - EIA 232:

EIA 232 - это 8-портовый асинхронный адаптер, позволяющий подключать до восьми асинхронных последовательных устройств EIA 232D (таких как модемы, терминалы, графопостроители и принтеры) к системному блоку.

В системе должна быть шина Micro Channel или ISA. Система должна поддерживать до восьми 8-портовых адаптеров.

Этот адаптер полностью программируемый и поддерживает только асинхронную связь. Он может добавлять и удалять старт- и стоп-биты, а также поддерживает контроль по четности, контроль по нечетности или отсутствие контроля. Программируемые генератор скорости передачи в бодах позволяет работать со скоростью от 50 до 38400 бит/с на шине Micro Channel и от 50 до 115200 бит/с на шине ISA. Адаптеры поддерживают символы длиной 5, 6, 7 или 8 битов с 1, 1,5 или 2 стоп-битами. Система прерываний позволяет контролировать передачу, прием, ошибки и состояние линии, а также прерывания набора данных.

Установка 8-портового асинхронного адаптера:

8-портовый асинхронный адаптер устанавливается в системе в разъеме Micro Channel. Для установки адаптера выполните следующие действия.

1. Убедитесь в том, что все пользователи вышли из системы, и выполните следующую команду:

   ```
   shutdown -F
   ```

2. После завершения выполнения команды **shutdown** переведите выключатель питания системы в положение "off" (выкл.).
3. Откройте корпус системного блока и вставьте 8-портовый асинхронный адаптер в свободный разъем Micro Channel.

4. Подключите 78-штырьковый D-образный разъем кабеля 8-портового интерфейса к 8-портовому адаптеру.
5. Закройте крышку корпуса системного блока.
6. Переведите выключатель питания системного блока в положение "on" (вкл.). Система распознает и настроит 8-портовый адаптер в ходе загрузки.
7. После завершения загрузки, войдите в систему как пользователь root и выполните следующую команду, чтобы убедиться в готовности адаптера:

   ```
   lsdev -Cc adapter | pg
   ```

Только доступные адаптеры готовы к использованию системой.

Если установленный адаптер *недоступен*, убедитесь в следующем:

- Адаптер правильно вставлен в разъем Micro Channel slot.
- Все провода правильно подключены.
- Запустите команду **errpt -a | pg** и изучите отчет системы о возможных ошибках, связанных с адаптером.
- Выполните команду: **cfgmgr -v | pg**. Эта команда попытается перенастроить адаптер без перезагрузки. Найдите информацию об ошибках в постраничном выводе команды.

Если выполнить **cfgmgr** не удалось, потребуется перезагрузка системы.

Сведения об аппаратном обеспечении 8-портовых асинхронных адаптеров:

Системный интерфейс предоставляет 3-разрядный адрес и 8-разрядные данные, а также линии управления микросхемой DUART. Данные из системного интерфейса сериализуются для передачи удаленному устройству. Последовательные данные могут содержать бит контроля четности на границе байта. И наоборот, данные от удаленного устройства десериализуются для передачи системному интерфейсу. Эти данные также могут содержать бит контроля четности, который может проверяться. В качестве дополнительной возможности канал может работать в режиме FIFO (первым пришел - первым обслужен).

В режиме FIFO в буфере передатчика и приемника может содержаться до 16 байт. Последовательный интерфейс применяет стартстопный протокол и для передачи, и для приема данных. Таким образом, каждый байт, плюс бит контроля четности, заключается между одним или несколькими старт-битами и стоп-битами, обеспечивающими синхронизацию на уровне отдельных символов (байтов).

Микросхема DUART использует осциллятор, работающий на частоте 12,288 МГц, для внутренней синхронизации передачи и приема. Канал поддерживает дуплексный режим. В каждом 8-портовом адаптере установлено 4 микросхемы DUART.

Предусмотрено 13 доступных для системы регистров. Ниже перечислены программируемые опции каждого канала:

- Длина символов: 5, 6, 7 или 8 бит
- Контроль четности: по четности, по нечетности, отсутствует
- Число стоп-битов: 1, 1,5 или 2
- Включение/выключение прерываний. Доступны полученные данные
- Сохранение пустого реестра передатчиком
- Состояние линии
- Ошибка переполнения
- Ошибка контроля четности
- Ошибка фрейма
- Разрыв.

В следующей таблице приведен обзор параметров портов (интерфейсов устройств) адаптеров.

Таблица 114. Характеристики порта 8-портового асинхронного адаптера

Параметр	EIA 232	MIL-STD 188	EIA 422A
Топология	Двухточечный	Двухточечный	Двухточечный
Максимальная скорость передачи данных	138,4 Кбит/с (MC)/115,2 (ISA)	138,4 Кбит/с	138,4 Кбит/с
Средство передачи	Мультипроводник	Мультипроводник	Мультипроводник
Число проводов кабеля	9, включая землю сигнала	9, включая землю сигнала	5, включая землю сигнала
Максимальная длина кабеля	61 м	130 м со скоростью передачи 38,4 Кбит/с	1200 м < 90 Кбит/с
Разъем устройства	25-штырьковый D-образный	25-штырьковый D-образный	25-штырьковый D-образный
Электрический интерфейс	Несбалансированный	Несбалансированный	Сбалансированный
Кодировка битов	Цифровая двухуровневая	Цифровая двухуровневая	Цифровая двухуровневая

Логика распределения прерываний присваивает адаптерам приоритет согласно следующей схеме:

```
Адаптер   Приоритет
    1     Наивысший
    2        |
    3        |
    4        |
    5        |
    6        |
    7        |
    8     Самый низкий
```

Приоритет канала связи:

Каналы DUART с ожидающими прерываниями обслуживаются согласно схеме фиксированных приоритетов.

Наивысший приоритет присваивается порту 0. Следующий приоритет присваивается порту 1, и т.д. Самый низкий приоритет имеет порт 7.

Описание логики прерываний 8-портовых асинхроннх адаптеров:

Логика прерываний делится на логику генерирования прерываний и логику распределения прерываний.

В каждом 8-портовом адаптере используются обе категории логики. Генерирование прерываний обеспечивает интерфейс для системы. Эта логика создает запросы системных прерываний и содержит схемы распределения прерываний.

Логика распределения прерываний служит для идентификации 8-портового адаптера с наивысшим приоритетом. После этого информация о прерываниях порта с наивысшим приоритетом помещается в регистр распределения прерываний. Это осуществляется за одну операцию чтения.

Не следует путать логику распределения прерываний 8-портовых адаптеров с логикой распределения Micro Channel.

Логика генерирования прерываний 8-портовых адаптеров:

Асинхронные адаптеры применяют восемь линий системных запросов прерываний.

Адаптер использует следующие 8 линий системных запросов прерывания:

- IRQ 3
- IRQ 5
- IRQ 9

- IRQ 10
- IRQ 11
- IRQ 12
- IRQ 14
- IRQ 15

Во время обычной работы активна только одна линия. Все 8-портовые адаптеры одной системы должны использовать одинаковый уровень прерываний для оптимальной производительности системы. Активная линия выбирается посредством записи в соответствующий регистр POS в ходе установки. Адаптер поддерживает совместное использование прерываний и применяет параметры открытой программы сбора статистики. В этом случае линия прерываний повышается системным резистором. Адаптер понижает линию для указания активного запроса прерывания.

Логика распределения прерываний 8-портовых адаптеров:

Логика распределения прерываний определяет приоритет обслуживания программ, когда два или более 8- или 16-портовых адаптера генерируют прерывания.

В системе могут находиться и одновременно работать до восьми 8-портовых адаптеров. Эта логика за одну операцию чтения предоставляет системе идентификаторы адаптера и порта, а также тип прерывания. При обнаружении запроса на прерывание система считывает 16-разрядный регистр распределения прерываний, расположенный по адресу ввода-вывода 0130.

Сигналы интерфейса MIL-STD 188 8-портовых асинхронных адаптеров:

Каждый порт адаптера использует следующие сигналы интерфейса.

Сигнал	Определение
Tx Data	Передать данные
RTS	Готовность к отправке
CTS	Готовность к приему
DSR	Сигнал готовности к отправке данных
Rx Data	Принять данные
DCD	Обнаружение несущей данных
DTR	Сигнал готовности терминала
RI	Индикатор звонка
Sig Gnd	Земля сигнала

Уровни напряжение сигнала 8-портового MIL-STD 188:

Уровни напряжения для адаптера MIL-STD 188 могут обладать нормальной или обратной полярностью mark и space.

