



Setting up intermediary services

Note

Before using this information, be sure to read the general information under “Notices” on page 423.

Compilation date: September 17, 2008

© Copyright International Business Machines Corporation 2008.

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

How to send your comments.	xi
Changes to serve you more quickly	xiii
Chapter 1. Overview and new features	1
Introduction: Web servers	1
Chapter 2. Communicating with Web servers	3
Installing IBM HTTP Server	5
Installing Web server plug-ins	6
Selecting a Web server topology diagram and roadmap	9
Plug-ins configuration	16
Web server configuration	24
Configuring a Web server and an application server on separate machines (remote)	28
Configuring multiple Web servers and remote stand-alone application servers	36
Configuring a Web server and an application server profile on the same machine	44
Configuring a Web server and a custom profile on the same machine	53
Configuring a Web server and a deployment manager profile on the same machine	60
Troubleshooting Web server plug-ins installation and removal	66
Web server plug-in response file	76
Editing Web server configuration files	81
Configuring Apache HTTP Server V2.0	82
Configuring Apache HTTP Server V2.2	85
Configuring Lotus Domino	87
Configuring IBM HTTP Server powered by Apache 2.x	90
Configuring IBM HTTP Server Version 6.x	92
Configuring IBM HTTP Server Version 7.0	94
Configuring Microsoft Internet Information Services (IIS)	95
Configuring the Sun Java System Web Server	98
Installing Web server plug-in maintenance	100
Uninstalling the Web server plug-ins for WebSphere Application Server	101
Manually uninstalling Web server plug-ins for WebSphere Application Server	103
Allowing Web servers to access the administrative console	104
Web server plug-in properties	106
Ignore DNS failures during Web server startup	106
Refresh configuration interval	107
Plug-in configuration file name	107
Automatically generate plug-in configuration file	107
Automatically propagate plug-in configuration file	108
Plug-in key store file name	108
Plug-in configuration directory and file name	108
Plug-in key store directory and file name	108
Plug-in logging	108
Web server plug-in request and response optimization properties	109
Web server plug-in caching properties	111
Web server plug-in request routing properties	112
Web server plug-in configuration service property	113
Enable automated Web server configuration processing	114
Application Server property settings for a Web server plug-in	114
Server Role	114
Connection timeout	114
Read/Write timeout	115
Maximum number of connections that can be handled by the Application Server	115

Use extended handshake to check whether application server is running	116
Send the header "100 Continue" before sending the request content	116
Web server plug-in configuration properties	116
Web server plug-in connections	119
Web server plug-in remote user information processing	120
Web server plug-ins	121
Checking your IBM HTTP Server version	121
Creating or updating a global Web server plug-in configuration file	122
Update the global Web server plug-in configuration setting	123
Gskit install images files	124
Plug-ins: Resources for learning	124
Web server plug-in tuning tips	124
Private headers.	126
plugin-cfg.xml file	127
Setting up a local Web server	137
Setting up a remote Web server	139
Web server definition.	142
Editing the Web server type	143
Web server collection	144
Web servers	144
Name	144
Web server type	144
Node	144
Version	145
Status	145
Web server configuration	145
Web server log file	146
Web server custom properties	146
Remote Web server management	147
Web server configuration file	147
Global directives	147
Web server virtual hosts collection.	148
Web server virtual hosts detail	148
Chapter 3. Using the DataPower appliance manager	151
WebSphere DataPower appliance manager overview	152
Adding DataPower appliances to the DataPower appliance manager	156
Modifying DataPower appliance settings	157
Removing a DataPower appliance	157
Appliance collection	158
New appliance settings	160
Appliance settings.	160
Adding new firmware versions to the DataPower appliance manager	164
Modifying settings for firmware versions.	165
Deleting firmware versions.	165
Firmware collection	165
Firmware settings	166
Managed set firmware settings	167
New firmware version settings	167
Adding a new managed set	168
Modifying a managed set	168
Removing a managed set from the DataPower appliance manager.	169
Managed set collection	170
Managed set settings	171
Edit membership for a managed set	174
Modifying DataPower appliance manager settings	175

Appliance manager settings	175
Importing an exported DataPower appliance manager configuration	176
Exporting the DataPower appliance manager configuration	177
Monitoring tasks that DataPower appliance manager is handling	177
Tasks collection	178
Administering managed domain versions	178
Domain history collection	179
Domain history collection	179
Managing versions of sharable appliance settings	180
Settings history collection	181
Settings version Collection	181
Administering DataPower appliance domains	182
Managed domain settings	183
Secure Socket Layer communication with DataPower	184
Chapter 4. Setting up the proxy server	187
Creating a proxy server	188
Proxy server collection	189
Proxy server configuration	190
Proxy server settings	191
Generic server clusters collection	197
Generic server clusters configuration	198
Generic server cluster ports collection	198
Generic server cluster members	198
URI groups	199
URI group configuration	199
Routing rules	199
Routing rules configuration	200
Rewriting rules collection	201
Rewriting rules configuration	202
HTTP proxy inbound channel settings	202
Starting a proxy server	203
Stopping a proxy server	204
HTTP proxy server custom properties	205
SIP proxy server custom properties	206
SIP container custom properties	211
Creating a proxy server cluster	221
Proxy server cluster collection	221
Proxy server cluster settings	222
Proxy server cluster member collection	223
Proxy cluster member settings	224
Proxy cluster member templates collection	225
Proxy cluster member template settings	226
Creating a proxy cluster: Basic proxy cluster settings	227
Creating a proxy cluster: Create first proxy cluster member	227
Creating a proxy cluster: Create additional proxy cluster members	228
Creating a proxy cluster: Summary settings	229
Managing a proxy server cluster	229
Migrating profiles for the proxy server	230
Customizing routing to applications	230
Web module proxy server configuration settings	231
Routing requests to ODC-compliant application servers in other cells	231
Configuring rules to route requests to Web servers	232
Modifying the HTTP endpoints that the proxy server listens on	232
Adding a new HTTP endpoint for the proxy server	233
Setting up caching in the proxy server	233

Static cache rules collection	234
Static cache rule settings	235
Routing requests from a plug-in to a proxy server	236
Creating a proxy server cluster using the wsadmin command	236
Monitoring the proxy server with PMI	238
Monitoring traffic through the proxy server	238
Overview of the custom error page policy	239
Request mapping	240
Session failover in the proxy server	242
Installing a Session Initiation Protocol proxy server.	242
Trusting SIP messages from external domains	243
Tracing a Session Initiation Protocol proxy server	244
High availability and workload management with Session Initiation Protocol proxy server.	245
Load balancing with the Session Initiation Protocol proxy server.	246
SIP proxy settings.	247
SIP external domains collection	251
SIP external domains	251
SIP routing rules collection	252
SIP routing rules set order.	253
SIP routing rules detail	253
SIP rule condition collection	254
SIP rule condition detail.	254
SIP proxy inbound channel detail	255
Troubleshooting the proxy server	256
Troubleshooting request routing and workload management through the proxy server	261
Session Initiation Protocol overload protection	262
Configuring SIP quorum support using the default core group.	264
Configuring the SIP proxy for network outage detection	266
Administering proxy actions	266
Proxy server actions	269
Proxy actions collection.	271
Caching action settings	273
HTTP compression action settings.	273
HTTP header action settings	274
Rewrite action settings	275
Route action settings.	277
Generic server cluster route action settings	278
Time mapping settings	279
Administering custom advisors for the proxy server	280
Custom advisor policies.	280
Custom advisors collection	281
Custom advisor policy settings	281
Configuring stand-alone application server mappings	282
Configuring application server cluster mappings	283
Configuring generic server cluster mappings	285
Creating custom advisors for the proxy server	286
Administering proxy virtual hosts	287
Proxy virtual hosts.	289
Proxy virtual hosts collection	289
Proxy virtual host settings	290
Proxy virtual host settings details	291
Administering proxy rule expressions	291
Proxy rule expressions	292
Proxy rule expressions collection	293
Proxy rule expression settings	294
Creating a custom filter and deploying it to a proxy server	295

Configuring denial of service protection for the proxy server	297
Denial of Service Protection	298
Tuning the security properties for the DMZ Secure Proxy Server for IBM WebSphere Application Server	298
DMZ Secure Proxy Server for IBM WebSphere Application Server start up user permissions	300
DMZ Secure Proxy Server for IBM WebSphere Application Server routing considerations	300
DMZ Secure Proxy Server for IBM WebSphere Application Server administration options	300
Error handling security considerations for the DMZ Secure Proxy Server for IBM WebSphere Application Server	301
Proxy security level properties	301
Configuring a DMZ Secure Proxy Server for IBM WebSphere Application Server using the administrative console	303
Configure secure routing for a DMZ Secure Proxy Server for IBM WebSphere Application Server	305
WebSphere DMZ Secure Proxy Server for IBM WebSphere Application Server	306
Chapter 5. Creating a proxy server cluster	309
Proxy server cluster collection	309
Name	309
Supported Protocols	309
Status	309
Proxy server cluster settings	310
Cluster name	310
Bounding node group name	310
Local Topology tab	311
Proxy server cluster member collection	311
Member name	311
Node	311
Version	311
Status	311
Proxy cluster member settings	312
Member name	312
Run in development mode.	312
Parallel start	312
Start components as needed.	313
Proxy cluster member templates collection.	313
Name	313
Platform	313
Version	313
Description	313
Proxy cluster member template settings.	313
Name	314
Run in development mode.	314
Parallel start	314
Start components as needed.	314
Creating a proxy cluster: Basic proxy cluster settings	314
Proxy cluster name	315
Supported Protocols	315
Creating a proxy cluster: Create first proxy cluster member	315
Member name	315
Select node	315
Generate unique HTTP ports.	315
Select basis for first proxy cluster member:	315
Creating a proxy cluster: Create additional proxy cluster members	316
Member name	316
Select node	316
Generate unique HTTP ports.	317

Creating a proxy cluster: Summary settings	317
Chapter 6. Managing a proxy server cluster	319
Chapter 7. Setting up caching in the proxy server	321
Static cache rules collection	321
URI Groups	322
Disable caching for this URI group.	322
Default expiration	322
Last modified factor	322
Name of the virtual host	322
Static cache rule settings	322
URI groups	323
Disable caching for this URI group.	323
Default expiration	323
Last modified factor	323
Name of the virtual host	323
Chapter 8. Using Session Initiation Protocol to provide multimedia and interactive services	325
SIP in WebSphere Application Server	325
SIP applications	326
SIP industry standards compliance	327
Runtime considerations for SIP application developers	329
SIP IBM Rational Application Developer for WebSphere framework.	330
SIP container	330
SIP converged proxy.	331
SIP high availability	331
SIP session affinity and failover.	332
SIP cluster routing.	336
SIP IP sprayer	337
Configuring the SIP container	337
SIP container custom properties	338
Using DNS procedures to locate SIP servers	347
SIP container settings	349
SIP stack settings	351
SIP timers settings	352
SIP digest authentication settings	354
Configuring SIP timers	356
Configuring digest authentication for SIP	357
Performing controlled failover of SIP applications	357
SIP PMI counters	358
Developing SIP applications	394
Developing a custom trust association interceptor	394
Developing SIP applications that support PRACK	396
Setting up SIP application composition	397
SIP servlets	399
Deploying SIP applications	405
Deploying SIP applications through the console	405
Deploying SIP applications through scripting	406
Securing SIP applications	406
Configuring security for the SIP container	406
Tracing a SIP container.	410
Upgrading SIP applications	410
Replicating SIP sessions	411
Troubleshooting SIP applications	413
Tuning SIP servlets for Linux.	415

SIP timer summary	417
Appendix. Directory conventions	419
Notices	423
Trademarks and service marks	425

How to send your comments

Your feedback is important in helping to provide the most accurate and highest quality information.

- To send comments on articles in the WebSphere Application Server Information Center
 1. Display the article in your Web browser and scroll to the end of the article.
 2. Click on the **Feedback** link at the bottom of the article, and a separate window containing an e-mail form appears.
 3. Fill out the e-mail form as instructed, and click on **Submit feedback** .
- To send comments on PDF books, you can e-mail your comments to: **wasdoc@us.ibm.com** or fax them to 919-254-5250.

Be sure to include the document name and number, the WebSphere Application Server version you are using, and, if applicable, the specific page, table, or figure number on which you are commenting.

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

Changes to serve you more quickly

Print sections directly from the information center navigation

PDF books are provided as a convenience format for easy printing, reading, and offline use. The information center is the official delivery format for IBM WebSphere Application Server documentation. If you use the PDF books primarily for convenient printing, it is now easier to print various parts of the information center as needed, quickly and directly from the information center navigation tree.

To print a section of the information center navigation:

1. Hover your cursor over an entry in the information center navigation until the **Open Quick Menu** icon is displayed beside the entry.
2. Right-click the icon to display a menu for printing or searching your selected section of the navigation tree.
3. If you select **Print this topic and subtopics** from the menu, the selected section is launched in a separate browser window as one HTML file. The HTML file includes each of the topics in the section, with a table of contents at the top.
4. Print the HTML file.

For performance reasons, the number of topics you can print at one time is limited. You are notified if your selection contains too many topics. If the current limit is too restrictive, use the feedback link to suggest a preferable limit. The feedback link is available at the end of most information center pages.

Under construction!

The Information Development Team for IBM WebSphere Application Server is changing its PDF book delivery strategy to respond better to user needs. The intention is to deliver the content to you in PDF format more frequently. During a temporary transition phase, you might experience broken links. During the transition phase, expect the following link behavior:

- Links to Web addresses beginning with `http://` work
- Links that refer to specific page numbers within the same PDF book work
- The remaining links will *not* work. You receive an error message when you click them

Thanks for your patience, in the short term, to facilitate the transition to more frequent PDF book updates.

Chapter 1. Overview and new features

Introduction: Web servers

An application server works with a Web server to handle requests for dynamic content, such as servlets, from Web applications. A Web server uses a Web server plug-ins to establish and maintain persistent HTTP and HTTPS connections with an application server.

The Supported Hardware and Software Web page provides the most current information about supported Web servers.

Chapter 2, “Communicating with Web servers,” on page 3 describes how to set up your Web server and Web server plug-in environment and how to create a Web server definition. The Web server definition associates a Web server with a previously defined managed or unmanaged node. After you define the Web server to a node, you can use the administrative console to perform the following functions for that Web server.

If a Web server is defined to a managed node, you can:

- Check the status of the Web server
- Generate a plug-in configuration file for that Web server.
- Propagate the plug-in configuration file after it is generated.

If the Web server is an IBM HTTP Server and the IBM HTTP Server Administration server is installed and properly configured, you can also:

- Display the IBM HTTP Server Error log (error.log) and Access log (access.log) files.
- Start and stop the server.
- Display and edit the IBM HTTP Server configuration file (httpd.conf).

If the Web server is defined to an unmanaged node, you can:

- Check the status of the Web server
- Generate a plug-in configuration file for that Web server.

You can not propagate an updated plug-in configuration file to a non-IBM HTTP Server that is defined to an unmanaged node. You must manually install an updated plug-in configuration file to a Web server that is defined to an unmanaged node. Web servers defined to an unmanaged node are typically remote Web servers. Remote Web servers are Web servers that are not located on the same machine as the WebSphere Application Server.

After you set up your Web server and Web server plug-in, whenever you deploy a Web application, you must specify a Web server as the deployment target that serves as a router for requests to the Web application. The configuration settings in the plug-in configuration file (plugin-cfg.xml) for each Web server are based on the applications that are routed through that Web server. If the Web server plug-in configuration service is enabled, a Web server plug-in's configuration file is automatically regenerated whenever a new application is associated with that Web server.

Note: Before starting the Web server, make sure you are authorized to run any Application Response Measurement (ARM) agent associated with that Web server.

Refer to your Web server documentation for information on how to administer that Web server. For tips on tuning your Web server plug-in, see “Web server plug-in tuning tips” on page 124.

Chapter 2. Communicating with Web servers

The product works with a Web server to route requests for dynamic content, such as servlets, from Web applications. The Web servers are necessary for directing traffic from browsers to the applications that run in an application server. The Web server plug-in uses the XML configuration file to determine whether a request is for an application server.

Before you begin

- Install your Web server if it is not already installed.

If you want to use the IBM® HTTP Server that is provided with the product, see the *Installing your application serving environment* PDF. Otherwise, see the installation information that is provided with your Web server.

- Verify that your Web server is configured to perform the operations that are required by Web applications, such as GET and POST. Typically, configuring your Web server to perform these operations involves setting a directive in the Web server configuration file. Refer to the Web server documentation for instructions.

If an operation is not enabled when a servlet or JSP file requiring the operation is accessed, an error message displays, such as this one from the IBM HTTP Server:

```
HTTP method POST is not supported by this URL.
```

- Make sure the appropriate plug-in file has been installed on your Web server and the `configureWeb_server_name` script has been run to create and configure the Web server definition for this Web server.

If you are using a distributed platform Web server, use the Plug-in Installation wizard to install the appropriate plug-in file to your Web server. Then run the `configureWeb_server_name` script created by the wizard to create and configure the Web server definition in the WebSphere® configuration repository.

About this task

The following steps are performed during the plug-in installation process. See the Plug-in Installation Roadmap for additional information.