В следующих разделах приведены сведения об уровнях напряжения сигналов адаптера MIL-STD 188:
- Нормальная полярность mark и space
- Обратная полярность mark и space

Сигнал находится в состоянии mark, когда напряжение постоянного тока на линии, измеренное в точке интерфейса, меньше -4 В относительно земли сигнала. Сигнал находится в состоянии space, когда напряжение постоянного тока больше +4 В относительно землю сигнала. Область между +4 В и -4 В называется областью перехода и не считается действительным уровнем. Напряжение ниже -6 В или выше +6 В также не считается действительным уровнем.

Во время передачи данных состояние mark выражается двоичной единицей, а состояние space - двоичным нулем.

В схеме управления интерфейсом функция включена, когда напряжение превышает +4 В относительно земли сигнала, и отключена, когда напряжение меньше -4 В относительно земли сигнала. Уровни сигналов MIL-STD 188 приведены в следующей таблице:

Таблица 115. Уровни сигналов MIL-STD 188

Напряжение	Двоичное состояние	Состояние сигнала	Функция управления интерфейсом
Напряжение +	0	Space	Включена
Напряжение -	1	Mark	Выключена

Военный стандарт MIL-STD 188 требует, чтобы адаптеры могли инвертировать полярность состояний mark и space линий передачи и приема. Эта возможность предоставляется независимо для каждого порта.

Для этой цели используется бит 3 (вывод 2) регистра управления модемом DUART. Когда бит 3 равен 1, полярность состояний mark и space нормальная. Когда бит 3 равен 0, полярность состояний mark и space обратная.

Сигнал находится в состоянии *space*, когда напряжение постоянного тока меньше -4 В относительно землю сигнала. Сигнал находится в состоянии mark, когда напряжение постоянного тока больше +4 В относительно землю сигнала.

Область между +4 В и -4 В называется *областью перехода* и не считается действительным уровнем. Напряжение ниже -6 В или выше +6 В также не считается действительным уровнем.

Электрические параметры 8-портового асинхронного адаптера MIL-STD 188 соответствуют разделам MIL-STD 188-114, посвященным интерфейсу с несбалансированным напряжением. Этот стандарт был принят 24 мая 1976 г.

Порты адаптера соответствуют функциональным требованиям для асинхронной работы (стартстопный протокол), описанным в стандарте EIA 232C, принятом в октябре 1969 г., и стандарте EIA 232D, принятом в январе 1987 г.

Сигналы интерфейса EIA 422A 8-портовых асинхронных адаптеров:

Каждый порт адаптера EIA 422A использует следующие сигналы интерфейса.

Сигнал	Определение
TxA	Передать данные
TxB	Передать данные
RxA	Принять данные
RxB	Принять данные
Sig Gnd	Земля сигнала

Уровни напряжение сигнала 8-портового EIA 422A:

Формирователь линии создает дифференциальное напряжение от 2 до 6 В (измеряется в точке интерфейса генератора). Дифференциальное напряжение в приемнике должно лежать в диапазоне от 200 мВ до 6 В (измеряется в точке нагрузки интерфейса).

Измерения проводятся на терминале A (положительный контакт) относительно терминала B (отрицательный контакт). В следующей таблице описаны состояния сигнала в зависимости от уровней напряжения:

Таблица 116. Состояния сигнала 8-портового EIA 422A

Напряжение	Двоичное состояние	Состояние сигнала
Напряжение +	0	Space
Напряжение -	1	Mark

8-портовый асинхронный адаптер EIA 422A поддерживает кабель длиной до 1200 м, предназначенный для прокладки внутри помещения. Кабели такой длины подвержены внезапным скачкам напряжения из-за наведенного напряжения, вызванного, например, непрямым ударом молнии. Для защиты от этих явлений в адаптере EIA 422A применяется вторичный контур защиты от скачков напряжения. Защитный контур реализован в линиях данных интерфейса адаптера.

Входные контакты каждого приемника EIA 422A снабжены контуром защиты от сбоев для предотвращения сбоев, которые могут возникнуть, когда приемник не подключен к формирователю (открытый кабель). Контур защиты от сбоев переводит приемник в состояние mark (двоичная единица) всякий раз, когда приемник не подключен к формирователю.

Электрические параметры портов 8-портового асинхронного адаптера EIA 422A соответствуют стандарту EIA 422A, принятому в декабре 1978 г.

Сигналы интерфейса EIA 232 8-портовых асинхронных адаптеров:

Каждый порт 8-портового асинхронного адаптера использует следующие сигналы интерфейса.

Каждый порт адаптера использует следующие сигналы интерфейса:

Сигнал	Определение
TxD	Передать данные
RTS	Готовность к отправке
CTS	Готовность к приему.
DSR	Сигнал готовности к отправке данных
RxD	Принять данные
DCD	Обнаружение несущей данных
DTR	Сигнал готовности терминала
RI	Индикатор звонка
Sig Gnd	Земля сигнала

Уровни напряжение сигнала 8-портового EIA 232:

Сигнал находится в состоянии mark, когда напряжение постоянного тока на линии, измеренное в точке интерфейса, меньше -3 В относительно земли сигнала. Сигнал находится в состоянии space, когда напряжение постоянного тока больше +3 В относительно земли сигнала. Область между +3 В и -3 В называется *областью перехода* и не считается действительным уровнем. Напряжение ниже -15 В или выше +15 В также не считается действительным уровнем.

Во время передачи данных состояние mark выражается двоичной единицей, а состояние space - двоичным нулем.

В схеме управления интерфейсом функция включена, когда напряжение превышает +3 В относительно земли сигнала, и отключена, когда напряжение меньше -3 В относительно земли сигнала. Информация об уровнях сигнала EIA 232 приведена в следующей таблице:

Таблица 117. Уровни сигнала EIA 232

Напряжение	Двоичное состояние	Состояние сигнала	Функция управления интерфейсом
Напряжение +	0	Space	Включена
Напряжение -	1	Mark	Выключена

Электрические параметры портов 8-портового асинхронного адаптера EIA 232 соответствуют стандарту EIA 232C, принятому в октябре 1969 г., и стандарту EIA 232D, принятому в январе 1987 г.

Порты адаптера соответствуют функциональным требованиям для асинхронной работы (стартстопный протокол), описанным в стандарте EIA 232C, принятом в октябре 1969 г., и стандарте EIA 232D, принятом в январе 1987 г.

Логика управления 8-портовыми асинхронными адаптерами:

Управляющая логика на основе PAL координирует деятельность всех основных функций адаптера.

Это выполняется с помощью генератора прямоугольных сигналов, работающего с частотой 40 Мгц. Она связывается с Micro Channel и включает такие функции, как декодирование адресов, контроль четности в адресах, ответ с правильными управляющими сигналами ввода-вывода и управление выбранной линий запросов прерываний (IRQ) (одной из восьми линий IRQ).

Управляющая логика связывается и с другими блоками логики адаптера. Это обеспечивает управляющие линии для каналов связи (DUART) и логику распределения прерываний. Управляющая логика также связывается с логикой драйверов шины и обеспечивает управление направлением потока данных и выбором байтов данных, помещаемых в локальную шину. Она управляет генератором контроля четности данных, блоком проверки четности и защелками.

16-портовые асинхронные адаптеры

Все адаптеры данного семейства разработаны на основе одних и тех же принципов. Параметры отдельных адаптеров, тем не менее, определяются поддерживаемыми интерфейсами устройств. Данное семейство состоит из двух адаптеров: 16-портового асинхронного адаптера EIA 422A и 16-портового асинхронного адаптера EIA 232.

Примечание: Информация следующего раздела неприменима к платформам на основе Itanium.

Семейство 16-портовых адаптеров создано на основе микросхемы двойного универсального асинхронного приемопередатчика (DUART), обеспечивающего два канала последовательной связи. Более подробная информация о микросхеме DUART и ее функциональных возможностях приведена в руководстве Информация об аппаратном обеспечении 16-портового асинхронного адаптера.

16-портовый асинхронный адаптер - EIA 422A:

16-портовый асинхронный адаптер EIA 232 поддерживает подключение до 16 асинхронных последовательных устройств EIA 232 (принтеры и терминалы) к системному блоку.

С одним системным блоком можно использовать до 8 адаптеров одного семейства (в любом сочетании).

Этот адаптер полностью программируемый и поддерживает только асинхронную связь. Он добавляет и удаляет старт-биты и стоп-биты. Адаптеры поддерживают контроль последовательных данных по четности, по нечетности или отсутствие контроля. Программируемый генератор скорости передачи в бодах поддерживает скорость от 50 до 38400 бит/с. Адаптеры поддерживают символы длиной 5, 6, 7 или 8 битов с 1, 1,5 или 2 стоп-битами. Система прерываний позволяет контролировать передачу, прием, ошибки и состояние линии, а также прерывания набора данных. 16 разъемов для подключения устройств организованы в 16-портовый набор EIA 422A.