1. A node is created.

An unmanaged node is created when the Web server is on a different computer from the Application Server. An unmanaged node is a node that does not have a node agent running on it. Using unmanaged nodes, the product can represent servers that are not application servers within its configuration topology. This representation enables connection information between those servers and application servers to be maintained. The *Using the administrative clients* PDF describes how to create a node.

2. A Web server definition is created.

You can also use either the administrative console or use the `ConfigureWebServerDefintion.jacl` script to create a Web server definition. If you use the administrative console:

- a. Select the node that was created in the preceding step, and in the Server name field, enter the local name of the Web server for which you are creating a Web server definition.
- b. Use the wizard to complete the Web server definition.

3. An application or modules are mapped to a Web server. If an application that you want to use with this Web server is already installed, the application is automatically mapped to the Web server. If the application is not installed, select this Web server during the Map modules to servers step of the application installation process.

4. The master repository is updated and saved.

When you install a plug-in, the configuration file for that plug-in is automatically created. You can change or fine tune the default settings for the properties in this configuration file. If change any of the settings, you must regenerate the file before your changes take affect.

Generating or regenerating the configuration file might take a while to complete. After it finishes, all objects in the administrative cell use their newest settings, which the Web server can access. If the Application Server is on the same physical machine as the Web server, the regeneration usually takes about 30 to 60 seconds to complete. The regeneration takes longer if they are not both on the same machine.

The following procedure describes the steps for updating the plug-in configuration file, including configuring for SSL and Web server tuning

1. Use the administrative console to change the settings in the plug-in configuration file.

When setting up your Web server plug-in, you must decide whether or not to have the configuration automatically generated in response to a configuration change. When the Web server plug-in configuration service is enabled and any of the following conditions occur, the plug-in configuration file is automatically generated:

- When the Web server is created or saved.
- When an application is installed.
- When an application is uninstalled.
- When the virtual host definition is updated

Note: When the plug-in configuration file is first generated, it does not include `admin_host` on the list of virtual hosts. The *Installing your application serving environment* PDF describes how to add it to the list.

You can either use the administrative console, or issue the `GenPluginCfg` command to regenerate your `plugin-cfg.xml` file. To use the administrative console:

- a. Select **Servers > Server Types > Web Servers > `web_server_name` > plug-in properties**.
- b. Select **Automatically generate plug-in configuration file** or click one or more of the following topics to manually configure the `plugin-cfg.xml` file:
 - Caching
 - Request and response
 - Request routing
 - Custom Properties

Web server plug-in configuration properties maps each property to one of these topics.

Note: It is recommended that you do not manually update the `plugin-cfg.xml` file. Any manual updates you make for a given Web server are overridden whenever the `plugin-cfg.xml` file for that Web server is regenerated.

- c. Click **OK**.
 - d. You might have to stop the application server and then start the application server again to enable the Web server to locate the `plugin-cfg.xml` file.
2. If you want to use Secure-socket layer (SSL) with this configuration, use the plug-in's installation wizard to install the appropriate GSKIT installation image file on your workstation.
 3. Tune your Web server. See the *Tuning guide* PDF for more information.
 4. Propagate the plug-in configuration. The plug-in configuration file (`plugin-cfg.xml`) is automatically propagated to the Web server if the Web server plug-in configuration service is enabled, and one of the following is true:
 - The Web server is a local Web server, which means that the Web server is located on the same machine as an application server.
 - The Web server is a remote IBM HTTP Server Version 7 that has a running IBM HTTP Server administration server.

If neither of these conditions is true, the plugin-cfg.xml file must be manually copied to the remote Web server's installation location. Copy plugin-cfg.xml in `<WASROOT>/profiles/<profilename>/config/cells/././nodes/./servers/<webservername>` to the Web server host location, which is `<PluginInstallRoot>/config/<webservername>/`.

Note: If you use the FTP function to perform the copy, and the configuration reload fails, check the file permissions on the plugin-cfg.xml file and make sure they are set to `rw-r--r--`. If the file permissions are not correct, the Web server is not able to access the new version of the file, which causes the configuration reload to fail.

If the file permissions are incorrect, issue the following command to change the file permissions to the appropriate settings:

```
chmod 644 plugin-cfg.xml
```

AIX The AIX® FTP function does not preserve file attributes. Therefore, if you need to manually copy the plugin-cfg.xml from an AIX operating system, you might want to use the AIX RCP function instead of the FTP function to copy the file.

The remote Web server installation location is the location you specified when you created the node for this Web server.

Results

The configuration is complete. To activate the configuration, stop and restart the Web server. If you encounter problems restarting your Web server, check the `http_plugin.log` file for information on what portion of the plugin-cfg.xml file contains an error. The log file states the line number on which the error occurred along with other details that might help you diagnose why the Web server did not start. You can then use the administrative console to update the plugin-cfg.xml file.

If applications are infrequently installed or uninstalled, which is usually the situation in a production environment, or if you can tolerate the performance impact of generating and distributing the plug-in configuration file each time any of the previously listed actions occur, you should consider enabling this service.

If you are making a series of simultaneous changes, like installing numerous applications, you might want the configuration service disabled until after you make the last change. The Web server plug-in configuration service is enabled by default. To disable this service, in the administrative console click **elect Servers > Server Types > WebSphere application servers > server_name > administration services > Web server plug-in configuration service**, and then unselect the **Enable automated Web server configuration processing** option.

Note: If your installation uses a firewall, make sure you configure the Web server plug-in to use a port that has been opened. See your security administrator for information on how to obtain an open port.

Installing IBM HTTP Server

This topic describes how to install IBM HTTP Server.

Before you begin

IBM HTTP Server on distributed platforms. Install the IBM HTTP Server product and its plug-in, or install a plug-in for another supported Web server to enable the Web server to work with WebSphere Application Server.

To use a Web server other than IBM HTTP Server, install and configure the Web server before or after installing the WebSphere Application Server product, but before installing the Web server plug-ins for WebSphere Application Server.

About this task

Refer to the Information center for IBM HTTP Server for detailed information on installation steps, configuring the Web server for SSL, the Fast Response Cache Accelerator, or Apache directives.

What to do next

After installing the Web server and the Application Server, install the appropriate Web server plug-in for a supported, installed Web server. No further configuration is required for most Web servers.

The Plug-ins installation wizard configures supported Web servers. You can also manually configure supported Web servers for WebSphere Application Server as described in “Editing Web server configuration files” on page 81.

Related tasks

Chapter 2, “Communicating with Web servers,” on page 3

The product works with a Web server to route requests for dynamic content, such as servlets, from Web applications. The Web servers are necessary for directing traffic from browsers to the applications that run in an application server. The Web server plug-in uses the XML configuration file to determine whether a request is for an application server.

Installing Web server plug-ins

It is possible to configure your Web server plug-in to route requests to WebSphere Application Server Version 4.x, Version 5.x, and Version 6.x releases. This topic describes configuring Web server plug-ins to route requests to WebSphere Application Server Version 7.0.

Before you begin

Go to <http://www-1.ibm.com/support/docview.wss?rs=180&context=SSEQTP&uid=swg21160581> for information about how to verify what Version 4.0, Version 5.0, Version 5.1, Version 6.0, Version 6.1, and Version 7.0 plug-in versions are installed on local or remote Web servers, and how to determine if the installation complies with supported configurations.

You must install a supported Web server before you can install a plug-in for the Web server. If the Web server is not already installed, you cannot install the plug-in for it. If the WebSphere Application Server product is not installed, you can install the plug-in. To create a Web server configuration for unmanaged nodes, WebSphere Application Server must be installed on your system.

The Plug-ins installation wizard installs the plug-in module, configures the Web server for communicating with the application server, and creates a Web server configuration definition in the application server, if possible.

Some topologies, such as the Web server on one machine and the application server on another machine, prevent the Plug-ins installation wizard from creating the Web server definition in the application server configuration on the remote machine. In such a case, the Plug-ins installation wizard creates a script that you can copy to the application server machine. Run the script to create the Web server configuration definition within the application server configuration.

About this task

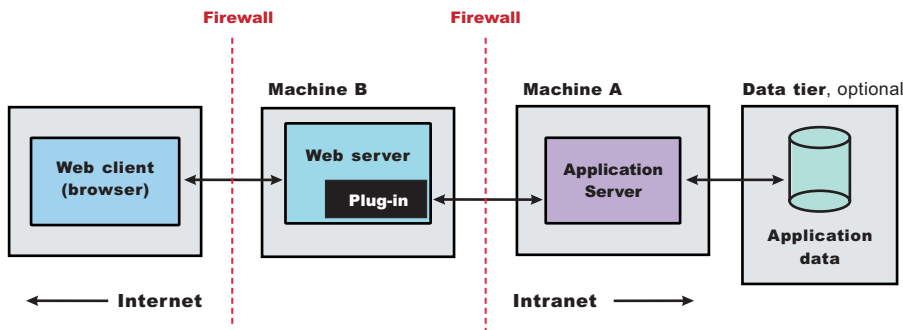
This topic describes installing a Web server plug-in for WebSphere Application Server. WebSphere Application Server products supply a unique binary plug-in module for each supported Web server. The plug-in configuration file, which the WebSphere Application Server products create and maintain, interacts with the binary module to provide information about the application server configuration to the Web server. The Web server uses the information to determine how to communicate with the application server, but to locate specific applications on the application server.

The Plug-ins installation wizard installs required files and configures the Web server and the application server to allow communication between the servers.

Select one of the following topology scenarios and follow the steps below the diagram to install the plug-in and configure both the Web server and the application server.

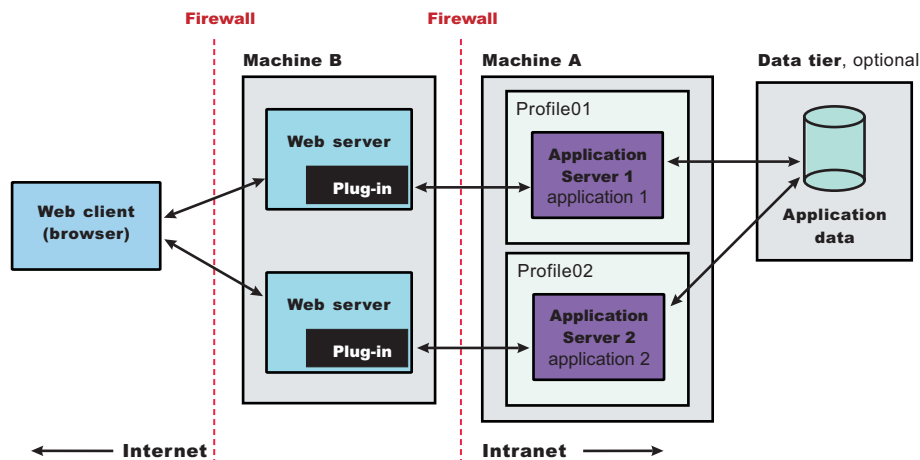
When multiple profiles exist, the plug-ins installer configures only the default profile. See “Plug-ins configuration” on page 16 for a description of the flow of logic that determines how the installer selects the profile to configure.

- **Scenario 1: Remote** The application server and the Web server are on separate machines or logical partitions.



See “Configuring a Web server and an application server on separate machines (remote)” on page 28 for the procedure that explains how to create this Web server topology.

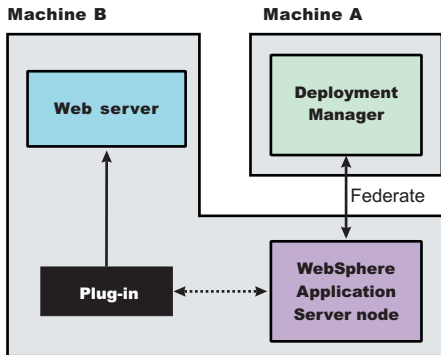
- **Scenario 2: Remote** Multiple stand-alone application servers are on one machine, and each application server has a dedicated Web server on a separate machine or logical partition.



See “Configuring multiple Web servers and remote stand-alone application servers” on page 36 for the procedure that explains how to create this Web server topology.

- **Scenario 3: Local Application Server profile** The application server and the Web server are on a single machine or logical partition.

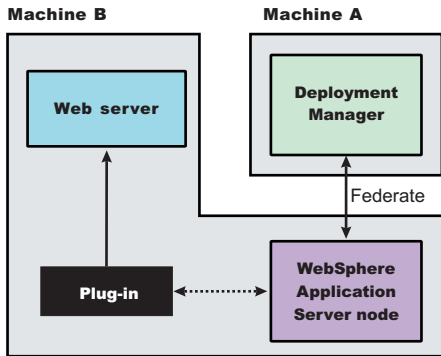
A local distributed installation includes the Web server plug-in, the Web server, and a *managed* application server on the same machine:



See “Configuring a Web server and an application server profile on the same machine” on page 44 for the procedure that explains how to create this Web server topology for an application server profile.

- **Scenario 4: Local custom profile** A managed node and the Web server are on the same machine or logical partition.

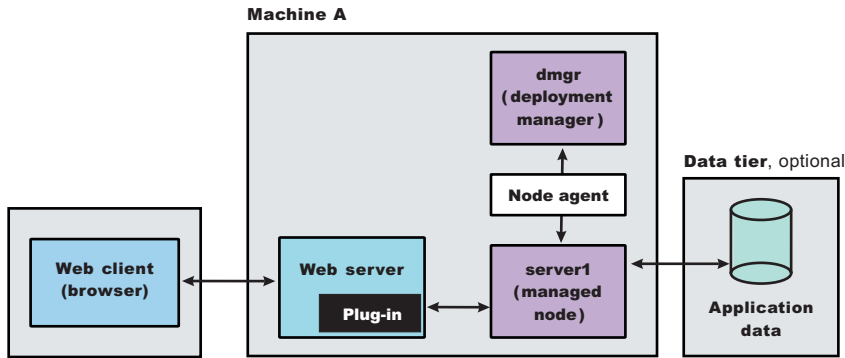
A local distributed installation includes the Web server plug-in, the Web server, and the managed custom node on the same machine:



See “Configuring a Web server and a custom profile on the same machine” on page 53 for the procedure that explains how to create this Web server topology for a federated custom profile.

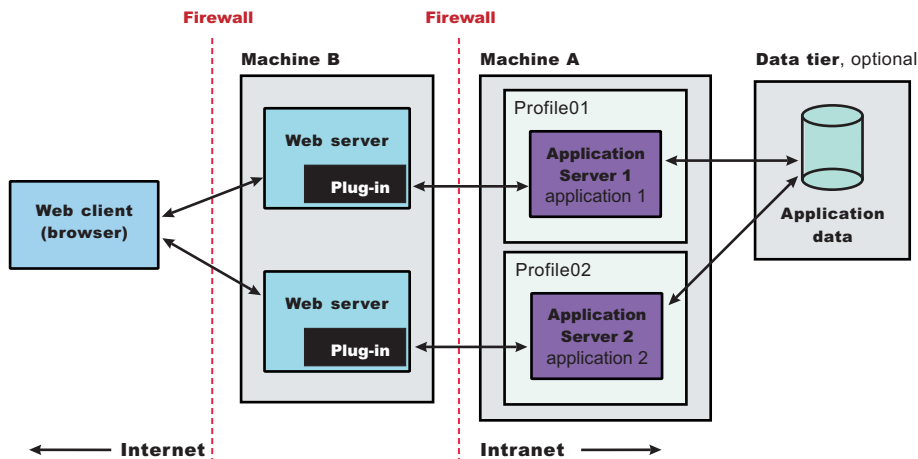
- **Scenario 5: Local deployment manager profile** A deployment manager node and the Web server are on a single machine or logical partition.

A local distributed installation includes the Web server plug-in, the Web server, and the application server on the same machine:



See “Configuring a Web server and a deployment manager profile on the same machine” on page 60 for the procedure that explains how to create this Web server topology for a deployment manager profile.

- **Scenario 6: Non-default profile** Creating a Web server definition for a profile that is not the default profile.



Results

You can install a Web server and the Web server plug-ins for various stand-alone application server topologies by following the procedures described in this topic.

What to do next

See “Selecting a Web server topology diagram and roadmap” for an overview of the installation procedure.

See “Web server configuration” on page 24 for more information about the files involved in configuring a Web server.

See Editing Web server configuration files for information about how the Plug-ins installation wizard configures supported Web servers.

Selecting a Web server topology diagram and roadmap

Install and configure Web server plug-ins for WebSphere Application Server to allow the application server to communicate with the Web server.

Before you begin

The primary production configuration for a Web server is an application server on one machine and a Web server on a separate machine. This configuration is referred to as a *remote* configuration. Contrast the remote configuration to the local configuration, where the application server and the Web server are on the same machine.

About this task

The Plug-ins installation wizard has four main tasks:

- Installs the binary plug-in module on the Web server machine.
- Configures the Web server configuration file on the Web server machine to point to the binary plug-in module and to the XML configuration file for the binary module.
- Installs a temporary XML configuration file for the binary module (plugin-cfg.xml) on the Web server machine in remote scenarios.
- Creates the configuration for a Web server definition on the application server machine. The wizard processes the creation of the Web server definition differently depending on the scenario:

Web server plug-in installation for stand-alone application server environments

- Recommended remote stand-alone application server installation:

Creates a configuration script that you run on the application server machine. Install the Web server and its plug-in on a different machine than the application server. This configuration is recommended for a production environment.

- Local stand-alone application server installation:

Detects the default profile on a local application server machine and creates the Web server definition for it directly. Install the Web server and its plug-in on the same machine with the application server. This configuration is for development and test environments.

Web server plug-in installation for distributed environments (cells)

- Recommended remote distributed installation:

Creates a configuration script that you run on the application server machine. Install the Web server and its plug-in on a different machine than the deployment manager or managed node. This configuration is recommended for a production environment.

- Local distributed installation:

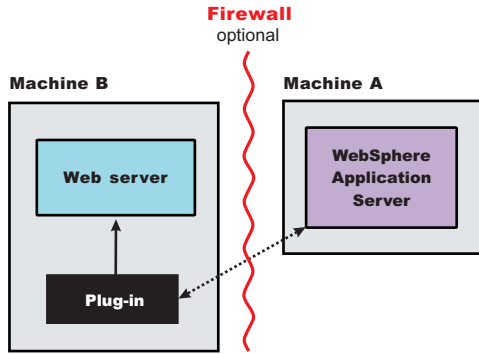
Creates a configuration script that you run when the deployment manager is running. Install the Web server and its plug-in on the same machine with the deployment manager or a managed node. This configuration is for development and test environments.

Select a link to go to the appropriate steps in the following procedure.

- **Set up a remote Web server installation.**

The remote Web server configuration is recommended for production environments.

The remote installation installs the Web server plug-in on the Web server machine when the application server is on a separate machine, such as shown in the following graphic:



Remote installation scenario

Table 1. Installation and configuration

Step	Machine	Task
1	A	Install the WebSphere Application Server Network Deployment product. Read the "Installing the product and additional software" topic.
2	A	Configure a stand-alone application server profile. See Creating an application server profile.
3	B	Install IBM HTTP Server or another supported Web server.
4	B	Install the binary plug-in module using the Plug-ins installation wizard. See "Configuring a Web server and an application server on separate machines (remote)" on page 28. The script for creating and configuring the Web server is created under the <i>plugins_root</i> /bin directory.
5	B	Copy the <code>configureweb_server_name</code> script to Machine A. See "Configuring a Web server and an application server on separate machines (remote)" on page 28 for information about cross-platform scripts and file encoding differences.
6	A	Paste the <code>configureweb_server_name</code> script from Machine B to the <i>app_server_root</i> /bin directory on Machine A. See "Configuring a Web server and an application server on separate machines (remote)" on page 28.
7	A	Start the application server, then run the script from a command line.
8	A	Verify that the application server is running. Open the administrative console and save the changed configuration.
9	B	AIX HP-UX Linux Solaris Source the <i>plugins_root</i> / <code>setupPluginCfg.sh</code> script for a Domino® Web Server before starting a Domino Web server. Otherwise, start the supported Web server.
10	B	Run the snoop servlet. Access the following URL in your browser: <code>http://host_name_of_machine_B:http_transport_port/snoop</code> To verify with your own application, regenerate and propagate the <code>plugin-cfg.xml</code> file after installing the application.

Regeneration of the plugin-cfg.xml file

The Web server plug-in configuration service regenerates the `plugin-cfg.xml` file automatically.

During the installation of the plug-ins, the temporary `plugin-cfg.xml` file is installed on Machine B in the *plugins_root* /`config` /`web_server_name` directory. To use the actual `plugin-cfg.xml` file from the application server, propagate the `plugin-cfg.xml` file as described in the next section.

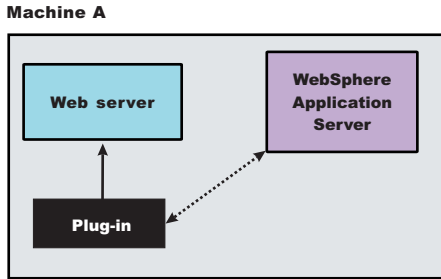
Propagation of the plugin-cfg.xml file

The Web server plug-in configuration service propagates the plugin-cfg.xml file automatically for IBM HTTP Server Version 6.0 or later. For all other Web servers, propagate the plug-in configuration file manually. Copy the plugin-cfg.xml file from the *profile_root/config/cells/cell_name/nodes/web_server_name_node/servers/web_server_name* directory on Machine A. Paste the file into the *plugins_root/config/web_server_name* directory on Machine B.

- **Set up a local Web server configuration.**

The local Web server configuration is recommended for a development or test environment.

A local installation includes the Web server plug-in, the Web server, and the Application Server on the same machine:



Local installation scenario

Table 2. Installation and configuration

Step	Machine	Task
1	A	Install your WebSphere Application Server product. Read the "Installing the product and additional software" topic.
2	A	Create an application server profile. See Creating profiles using the graphical user interface.
3	A	Install IBM HTTP Server or another supported Web server.
4	A	Install the binary plug-in module using the Plug-ins installation wizard. See "Configuring a Web server and an application server profile on the same machine" on page 44. The Web server definition is automatically created and configured during the installation of the plug-ins.
5	A	Verify that the application server is running. Open the administrative console and save the changed configuration.
6	A	<div style="display: flex; justify-content: space-around; margin-bottom: 5px;"> AIX HP-UX Linux Solaris </div> Run the <i>plugins_root/setupPluginCfg.sh</i> script for a Domino Web Server before starting a Domino Web server. Start the Web server.
7	A	Run the snoop servlet. Access the following URL in your browser: <i>http://host_name_of_machine_A:http_transport_port/snoop</i> To verify with your own application, regenerate and propagate the plugin-cfg.xml file after installing the application.

Regeneration of the plugin-cfg.xml file

The Web server plug-in configuration service regenerates the plugin-cfg.xml file automatically.

The plugin-cfg.xml file is generated in the *profile_root/config/cells/cell_name/nodes/web_server_name_node/servers/web_server_name* directory. The generation occurs when the Web server definition is created.

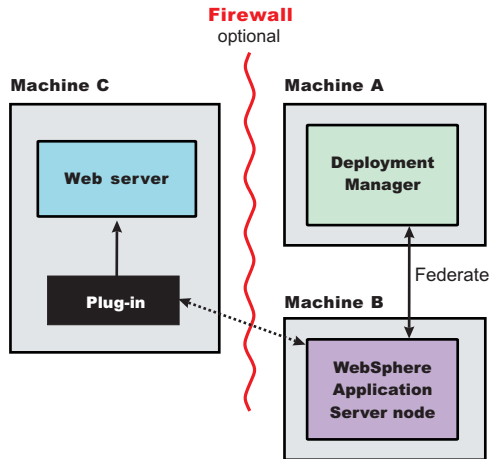
Propagation of the plugin-cfg.xml file

The local file does not require propagation.

- **Set up a remote Web server installation in a cell.**

The remote Web server configuration is recommended for production environments.

The remote distributed installation installs the Web server plug-in on the Web server machine when the application server is on a separate machine, such as shown in the following graphic:



Remote distributed installation scenario

Table 3. Installation and configuration

Step	Machine	Task
1	A	Install WebSphere Application Server Network Deployment. Read the "Installing the product and additional software" topic.
2	A	Create a deployment manager profile. See Creating profiles using the graphical user interface.
3	A	Start the deployment manager with the <code>profile_root/bin/startManager.sh</code> command or its Windows® equivalent.
4	B	Install WebSphere Application Server Network Deployment. Read the "Installing the product and additional software" topic.
5	B	Create an application server profile. See Creating profiles using the graphical user interface.
6	B	Federate the node with the <code>profile_root/bin/addNode.sh dmgrhost 8879 -includeapps</code> command or its Windows equivalent. Federating the node starts the nodeagent process, which is required to be running during this configuration.
7	C	Install IBM HTTP Server or another supported Web server.
8	C	Install the binary plug-in module using the Plug-ins installation wizard. See "Configuring a Web server and an application server on separate machines (remote)" on page 28. The script for creating and configuring the Web server is created under the <code>plugins_root/bin</code> directory.
9	C	Copy the <code>configureweb_server_name</code> script to Machine A. If one machine is running under an operating system such as AIX or Linux® and the other machine is running under Windows, copy the script from the <code>plugins_root/bin/crossPlatformScripts</code> directory. See "Configuring a Web server and an application server on separate machines (remote)" on page 28 for information about cross-platform scripts and file encoding differences.

Table 3. Installation and configuration (continued)

Step	Machine	Task
10	A	Paste the configureweb_server_name script from Machine C to the app_server_root/bin directory on Machine A.
11	A	Start the node agent and the deployment manager if they are not already running, then run the script from a command line. If you have enabled security or changed the default JMX connector type, edit the script and include the appropriate parameters on the wsadmin command.
12	A/B	Use the administrative console of the deployment manager on Machine A to start the application server on Machine B. Wait for synchronization to occur and save the new configuration.
13	C	AIX HP-UX Linux Solaris Run the plugins_root/setupPluginCfg.sh script for a Domino Web Server before starting a Domino Web server. Otherwise, start the Web server.
14	C	Run the snoop servlet. Access the following URL in your browser: <code>http://host_name_of_machine_C:http_transport_port/snoop</code> To verify with your own application, regenerate and propagate the plugin-cfg.xml file after installing the application.

Regeneration of the plugin-cfg.xml file

The Web server plug-in configuration service regenerates the plugin-cfg.xml file automatically.

During the installation of the plug-ins, the temporary plugin-cfg.xml file is installed on Machine B in the plugins_root/config/web_server_name directory.

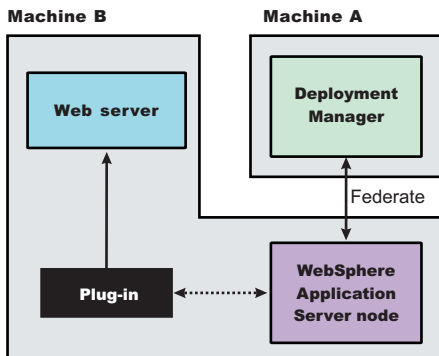
Propagation of the plugin-cfg.xml file

The Web server plug-in configuration service propagates the plugin-cfg.xml file automatically for IBM HTTP Server Version 6.0 or later. For all other Web servers, propagate the plug-in configuration file, by manually copying the plugin-cfg.xml file from the profile_root/config/cells/cell_name/nodes/node_name/servers/web_server_name directory on Machine A to the plugins_root/config/web_server_name directory on Machine C.

- **Set up a local distributed Web server configuration.**


The local Web server configuration is recommended for a development or test environment.

A local distributed installation includes the Web server plug-in, the Web server, and the managed application server on the same machine:



Local distributed installation scenario

Table 4. Installation and configuration

Step	Machine	Task
1	A	Install WebSphere Application Server Network Deployment. Read the "Installing the product and additional software" topic..
2	A	Create a deployment manager profile. See Creating profiles using the graphical user interface.
3	A	Start the deployment manager with the <i>profile_root/bin/startManager.sh</i> command or its Windows equivalent.
4	B	Install WebSphere Application Server Network Deployment. Read the "Installing the product and additional software" topic.
5	B	Create an application server profile. See Creating profiles using the graphical user interface.
6	B	Federate the node with the <i>profile_root/bin/addNode.sh dmgrhost 8879 -includeapps</i> command or its Windows equivalent. Federating the node starts the nodeagent process, which is required to be running during this configuration.
7	B	Install IBM HTTP Server or another supported Web server.
8	B	Install the binary plug-in module using the Plug-ins installation wizard. See "Configuring a Web server and an application server profile on the same machine" on page 44. The script for creating and configuring the Web server is created in the <i>plugins_root/bin</i> directory.
11	B	After verifying that the deployment manager and the node agent are running on Machine A, run the <i>configureweb_server_name</i> script from a command line in the <i>plugins_root/bin</i> directory on Machine B. If you have enabled security or changed the default JMX connector type, edit the script and include the appropriate parameters.
12	A/B	Use the administrative console of the deployment manager on Machine A to start the application server on Machine B. Wait for synchronization to occur and save the new configuration.
13	B	 Run the <i>plugins_root/setupPluginCfg.sh</i> script for a Domino Web Server before starting a Domino Web server. Otherwise, start the Web server.
14	B	Run the snoop servlet. Access the following URL in your browser: <i>http://host_name_of_machine_B:http_transport_port/snoop</i>

Regeneration of the plugin-cfg.xml file

The Web server plug-in configuration service regenerates the plugin-cfg.xml file automatically.

During the installation of the plug-ins, the temporary plugin-cfg.xml file is installed on Machine B in the *plugins_root/config/web_server_name* directory.

The plugin-cfg.xml file is generated at the location *profile_root/config/cells/cell_name/nodes/node_name/servers/webServerName* directory, when the Web server definition is created.

Regenerate the plugin-cfg.xml file in the Web server definition in the application server whenever the configuration changes. The Web server has immediate access to the file whenever it is regenerated.

When the Web server plug-in configuration service (an administration service) is enabled on Machine A, the plugin-cfg.xml file is automatically generated for all Web servers.

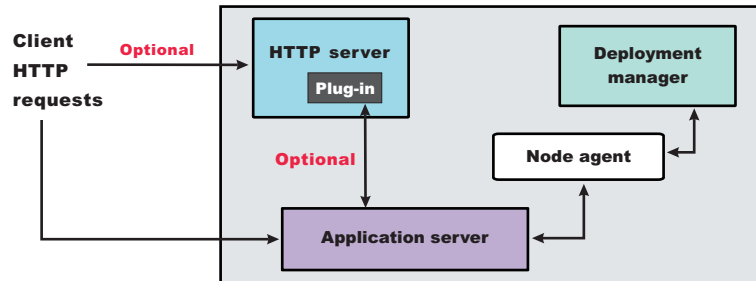
Propagation of the plugin-cfg.xml file

Node synchronization is used to propagate the plugin-cfg.xml file from Machine A to Machine B.

When the Web server plug-in configuration service (an administration service) is enabled on Machine A, the plugin-cfg.xml file is automatically propagated for all Web servers.

Alternate configuration

This procedure describes installing the plug-ins on two machines. However, you can perform this procedure on a single machine as shown in the following graphic. A local distributed installation also includes the Web server plug-in, the Web server, the application server, and the deployment manager on the same machine:



Results

You can set up a remote or local Web server by installing Application Server, the Web server, and then the Web server plug-ins.

What to do next

See “Web server configuration” on page 24 for more information about the files involved in configuring a Web server.

See “Plug-ins configuration” for information about the logic behind the processing scenarios for the Plug-ins installation wizard.

See “Editing Web server configuration files” on page 81 for information about how the Plug-ins installation wizard configures supported Web servers.

See “Installing Web server plug-ins” on page 6 for information about other installation scenarios for installing Web server plug-ins.

Plug-ins configuration

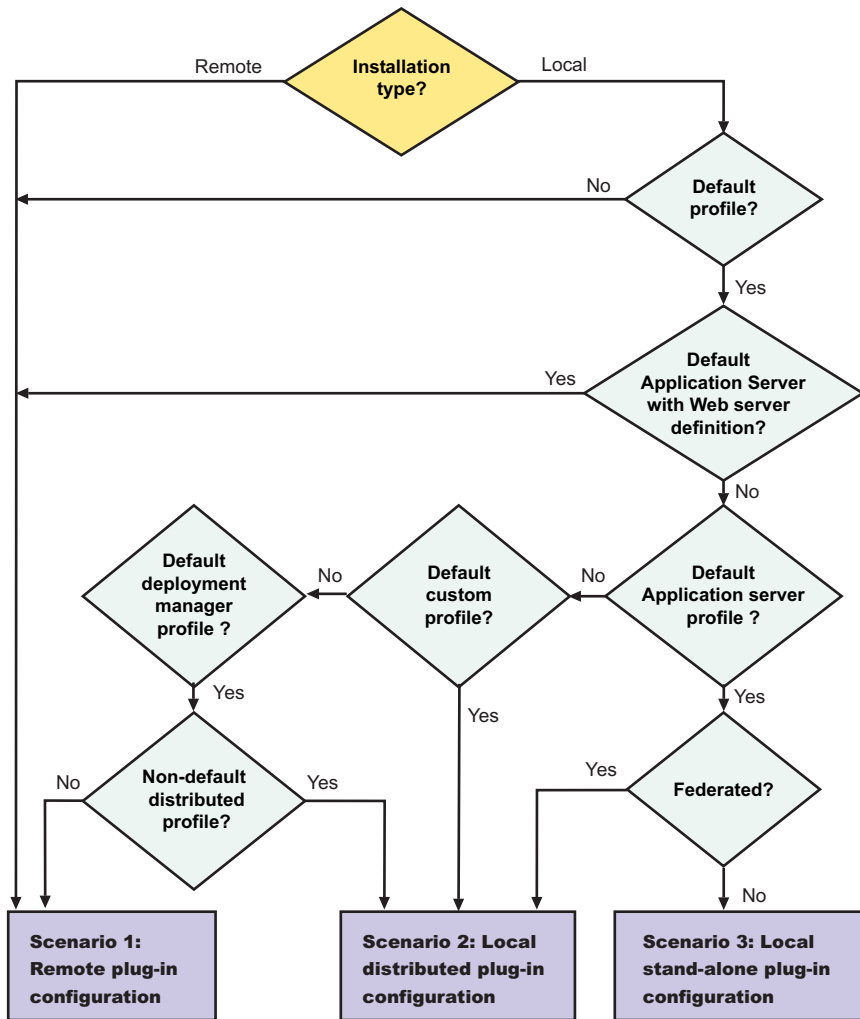
The Plug-ins installation wizard installs a binary plug-in module and a plug-in configuration file for the Web server. The wizard then configures the supported Web server for the Application Server and creates a Web server definition in the configuration of the application server. This overview shows the different processing paths that the wizard uses.