16-портовые асинхронные адаптеры EIA 422A обладают следующими характеристиками:

- Стандартная карта формата Micro Channel.
- Скорость передачи данных до 38,4 Кбит/с для каждого порта.
- 16-байтовая буферизация при отправке и приеме.
- Одинарный 78-штырьковый разъем вывода (к нему присоединяются кабели с различными интерфейсами).
- Контур защиты от скачков напряжения.
- Поддержка кабеля длиной до 1200 м.
- Поддержка сигналов интерфейса TxD и RxD.
- 8-разрядный/16-разрядный подчиненный интерфейс Micro Channel.

Установка 16-портового асинхронного адаптера:

16-портовый асинхронный адаптер устанавливается на сервере в разъеме Micro Channel. Для установки адаптера выполните следующие действия.

1. Убедитесь в том, что все пользователи вышли из системы, и выполните следующую команду:
   ```
   shutdown -F
   ```
2. После завершения выполнения команды **shutdown** переведите выключатель питания системы в положение "off" (выкл.).
3. Откройте корпус сервера и вставьте 16-портовый асинхронный адаптер в свободный разъем MicroChannel.
4. Подключите 78-штырьковый D-образный разъем кабеля 16-портового интерфейса к 16-портовому адаптеру.
5. Закройте крышку корпуса системного блока.
6. Переведите выключатель питания системного блока в положение "On" (вкл.). Система распознает и настроит 16-портовый адаптер в ходе загрузки.

После завершения загрузки войдите в систему как пользователь root и выполните следующую команду, чтобы убедиться в готовности адаптера:

```
lsdev -Cc adapter | pg
```

Только доступные адаптеры готовы к использованию системой.

Если установленный адаптер недоступен, убедитесь в следующем:

1. Адаптер правильно вставлен в разъем Micro Channel slot.
2. Все провода правильно подключены.
3. Запустите команду **errpt -a | pg** и изучите отчет системы о возможных ошибках, связанных с адаптером.
4. Выполните команду: **cfgmgr -v | pg**. Эта команда попытается перенастроить адаптер без перезагрузки. Найдите информацию об ошибках в постраничном выводе команды.
5. Если выполнить **cfgmgr** не удалось, потребуется перезагрузка системы.

Сведения об аппаратном обеспечении 16-портовых асинхронных адаптеров:

Системный интерфейс предоставляет 3-разрядный адрес и 8-разрядные данные, а также линии управления микросхемой. Данные из системного интерфейса сериализуются для передачи удаленному устройству. Последовательные данные могут содержать бит контроля четности на границе байта. И наоборот, данные от удаленного устройства десериализуются для передачи системному интерфейсу. Эти данные также могут содержать бит контроля четности, который может проверяться. В качестве дополнительной возможности канал может работать в режиме FIFO (первым пришел - первым обслужен).

В режиме FIFO в буфере передатчика и приемника может содержаться до 16 байт. Последовательный интерфейс применяет стартстопный протокол и для передачи, и для приема данных. Таким образом, каждый

байт, плюс бит контроля четности, заключается между старт-битом и стоп-битом, обеспечивающими синхронизацию на уровне отдельных символов (байтов).

Микросхема DUART использует осциллятор, работающий на частоте 12,288 МГц, для внутренней синхронизации передачи и приема. Канал поддерживает дуплексный режим. В каждом 16-портовом адаптере установлено 8 микросхем DUART.

Предусмотрено 13 доступных для системы регистров. Ниже перечислены программируемые опции каждого канала:

- Длина символов: 5, 6, 7 или 8 бит
- Контроль четности: по четности, по нечетности, отсутствует
- Число стоп-битов: 1, 1,5 или 2
- Включение/выключение прерываний. Доступны полученные данные
- Сохранение пустого реестра передатчиком
- Состояние линии
- Ошибка переполнения
- Ошибка контроля четности
- Ошибка фрейма
- Разрыв.

В следующей таблице приведен обзор параметров портов (интерфейсов устройств) адаптеров.

Таблица 118. Характеристики порта 16-портового асинхронного адаптера

Параметр	**EIA 232**	**EIA 422A**
Топология	Двухточечный	Двухточечный
Максимальная скорость передачи данных (стандартная)	20 Кбит/с	2 Мбит/с
Максимальная скорость передачи данных (плата)	38,4 Кбит/с	38,4 Кбит/с
Средство передачи	Мультипроводник	Мультипроводник
Число проводов кабеля	5, включая землю сигнала	5, включая землю сигнала
Максимальная длина кабеля	61 м	1200 м < 90 Кбит/с
Разъем устройства	25-штырьковый D-образный	25-штырьковый D-образный
Электрический интерфейс	Несбалансированный	Сбалансированный
Кодировка битов	Цифровая двухуровневая	Цифровая двухуровневая

Приоритет платы 16-портового асинхронного адаптера:

Логика распределения прерываний присваивает адаптерам приоритет согласно определенной схеме.

```
Адаптер   Приоритет
    0     Наивысший
    1        |
    2        |
    3        |
    4        |
    5        |
    6        |
    7        |
    8        |
    9        |
    10       |
    11       |
```

Каналы DUART с ожидающими прерываниями обслуживаются согласно схеме фиксированных приоритетов. Наивысший приоритет присваивается порту 0. Следующий приоритет присваивается порту 1, и т.д. Самый низкий приоритет имеет порт 15.

Логика прерываний 16-портовых асинхронных адаптеров:

Для 16-портовых адаптеров логика прерываний делится на логику генерирования прерываний и логику распределения прерываний.

В каждом 16-портовом адаптере используются обе категории логики. Генерирование прерываний обеспечивает интерфейс для системы. Эта логика создает запросы системных прерываний и содержит схемы распределения прерываний.

Логика распределения прерываний служит для идентификации 16-портового адаптера с наивысшим приоритетом. После этого информация о прерываниях порта с наивысшим приоритетом помещается в регистр распределения прерываний. Это осуществляется за одну операцию чтения.

Не следует путать логику распределения прерываний 16-портовых адаптеров с логикой распределения Micro Channel.

Логика генерирования прерываний 16-портовых адаптеров:

16-портовые асинхронные адаптеры применяют восемь линий системных запросов прерываний.

Адаптер использует следующие 8 линий системных запросов прерывания (IRQ):
- IRQ 3
- IRQ 5
- IRQ 9
- IRQ 10
- IRQ 11
- IRQ 12
- IRQ 14
- IRQ 15

Во время обычной работы активна только одна линия. Все 16-портовые адаптеры одной системы должны использовать одинаковый уровень прерываний для оптимальной производительности системы. Активная линия выбирается посредством записи в соответствующий регистр POS в ходе установки. Адаптер поддерживает совместное использование прерываний и применяет параметры открытой программы сбора статистики, как указано в архитектуре Micro Channel. В этом случае линия прерываний повышается системным резистором. Адаптер понижает линию для указания активного запроса прерывания.

Логика распределения прерываний 16-портовых адаптеров:

Логика распределения прерываний определяет приоритет обслуживания программ, когда два или более 8- или 16-портовых адаптера генерируют прерывания.

В системе могут находиться и одновременно работать до восьми 8-портовых или 16-портовых адаптеров. Эта логика за одну операцию чтения предоставляет системе идентификаторы адаптера и порта, а также тип прерывания. При обнаружении запроса на прерывание система считывает 16-разрядный регистр распределения прерываний, расположенный по адресу ввода-вывода 0130.

Сигналы интерфейса EIA 232 16-портовых асинхронных адаптеров:

Каждый порт 16-портового асинхронного адаптера использует следующие сигналы интерфейса.

Сигнал	Определение
TxD	Передать данные
DCD	Обнаружение несущей
DTR	Сигнал готовности терминала
RxD	Принять данные
Sig Gnd	Земля сигнала

Уровни напряжение сигнала 16-портового EIA 232:

Сигнал находится в состоянии mark, когда напряжение постоянного тока на линии, измеренное в точке интерфейса, меньше -3 В относительно земли сигнала. Сигнал находится в состоянии space, когда напряжение постоянного тока больше +3 В относительно земли сигнала. Область между +3 В и -3 В называется областью перехода и не считается действительным уровнем. Напряжение ниже -15 В или выше +15 В также не считается действительным уровнем.

Во время передачи данных состояние mark выражается двоичной единицей, а состояние space - двоичным нулем.