This topic describes the three ways that the Plug-ins installation wizard configures a Web server to locate the plugin-cfg.xml file, which is the plug-in configuration file.

Configuration flows for the Network Deployment product

The Plug-ins installation wizard resolves all configurations of Web server and WebSphere Application Server to three scenarios: a remote application server, local distributed application Server, and local stand-alone Application Server. The logic implemented in determining which scenario applies to a configuration is shown in the following diagram.

Web server plug-ins for WebSphere Application Server



Legend:

Default application server with Web server definition?

If the default profile is an application server with an existing Web server definition, then the installation is considered a remote installation. You cannot have more than one Web server definition in a stand-alone application server.

Use the same name for the Web server to configure a new Web server to use the existing Web server definition.

Use a different Web server name to create a script that creates a new Web server definition in a federated application server profile.

Default profile?

If the product is installed but the Profile Management Tool has not yet created a profile, the scenario is considered to be a remote installation. When multiple profiles exist, the plug-ins installer configures only the default profile.

Default *profile_type*?

The Plug-ins installation wizard can configure only one profile at a time. The wizard always works with the default profile. These three paths show how processing varies for different types of profiles.

Federated?

If the application server node is federated, the Plug-ins installation wizard configures the Web server definition on the managed node. This has advantages. Suppose the Web server and the managed node are on a separate machine. The `plugin-cfg.xml` file is automatically propagated to the remote node during node synchronization because the Web server definition is part of the node configuration.

Installation type?

The installation type is either remote or local.

Non-default distributed profile?

If the deployment manager has a federated custom node (custom profile), the Plug-ins installation wizard configures the Web server definition on the managed node. This has advantages. Suppose the Web server and the managed node are on a separate machine. The `plugin-cfg.xml` file is automatically propagated to the remote node during node synchronization because the Web server definition is part of the node configuration.

If a federated custom profile is not found, the Plug-ins installation wizard looks for and configures the first federated application server node (application server profile) that it finds. So, the logic is:

1. Look for a federated managed (custom) profile and configure the first one found.
2. If no federated managed profile is found, look for a federated application server profile and configure the first one found.

Scenario 1. Remote plug-in configuration

The Plug-ins installation wizard does not automatically create a Web server definition within the default distributed profile on a remote machine. The wizard creates the `configureweb_server_name` script instead.

The Plug-ins installation wizard configures the Web server to use the `plugin-cfg.xml` file that will be maintained on the Web server machine in the `plugins_root/config/web_server_name` directory. This file requires periodic propagation. Propagation is copying the current `plugin-cfg.xml` file from the Application Server machine to replace the `plugins_root/config/web_server_name/plugin-cfg.xml` file.

After installing the binary plug-in for the local Web server, you do not have to run the script before you can start the application server and the Web server. However, you do not have the benefits of a Web server definition in the application server node until you run the script.

Four configurations qualify for the remote application server scenario:

Profile type	Federation status	Creation of Web server definition?	Web server already defined in Application Server configuration?
Any profile anywhere if you select a remote installation type in the Plug-ins installation wizard	N/A	By script	N/A
No default profile detected	N/A	By script	N/A
Default unfederated stand-alone application server profile with an existing Web server definition	Not federated	By script	Yes
Default deployment manager profile with no managed nodes	N/A	By script	N/A

Testing the application server without a Web server definition: The following overview shows the procedure for verifying the temporary `plugins_root/config/web_server_name/plugin-cfg.xml` file.

The Web server communicates with the remote Application Server using the temporary `plugin-cfg.xml` file.

If the application server has an HTTP Transport port assignment other than 9080, the test is not successful. Continue to the next section to create the Web server definition on the application server and complete your test of the configuration.

1. Start the Web server with the proper procedure for your Web server.

For example, start the IBM HTTP Server from a command line:

- `AIX HP-UX Linux Solaris ./IHS_root/bin/apachectl start`
- `Windows IHS_root\bin\apache`

2. Start the application server on the remote machine.

Change directories to the *profile_root/bin* directory and run the `startServer` command:

- `AIX HP-UX Linux Solaris ./profile_root/bin/startServer.sh server1`
- `Windows profile_root\bin\startServer server1`

3. Point your browser to `http://localhost:9080/snoop` to test the internal HTTP transport provided by the application server. Point your browser to `http://Host_name_of_Web_server_machine/snoop` to test the Web server plug-in.
4. Verify that both Web addresses display the Snoop Servlet - Request/Client Information page.

Completing the installation by configuring a Web server definition: The following overview shows the procedure for completing the configuration. The configuration is not complete until the Web server definition exists in the configuration of the Application Server node. The Web server definition is a central element in the regeneration of a valid plug-in configuration file, `plugin-cfg.xml`.

1. Start the deployment manager if you are configuring the deployment manager or a managed node.
2. Federate a remote application server node or custom node now if you are planning to federate the node at some point. If a Web server definition already exists when you federate a node, the definition is lost.
3. Create the Web server definition in the application server. You have two options for a managed node. Use the script option for a deployment manager node without managed nodes.
 - Use the administrative console of the deployment manager to create a Web server definition for a managed node. Click **Servers > Web servers > New** and use the Create new Web server entry wizard to create the Web server definition.
 - Run the script to manually create the Web server definition within the configuration of the Application Server node:
 - a. Copy the script from the *plugins_root/bin* directory to the remote *app_server_root/bin* directory.
 - b. Open a command window and run the script:

- `AIX HP-UX Linux Solaris ./configureweb_server_name.sh`
- `Windows configureweb_server_name.bat`

Note: The *webserverNodeName* value in the script is a concatenation of the nick name you have chosen for the web server and the suffix *-node*. It is automatically created during plug-in installation and cannot be changed. For example, if you named your web server *myserver* during plug-in installation, the value for the associated Web server definition created after you ran the script would be *myserver-node*.

If you have enabled security or changed the default Java™ Management Extensions (JMX) connector type, edit the script and include the appropriate parameters.

4. Open the administrative console of the deployment manager if the node is federated. Wait for node synchronization to occur on the managed node and save the changed configuration that includes the new Web server definition. If the remote node is not federated, open the administrative console of the application server and save the changed configuration.
5. Copy the current plug-in configuration file, `plugin-cfg.xml`, in the *profile_root/config/cells/cell_name/nodes/web_server_name_node/servers/web_server_name* directory. Paste the file on the Web server

machine to replace the temporary *plugins_root/config/web_server_name/plugin-cfg.xml* file. The IBM HTTP Server supports automatic propagation. Other Web servers require manual propagation.

6. Start the Web server with the proper procedure for your Web server.
7. Point your browser to <http://localhost:9080/snoop> to test the internal HTTP transport provided by the Application Server. Point your browser to http://Host_name_of_Web_server_machine/snoop to test the Web server plug-in.
8. Verify that both Web addresses display the Snoop Servlet - Request/Client Information page.

Scenario 2. Local distributed plug-in configuration

The Plug-ins installation wizard does not automatically create a Web server definition within a federated application server profile. The wizard creates the *configureweb_server_name* script instead in the *plugins_root/bin* directory.

The Plug-ins installation wizard configures the Web server to use the *plugin-cfg.xml* file that will be created within the application server profile when you run the script. The deployment manager regenerates the *plugin-cfg.xml* file in the *profile_root/config/cells/cell_name/nodes/node_name/servers/web_server_name* directory. Regeneration occurs whenever a change occurs in the application server configuration that affects deployed applications on the managed node.

After installing the binary plug-in for the local Web server, you must run the script before you can start the Web server. The Web server has already been configured to use the *plugin-cfg.xml* file in the application server configuration. That file does not exist until you run the *configureweb_server_name* script.

Four configurations qualify for the local distributed application server scenario:

Profile type	Federation status	Creation of Web server definition?	Web server already defined in Application Server configuration?
Default application server profile	Federated	By script	N/A
Default Custom profile	Not federated	By script	N/A
Default Custom profile	Federated	By script	N/A
Default deployment manager profile with a managed node (non-default distributed profile)	N/A	By script	N/A

The following overview shows the procedure for completing the configuration and verifying the Web server configuration:

1. Start the deployment manager.
2. If you are planning to add an application server node into a deployment manager cell but have not done so yet, federate the node before installing the plug-ins. If the Web server definition exists when you federate the node, the Web server definition is lost when you federate.
3. Create the Web server definition in the application server. You have two options:
 - Use the administrative console of the deployment manager to create a Web server definition for a managed node. Click **Servers > Web servers > New** and use the Create new Web server entry wizard to create the Web server definition.
 - Run the script to manually create the Web server definition within the configuration of the deployment manager. Run the script from the *plugins_root/bin* directory. The script can address the deployment manager on the same machine.

Open a command window to run the appropriate script:

- **AIX** **HP-UX** **Linux** **Solaris** `./configureweb_server_name.sh`
- **Windows** `configureweb_server_name.bat`

Note: The `webserverNodeName` value in the script is a concatenation of the nick name you have chosen for the web server and the suffix `-node`. It is automatically created during plug-in installation and cannot be changed. For example, if you named your web server `myserver` during plug-in installation, the value for the associated Web server definition created after you ran the script would be `myserver-node`.

If you have enabled security or changed the default JMX connector type, edit the script and include the appropriate parameters.

4. Start the Web server with the proper procedure for your Web server.

For example, start the IBM HTTP Server from a command line:

- **AIX** **HP-UX** **Linux** **Solaris** `./IHS_root/bin/apachectl start`
- **Windows** `IHS_root\bin\apache`

5. Start the application server.

Change directories to the `profile_root/bin` directory and run the `startServer` command:

- **AIX** **HP-UX** **Linux** **Solaris** `./profile_root/bin/startServer.sh server1`
- **Windows** `profile_root\bin\startServer server1`

6. Open the administrative console of the deployment manager. Wait for node synchronization to occur and save the changed configuration that includes the new Web server definition.
7. Point your browser to `http://localhost:9080/snoop` to test the internal HTTP transport provided by the application server. Point your browser to `http://Host_name_of_Web_server_machine/snoop` to test the Web server plug-in.
8. Verify that both Web addresses display the Snoop Servlet - Request/Client Information page.

Scenario 3. Local stand-alone plug-in configuration

The Plug-ins installation wizard creates a Web server definition within the application server profile.

The Plug-ins installation wizard configures the Web server to use the `plugin-cfg.xml` file that is within the application server profile. The stand-alone application server regenerates the `profile_root/config/cells/cell_name/nodes/web_server_name_node/servers/web_server_name/plugin-cfg.xml` file whenever a change occurs in the application server configuration that affects deployed applications.

After installing the binary plug-in for the local Web server, you can start the Application Server and the Web server immediately upon completion of the installation.

Suppose that you create a Web server definition on a stand-alone application server and then federate the node. The Web server definition is not federated into the cell because the Web server definition is defined as a separate node in a stand-alone Application Server. You must recreate the Web server definition on the managed node. See Scenario 2.

Only one configuration qualifies for the local stand-alone application server scenario:

Profile type	Federation status	Automatic creation of Web server definition?	Web server already defined in Application Server configuration?
Application server	Not federated	Yes	No

Redirection to Scenario 1

An unfederated default standalone application server that has an existing Web server definition is processed as a remote plug-in configuration.

An existing Web server definition on a stand-alone application server causes the Plug-ins installation wizard to follow the remote installation path. A stand-alone application server can have just one Web server definition. Specify the same nick name for the Web server if you want to configure a new Web server.

You can use the plugin-cfg.xml file that is within the Web server definition in the configuration of the Application Server. Simply click **Browse** on the appropriate panel in the Plug-ins installation wizard to select the file. This file must exist. Otherwise, the Plug-ins installation wizard displays a warning and prevents you from proceeding until you select an existing file. The Web server is configured to use this existing plugin-cfg.xml file.

See Scenario 1 for a description of this type of node.

Redirection to Scenario 2

A federated default standalone Application Server is processed as a local distributed plug-in configuration. See Scenario 2 for a description of this type of node.

Overview of the verification procedure

The following overview shows the procedure for verifying the Web server configuration after installing the binary plug-in module:

1. Start the Web server with the proper procedure for your Web server.

For example, start the IBM HTTP Server from a command line:

- **AIX** **HP-UX** **Linux** **Solaris** `./IHS_root/bin/apachectl start`
- **Windows** `IHS_root\bin\apache`

2. Start the application server.

Change directories to the *profile_root*/bin directory and run the startServer command:

- **AIX** **HP-UX** **Linux** **Solaris** `./profile_root/bin/startServer.sh server1`
- **Windows** `profile_root\bin\startServer server1`

Open the administrative console and save the changed configuration.

3. Point your browser to `http://localhost:9080/snoop` to test the internal HTTP transport provided by the application server. Point your browser to `http://Host_name_of_Web_server_machine/snoop` to test the Web server plug-in.
4. Verify that both Web addresses display the Snoop Servlet - Request/Client Information page.

Summary

Three scenarios exist for Web server plug-ins for WebSphere Application Server. Each scenario revolves around a unique location for the plug-in configuration file, plugin-cfg.xml. The application server generates the plug-in configuration file. The purpose of the file is to publish the location of all of the application server elements that are relevant to a Web server. Such elements include applications, virtual hosts for serving applications, clusters, and cluster members, for example.

If the Web server cannot get to the file on the application server machine, you must take the file to the Web server. That process is called propagation. Propagation is reserved for the remote plug-in configuration scenario, which is **Scenario 1** in this topic.

In each of the local scenarios, the Web server can get to the plugin-cfg.xml file because it is on the same machine as the file. Two local scenarios exist because of two distinct locations for a local plugin-cfg.xml file.

The configuration scheme for Version 6 of WebSphere Application Server puts the plug-in configuration file in a Web server definition that is either within a Web server node or a managed node. The type of node is the difference between Scenario 2 and Scenario 3 in this topic. All **Scenario 2** configurations require the Web server definition to exist within a managed application server node. All **Scenario 3** configurations have the Web server definition within its own Web server node.

Limited management options do not let you create or delete the one Web server definition in the administrative console of a stand-alone application server. The inability of a stand-alone application server to create a Web server definition is the basis for the configuration scripts created by the Web server plug-ins for WebSphere Application Server. Without the scripts you could not easily create a Web server definition on a stand-alone application server node.

The location of the plugin-cfg.xml file for each configuration described in this topic is shown in the following table:

Table 5. Plug-in configuration file locations

Scenario	Profile type	Location of the plugin-cfg.xml file		
		Plugins_ install_ root	profiles_ root: within the managed node	profiles_ root: within the Web server node
1	Any profile anywhere if you select a remote installation type in the Plug-ins installation wizard	X		
	No default profile detected	X		
	Default unfederated (stand-alone) Application Server profile with an existing Web server definition	X		
	Default deployment manager profile with no managed nodes	X		
2	Default application server profile		X	
	Default custom profile		X	
	Default deployment manager profile with a managed node (non-default distributed profile)		X	
3	Default application server profile			X

Legend:

plugins_root

plugins_root
/config/
web_server_name/plugin-cfg.xml

profiles_root: within the managed node

profile_root
/config/cells/*cell_name*/nodes/
node_name_of_AppServer/servers/
web_server_name/plugin-cfg.xml

profiles_ root: within the Web server node

```
profile_root  
/config/cells/cell_name/nodes/  
web_server_name_node/servers/  
web_server_name/plugin-cfg.xml
```

Web server configuration

Plug-in configuration involves configuring the Web server to use the binary plug-in module that WebSphere Application Server provides. Plug-in configuration also includes updating the plug-in XML configuration file to reflect the current application server configuration. The binary module uses the XML file to help route Web client requests.

After installing a supported Web server, you must install a binary plug-in module for the Web server. The plug-in module lets the Web server communicate with the application server. The Plug-ins installation wizard installs the Web server plug-in. The wizard configures the Web server. The wizard also creates a Web server definition in the configuration of the application server. The Plug-ins installation wizard uses the following files to configure a plug-in for the Web server that you select:

- The **Web server configuration file** on the Web server machine, such as the httpd.conf file for IBM HTTP Server.
- The **binary Web server plug-in file** that the Plug-ins installation Wizard installs on the Web server machine.
- The **plug-in configuration file, plugin-cfg.xml**, on the application server machine that you propagate (copy) to a Web server machine.
- The **default (temporary) plug-in configuration file, plugin-cfg.xml**, on the Web server machine.
- The **configureweb_server_name script** that you copy from the Web server machine to the application server machine.

See the following descriptions of each file.

Web server configuration file

The Web server configuration file is installed as part of the Web server.

The wizard must reconfigure the configuration file for a supported Web server.

Configuration consists of adding directives that identify file locations of two files:

- The binary plug-in file
- The plugin-cfg.xml configuration file

The binary Web server plug-in file

See “Web server plug-ins” on page 121 for a description of the binary plug-in module.

An example of a binary plug-in module is the mod_ibm_app_server_http.dll file for IBM HTTP Server on the Windows platform.

The binary plug-in file does not change. However, the configuration file for the binary plug-in is an XML file. The application server changes the configuration file when certain changes to your WebSphere Application Server configuration occur. See “Web server plug-in configuration service property” on page 113 for examples of when the file gets regenerated and when it does not.