В схеме управления интерфейсом функция включена, когда напряжение превышает +3 В относительно земли сигнала, и отключена, когда напряжение меньше -3 В относительно земли сигнала. Информация об уровнях сигнала EIA 232 приведена в следующей таблице:

Таблица 119. Уровни сигнала EIA 232

Напряжение	Двоичное состояние	Состояние сигнала	Функция управления интерфейсом
Напряжение +	0	Space	Включена
Напряжение -	1	Mark	Выключена

Электрические параметры портов 16-портового асинхронного адаптера EIA 232 соответствуют стандарту EIA 232C, принятому в октябре 1969 г., и стандарту EIA 232D, принятому в январе 1987 г.

Порты адаптера соответствуют функциональным требованиям для асинхронной работы (стартстопный протокол), описанным в стандарте EIA 232C, принятом в октябре 1969 г., и стандарте EIA 232D, принятом в январе 1987 г.

Сигналы интерфейса EIA 422A 16-портовых асинхронных адаптеров:

Каждый порт 16-портового асинхронного адаптера EIA 422A использует следующие сигналы интерфейса.

Сигнал	Определение
TxA	Передать данные
TxB	Передать данные
RxA	Принять данные
RxB	Принять данные
Sig Gnd	Земля сигнала

Уровни напряжение сигнала 16-портового EIA 422A:

Формирователь линии создает дифференциальное напряжение от 2 до 6 В (измеряется в точке интерфейса генератора). Дифференциальное напряжение в приемнике должно лежать в диапазоне от 200 мВ до 6 В (измеряется в точке нагрузки интерфейса).

Измерения проводятся на терминале A (положительный контакт) относительно терминала B (отрицательный контакт). В следующей таблице описаны состояния сигнала в зависимости от уровней напряжения:

Таблица 120. Состояния сигнала 16-портового EIA 422A

Напряжение	Двоичное состояние	Состояние сигнала
Напряжение +	0	Space
Напряжение -	1	Mark

16-портовый асинхронный адаптер EIA 422A поддерживает кабель длиной до 1200 м, предназначенный для прокладки внутри помещения. Кабели такой длины подвержены внезапным скачкам напряжения из-за наведенного напряжения, вызванного, например, непрямым ударом молнии. Для защиты от этих явлений в адаптере EIA 422A применяется вторичный контур защиты от скачков напряжения. Защитный контур реализован в линиях данных интерфейса адаптера.

Входные контакты каждого приемника EIA 422A снабжены контуром защиты от сбоев для предотвращения сбоев, которые могут возникнуть, когда приемник не подключен к формирователю (открытый кабель). Контур защиты от сбоев переводит приемник в состояние mark (двоичная единица) всякий раз, когда приемник не подключен к формирователю.

Электрические параметры портов 16-портового асинхронного адаптера EIA 422A соответствуют стандарту EIA 422A, принятому в декабре 1978 г.

Таблица преобразования между значениями ASCII, шестнадцатеричными, восьмеричными и двоичными значениями

Полезная информация для преобразования между значениями ASCII, десятичными, шестандацеричными, восьмеричными и двоичными.

Таблица 121. Преобразование между значениями ASCII, десятичными, шестнадцатеричными, восьмеричными и двоичными значениями

ASCII	Десятичное значение	Шестн. значение	Восьмеричное значение	Двоичное значение
null	0	0	0	0
начало заголовка	1	1	1	1
начало текста	61 см	61 см	61 см	10
конец текста	3	3	3	11
конец передачи	4	4	4	100
запрос	5	5	5	101
подтверждение	6	6	6	110
сигнал	7	7	7	111
забой	8	8	10	1000
горизонтальная табуляция	9	9	11	1001
перевод строки	10	A	12	1010
вертикальная табуляция	11	B	13	1011
перевод страницы	12	C	14	1100
возврат каретки	13	Д	15	1101
открывающий скобочный символ	14	E	16	1110
закрывающий скобочный символ	15	F	17	1111
переход при передаче данных	16	10	20	10000

Таблица 121. Преобразование между значениями ASCII, десятичными, шестнадцатеричными, восьмеричными и двоичными значениями (продолжение)

ASCII	Десятичное значение	Шестн. значение	Восьмеричное значение	Двоичное значение
управляющий сигнал 1 устройства/Xon	17	11	21	10001
управляющий сигнал 2 устройства	18	12	22	10010
управляющий сигнал 3 устройства/Xoff	19	13	23	10011
управляющий сигнал 4 устройства	20	14	24	10100
отрицательное подтверждение	21	15	25	10101
синхронный простой	22	16	26	10110
конец блока передачи	23	17	27	10111
отмена	24	18	30	11000
конец носителя	25	19	31	11001
конец файла/ заменяющий символ	26	1A	32	11010
escape	27	1B	33	11011
разделитель файлов	28	1C	34	11100
разделитель групп	29	1D	35	11101
разделитель записей	30	1E	36	11110
разделитель элементов	31	1F	37	11111
пробел	32	20	40	100000
!	33	21	41	100001
"	34	22	42	100010
#	35	23	43	100011
$	36	24	44	100100
%	37	25	45	100101
&	38	26	46	100110
'	39	27	47	100111
(	40	28	50	101000
)	41	29	51	101001
*	42	2A	52	101010
+	43	2B	53	101011
,	44	2C	54	101100
-	45	2D	55	101101
.	46	2E	56	101110
/	47	2F	57	101111
0	48	30	60	110000
1	49	31	61	110001
61 см	50	32	62	110010
3	51	33	63	110011
4	52	34	64	110100
5	53	35	65	110101
6	54	36	66	110110
7	55	37	67	110111
8	56	38	70	111000

Таблица 121. Преобразование между значениями ASCII, десятичными, шестнадцатеричными, восьмеричными и двоичными значениями (продолжение)

ASCII	Десятичное значение	Шестн. значение	Восьмеричное значение	Двоичное значение
9	57	39	71	111001
:	58	3A	72	111010
;	59	3B	73	111011
<	60	3C	74	111100
=	61	3D	75	111101
>	62	3E	76	111110
?	63	3F	77	111111
@	64	40	100	1000000
A	65	41	101	1000001
B	66	42	102	1000010
C	67	43	103	1000011
Д	68	44	104	1000100
E	69	45	105	1000101
F	70	46	106	1000110
G	71	47	107	1000111
H	72	48	110	1001000
I	73	49	111	1001001
J	74	4A	112	1001010
K	75	4B	113	1001011
L	76	4C	114	1001100
M	77	4D	115	1001101
N	78	4E	116	1001110
O	79	4F	117	1001111
P	80	50	120	1010000
Q	81	51	121	1010001
R	82	52	122	1010010
C	83	53	123	1010011
T	84	54	124	1010100
U	85	55	125	1010101
V	86	56	126	1010110
W	87	57	127	1010111
X	88	58	130	1011000
Y	89	59	131	1011001
Z	90	5A	132	1011010
[	91	5B	133	1011011
\	92	5C	134	1011100
]	93	5D	135	1011101
^	94	5E	136	1011110
_	95	5F	137	1011111
`	96	60	140	1100000
a	97	61	141	1100001
b	98	62	142	1100010
c	99	63	143	1100011
d	100	64	144	1100100

Таблица 121. Преобразование между значениями ASCII, десятичными, шестнадцатеричными, восьмеричными и двоичными значениями (продолжение)

ASCII	Десятичное значение	Шестн. значение	Восьмеричное значение	Двоичное значение
e	101	65	145	1100101
f	102	66	146	1100110
g	103	67	147	1100111
h	104	68	150	1101000
i	105	69	151	1101001
j	106	6A	152	1101010
k	107	6B	153	1101011
l	108	6C	154	1101100
m	109	6D	155	1101101
n	110	6E	156	1101110
o	111	6F	157	1101111
p	112	70	160	1110000
q	113	71	161	1110001
r	114	72	162	1110010
s	115	73	163	1110011
t	116	74	164	1110100
u	117	75	165	1110101
v	118	76	166	1110110
w	119	77	167	1110111
x	120	78	170	1111000
y	121	79	171	1111001
z	122	7A	172	1111010
{	123	7B	173	1111011
|	124	7C	174	1111100
}	125	7D	175	1111101
~	126	7E	176	1111110
DEL	127	7F	177	1111111
	128	80	200	10000000
	129	81	201	10000001
	130	82	202	10000010
	131	83	203	10000011
	132	84	204	10000100
	133	85	205	10000101
	134	86	206	10000110
	135	87	207	10000111
	136	88	210	10001000
	137	89	211	10001001
	138	8A	212	10001010
	139	8B	213	10001011
	140	8C	214	10001100
	141	8D	215	10001101
	142	8E	216	10001110
	143	8F	217	10001111
	144	90	220	10010000

Таблица 121. Преобразование между значениями ASCII, десятичными, шестнадцатеричными, восьмеричными и двоичными значениями (продолжение)