The binary module reads the XML file to adjust settings and to route requests to the application server.

The plug-in configuration file, plugin-cfg.xml

The plug-in configuration file is an XML file with settings that you can tune in the administrative console. The file lists all of the applications installed on the Web server definition. The binary module reads the XML file to adjust settings and to route requests to the application server.

The stand-alone application server regenerates the plugin-cfg.xml file in the *profile_root/config/cells/cell_name/nodes/web_server_name_node/servers/web_server_name* directory. Regeneration occurs whenever a change occurs in the application server configuration that affects deployed applications.

The deployment manager regenerates the plugin-cfg.xml file in the *profile_root/config/cells/cell_name/nodes/node_name/servers/web_server_name* directory whenever a change occurs in application server configuration that affects deployed applications on the managed node.

After regeneration, propagate (copy) the file to the Web server machine. The binary plug-in then has access to the most current copy of its configuration file.

The Web server plug-in configuration service automatically regenerates the plugin-cfg.xml file after certain events that change the configuration. The configuration service automatically propagates the plugin-cfg.xml file to an IBM HTTP Server machine when the file is regenerated. You must manually copy the file on other Web servers.

See “Web server plug-in configuration service property” on page 113 for more information.

Default plug-in configuration file, plugin-cfg.xml

The Plug-ins installation wizard creates the temporary plugin-cfg.xml file in the *plugins_root/config/web_server_name* directory. The wizard creates the file for every remote installation scenario. The wizard creates the file at the same time that it installs the binary plug-in module for the Web server.

The default file is a placeholder that you must replace with the plugin-cfg.xml file from the Web server definition on the application server. The default file is a replica of the file that the application server creates for a default stand-alone application server that has the samples installed.

Run the *configureweb_server_name* script from the *app_server_root/bin* directory of the application server machine for a remote installation, or directly from the *plugins_root/bin* directory for a local installation. The script creates the Web server definition in the configuration files of the default profile. To configure a different profile than the default, edit the *configureweb_server_name* script. Use the *-profileName* parameter to identify a different profile than the default.

After the Web server definition is created, the Web server plug-in configuration service within the application server creates the first plugin-cfg.xml file in the Web server definition on the application server machine. If you install an application, create a virtual host, or do anything that changes the configuration, you must propagate the updated plugin-cfg.xml file from the application server machine to the Web server machine to replace the default file.

The *configureweb_server_name* script for the Web server definition

The Plug-ins installation wizard creates the *configureweb_server_name* script on the Web server machine in the *plugins_root/bin* directory. If one machine in a remote scenario is running under an operating system like AIX or Linux and the other machine is running under Windows, use the script created in the *plugins_root/bin/crossPlatformScripts* directory. The script is created for remote installation scenarios only.

Copy the script from the Web server machine to the *app_server_root/bin* directory on a remote application server machine. You do not have to copy the script on a local installation. Run the script to create a Web server definition in the configuration of the application server.

When using the IBM HTTP Server, configure the IBM HTTP Administration Server also. The IBM HTTP Administration Server works with the administrative console to manage Web server definitions. Also, use the administrative console to update your Web server definition with remote Web server management options. Click **Servers > Web servers > *web_server_name*** to see configuration options. For example, click **Remote Web server management** to change such properties as:

- Host name
- Administrative port
- User ID
- Password

Note: Always open a new command window before running this script. You can avoid a potential problem by doing so.

The problem is a potential conflict between a shell environment variable, the `WAS_USER_SCRIPT` environment variable, and the actual default profile. The script always works against the default profile. However, if the `WAS_USER_SCRIPT` environment variable is set, a conflict arises as the script attempts to work on the profile identified by the variable.

The variable is easy to set accidentally. Issue any command from the `profile_root/bin` directory of any profile and the variable is set to that profile.

If you have more than one profile on your system, the potential exists that the default profile and the profile identified by the variable are different profiles. If so, a conflict occurs and the script might not create the Web server definition in the correct profile, or might not create the Web server definition at all.

Reset the variable in either of two ways:

- Close the command window where the variable is set and open a new one.
- Change directories to the `profile_root/bin` directory of the default profile and source the `setupCmdLine.sh` script:

Windows

1. Open a command prompt window.
2. Change directories to the `app_server_root\bin` directory.
3. Issue the `setupCmdLine.bat` command.
4. Use the same command prompt window to start the update installer, as described in the appropriate procedure.

AIX

HP-UX

Linux

Solaris

1. Open a command shell window.
2. Change directories to the `app_server_root/bin` directory.
3. Issue the `./setupCmdLine.sh` command. Notice the space between the periods. The special format for this command sources the command to make the setting active for all processes started from the command shell.
4. Use the same command shell window to start the update installer, as described in the appropriate procedure.

If a Web server definition already exists for a stand-alone application server, running the script does not add a new Web server definition. Each stand-alone application server can have only one Web server definition.

You cannot use the administrative console of a stand-alone application server to add or delete a Web server definition. However, you can do both tasks using the administrative scripting interface:

- Add a Web server definition through the wsadmin facility using the `configureweb_server_name` script. The script uses a Java Command Language (Jacl) script named `configureWebserverDefintion.jacl` to create and configure the Web server definition.
- Delete a Web server definition using wsadmin commands. The Web server is named `webserver1` in the following example:

```
set webserverName webserver1
set webserverNodeSuffix _node
set webserverNodeName $webserverName$webserverNodeSuffix
$AdminConfig remove [$AdminConfig getid /Node:$webserverNodeName/Server:$webserverName]
$AdminConfig remove [$AdminConfig getid /Node:$webserverNodeName]
$AdminConfig save
```

A managed node, on the other hand, can have multiple Web server definitions. The script creates a new Web server definition unless the Web server name is the same.

Replacing the default plug-in configuration file with the file from the Web server definition (propagation)

The default file uses fixed parameter values that might not match the parameter values in the actual file on the application server. The default file is a placeholder only.

The file cannot reflect changes that occur in the application server configuration. The file also cannot reflect non-default values that might be in effect on the application server.

The application server must have the following values in the actual `plugin-cfg.xml` file. If so, the default file can successfully configure the binary plug-in module. Then, the plug-in module can successfully communicate with the Web server and the application server.

Suppose that the application server does not have the following values in the actual `plugin-cfg.xml` file. In that case, the default file configures the binary plug-in module incorrectly. The plug-in module can always communicate with the Web server. But with an improper configuration file, the plug-in module cannot communicate successfully with the application server.

The following are fixed parameter values in the temporary plug-in configuration file.

- **Virtual host name**

Default value: `default_host`

This virtual host is configured to serve the `DefaultApplication` and the `Sample` applications. This value is probably the same as the value in the real `plugin-cfg.xml` file. However, suppose that you create another virtual host for serving applications and install the `DefaultApplication` on it. If so, the actual `plugin-cfg.xml` file is regenerated. The Web server cannot access the `DefaultApplication`. (The application includes the `snoop` servlet and the `hitcount` servlet.)

To access applications on the new virtual host, propagate the real `plugin-cfg.xml` file. Propagation is copying the updated file from the application server machine to the Web server machine.

- **HTTP transport port**

Default value: `9080`

The `9080` value is the default value for the HTTP transport port for the `default_host` virtual host. This value is probably the same as the value in the updated file. However, this value changes for every profile on the application server machine. The HTTP transport port value must be unique for every application server.

To communicate over a different port, propagate the real `plugin-cfg.xml` file.

- **Web server listening port**

Default value: `80`

The 80 value is the default value for the port that controls communication with the Web server. However, each application server profile must have a unique port value to communicate to a Web server. The actual port value might be 81 or another number.

To communicate over a different port, propagate the real plugin-cfg.xml file.

- **HTTPS transport port**

Default value: 9443

The 9443 value is the default value for the HTTPS (secure) transport port for the default_host virtual host. This value is probably the same as the value in the updated file. However, this value changes for every profile on the application server machine. The HTTPS transport port value must be unique for every application server.

To communicate over a different secure port, propagate the real plugin-cfg.xml file.

- **Applications installed on the server1 application server**

All of the default servlets and applications are included in the default file, including the WSsamples application and the SamplesGallery application.

The default file lists all of the default applications and samples. The list can be inaccurate. If you performed a custom installation and did not install the samples, for example, the list is inaccurate.

To serve an application that you developed with the Web server, propagate the real plugin-cfg.xml file.

Configuring a Web server and an application server on separate machines (remote)

This topic describes installing a Web server plug-in that WebSphere Application Server provides to communicate with a particular brand of Web server. This procedure describes installing the Web server and its Web server plug-in for WebSphere Application Server on one machine and configuring the application server in the default profile on another machine to communicate with the Web server.

Before you begin

When multiple profiles exist, the plug-ins installer configures only the default profile. See “Plug-ins configuration” on page 16 for a description of the flow of logic that determines how the installer selects the profile to configure.

If the WebSphere Application Server product family supports a particular brand of Web server, such as IBM HTTP Server or Microsoft Internet Information Services (IIS), then your WebSphere Application Server product provides a binary plug-in for the Web server that you must install.

If the WebSphere Application Server product family does not provide a binary plug-in for a particular brand of Web server, then the Web server is not supported. The purpose of the binary plug-in is to provide the communication protocol between the Web server and the application server.

Suppose that you create a new profile. Suppose also that you want to use a Web server. You must install a new Web server for the new profile and use the Plug-ins installation wizard to install the binary plug-in module and to configure both the Web server and the application server.

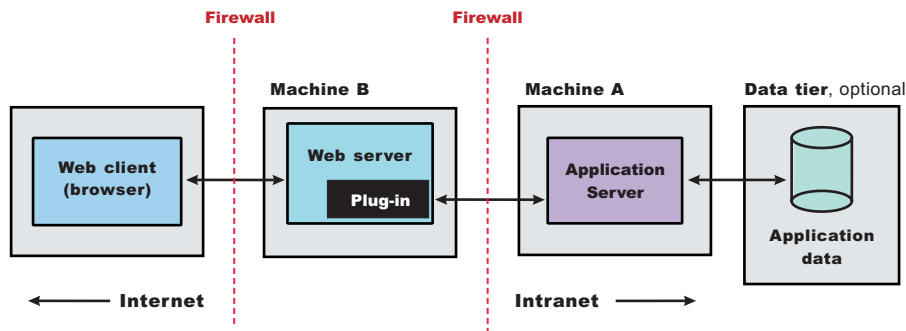
If the Web server is not already installed, you can still install the plug-ins for future use. If the WebSphere Application Server product is not installed, you can still install the plug-ins. However, it is recommended that you install the Web server and the WebSphere Application Server product before installing the plug-ins for the supported Web server.

About this task

The Plug-ins installation wizard installs the plug-in module, configures the Web server for communicating with the application server, and creates a Web server configuration definition in the application server, if possible.

This procedure configures the application server profile that is the default profile on the machine. A one-to-one relationship exists between a Web server and the application server.

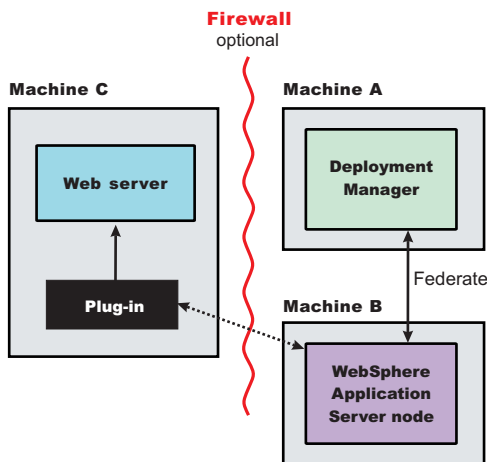
This topic describes how to create the following topology:



Note:

If you are planning to add the application server node into a deployment manager cell but have not done so yet, start the deployment manager and federate the node before installing the plug-in. You cannot add an application server with a Web server definition into the deployment manager cell.

The following topology is considered a remote topology because the Web server is on a separate machine. The diagram shows a typical remote topology for a distributed environment:



This topic describes the installation of a Web server on one machine and the application server on a separate machine. In this situation, the Plug-ins installation wizard on one machine cannot create the Web server definition in the application server configuration on the other machine.

In such a case, the Plug-ins installation wizard creates a script on the Web server machine that you can copy to the application server machine. Run the script on the application server machine to create the Web server configuration definition within the application server configuration.

Perform the following procedure to install the plug-in and configure both the Web server and the application server.

1. Log on to the operating system. If you are installing as a non-root or non-administrative user, then there are certain limitations. See the documentation for non-root installation for more information.

AIX

HP-UX

Linux

Solaris

In addition, select a umask that allows the owner to read/write to the files, and allows others to access them according to the prevailing system policy. For root, a umask of 022 is recommended. For non-root users, a umask of 002 or 022 could be used, depending on whether or not the users share the group. To verify the umask setting, issue the following command:

```
umask
```

To set the umask setting to 022, issue the following command:

```
umask 022
```

Windows

When installing as an administrative user on a Windows operating system, a Windows service is automatically created to autostart the application server. The installer user account must have the following advanced user rights:

- Act as part of the operating system
- Log on as a service

For example, on some Windows operating systems, click **Administrative Tools > Local Security Policy > User Rights Assignments** to set the advanced options. See your Windows operating system documentation for more information.

Windows

If you plan to run the application server as a Windows service, do not install from a user ID that contains spaces. A user ID with spaces cannot be validated. Such a user ID is not allowed to continue the installation. To work around this restriction, install with a user ID that does not contain spaces.

2. Install WebSphere Application Server Network Deployment on Machine A.

Read the "Installing the product and additional software" topic.

3. Install the IBM HTTP Server or another supported Web server on Machine B.

4. Optional: Create a new host alias for the default virtual host.

If you configured the Web server to use a port other than port 80, then you must add a new host alias for that port for the default host. For example, when running as non-root, IBM HTTP Server is configured with a default port value of 8080. Read the Mapping virtual hosts for Web modules topic for more information.

5. Launch the Plug-ins installation wizard on the machine with the Web server.

Select the Plug-ins installation wizard from the launchpad or change directories to the plugin directory on the product disc or in the downloaded installation image and issue the install command.

6. Clear the check box for the roadmap or select the check box to view the roadmap, then click **Next**.

If you are unsure of which installation scenario to follow, display the roadmap instead. Print and keep the roadmap as a handy overview of the installation steps.

Press **Ctrl-P** to print the roadmap if the Web browser navigation controls and the menu bar are not present on the browser window that displays the Plug-ins roadmap. Press **Ctrl-W** to close the browser window if the navigation controls and the menu bar do not display. Or close the browser window with the window control in the title bar.

7. Read the license agreement and accept the agreement if you agree to its terms. Click **Next** when you are finished.

8. If your system does not pass the prerequisites check, stop the installation, correct any problems, and restart the installation. If your system passes the prerequisites check, click **Next**.

Look for the appropriate log file for information about missing prerequisites:

- If you stop the installation, see the temporaryPluginInstallLog.txt file in the temporary directory of the user who installed the plug-ins. For example, the /tmp/temporaryPluginInstallLog.txt file might exist if the root user installed the plug-ins on an operating system such as AIX or Linux.
- If you continue the installation in spite of warnings about missing prerequisites, see the *plugins_root/logs/install/log.txt* file after the installation is complete.

Read the "Troubleshooting installation" topic for more information about log files.

9. Select the type of Web server that you are configuring and click **Next**.

The Plug-ins installation wizard panel prompts you to identify the *Web servers* to configure. Actually you can select only one Web server each time you run the Plug-ins installation wizard.

Stop any Web server while you are configuring it. A step later in the procedure directs you to start the Web server as you begin the snoop servlet test.

If you select the Web server identification option labeled **None**, the Web server installs the binary plug-ins but does not configure the Web server.

10. Select **Web server machine (remote)** and click **Next**.

11. Accept the default location for the installation root directory for the plug-ins. Click **Next**.

You can type another new directory or click **Browse** to select an empty directory. The fully qualified path identifies the plug-ins installation root directory.

The default location is shown in "Directory conventions," on page 419.

Note: The installation directory cannot contain any unsupported characters. See "Object names: what the name string cannot contain" for more information.

A possibility exists that the Web server might run on a platform that WebSphere Application Server does not support.

12. Click **Browse** to select the configuration file for your Web server, verify that the Web server port is correct, and then click **Next** when you are finished.

Select the file and not just the directory of the file. Some Web servers have two configuration files and require you to browse for each file.

The following list shows configuration files for supported Web servers:

Apache HTTP Server

apache_root/config/httpd.conf

Domino Web Server

names.nsf and Notes.jar

The wizard prompts for the notes.jar file. The actual name is Notes.jar.

The Plug-ins installation wizard verifies that the files exist but the wizard does not validate either file.

IBM HTTP Server

Microsoft Internet Information Services (IIS)

The Plug-ins installation wizard can determine the correct files to edit.

Sun Java System Web Server (formerly Sun ONE Web Server and iPlanet Web Server) Version 6.0 and later

obj.conf and magnus.conf

The wizard displays a naming panel for the nickname of the Web server definition.

13. Specify a nickname for the Web server. Click **Next** when you are finished.

The wizard uses the value to name configuration folders in the plug-ins installation root directory. The wizard also uses the name in the configuration script for the application server to name the Web server definition.