ASCII	Десятичное значение	Шестн. значение	Восьмеричное значение	Двоичное значение
	145	91	221	10010001
	146	92	222	10010010
	147	93	223	10010011
	148	94	224	10010100
	149	95	225	10010101
	150	96	226	10010110
	151	97	227	10010111
	152	98	230	10011000
	153	99	231	10011001
	154	9A	232	10011010
	155	9B	233	10011011
	156	9C	234	10011100
	157	9D	235	10011101
	158	9E	236	10011110
	159	9F	237	10011111
	160	A0	240	10100000
	161	A1	241	10100001
	162	A2	242	10100010
	163	A3	243	10100011
	164	A4	244	10100100
	165	A5	245	10100101
	166	A6	246	10100110
	167	A7	247	10100111
	168	A8	250	10101000
	169	A9	251	10101001
	170	AA	252	10101010
	171	AB	253	10101011
	172	AC	254	10101100
	173	AD	255	10101101
	174	AE	256	10101110
	175	AF	257	10101111
	176	B0	260	10110000
	177	B1	261	10110001
	178	B2	262	10110010
	179	B3	263	10110011
	180	B4	264	10110100
	181	B5	265	10110101
	182	B6	266	10110110
	183	B7	267	10110111
	184	B8	270	10111000
	185	B9	271	10111001
	186	BA	272	10111010
	187	BB	273	10111011
	188	BC	274	10111100

Таблица 121. Преобразование между значениями ASCII, десятичными, шестнадцатеричными, восьмеричными и двоичными значениями (продолжение)

ASCII	Десятичное значение	Шестн. значение	Восьмеричное значение	Двоичное значение
	189	BD	275	10111101
	190	BE	276	10111110
	191	BF	277	10111111
	192	C0	300	11000000
	193	C1	301	11000001
	194	C2	302	11000010
	195	C3	303	11000011
	196	C4	304	11000100
	197	C5	305	11000101
	198	C6	306	11000110
	199	C7	307	11000111
	200	C8	310	11001000
	201	C9	311	11001001
	202	CA	312	11001010
	203	CB	313	11001011
	204	CC	314	11001100
	205	CD	315	11001101
	206	CE	316	11001110
	207	CF	317	11001111
	208	D0	320	11010000
	209	D1	321	11010001
	210	D2	322	11010010
	211	D3	323	11010011
	212	D4	324	11010100
	213	D5	325	11010101
	214	D6	326	11010110
	215	D7	327	11010111
	216	D8	330	11011000
	217	D9	331	11011001
	218	DA	332	11011010
	219	DB	333	11011011
	220	DC	334	11011100
	221	DD	335	11011101
	222	DE	336	11011110
	223	DF	337	11011111
	224	E0	340	11100000
	225	E1	341	11100001
	226	E2	342	11100010
	227	E3	343	11100011
	228	E4	344	11100100
	229	E5	345	11100101
	230	E6	346	11100110
	231	E7	347	11100111
	232	E8	350	11101000

Таблица 121. Преобразование между значениями ASCII, десятичными, шестнадцатеричными, восьмеричными и двоичными значениями (продолжение)

ASCII	Десятичное значение	Шестн. значение	Восьмеричное значение	Двоичное значение
	233	E9	351	11101001
	234	EA	352	11101010
	235	EB	353	11101011
	236	EC	354	11101100
	237	ED	355	11101101
	238	EE	356	11101110
	239	EF	357	11101111
	240	F0	360	11110000
	241	F1	361	11110001
	242	F2	362	11110010
	243	F3	363	11110011
	244	F4	364	11110100
	245	F5	365	11110101
	246	F6	366	11110110
	247	F7	367	11110111
	248	F8	370	11111000
	249	F9	371	11111001
	250	FA	372	11111010
	251	FB	373	11111011
	252	FC	374	11111100
	253	FD	375	11111101
	254	FE	376	11111110
	255	FF	377	11111111

uDAPL (библиотека программирования прямого доступа пользовательского уровня)

uDAPL (пользовательская библиотека программирования прямого доступа) — это среда прямого доступа, работающая на транспортных протоколах, которые поддерживают прямой доступ к данным аналогично InfiniBand, RNIC и т.д.

DAT Collaborative определяет **uDAPL** API http://www.datcollaborative.org .

Основание кода uDAPL из Open Fabrics связывается с AIX и в настоящее время поддерживается адаптерами GX++ HCA и 4X DDR Expansion card (CFFh) InfiniBand.

uDAPL версии 1.2 поддерживается в AIX 6.1 с пакетом обслуживания 6100-06 и следующих версий. Установочный образ **uDAPL** поставляется на пакет расширения, такой как *udapl.rte*. Этот образ предоставляет файлы заголовков DAT, которые находятся в **/usr/include/dat**. Установочный образ также содержит две библиотеки: *libdat.a* и *libdapl.a*.

Приложения включают файлы заголовков DAT и ссылку с библиотекой DAT *libdat.a* в **/usr/include/dat**). Уровень DAT определяет соответствующие основные библиотеки для транспортных протоколов.

Провайдер AIX **uDAPL** регистрируется с использованием записей *dat.conf* в реестре DAT. Файл */etc/dat.conf* поставляется с записями по умолчанию и сведениями о формате записей.

Для целей отладки библиотеки **uDAPL** поддерживают трассировку системных событий AIX. ИД точек трассировки системных событий **uDAPL** включают 5C3 (для событий DAPL), 5C4 (для событий ошибок DAPL), 5C7 (для событий DAT) и 5C8 (для событий ошибок DAT). Первоначальный уровень трассировки можно изменять с помощью переменных среды *DAT_TRACE_LEVEL* и *DAPL_TRACE_LEVEL*, для которых можно задавать численные значения от 0 до 10. Чем выше уровень, тем больше число отслеживаемых событий и объем данных при следующих основных уровнях трассировки

```
TRC_LVL_ERROR  = 1,
TRC_LVL_NORMAL = 3,
TRC_LVL_DETAIL = 7
```

Другие стандартные сервисные компоненты AIX, такие как протокол ошибок AIX, могут быть полезны для определения неполадок. Сервисные компоненты основного уровня транспортного протокола, такие как команда ibstat и трассировка компонентов InfiniBand, также удобны для целей диагностики.

API DAT возвращают стандартные коды возврата, которые можно декодировать с помощью файла */usr/include/dat/dat_error.h*. Подробное описание кодов возврата содержится в спецификации **uDAPL** от DAT Collaborative.

“Протокол IP для InfiniBand (IPoIB)” на стр. 399

API uDAPL, поддерживаемые в AIX

Из многих API **uDAPL**, определяемых DAT Collaborative, имеются несколько API, которые не поддерживаются в AIX.

Следующие API, которые распространены в отраслевых реализациях **uDAPL**, не поддерживаются и также не будут поддерживаться в AIX.

Элемент	Описание
dat_cr_handoff	// B DAT 1.2
dat_ep_create_with_srq	// B DAT 1.2
dat_ep_recv_query	// B DAT 1.2
dat_ep_set_watermark	// B DAT 1.2
dat_srq_create	// B DAT 1.2
dat_srq_post_recv	// B DAT 1.2
dat_srq_resize	// B DAT 1.2
dat_srq_set_lw	// B DAT 1.2
dat_srq_free	// B DAT 1.2
dat_srq_query	// B DAT 1.2

Другие API, которые не поддерживает AIX, следующие:

- dat_lmr_sync_rdma_read
- dat_lmr_sync_rdma_write
- dat_registry_add_provider
- dat_registry_add_provider

Для всех неподдерживаемых API AIX следует определенным механизмам, описанным в спецификации DAT в разделе неподдерживаемых компонентов. Они также включают значения атрибутов (например, max_srq, равный нулю) и определенные коды возвратов (например DAT_MODEL_NOT_SUPPORTED). В соответствии с отраслевой реализацией и спецификацией DAT DAT_NOT_IMPLEMENTED также может возвращаться для неподдерживаемых функций.

Поддержка RMR, связанных с API, таких как *dat_rmr_create* ,*dat_rmr_bind* ,*dat_rmr_free* и *dat_rmr_query*, зависит от основных функциональных возможностей HCA, а успешность или неудача определяется основной средой IB. В настоящее время адаптеры GX++ HCA и 4X DDR Expansion card (CFFh) InfiniBand не поддерживают эти операции RMR.

“uDAPL (библиотека программирования прямого доступа пользовательского уровня)” на стр. 686

“Зависящие от поставщика атрибуты для uDAPL”

“Протокол IP для InfiniBand (IPoIB)” на стр. 399

Зависящие от поставщика атрибуты для uDAPL

В AIX существуют несколько зависящих от поставщика атрибутов . Имена этих атрибутов следующие: **delayed_ack_supported** , **vendor_extension**, **vendor_ext_version**, **debug_query** и **debug_modify**.

delayed_ack_supported

Провайдер AIX транспортного протокола InfiniBand (IB) включает зависящий от поставщика атрибут адаптера интерфейса (IA) с именем **delayed_ack_supported**. Для этого атрибута допустимы значения **true** и **false**. При значении **true** конечные точки, связанные с этим IA, имеют изменяемый зависящий от провайдера атрибут с именем **delayed_ack**. Если атрибут **delayed_ack_supported** имеет значение **false**, то зависящий от провайдера атрибут **delayed_ack** конечной точки изменяться не может. По умолчанию атрибут **delayed_ack** конечной точки имеет значение **false**. Задание значения **true** (через dat_ep_modify) разрешает функцию отложенного подтверждения адаптера канала хоста IB Host Channel (HCA) для конкретной пары очереди IB, связанной с конечной точкой. Эта аппаратная функция не реализуется всеми HCA и поэтому недоступна всем IA. При разрешении этой функции HCA задерживает отправку подтверждения, пока операция передачи данных видна в системной памяти сервера. Это немного более строгая семантика, чем предусматриваемая спецификацией IB, может быть связана с очень небольшим увеличением времени ожидания.

vendor_extension, vendor_ext_version, debug_query и debug_modify

Для целей отладки библиотеки **uDAPL** поддерживают трассировку системных событий AIX. Первоначальный уровень трассировки можно изменять с помощью переменных среды *DAT_TRACE_LEVEL* и *DAPL_TRACE_LEVEL*. Для динамичного изменения уровней трассировки с помощью API предусмотрена динамическая поддержка уровней трассировки в AIX. Для проверки динамической поддержки уровней трассировки библиотекой приложения могут запрашивать зависящий от поставщика атрибут IA **vendor_extension**. По результату запроса присутствие атрибута **vendor_extension** означает динамическую поддержку уровней трассировки. Независимо от задания атрибуту значения **true** присутствие этого атрибута означает поддержку. Когда атрибут **vendor_extension** присутствует, приложения могут получать указатели функции на **dat_trclvl_query()** и **dat_trclvl_modify()** по запросу зависящих от поставщика атрибутов IA: **debug_query** и **debug_modify**. Значение этих атрибутов будет иметь указатель на соответствующие функции. Для возможности расширения интерфейса **vendor_extension** в будущем предусмотрен еще один зависящий от поставщика атрибут IA с именем **vendor_ext_version**. Поскольку сейчас поддерживается только одна версия, значением этого атрибута будет **1.0**. Если атрибут **vendor_extension** отсутствует, приложения не могут динамически изменять уровни трассировки.

Пример управления этими атрибутами включен в пример кода **uDAPL**, который устанавливается с реализацией AIX.

“uDAPL (библиотека программирования прямого доступа пользовательского уровня)” на стр. 686

“Протокол IP для InfiniBand (IPoIB)” на стр. 399

Поддержка PCIe2 10 GbE RoCE Adapter

Адаптер PCIe2 10GbE RDMA Over Converged Ethernet (RoCE) впервые поддерживался в операционной системе AIX как удаленное устройство с поддержкой прямого доступа к памяти (RDMA). Поддержка адаптера осуществлялась собственным программным обеспечением IBM на основе стека AIX InfiniBand. Это средство поддержки называлось AIX RoCE. AIX 7 с пакетом обслуживания 7100-02 поддерживает этот адаптер в двух режимах: с помощью AIX RoCE и с помощью 10G Ethernet, называемого также картой

сетевого интерфейса (AIX NIC). Новый AIX 7 с пакетом обслуживания 7100-03 теперь поддерживает режим RDMA вместе с NIC и OpenFabrics Enterprise Distribution (OFED). Адаптер шины хоста (HBA), который не был доступен в предыдущих версиях операционной системы AIX, управляет выбором включаемого режима.

В следующей таблице показана эволюция программного обеспечения адаптера PCIe2 10GbE Adapter:

Уровень AIX	РЕЖИМ 1	РЕЖИМ 2
До AIX 7 с пакетом обслуживания 7100-02	AIX RoCE	NA
AIX 7 с пакетом обслуживания 7100-02	AIX RoCE	AIX NIC
AIX 7 с пакетом обслуживания 7100-03	AIX RoCE	AIX NIC + OFED RoCE

Для загрузки последнего драйвера адаптера выполните следующие действия:

1. Откройте веб-сайт IBM (www.ibm.com)
2. Выберите **Поддержка и загрузки**.
3. Загрузите последнее встроенное программное обеспечение в расположение хоста AIX (`/etc/microcode`)
4. Запустите инструмент **diag** для обновления встроенного программного обеспечения, выбрав одну из следующих процедур:
   - Короткая процедура
     a. Введите следующую команду:
        ```
        *diag -d entX -T download
        ```
        Примечание: Замените **entX** на **roceX** в случае применения стека RoCE из предыдущей версии.
     b. Выберите микрокод, сохраненный в каталоге `/etc/microcode`.
   - Длинная процедура
     a. Введите следующую команду:
        ```
        *diag
        ```
     b. Выберите: **Выбор задачи > Задачи микрокода > Загрузить микрокод**.
     c. Выберите **entX** или **roceX**.
     d. Выберите микрокод, сохраненный в каталоге `/etc/microcode`.

По умолчанию для адаптера настроена поддержка режима AIX RoCE. Для изменения режима выполните инструкции из раздела “AIX NIC + OFED RDMA”.

AIX NIC + OFED RDMA

Начиная с AIX 7 с пакетом обслуживания 7100-02, адаптер PCIe2 10 GbE RoCE можно настроить для работы в конфигурации AIX NIC. Начиная с AIX 7 с пакетом обслуживания 7100-03, в конфигурацию AIX NIC можно также добавить функции OFED RDMA. Если отсутствуют приложения, эффективность которых зависит от RDMA, то адаптер можно использовать только как карту сетевого адаптера (NIC).

Для использования адаптера PCIe2 10 GbE RoCE Adapter в конфигурации AIX NIC + OFED RoCE или в конфигурации AIX RoCE требуются следующие наборы файлов, которые доступны на компакт-диске базовой операционной системы AIX 7 с пакетом обслуживания 7100-03.

devices.ethernet.mlx
: Основной драйвер устройства Converged Ethernet Adapter (mlxentdd) для поддержки конфигурации AIX NIC + OFED RoCE.

devices.pciex.b315506b3157265
: Поддержка пакетов для NGP ITE Converge Ethernet Adapter ASIC2.

devices.pciex.b3155067b3157365
: Поддержка пакетов для NGP ITE Converge Ethernet Adapter ASIC1.

devices.pciex.b315506714101604
Пакет для адаптера Mellanox 2 Ports 10 GbE Converge Ethernet с приемопередатчиками типа SFP+.

devices.pciex.b315506714106104
Пакет для адаптера Mellanox 2 Ports 10 GbE Converge Ethernet, поддерживающего приемопередатчики SFP+.

devices.common.IBM.ib
Драйвер устройства ICM, необходимый для использования конфигурации AIX RoCE.

devices.pciex.b3154a63
Драйвер устройства Mellanox 10 GbE Converge Ethernet Adapter, необходимый для использования конфигурации AIX RoCE.

ofed.core
Набор файлов среды выполнения OFED Core Runtime Environment, который необходим только в том случае, если требуется OFED RDMA.