If the application server profile already has a Web server definition, delete the Web server definition before continuing. Use the following commands to delete the Web server definition:

```
$AdminTask deleteServer { -serverName webserver1 -nodeName webserver1_node }
$AdminTask removeUnmanagedNode { -nodeName webserver1_node }
$AdminConfig save
```

In these commands, *webserver1* is the Web server name.

14. Accept the default location for the plugin-cfg.xml file that the wizard creates on the Web server machine, then click **Next**.

You can type a change to the value or click **Browse** to select a file in another location. If you do not accept the default location, the plugin-cfg.xml file must exist.

15. Identify the host name or IP address of Machine A, which is the application server machine, then click **Next**.

16. Examine the summary panel. Click **Next** when you are finished.

The panel notifies you that you have manual steps to perform to complete the installation and configuration. The type of Web server, the nickname of the Web server, and the location of the plugin-cfg.xml file displays on the panel.

The Plug-ins installation wizard creates the configure*Web_server_name* script in the *plugins_root/bin/* directory on Machine B (the machine with the Web server).

The Plug-ins installation wizard also creates the plugin-cfg.xml file in the *plugins_root/config/ Web_server_name* directory.

The Web server reads the plugin-cfg.xml file to determine the applications that the application server on Machine A can serve to the Web server on Machine B. Whenever the configuration changes, the application server regenerates the file. When regeneration occurs, propagate, or copy the actual plugin-cfg.xml file from the application server machine to the Web server machine. You can automatically propagate the file to the IBM HTTP Server product.

17. Click **Next** on the pre-installation summary panel to begin the installation or click **Back** to change any characteristics of the installation.

The panel specifies the plug-ins installation root directory, the Web server plug-ins feature, and the disk size of the code that installs when you click **Next**.

18. After the wizard installs the code and creates the uninstaller program, examine the post-installation summary panel. Click **Next** when you are finished to display the Plug-ins installation roadmap.

The Plug-ins installation wizard installs the binary plug-in module. On a Linux system, for example, the installation creates the *plugins_root* directory. The *plugins_root/config/ Web_server_name* directory contains the plugin-cfg.xml file.

The wizard displays the name and location of the configuration script and the plugin-cfg.xml file. The wizard also displays the type of Web server that is configured and the nickname of the Web server.

If a problem occurs and the installation is unsuccessful, examine the logs in the *plugins_root/logs* directory. Correct any problems and reinstall.

19. Close the road map and click **Finish** to exit the wizard.

Log files from the installation are in the *plugins_root/logs/install* directory.

20. Copy the configure*Web_server_name* script from Machine B (the machine with the Web server) to the *app_server_root /bin* directory on Machine A (the application server machine).

Web_server_name is the nickname of the Web server that you specified in step 12.

Web_server_name is not a vendor name, such as IIS or Apache.

On an operating system such as AIX or Linux, the file is configure*Web_server_name*.sh. On a Windows system, the file is configure*Web_server_name*.bat. For example, on a Linux system with an IBM HTTP Server named web_server_1 in the default location, copy *plugins_root/bin/ configureweb_server_1.sh* from Machine B (the machine with the Web server) to the *app_server_root/bin* directory on Machine A (the application server machine).

If one platform is a system such as AIX or Linux and the other is a Windows platform, copy the script from the crossPlatformScripts directory. For example:

- AIX HP-UX Linux Solaris *plugins_root/bin/configureWeb_server_name.sh*
- Windows *plugins_root/bin/crossPlatformScripts/configureWeb_server_name.bat*

21. Compensate for file encoding differences to prevent script failure.

The content of the configure*Web_server_name*.bat script or the configure*Web_server_name*.sh script can be corrupt if the default file encoding of the two machines differs. This scenario is possible when one machine is set up for a double-byte character set (DBCS) locale and the other machine is not.

Determine the file encoding and use one of the following procedures to circumvent the failure. To determine the default file encoding, run the appropriate command.

- Run the following command on a system such as AIX or Linux:

```
locale
```

- Run the following command on a Windows machine:

```
CHCP
```

Use the result of the command on each machine as the value of the *web_server_machine_encoding* variable and the *application_server_machine_encoding* variable in one of the following procedures.

Procedures for compensating for encoding differences

Suppose that the Web server is running on a Linux machine and Network Deployment is running on a Windows machine.

Web server running on a system such as AIX or Linux

Run the following command on the system to encode the script file that configures the Web server definition, before you FTP the file to the Windows machine in binary mode:

```
iconv -f web_server_machine_encoding \  
-t application_server_machine_encoding \  
configureWeb_server_name.bat
```

Omit the continuation characters (\) if you enter the command on one line.

Note: The name of the Web server (nick name) is used in the name of the script file. The name cannot contain characters from a double-byte character set (DBCS) if you intend to set up IBM HTTP Server for automatic propagation.

Suppose that the Web server is running on a Windows machine and Network Deployment is running on a machine with a system such as AIX or Linux.

Web server running on a Windows machine

Run the following command on the machine with a system such as AIX or Linux to encode the script file that configures the Web server definition, after you FTP the file in binary mode:

```
iconv -f web_server_machine_encoding \  
-t application_server_machine_encoding \  
configureWeb_server_name.sh
```

Omit the continuation characters (\) if you enter the command on one line.

If the conversion mapping is not supported by the iconv command on your system, copy the contents of the Web server configuration script to a clip board and paste it onto the machine where the application server is running.

22. Start the application server on Machine A.

Use the startServer command, for example:

- **AIX** **HP-UX** **Linux** **Solaris** `profile_root/bin/startServer.sh server1`
- **Windows** `profile_root\bin\startServer server1`

23. Open a command window and change to the profile directory where the Web server should be assigned. Run the script that you copied to Machine A (the application server machine). You will need the following parameters:

```
Profile Name  
(Optional) Admin user ID  
(Optional) Admin user password
```

For example:

```
configurewebserver1.sh -ProfileName Dmgr01 -user myUserID -password myPassword
```

The webserver will be configured via wsadmin.

24. From the administrative console of the deployment manager, click **System administration > Save Changes to Master Repository > Synchronize changes with Nodes > Save**.

25. **Domino Web server only:** Set the WAS_PLUGIN_CONFIG_FILE environment variable.

See “Configuring Lotus Domino” on page 87

26. Regenerate the plugin-cfg.xml file on Machine A (the application server machine) using the administrative console. Click **Servers > Web server**. Select the Web server, then click **Generate Plug-in**.

During the installation of the plug-ins, the default plugin-cfg.xml file is installed on Machine B (the machine with the Web server) in the *plugins_root/config/Web_server_name* directory. The Web server plug-in configuration service regenerates the plugin-cfg.xml file automatically. To use the current plugin-cfg.xml file from the application server, propagate the plugin-cfg.xml file as described in the next step.

This step shows you how to regenerate the plugin-cfg.xml file. WebSphere Application Server products are configured to automatically regenerate the file each time a significant event occurs. Such events include installing applications on the application server and the Web server, for example. Creating a new virtual host is another such event.

27. Propagate the plugin-cfg.xml file from the application server to the Web server using the administrative console. Click **Servers > Web server**. Select the Web server, then click **Propagate Plug-in**. Web servers other than IBM HTTP Server require manual propagation.

The Web server plug-in configuration service propagates the plugin-cfg.xml file automatically for IBM HTTP Server 7.0 only. For all other Web servers, propagate the plug-in configuration file by manually copying the plugin-cfg.xml file from the *profile_root/config/cells/cell_name/nodes/node_name/servers/Web_server_name* directory on Machine A (the application server machine) to the *plugins_root/config/Web_server_name* directory on Machine B (the machine with the Web server).

28. Start the Snoop servlet to verify the ability of the Web server to retrieve an application from the Application Server.

Test your environment by starting your Application Server, your Web server, and using the snoop servlet with an IP address.

a. Start the Application Server. In a Network Deployment environment, the Snoop servlet is available in the cell only if you included the DefaultApplication when adding the Application Server to the cell. The -includeapps option for the addNode command migrates the DefaultApplication to the cell. If the application is not present, skip this step.

Change directories to the *profile_root/bin* directory and run the startServer command:

- **AIX** **HP-UX** **Linux** **Solaris** `./startServer.sh server1`
- **Windows** `startServer server1`

b. Start the IBM HTTP Server or the Web server that you are using.

To start the IBM HTTP Server from the command line:

Access the apache and apachectl commands in the IBMHttpServer/bin directory.

- **AIX** **HP-UX** **Linux** **Solaris** `./apachectl start`
- **Windows** `apache`

c. Point your browser to `http://localhost:9080/snoop` to test the internal HTTP transport provided by the Application Server. Point your browser to `http://Host_name_of_Web_server_machine/snoop` to test the Web server plug-in.

The HTTP Transport port is 9080 by default and must be unique for every profile. The port is associated with a virtual host named `default_host`, which is configured to host the installed DefaultApplication and any installed Samples. The snoop servlet is part of the DefaultApplication. Change the port to match your actual HTTP Transport port.

d. Verify that snoop is running.

Either Web address should display the Snoop Servlet - Request/Client Information page.

e. **Remote IBM HTTP Server only:**

Verify that the automatic propagation function can work on a remote IBM HTTP Server by using the following steps. This procedure is not necessary for local Web servers.

- 1) Create a user=adminUser, password=adminPassword in the *IHS_root* /conf/admin.passwd file. For example: `c:\ws\ihs60\bin\htpasswd -cb c:\ws\ihs60\conf\admin.passwd adminUser adminPassword`
- 2) Use the administrative console of the deployment manager or the Application Server to enter the User ID and password information that you created for the administrative user of IBM HTTP Server. Go to **Servers > Web server > Web_server_definition > Remote Web server administration**. Set the following values: admin Port=8008, User Id=adminUser, Password=adminPassword.
- 3) Set the correct read/write permissions for the httpd.conf file and the plugin-cfg.xml file. See the *IHS_root* /logs/admin_error.log file for more information.

Automatic propagation of the plug-in configuration file requires the IBM HTTP administrative server to be up and running. If you are managing an IBM HTTP Server using the WebSphere Application Server administrative console, the following error might display:

"Could not connect to IHS Administration server error"

Perform the following procedure to correct the error:

- 1) Verify that the IBM HTTP Server administration server is running.
- 2) Verify that the Web server host name and the port that is defined in the WebSphere Application Server administrative console matches the IBM HTTP Server administration host name and port.
- 3) Verify that the fire wall is not preventing you from accessing the IBM HTTP Server administration server from the WebSphere Application Server administrative console.
- 4) Verify that the user ID and password that is specified in the WebSphere Application Server administrative console under remote managed, is created in the admin.passwd file, using the htpasswd command.
- 5) If you are trying to connect securely, verify that you export the IBM HTTP Server administration server keydb personal certificate into the WebSphere Application Server key database as a signer certificate. This key database is specified by the com.ibm.ssl.trustStore directive in the sas.client.props file in the profile where your administrative console is running. This consideration is primarily for self-signed certificates.
- 6) If you still have problems, check the IBM HTTP Server admin_error.log file and the WebSphere Application Server logs (trace.log file) to determine the cause of the problem.

Results

This procedure results in the installation of the Web server plug-ins for WebSphere Application Server on a Web server machine. The Plug-ins installation wizard also configures the Web server to support an application server on a separate machine.

The installation of the binary plug-in modules results in the creation of the Plugins directory and several subdirectories. The following directories are among those created on a Linux system, for example:

- *plugins_root*/uninstall contains the uninstaller program
- *plugins_root*/bin contains the binary plug-ins for all supported Web servers
- *plugins_root*/logs contains log files
- *plugins_root*/properties contains version information
- *plugins_root*/roadmap contains the roadmap for the Plug-ins installation wizard

What to do next

See “Selecting a Web server topology diagram and roadmap” on page 9 for an overview of the installation procedure.

See “Web server configuration” on page 24 for more information about the files involved in configuring a Web server.

See “Plug-ins configuration” on page 16 for information about the location of the plug-in configuration file.

See “Editing Web server configuration files” on page 81 for information about how the Plug-ins installation wizard configures supported Web servers.

Configuring multiple Web servers and remote stand-alone application servers

This topic describes installing a Web server plug-in that WebSphere Application Server provides to communicate with a particular brand of Web server. This procedure describes installing multiple Web servers and their Web server plug-ins for WebSphere Application Server on one machine and on multiple application servers on another machine.

Before you begin

When multiple profiles exist, the plug-ins installer configures only the default profile. See “Plug-ins configuration” on page 16 for a description of the flow of logic that determines how the installer selects the profile to configure.

If the WebSphere Application Server product family supports a particular brand of Web server, such as IBM HTTP Server or Microsoft Internet Information Services (IIS), then your WebSphere Application Server product provides a binary plug-in for the Web server that you must install.

If the WebSphere Application Server product family does not provide a binary plug-in for a particular brand of Web server, then the Web server is not supported. The purpose of the binary plug-in is to provide the communication protocol between the Web server and the application server.

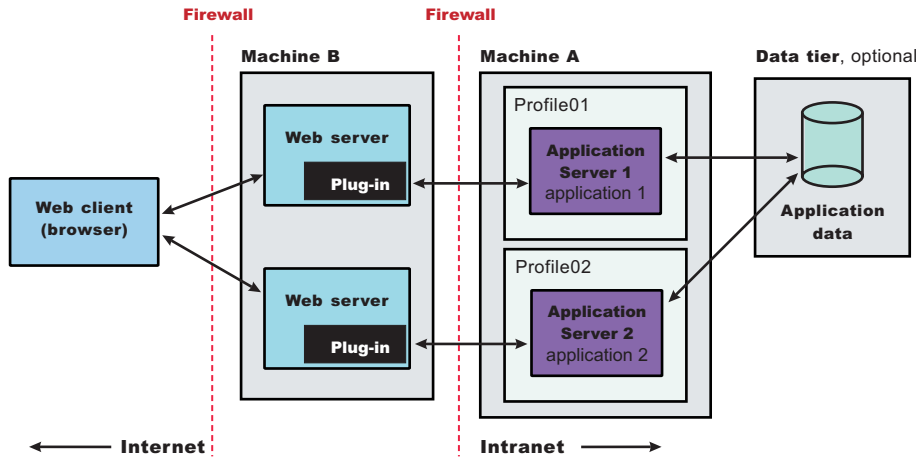
Suppose that you create a new profile. Suppose also that you want to use a Web server. You must install a new Web server for the new profile and use the Plug-ins installation wizard to install the binary plug-in module and to configure both the Web server and the application server.

If the Web server is not already installed, you can still install the plug-ins for future use. If the WebSphere Application Server product is not installed, you can still install the plug-ins. However, it is recommended that you install the Web server and the WebSphere Application Server product before installing the plug-ins for the supported Web server.

About this task

The Plug-ins installation wizard installs the plug-in module, configures the Web server for communicating with the application server, and creates a Web server configuration definition in the application server, if possible.

This topic describes how to create the following topology:



Perform the following procedure to install the plug-ins and configure both Web servers and both application servers.

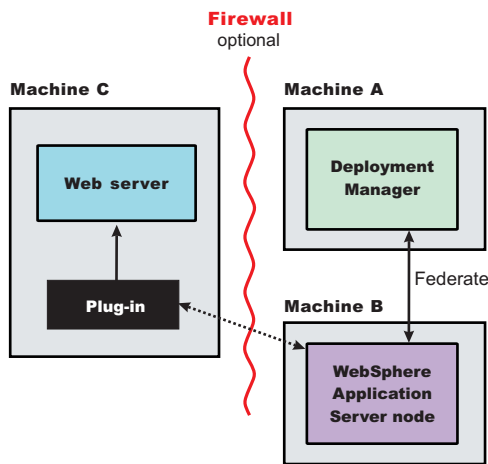
This topology lets each profile have unique applications, configuration settings, data, and log files, while sharing the same set of system files. Creating multiple profiles creates multiple application server environments that you can then dedicate to different purposes.

For example, each application server on a Web site can serve a different application. In another example, each application server can be a separate test environment that you assign to a programmer or a development team.

Note:

If you are planning to add the application server node into a deployment manager cell but have not done so yet, start the deployment manager and federate the node before installing the plug-in. You cannot add an application server with a Web server definition into the deployment manager cell.

The following topology is considered a remote topology because the Web server is on a separate machine. The diagram shows a typical remote topology for a distributed environment:



A deployment manager by itself is also considered a remote scenario if the deployment manager has no managed nodes. Although multiple application servers are not shown in the preceding diagram, Machine B could have more than one application server profile.

1. Log on to the operating system. If you are installing as a non-root or non-administrative user, then there are certain limitations. See the documentation for non-root installation for more information.

AIX

HP-UX

Linux

Solaris

In addition, select a umask that allows the owner to read/write to the files, and allows others to access them according to the prevailing system policy. For root, a umask of 022 is recommended. For non-root users, a umask of 002 or 022 could be used, depending on whether or not the users share the group. To verify the umask setting, issue the following command:

```
umask
```

To set the umask setting to 022, issue the following command:

```
umask 022
```

Windows

When installing as an administrative user on a Windows operating system, a Windows service is automatically created to autostart the application server. The installer user account must have the following advanced user rights:

- Act as part of the operating system
- Log on as a service

For example, on some Windows operating systems, click **Administrative Tools > Local Security Policy > User Rights Assignments** to set the advanced options. See your Windows operating system documentation for more information.