После обновления существующих наборов файлов AIX RoCE устройства roce и ent могут быть показаны как настроенные. Если оба устройства показаны как настроенные при выполнении команды **lsdev** на адаптерах, выполните следующие действия:

1. Удалите экземпляры *roceX*, связанные с адаптером PCIe2 10 GbE RoCE. Для этого введите следующую команду:
   ```
   # rmdev -dl roce0[, roce1][, roce2,...]
   ```
2. Удалите экземпляры *entX*, связанные с адаптером PCIe2 10 GbE RoCE. Для этого введите следующую команду:
   ```
   # rmdev -dl ent1[,ent2][, ent3...]
   ```
3. Если с адаптером PCIe2 10 GbE RoCE связаны объединенные адаптеры шины хоста (hbaX), удалите их с помощью следующей команды:
   ```
   # rmdev -dl hba0[, hba1][,hba2...]
   ```
4. Запустите администратор конфигурации для применения изменений. Для этого выполните следующую команду:
   ```
   # cfgmgr
   ```

Выполните следующие действия для переключения с конфигурации AIX RoCE на конфигурацию AIX NIC + OFED RoCE:

1. Завершите работу всех приложений RDMA, связанных с адаптером PCIe2 10 GbE RoCE.
2. Удалите или переопределите экземпляры *roceX*. Для этого введите следующую команду:
   - `# rmdev -d -l roce0`
   - `# rmdev -l roce0`

   Команда `rmdev -l roce0` сохраняет определение конфигурации `roce0`, позволяя использовать его в следующий раз при создании экземпляров.
3. Измените значение атрибута stack_type hba с `aix_ib` (AIX RoCE) на `ofed` (AIX NIC + OFED RoCE). Для этого введите следующую команду:
   ```
   # chdev -l hba0 -a stack_type=ofed
   ```
4. Запустите администратор конфигурации, чтобы адаптер шины хоста мог настроить адаптер PCIe2 10 GbE RoCE в качестве адаптера NIC. Для этого выполните следующую команду:
   ```
   # cfgmgr
   ```
5. Убедитесь, что адаптер работает в конфигурации NIC. Для этого введите следующую команду:
   ```
   # lsdev -C -c adapter
   ```

   В следующем примере показаны результаты выполнения команды **lsdev** для адаптера, настроенного в режиме AIX NIC + OFED RoCE:

```
ent1 Available 00-00-01 PCIe2 10GbE RoCE Converged Network Adapter
ent2 Available 00-00-02 PCIe2 10GbE RoCE Converged Network Adapter
hba0 Available 00-00 PCIe2 10GbE RoCE Converged Host Bus Adapter (b315506714101604)
```

Рисунок 45. Пример вывода команды **lsdev** *для адаптера с конфигурацией AIX NIC + OFED RoCE*

Поскольку начиная с AIX 7 с пакетом обслуживания 7100-03 AIX поддерживает также OFED RDMA в режиме AIX NIC, то для включения OFED RDMA необходимо выполнить два следующих дополнительных действия:

1. Установите пакет ofed.core.
2. Задайте режим RDMA для устройств ent1, ent2 с помощью следующей команды:

   ```
   # chdev -l ent1 -a rdma=desired
   # chdev -l ent2 -a rdma=desired
   ```

   Режим RDMA задается до настройки интерфейсов en1 или en2.
3. Режим RDMA можно выключить с помощью следующей команды:

   ```
   # chdev -l ent1 -a rdma=disabled
   # chdev -l ent2 -a rdma=disabled
   ```

AIX RoCE

Адаптер PCIe2 10 GbE RoCE по умолчанию настроен для работы в режиме AIX RoCE. Сеть с поддержкой RDMA обеспечивает более высокую производительность по сравнению с адаптером, используемым как NIC, для приложений, создающих высокую нагрузку на сеть. Этот режим часто эффективен для сетевых хранилищ или высокопроизводительных вычислений.

Для использования конфигурации AIX RoCE требуются следующие библиотеки или интерфейсы:

- Библиотека Direct Access Programming Library (uDAPL), применяемая системой базы данных DB2.
- Интерфейс передачи сообщений (MPI), применяемый в средах высокопроизводительных вычислений (HPC).

На рис. 46 на стр. 692 показан вывод адаптера, работающего в режиме AIX RoCE.

В режиме AIX RoCE доступен только один экземпляр адаптера PCIe2 10 GbE RoCE Adapter, но он может иметь до двух портов. С помощью команды **ibstat** определите число настроенных портов. Для этого выполните следующие действия:

1. Определите, настроено ли расширение ядра `icm`. Для этого введите следующую команду:

   ```
   # lsdev -C | grep icm
   ```
2. Если ядро icm не настроено, настройте его с помощью следующей команды:

   ```
   # mkdev -c management -s infiniband -t icm
   ```
3. Выполните следующую команду **ibstat**:

   ```
   # ibstat roce0
   ```

Несмотря на то что адаптер PCIe2 10 GbE RoCE изначально настроен для работы в режиме AIX RoCE, может потребоваться переключиться из конфигурации AIX NIC + OFED RoCE. Для переключения из конфигурации AIX NIC + OFED RoCE на конфигурацию AIX RoCE выполните следующие действия:

1. Убедитесь, что адаптер работает в режиме AIX NIC + OFED RoCE. Для этого введите следующую команду:

   ```
   # lsdev -C -c adapter
   ```

   Вывод команды **lsdev** аналогичен примеру, приведенному в разделе рис. 45.
2. Остановите передачу трафика TCP/IP и отключите интерфейсы IP с помощью следующей команды:

   ```
   # ifconfig en1 down detach; ifconfig en2 down detach
   ```

3. Удалите или переведите экземпляры NIC в определенное состояние. Для этого введите следующую команду:
   - `# rmdev -d -l ent1; rmdev -d -l ent2`
   - `# rmdev -l ent1; rmdev -l ent2`

   Команда `rmdev -l ent1; rmdev -l ent2` сохраняет определение устройств Ethernet, позволяя использовать их в следующий раз при создании экземпляров.
4. Измените значение атрибута stack_type hba с `ofed` (AIX NIC + OFED RoCE) на `aix_ib` (AIX RoCE) с помощью следующей команды:
   ```
   # chdev -l hba0 -a stack_type=aix_ib
   ```
5. Запустите администратор конфигурации, чтобы адаптер шины хоста мог настроить адаптер PCIe2 10 GbE RoCE в качестве адаптера AIX RoCE. Для этого выполните следующую команду:
   ```
   # cfgmgr
   ```
6. Убедитесь, что адаптер работает в конфигурации AIX RoCE. Для этого введите следующую команду:
   ```
   # lsdev -C -c adapter
   ```

   В следующем примере показаны результаты выполнения команды **lsdev** для адаптеров, если адаптер настроен в режиме AIX RoCE.

```
roce0 Avaliable 00-00-00 PCIe2 10GbE RoCE Converged Network Adapter
hba0 Available 00-00-00 PCIe2 10GbE RoCE Converged Host Bus Adapter (b315506714101604)
```

Рисунок 46. Пример вывода команды **lsdev** *для адаптеров в случае применения конфигурации AIX RoCE.*

Поддержка PCIe3 40 GbE RoCE Adapter

Адаптер PCIe3 40 GbE RDMA Over Converged Ethernet (RoCE) поддерживает удаленный прямой доступ к памяти (RDMA) с помощью OpenFabrics Enterprise Distribution (OFED) в обычном режиме NIC. Если установлено программное обеспечение OpenFabrics, то RDMA поддерживается и включается по умолчанию.

Для загрузки последнего драйвера устройства для этого адаптера выполните следующие действия:

1. Перейдите на веб-сайт IBM (www.ibm.com).
2. Выберите **Поддержка и загрузки**.
3. Загрузите последнее встроенное программное обеспечение в расположение хоста AIX (`/etc/microcode`).
4. Запустите инструмент **diag** для обновления встроенного программного обеспечения, выбрав одну из следующих процедур:
   - Короткая процедура
     a. Введите команду
        ```
        *diag -d entX -T download
        ```

        Примечание: Если устройство Ethernet принадлежит тому же адаптеру шины хоста (например hba0, hba1 и т. д.), то загрузите встроенное ПО на одно из устройств **ent**.
     b. Выберите микрокод, сохраненный в каталоге `/etc/microcode`.
   - Длинная процедура
     a. Введите следующую команду:
        ```
        *diag
        ```
     b. Выберите **Выбор задачи > Задачи микрокода > Загрузить микрокод**.
     c. Выберите **entX**.
     d. Выберите микрокод, сохраненный в каталоге `/etc/microcode`.

Для работы с адаптером PCIe3 40 GbE RoCE Adapter и AIX NIC + OFED RoCE требуются следующие наборы файлов. Они доступны на компакт-диске базовой операционной системы AIX 7 с пакетом обслуживания 7100-03.

devices.ethernet.mlx	Основной драйвер устройства Converged Ethernet Adapter (mlxentdd) для поддержки конфигурации AIX NIC + OFED RoCE.
devices.pciex.b31503101410b504	Пакет для адаптера Mellanox 2 Ports 40 Gb Converged Ethernet Adapter, который использует порты с пассивным подключением по медному кабелю Quad Small Form-factor Pluggable (QSFP).
ofed.core	Набор файлов среды выполнения OFED Core Runtime Environment, который необходим только в том случае, если требуются функции OFED RDMA.

Для выключения функций RDMA введите следующую команду:

```
chdev  -l <устройство-Ethernet> rdma=disabled
```

Пример:

```
# chdev -l ent1 -a rdma=disabled
# chdev -l ent2 -a rdma=disabled
```

Для включения функций RDMA введите следующую команду:

```
chdev  -l <устройство-Ethernet> rdma=desired
```

Примечания

Данная информация была разработана для продуктов и услуг, предлагаемых на территории США.