Windows

If you plan to run the application server as a Windows service, do not install from a user ID that contains spaces. A user ID with spaces cannot be validated. Such a user ID is not allowed to continue the installation. To work around this restriction, install with a user ID that does not contain spaces.

2. Install WebSphere Application Server Network Deployment on Machine A.
Read the "Installing the product and additional software" topic.
3. Create the first application server profile using the Profile Management Tool on Machine A.
4. Install the IBM HTTP Server or another supported Web server on Machine B.
5. Launch the Plug-ins installation wizard on the machine with the Web server.
Select the Plug-ins installation wizard from the launchpad or change directories to the plugin directory on the product disc or in the downloaded installation image and issue the install command.
6. Clear the check box for the roadmap or select the check box to view the roadmap, then click **Next**.
If you are unsure of which installation scenario to follow, display the roadmap instead. Print and keep the roadmap as a handy overview of the installation steps.
Press **Ctrl-P** to print the roadmap if the Web browser navigation controls and the menu bar are not present on the browser window that displays the Plug-ins roadmap. Press **Ctrl-W** to close the browser window if the navigation controls and the menu bar do not display. Or close the browser window with the window control in the title bar.
7. Read the license agreement and accept the agreement if you agree to its terms. Click **Next** when you are finished.
8. If your system does not pass the prerequisites check, stop the installation, correct any problems, and restart the installation. If your system passes the prerequisites check, click **Next**.
Look for the appropriate log file for information about missing prerequisites:
 - If you stop the installation, see the temporaryPluginInstallLog.txt file in the temporary directory of the user who installed the plug-ins. For example, the /tmp/temporaryPluginInstallLog.txt file might exist if the root user installed the plug-ins on an operating system such as AIX or Linux.
 - If you continue the installation in spite of warnings about missing prerequisites, see the *plugins_root/logs/install/log.txt* file after the installation is complete.Read the "Troubleshooting installation" topic for more information about log files.
9. Select the type of Web server that you are configuring and click **Next**.

The Plug-ins installation wizard panel prompts you to identify the *Web servers* to configure. Actually you can select only one Web server each time you run the Plug-ins installation wizard.

Stop any Web server while you are configuring it. A step later in the procedure directs you to start the Web server as you begin the snoop servlet test.

If you select the Web server identification option labeled **None**, the Web server installs the binary plug-ins but does not configure the Web server.

10. Select **Web server machine (remote)** and click **Next**.

11. Accept the default location for the installation root directory for the plug-ins. Click **Next**.

You can type another new directory or click **Browse** to select an empty directory. The fully qualified path identifies the plug-ins installation root directory.

The default location is shown in "Directory conventions," on page 419.

Note: The installation directory cannot contain any unsupported characters. See "Object names: what the name string cannot contain" for more information.

A possibility exists that the Web server might run on a platform that WebSphere Application Server does not support.

12. Click **Browse** to select the configuration file for your Web server, verify that the Web server port is correct, and then click **Next** when you are finished.

Select the file and not just the directory of the file. Some Web servers have two configuration files and require you to browse for each file.

The following list shows configuration files for supported Web servers:

Apache HTTP Server

apache_root/config/httpd.conf

Domino Web Server

names.nsf and Notes.jar

The wizard prompts for the notes.jar file. The actual name is Notes.jar.

The Plug-ins installation wizard verifies that the files exist but the wizard does not validate either file.

IBM HTTP Server

Microsoft Internet Information Services (IIS)

The Plug-ins installation wizard can determine the correct files to edit.

Sun Java System Web Server (formerly Sun ONE Web Server and iPlanet Web Server) Version 6.0 and later

obj.conf and magnus.conf

The wizard displays a naming panel for the nickname of the Web server definition.

13. Specify a nickname for the Web server. Click **Next** when you are finished.

The wizard uses the value to name configuration folders in the plug-ins installation root directory. The wizard also uses the name in the configuration script for the application server to name the Web server definition.

If the application server profile already has a Web server definition, delete the Web server definition before continuing. Use the following commands to delete the Web server definition:

```
$AdminTask deleteServer { -serverName webserver1 -nodeName webserver1_node }
$AdminTask removeUnmanagedNode { -nodeName webserver1_node }
$AdminConfig save
```

In these commands, *webserver1* is the Web server name.

14. Accept the default location for the plugin-cfg.xml file that the wizard creates on the Web server machine, then click **Next**.

You can type a change to the value or click **Browse** to select a file in another location. If you do not accept the default location, the plugin-cfg.xml file must exist.

15. Identify the host name or IP address of Machine A, which is the application server machine, then click **Next**.

16. Examine the summary panel. Click **Next** when you are finished.

The panel notifies you that you have manual steps to perform to complete the installation and configuration. The type of Web server, the nickname of the Web server, and the location of the plugin-cfg.xml file displays on the panel.

The Plug-ins installation wizard creates the configure*Web_server_name* script in the *plugins_root/bin/* directory on Machine B (the machine with the Web server).

The Plug-ins installation wizard also creates the plugin-cfg.xml file in the *plugins_root/config/Web_server_name* directory.

The Web server reads the plugin-cfg.xml file to determine the applications that the application server on Machine A can serve to the Web server on Machine B. Whenever the configuration changes, the application server regenerates the file. When regeneration occurs, propagate, or copy the actual plugin-cfg.xml file from the application server machine to the Web server machine. You can automatically propagate the file to the IBM HTTP Server product.

17. Click **Next** on the pre-installation summary panel to begin the installation or click **Back** to change any characteristics of the installation.

The panel specifies the plug-ins installation root directory, the Web server plug-ins feature, and the disk size of the code that installs when you click **Next**.

18. After the wizard installs the code and creates the uninstaller program, examine the post-installation summary panel. Click **Next** when you are finished to display the Plug-ins installation roadmap.

The Plug-ins installation wizard installs the binary plug-in module. On a Linux system, for example, the installation creates the *plugins_root* directory. The *plugins_root/config/Web_server_name* directory contains the plugin-cfg.xml file.

The wizard displays the name and location of the configuration script and the plugin-cfg.xml file. The wizard also displays the type of Web server that is configured and the nickname of the Web server.

If a problem occurs and the installation is unsuccessful, examine the logs in the *plugins_root/logs* directory. Correct any problems and reinstall.

19. Close the road map and click **Finish** to exit the wizard.

Log files from the installation are in the *plugins_root/logs/install* directory.

20. Copy the configure*Web_server_name* script from Machine B (the machine with the Web server) to the *app_server_root/bin* directory on Machine A (the application server machine).

Web_server_name is the nickname of the Web server that you specified in step 12.

Web_server_name is not a vendor name, such as IIS or Apache.

On an operating system such as AIX or Linux, the file is configure*Web_server_name*.sh. On a Windows system, the file is configure*Web_server_name*.bat. For example, on a Linux system with an IBM HTTP Server named web_server_1 in the default location, copy *plugins_root/bin/configureweb_server_1.sh* from Machine B (the machine with the Web server) to the *app_server_root/bin* directory on Machine A (the application server machine).

If one platform is a system such as AIX or Linux and the other is a Windows platform, copy the script from the crossPlatformScripts directory. For example:

- AIX HP-UX Linux Solaris *plugins_root/bin/configureWeb_server_name.sh*
- Windows *plugins_root/bin/crossPlatformScripts/configureWeb_server_name.bat*

21. Compensate for file encoding differences to prevent script failure.

The content of the configure*Web_server_name*.bat script or the configure*Web_server_name*.sh script can be corrupt if the default file encoding of the two machines differs. This scenario is possible when one machine is set up for a double-byte character set (DBCS) locale and the other machine is not.

Determine the file encoding and use one of the following procedures to circumvent the failure. To determine the default file encoding, run the appropriate command.

- Run the following command on a system such as AIX or Linux:

```
locale
```

- Run the following command on a Windows machine:

```
CHCP
```

Use the result of the command on each machine as the value of the *web_server_machine_encoding* variable and the *application_server_machine_encoding* variable in one of the following procedures.

Procedures for compensating for encoding differences

Suppose that the Web server is running on a Linux machine and Network Deployment is running on a Windows machine.

Web server running on a system such as AIX or Linux

Run the following command on the system to encode the script file that configures the Web server definition, before you FTP the file to the Windows machine in binary mode:

```
iconv -f web_server_machine_encoding \  
-t application_server_machine_encoding \  
configureWeb_server_name.bat
```

Omit the continuation characters (\) if you enter the command on one line.

Note: The name of the Web server (nick name) is used in the name of the script file. The name cannot contain characters from a double-byte character set (DBCS) if you intend to set up IBM HTTP Server for automatic propagation.

Suppose that the Web server is running on a Windows machine and Network Deployment is running on a machine with a system such as AIX or Linux.

Web server running on a Windows machine

Run the following command on the machine with a system such as AIX or Linux to encode the script file that configures the Web server definition, after you FTP the file in binary mode:






```
iconv -f web_server_machine_encoding \  
-t application_server_machine_encoding \  
configureWeb_server_name.sh
```

Omit the continuation characters (\) if you enter the command on one line.

If the conversion mapping is not supported by the iconv command on your system, copy the contents of the Web server configuration script to a clip board and paste it onto the machine where the application server is running.

22. Start the application server on Machine A.

Use the startServer command, for example:

-     `profile_root/bin/startServer.sh server1`
-  `profile_root\bin\startServer server1`

23. Open a command window and change to the profile directory where the Web server should be assigned. Run the script that you copied to Machine A (the application server machine). You will need the following parameters:

```
Profile Name  
(Optional) Admin user ID  
(Optional) Admin user password
```

For example:

```
configurewebserver1.sh -ProfileName Dmgr01 -user myUserID -password myPassword
```

The webserver will be configured via wsadmin.

24. From the administrative console of the deployment manager, click **System administration > Save Changes to Master Repository > Synchronize changes with Nodes > Save**.

25. **Domino Web server only:** Set the WAS_PLUGIN_CONFIG_FILE environment variable.

On platforms such as AIX or Linux, sourcing a script to the parent shell allows child processes to inherit the exported variables. On Windows systems, run the script as you would run any other command. Sourcing is automatic on Windows systems.

a. Open a command window.

b. Change directories to the plug-ins installation root directory.

c. Issue the appropriate command for the *plugins_root/bin/setupPluginCfg.sh* script:

- **AIX** **HP-UX** **Solaris** `. plugins_root/bin/setupPluginCfg.sh` (Notice the space between the period and the installation root directory.)
- **Linux** `source plugins_root/bin/setupPluginCfg.sh`

The script is also in the *lotus_root/notesdata* directory on operating systems such as AIX or Linux.

Issue the appropriate command for the script before starting the Domino Web Server.

26. Regenerate the plugin-cfg.xml file on Machine A (the application server machine) using the administrative console. Click **Servers > Web server**. Select the Web server, then click **Generate Plug-in**.

During the installation of the plug-ins, the default plugin-cfg.xml file is installed on Machine B (the machine with the Web server) in the *plugins_root/config/Web_server_name* directory. The Web server plug-in configuration service regenerates the plugin-cfg.xml file automatically. To use the current plugin-cfg.xml file from the application server, propagate the plugin-cfg.xml file as described in the next step.

This step shows you how to regenerate the plugin-cfg.xml file. WebSphere Application Server products are configured to automatically regenerate the file each time a significant event occurs. Such events include installing applications on the application server and the Web server, for example. Creating a new virtual host is another such event.

27. Propagate the plugin-cfg.xml file from the application server to the Web server using the administrative console. Click **Servers > Web server**. Select the Web server, then click **Propagate Plug-in**. Web servers other than IBM HTTP Server require manual propagation.

The Web server plug-in configuration service propagates the plugin-cfg.xml file automatically for IBM HTTP Server 7.0 only. For all other Web servers, propagate the plug-in configuration file by manually copying the plugin-cfg.xml file from the *profile_root/config/cells/cell_name/nodes/node_name/servers/Web_server_name* directory on Machine A (the application server machine) to the *plugins_root/config/Web_server_name* directory on Machine B (the machine with the Web server).

28. Start the Snoop servlet to verify the ability of the Web server to retrieve an application from the Application Server.

Test your environment by starting your Application Server, your Web server, and using the snoop servlet with an IP address.

a. Start the Application Server. In a Network Deployment environment, the Snoop servlet is available in the cell only if you included the DefaultApplication when adding the Application Server to the cell. The `-includeapps` option for the `addNode` command migrates the DefaultApplication to the cell. If the application is not present, skip this step.

Change directories to the *profile_root/bin* directory and run the `startServer` command:

- **AIX** **HP-UX** **Linux** **Solaris** `./startServer.sh server1`
- **Windows** `startServer server1`

b. Start the IBM HTTP Server or the Web server that you are using.

To start the IBM HTTP Server from the command line:

Access the `apache` and `apachectl` commands in the `IBMHttpServer/bin` directory.

- **AIX** **HP-UX** **Linux** **Solaris** `./apachectl start`

- **Windows** apache

- c. Point your browser to `http://localhost:9080/snoop` to test the internal HTTP transport provided by the Application Server. Point your browser to `http://Host_name_of_Web_server_machine/snoop` to test the Web server plug-in.

The HTTP Transport port is 9080 by default and must be unique for every profile. The port is associated with a virtual host named `default_host`, which is configured to host the installed `DefaultApplication` and any installed `Samples`. The `snoop` servlet is part of the `DefaultApplication`. Change the port to match your actual HTTP Transport port.

- d. Verify that `snoop` is running.

Either Web address should display the `Snoop Servlet - Request/Client Information` page.

- e. **Remote IBM HTTP Server only:**

Verify that the automatic propagation function can work on a remote IBM HTTP Server by using the following steps. This procedure is not necessary for local Web servers.

- 1) Create a user=`adminUser`, password=`adminPassword` in the `IHS_root/conf/admin.passwd` file. For example: `c:\ws\ihs60\bin\htpasswd -cb c:\ws\ihs60\conf\admin.passwd adminUser adminPassword`
- 2) Use the administrative console of the deployment manager or the Application Server to enter the User ID and password information that you created for the administrative user of IBM HTTP Server. Go to **Servers > Web server > Web_server_definition > Remote Web server administration**. Set the following values: `admin Port=8008`, `User Id=adminUser`, `Password=adminPassword`.
- 3) Set the correct read/write permissions for the `httpd.conf` file and the `plugin-cfg.xml` file. See the `IHS_root/logs/admin_error.log` file for more information.

Automatic propagation of the plug-in configuration file requires the IBM HTTP administrative server to be up and running. If you are managing an IBM HTTP Server using the WebSphere Application Server administrative console, the following error might display:

"Could not connect to IHS Administration server error"

Perform the following procedure to correct the error:

- 1) Verify that the IBM HTTP Server administration server is running.
 - 2) Verify that the Web server host name and the port that is defined in the WebSphere Application Server administrative console matches the IBM HTTP Server administration host name and port.
 - 3) Verify that the fire wall is not preventing you from accessing the IBM HTTP Server administration server from the WebSphere Application Server administrative console.
 - 4) Verify that the user ID and password that is specified in the WebSphere Application Server administrative console under `remote managed`, is created in the `admin.passwd` file, using the `htpasswd` command.
 - 5) If you are trying to connect securely, verify that you export the IBM HTTP Server administration server `keydb` personal certificate into the WebSphere Application Server key database as a `signer` certificate. This key database is specified by the `com.ibm.ssl.trustStore` directive in the `sas.client.props` file in the profile where your administrative console is running. This consideration is primarily for self-signed certificates.
 - 6) If you still have problems, check the IBM HTTP Server `admin_error.log` file and the WebSphere Application Server logs (`trace.log` file) to determine the cause of the problem.
29. Create the second application server profile using the Profile Management tool on Machine A. Make the profile the default profile during the profile creation by selecting the check box on the appropriate panel.
The script that the Plug-ins installation wizard creates only works on the default profile. So, this script can create only a Web server definition on the profile that is the default profile at the time that the script runs.
 30. Install a second IBM HTTP Server or another supported Web server on Machine B.

31. On Machine B, install the Web server plug-ins to configure the second Web server using the Plug-ins installation wizard. Both Web servers share a single installation of the plug-in binaries but must be configured individually.
32. The Plug-ins installation wizard creates a script named `configureweb_server_name` for the second Web server. The script is in the `plugins_root/bin` directory on Machine B. Copy the script to the `app_server_root/bin` directory on Machine A.
33. Start the second application server.
34. Run the `configureweb_server_name` script to create a Web server definition in the administrative console. You can then use the administrative console to manage the Web server.
35. Propagate the `plugin-cfg.xml` file from the second application server to the Web server using the administrative console. Click **Servers > Web server > Propagate Plug-in**. Web servers other than IBM HTTP Server require manual propagation.
36. Run the snoop servlet on the second Web server to verify that it is operational.

Results

This procedure results in installing two or more application servers on one machine and installing dedicated Web servers on another machine. This procedure installs the Web server plug-ins for both Web servers and configures both Web servers and both application servers.

What to do next

See “Selecting a Web server topology diagram and roadmap” on page 9 for an overview of the installation procedure.

See “Editing Web server configuration files” on page 81 for information about how the Plug-ins installation wizard configures supported Web servers.

See “Web server configuration” on page 24 for more information about the files involved in configuring a Web server.