Компания IBM может не предоставлять в других странах продукты и услуги, обсуждаемые в данном документе. Информацию о продуктах и услугах, распространяемых в вашей стране, вы можете получить в местном представительстве IBM. Ссылки на продукты, программы или услуги IBM не означают, что можно использовать только указанные продукты, программы или услуги IBM. Вместо них можно использовать любые другие функционально эквивалентные продукты, программы или услуги, не нарушающие прав IBM на интеллектуальную собственность. Однако ответственность за проверку действия любых продуктов, программ и услуг других компаний лежит на пользователе.

Компания IBM может обладать заявками на патенты или патентами на предметы обсуждения в данном документе. Обладание данным документом не предоставляет лицензии на эти патенты. Запросы на получение лицензии можно отправлять в письменном виде по адресу:

IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
US

За получением лицензий, имеющих отношение к двухбайтовому набору символов (DBCS), обращайтесь в местное отделение компании IBM по интеллектуальной собственности или направьте запрос в письменной форме по следующему адресу:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

КОМПАНИЯ IBM ПРЕДОСТАВЛЯЕТ НАСТОЯЩУЮ ПУБЛИКАЦИЮ НА УСЛОВИЯХ "КАК ЕСТЬ", БЕЗ КАКИХ-ЛИБО ЯВНЫХ ИЛИ ПОДРАЗУМЕВАЕМЫХ ГАРАНТИЙ, ВКЛЮЧАЯ, НО НЕ ОГРАНИЧИВАЯСЬ ЭТИМ, НЕЯВНЫЕ ГАРАНТИИ СОБЛЮДЕНИЯ ПРАВ, КОММЕРЧЕСКОЙ ЦЕННОСТИ И ПРИГОДНОСТИ ДЛЯ КАКОЙ-ЛИБО ЦЕЛИ. В некоторых юрисдикциях освобождение от явных и подразумеваемых гарантий запрещено в некоторых сделках, поэтому это заявление может к вам не относиться.

Эта информация может содержать технические неточности или типографические ошибки. В информацию периодически вносятся изменения, которые будут учтены во всех последующих изданиях этой книги. IBM может вносить обновления или изменения в этот документ без предварительного уведомления.

Любые ссылки на веб-сайты других компаний приведены в данной публикации исключительно для удобства пользователей и не должны рассматриваться как рекомендация этих веб-сайтов. Материалы, размещенные на этих веб-сайтах, не являются частью информации по данному продукту IBM, и ответственность за применение этих материалов лежит на пользователе.

IBM может использовать и распространять предоставленную вами информацию любым способом без каких-либо обязательств перед вами.

Лицам, обладающим лицензией на данную программу и желающим получить информацию о ней с целью: (i) настройки обмена данными между независимо разработанными программами и другими программами (включая данную) и (ii) использования информации, полученной в результате обмена, этими программами, следует обращаться по адресу:

IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
US

Такая информация может быть предоставлена на определенных условиях, а в некоторых случаях - и за дополнительную плату.

Описанная в этом документе лицензионная программа и все связанные с ней лицензионные материалы предоставляются IBM в соответствии с условиями Соглашения с заказчиком IBM, Международного соглашения о лицензии на программу IBM или любого другого эквивалентного соглашения.

Данные о производительности и примеры клиентов приведены исключительно в иллюстративных целях. Фактические результаты производительности зависят от конкретных конфигураций и рабочих сред.

Информация о продуктах других компаний была получена от поставщиков этих продуктов, их опубликованных материалов или других общедоступных источников. Компания IBM не проверяла эти продукты и не может подтвердить правильность их работы, совместимость или другие заявленные характеристики продуктов других компаний. По вопросам о возможностях продуктов других компаний следует обращаться к поставщикам этих продуктов.

Заявления относительно будущих намерений IBM могут быть изменены или отозваны без дополнительного уведомления и отражают только текущие цели и задачи.

Все указанные цены IBM являются рекомендуемыми розничными ценами IBM на данный момент и могут быть изменены без предварительного уведомления. Цены дилеров могут быть другими.

Данная информация предназначена только для планирования. Она может быть изменена до выпуска описанных в данном документе продуктов.

Настоящая документация содержит примеры данных и отчетов, применяемых в повседневной деятельности компаний. Для большего сходства с реальностью примеры содержат имена людей, названия компаний, товарных знаков и продуктов. Все эти имена и названия вымышленные. Любые совпадения с реально существующими физическими или юридическими лицами совершенно случайны.

Лицензия на авторские права:

Настоящая документация содержит примеры исходного кода программ, иллюстрирующие приемы программирования в различных операционных системах. Вы имеете право копировать, изменять и распространять эти примеры программ в любой форме без уплаты вознаграждения фирме IBM в целях разработки, применения, сбыта или распространения прикладных программ, соответствующих интерфейсу прикладных программ операционной системы, для которой предназначены эти примеры. Эти примеры не были тщательно и всесторонне протестированы. В связи с этим IBM не может гарантировать их надежность, удобство обслуживания и отсутствие ошибок. Примеры программ предоставляются "КАК ЕСТЬ", без каких-либо гарантий. IBM не несет ответственности за ущерб, который может возникнуть в результате использования эти образцов программ.

Во все копии или фрагменты этих примеров программ, а также программы созданные на их основе, следует добавлять следующее замечание об авторских правах:

© (название вашей компании) (год).

Некоторые фрагменты исходного кода получены из примеров программ фирмы IBM Corp.

© Copyright IBM Corp. _год или годы_.

Замечания о правилах работы с личными данными

Продукты IBM Software, включая решения программного обеспечения как услуг, (“Предложения программного обеспечения”) могут использовать cookie или другие технологии для сбора информации об использовании продукта в целях усовершенствования пользовательского интерфейса, для приспособления взаимодействий к конечному пользователю или для других целей. Во многих случаях Предложениями программного обеспечения собирается информация, в которой невозможно опознать персональные данные. Некоторые из наших Предложений программного обеспечения могут позволить вам собирать опознаваемую персональную информацию. Если это Предложение программного обеспечения использует cookie для сбора опознаваемой персональной информации, то специфическая информация об этом использовании cookie в предложении приведена далее.

Это Предложение программного обеспечения не использует cookie или другие технологии для сбора опознаваемой персональной информации.

Если конфигурации, развернутые для этого Предложения программного обеспечения предоставляют вам как клиенту возможность собирать опознаваемую персональную информацию о конечных пользователях посредством cookie и других технологий, вы должны самостоятельно проконсультироваться с юристом о всех законах, применимых к такому сбору данных, включая требования к уведомлению и согласию.

Более подробная информация об использовании различных технологий, включая cookie, для этих целей, приведена в Политике конфиденциальности IBM (http://www.ibm.com/privacy) и Заявлении IBM о конфиденциальности в Интернет (http://www.ibm.com/privacy/details), а также в разделах “Cookies, Web Beacons and Other Technologies” и “IBM Software Products and Software-as-a-Service Privacy Statement” на странице http://www.ibm.com/software/info/product-privacy.

Товарные знаки

IBM, эмблема IBM и ibm.com являются товарными знаками или зарегистрированными товарными знаками International Business Machines Corp. во всем мире. Названия других продуктов и услуг могут быть товарными знаками IBM и других компаний. Текущий список товарных знаков IBM опубликован на веб-странице Copyright and trademark information по следующему адресу: www.ibm.com/legal/copytrade.shtml.

INFINIBAND, InfiniBand Trade Association и дизайнерские знаки INFINIBAND являются товарными или сервисными знаками INFINIBAND Trade Association.

Intel, эмблема Intel, Intel Inside, эмблема Intel Inside, Intel Centrino, эмблема Intel Centrino, Celeron, Intel Xeon, Intel SpeedStep, Itanium и Pentium являются товарными знаками или зарегистрированными товарными знаками Intel Corporation и ее дочерних фирм в США и/или других странах.

Linux является зарегистрированным товарным знаком Линуса Торвальдса в США и других странах.

Microsoft, Windows, Windows NT и эмблема Windows являются товарными знаками Microsoft Corporation в США и/или других странах.

Java и все основанные на Java названия и эмблемы являются товарными знаками или зарегистрированными товарными знаками Oracle и/или дочерних компаний.

UNIX - зарегистрированный товарный знак The Open Group в США и других странах.

Индекс

Спец. символы

Числа

A

B

C

D

E

F

G

H

I

K

L

M

N

P

Q

R

S

T

U

V

W

X

Z

A

Б

В

Г

Д

З

И

К

Л

М

Н

О

П

P

C

Т

У

Ф

Х

Ч

Ш

Э

Напечатано в Дании