For IHS Web servers, you can stop and start the Web server and propagate the `plugin-cfg.xml` file from the WebSphere Application Server machine to the Web server machine. For all other Web servers, you can not start/stop or propagate the `plugin-cfg.xml` file in the admin console. You will need to propagate the `plugin-cfg.xml` file manually. The following three steps describes how to perform manual propagation:

1. After completion of configuration with Web servers other than IHS 6.x, verify that the `plugin-cfg.xml` file exists at `<WAS_HOME>/profiles/<PROFILE_HOME>/config/cells/<CELL_NAME>/nodes/<SERVER_NAME>/servers/<WEBSERVER_DEFINITION>`
2. Transfer the above `plugin-cfg.xml` to replace `<PLUGIN_HOME>/config/<WEBSERVER_DEFINITION>/plugin-xfg.xml`
3. Restart the Web server and corresponding profile.

Configuring a Web server and an application server profile on the same machine

This topic describes installing a Web server plug-in that WebSphere Application Server provides to communicate with a particular brand of Web server. This procedure describes installing the Web server and its Web server plug-in for WebSphere Application Server and the application server on the same machine.

Before you begin

When multiple profiles exist, the plug-ins installer configures only the default profile. See “Plug-ins configuration” on page 16 for a description of the flow of logic that determines how the installer selects the profile to configure.

If the WebSphere Application Server product family supports a particular brand of Web server, such as IBM HTTP Server or Microsoft Internet Information Services (IIS), then your WebSphere Application Server product provides a binary plug-in for the Web server that you must install.

If the WebSphere Application Server product family does not provide a binary plug-in for a particular brand of Web server, then the Web server is not supported. The purpose of the binary plug-in is to provide the communication protocol between the Web server and the application server.

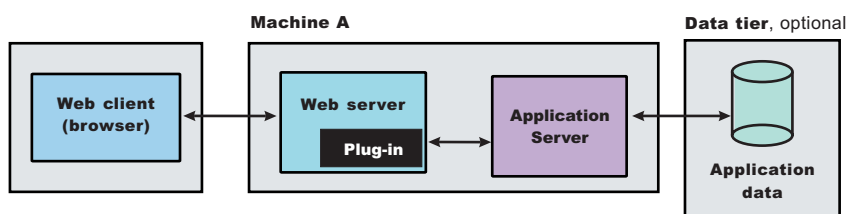
Suppose that you create a new profile. Suppose also that you want to use a Web server. You must install a new Web server for the new profile and use the Plug-ins installation wizard to install the binary plug-in module and to configure both the Web server and the application server.

If the Web server is not already installed, you can still install the plug-ins for future use. If the WebSphere Application Server product is not installed, you can still install the plug-ins. However, it is recommended that you install the Web server and the WebSphere Application Server product before installing the plug-ins for the supported Web server.

This procedure configures the application server profile that is the default profile on the machine. A one-to-one relationship exists between a Web server and the application server.

However, a stand-alone application server profile and a managed profile can each have multiple Web servers defined, each in a separate Web server definition.

This topic describes how to create the following topology:



The set of steps leading up to the next diagram show how to configure a stand-alone application server. The set of steps after the next diagram show how to configure an application server that is federated into a deployment manager cell.

Note: Non-root installation for the plug-in component is only supported if the application server was also installed by the same non-root user. Otherwise the Web server configuration scripts will fail to run against the application server installation.

About this task

The wizard performs three steps to properly configure a Web server. The wizard performs the steps in the following order:

1. The wizard installs the unique binary plug-in module for the supported Web server after collecting the following information:

- The type of Web server
- The location of the configuration file for the Web server that the wizard configures
- The plug-ins installation root directory for the Web server plug-in modules that the wizard installs
- The installation root directory of the WebSphere Application Server product, where the wizard creates a Web server definition

If administrative security is enabled, the Plug-ins installation wizard prompts for the administrative user ID and password for the profile.

2. The wizard prompts you for the location of the configuration file or files for the Web server. You must browse for and select the correct file.

The wizard edits the configuration file or files for a Web server by creating directives that point to the location of the binary plug-in module and the plug-in configuration file.

The name of the binary plug-in module varies per Web server type. The plug-in configuration file is always the plugin-cfg.xml file.

3. The wizard creates a Web server definition in the configuration of the application server unless one already exists.

You can use the administrative console to manage the Web server configuration. For example, when you install an application on the application server, you can also choose to install it on the Web server definition. If so, the updated plugin-cfg.xml file shows that the new application is available. When the Web server reads the updated plug-in configuration file, the Web server becomes aware of the new application that it can serve to Web clients.

If you choose not to install the new application on the Web server definition, the application is not added to the plug-in configuration file. The Web server is not aware of the application and cannot serve it to Web clients.

Use the following procedure to install the Web server plug-in, configure the Web server, and create a Web server definition in the default application server profile.

1. Log on to the operating system. If you are installing as a non-root or non-administrative user, then there are certain limitations. See the documentation for non-root installation for more information.

AIX **HP-UX** **Linux** **Solaris** In addition, select a umask that allows the owner to read/write to the files, and allows others to access them according to the prevailing system policy. For root, a umask of 022 is recommended. For non-root users, a umask of 002 or 022 could be used, depending on whether or not the users share the group. To verify the umask setting, issue the following command:

```
umask
```

To set the umask setting to 022, issue the following command:

```
umask 022
```

Windows When installing as an administrative user on a Windows operating system, a Windows service is automatically created to autostart the application server. The installer user account must have the following advanced user rights:

- Act as part of the operating system
- Log on as a service

For example, on some Windows operating systems, click **Administrative Tools > Local Security Policy > User Rights Assignments** to set the advanced options. See your Windows operating system documentation for more information.

Windows If you plan to run the application server as a Windows service, do not install from a user ID that contains spaces. A user ID with spaces cannot be validated. Such a user ID is not allowed to continue the installation. To work around this restriction, install with a user ID that does not contain spaces.

2. Install WebSphere Application Server Network Deployment on the machine.

Read the "Installing the product and additional software" topic for more information.

3. Install the IBM HTTP Server or another supported Web server on the machine.
4. Stop the stand-alone application server before installing the Web server plug-ins. For example, assuming that the profile name is default, use one of the following commands.
 - **AIX** /usr/IBM/WebSphere/AppServer/profiles/default/bin/stopServer.sh server1
 - **HP-UX** **Linux** **Solaris** /opt/IBM/WebSphere/AppServer/profiles/default/bin/stopServer.sh server1
 - **Windows** C: Program Files\ IBM\WebSphere\ AppServer\profiles\ default\bin\stopServer.sh server1

5. Launch the Plug-ins installation wizard on the machine.

Select the Plug-ins installation wizard from the launchpad or change directories to the plugin directory on the product disc or in the downloaded installation image and issue the install command.

6. Clear the check box for the roadmap or select the check box to view the roadmap, then click **Next**.

If you are unsure of which installation scenario to follow, display the roadmap instead. Print and keep the roadmap as a handy overview of the installation steps.

Press **Ctrl-P** to print the roadmap if the Web browser navigation controls and the menu bar are not present on the browser window that displays the Plug-ins roadmap. Press **Ctrl-W** to close the browser window if the navigation controls and the menu bar do not display. Or close the browser window with the window control in the title bar.

7. Read the license agreement and accept the agreement if you agree to its terms. Click **Next** when you are finished.
8. If your system does not pass the prerequisites check, stop the installation, correct any problems, and restart the installation. If your system passes the prerequisites check, click **Next**.

Look for the appropriate log file for information about missing prerequisites:

- If you stop the installation, see the temporaryPluginInstallLog.txt file in the temporary directory of the user who installed the plug-ins. For example, the /tmp/temporaryPluginInstallLog.txt file might exist if the root user installed the plug-ins on an operating system such as AIX or Linux.
- If you continue the installation in spite of warnings about missing prerequisites, see the *plugins_root/logs/install/log.txt* file after the installation is complete.

Read the "Troubleshooting installation" topic for more information about log files.

9. Select the type of Web server that you are configuring and click **Next**.

The Plug-ins installation wizard panel prompts you to identify the *Web servers* to configure. Actually you can select only one Web server each time you run the Plug-ins installation wizard.

Stop any Web server while you are configuring it. A step later in the procedure directs you to start the Web server as you begin the snoop servlet test.

If you select the Web server identification option labeled **None**, the Web server installs the binary plug-ins but does not configure the Web server.

10. Select **Application Server machine (local)** and click **Next**.

11. Accept the default location for the installation root directory for the plug-ins. Click **Next**.

You can type another new directory or click **Browse** to select an empty directory. The fully qualified path identifies the plug-ins installation root directory.

The default location is shown in "Directory conventions," on page 419.

Note: The installation directory cannot contain any unsupported characters. See "Object names: what the name string cannot contain" for more information.

A possibility exists that the Web server might run on a platform that WebSphere Application Server does not support.

12. Click **Browse** on the Application Server Installation Location panel to browse for the location of the application server profile if necessary. Click **Next** when the installation root directory is correct.

The fully qualified path identifies the installation root directory for the WebSphere Application Server product, which is referred to as the *app_server_root* throughout the information center.

13. Enter an administrative user ID and password if administrative security is enabled on the application server.
14. Click **Browse** to select the configuration file for your Web server, verify that the Web server port is correct, and then click **Next** when you are finished.

Select the file and not just the directory of the file. Some Web servers have two configuration files and require you to browse for each file.

The following list shows configuration files for supported Web servers:

Apache HTTP Server

apache_root/config/httpd.conf

Domino Web Server

names.nsf and Notes.jar

The wizard prompts for the notes.jar file. The actual name is Notes.jar.

The Plug-ins installation wizard verifies that the files exist but the wizard does not validate either file.

IBM HTTP Server

Microsoft Internet Information Services (IIS)

The Plug-ins installation wizard can determine the correct files to edit.

Sun Java System Web Server (formerly Sun ONE Web Server and iPlanet Web Server) Version 6.0 and later

obj.conf and *magnus.conf*

The wizard displays a naming panel for the nickname of the Web server definition.

15. Specify a nickname for the Web server. Click **Next** when you are finished.
The wizard uses the value to name configuration folders in the plug-ins installation root directory. The wizard also uses the name in the configuration script for the application server to name the Web server definition.

If the application server profile already has a Web server definition, delete the Web server definition before continuing. Use the following commands to delete the Web server definition:

```
$AdminTask deleteServer { -serverName webserver1 -nodeName webserver1_node }  
$AdminTask removeUnmanagedNode { -nodeName webserver1_node }  
$AdminConfig save
```

In these commands, *webserver1* is the Web server name.

16. Specify the location for the plugin-cfg.xml file and click **Next**.

This is a critical selection.

See “Plug-ins configuration” on page 16 for a description of the logic that determines what path is configured by default. The following possibilities exist for the default location of the plug-in configuration file. The wizard determines the characteristics of the application server to determine the best path for the file:

- An application server that has an existing Web server definition has the following path:

```
plugins_root/config/  
web_server_name/plugin-cfg.xml
```

- A stand-alone application server that does not have a Web server definition has the following path:

```
profile_root  
/config/cells/cell_name/nodes/  
web_server_name_node/servers/  
web_server_name/plugin-cfg.xml
```

You can accept the default value if the application server does not have a Web server definition.

Using an existing Web server definition

If the application server has a Web server definition, the wizard cannot create a new Web server definition within the application server configuration. However, the wizard can reconfigure the Web server. Click **Browse** and select the existing plugin-cfg.xml file in the application server configuration.

To find the plug-in configuration file in a stand-alone application server, follow this file path:

```
profile_root
  /config/cells/cell_name/nodes/
    web_server_name_node/servers/
      web_server_name/plugin-cfg.xml
```

If the existing *web_server_name* is different than the nickname that you gave the Web server in the wizard, click **Back** to return to the naming panel for the Web server and change the name to match the existing Web server definition name.

If you cannot find an existing plugin-cfg.xml file after all, you must install the temporary plugin-cfg.xml file. In such a case, type the path to the plug-ins installation root directory so that the wizard can install the temporary plug-in configuration file:

```
plugins_root/config/
  web_server_name/plugin-cfg.xml
```

17. Click **Next** after verifying the characteristics of the plug-ins installation or click **Back** to make changes.

Once created, a Web server definition on a stand-alone application server node cannot be removed except through scripting. (See “Uninstalling the Web server plug-ins for WebSphere Application Server” on page 101 for the procedure.)

You can, however, reuse the same definition for a different type of Web server. Run the Plug-ins installation wizard to configure a new Web server in that situation. The Plug-ins installation wizard configures the new Web server to use the existing plugin-cfg.xml file.

18. Click **Next** on the pre-installation summary panel to begin the installation or click **Back** to change any characteristics of the installation. The wizard begins installing the plug-ins and configuring the Web server and the application server.

The wizard shows an installation status panel as it installs the plug-ins.

The wizard displays the Installation summary panel at the completion of the installation.

19. Verify the success of the installation on the Installation summary panel and click **Finish** to exit the wizard.

If a problem occurs and the installation is unsuccessful, examine the logs in the *plugins_root/logs* directory. Correct any problems and reinstall.

20. **Domino Web server only:** Set the WAS_PLUGIN_CONFIG_FILE environment variable.

On platforms such as AIX or Linux, sourcing a script to the parent shell allows child processes to inherit the exported variables. On Windows systems, run the script as you would run any other command. Sourcing is automatic on Windows systems.

- a. Open a command window.
- b. Change directories to the plug-ins installation root directory.
- c. Issue the appropriate command for the *plugins_root/bin/setupPluginCfg.sh* script:

- **AIX** **HP-UX** **Solaris** `. plugins_root/bin/setupPluginCfg.sh` (Notice the space between the period and the installation root directory.)
- **Linux** `source plugins_root/bin/setupPluginCfg.sh`

The script is also in the *lotus_root/notesdata* directory on operating systems such as AIX or Linux. Issue the appropriate command for the script before starting the Domino Web Server.

21. Start the Snoop servlet to verify the ability of the Web server to retrieve an application from the Application Server.

Test your environment by starting your Application Server, your Web server, and using the snoop servlet with an IP address.

- a. Start the Application Server. In a Network Deployment environment, the Snoop servlet is available in the cell only if you included the DefaultApplication when adding the Application Server to the cell. The `-includeapps` option for the `addNode` command migrates the DefaultApplication to the cell. If the application is not present, skip this step.

Change directories to the `profile_root/bin` directory and run the `startServer` command:

- `AIX` `HP-UX` `Linux` `Solaris` `./startServer.sh server1`
- `Windows` `startServer server1`

- b. Start the IBM HTTP Server or the Web server that you are using.

To start the IBM HTTP Server from the command line:

Access the `apache` and `apachectl` commands in the `IBMHttpServer/bin` directory.

- `AIX` `HP-UX` `Linux` `Solaris` `./apachectl start`
- `Windows` `apache`

- c. Point your browser to `http://localhost:9080/snoop` to test the internal HTTP transport provided by the Application Server. Point your browser to `http://Host_name_of_Web_server_machine/snoop` to test the Web server plug-in.

The HTTP Transport port is 9080 by default and must be unique for every profile. The port is associated with a virtual host named `default_host`, which is configured to host the installed DefaultApplication and any installed Samples. The snoop servlet is part of the DefaultApplication. Change the port to match your actual HTTP Transport port.

- d. Verify that snoop is running.

Either Web address should display the Snoop Servlet - Request/Client Information page.

- e. **Remote IBM HTTP Server only:**

Verify that the automatic propagation function can work on a remote IBM HTTP Server by using the following steps. This procedure is not necessary for local Web servers.

- 1) Create a user=`adminUser`, password=`adminPassword` in the `IHS_root/conf/admin.passwd` file. For example: `c:\ws\ihs60\bin\htpasswd -cb c:\ws\ihs60\conf\admin.passwd adminUser adminPassword`
- 2) Use the administrative console of the deployment manager or the Application Server to enter the User ID and password information that you created for the administrative user of IBM HTTP Server. Go to **Servers > Web server > Web_server_definition > Remote Web server administration**. Set the following values: `admin Port=8008`, `User Id=adminUser`, `Password=adminPassword`.
- 3) Set the correct read/write permissions for the `httpd.conf` file and the `plugin-cfg.xml` file. See the `IHS_root/logs/admin_error.log` file for more information.

Automatic propagation of the plug-in configuration file requires the IBM HTTP administrative server to be up and running. If you are managing an IBM HTTP Server using the WebSphere Application Server administrative console, the following error might display:

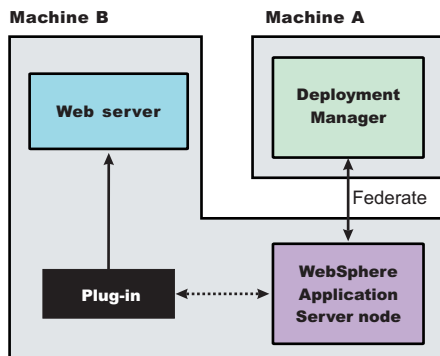
"Could not connect to IHS Administration server error"

Perform the following procedure to correct the error:

- 1) Verify that the IBM HTTP Server administration server is running.
- 2) Verify that the Web server host name and the port that is defined in the WebSphere Application Server administrative console matches the IBM HTTP Server administration host name and port.
- 3) Verify that the fire wall is not preventing you from accessing the IBM HTTP Server administration server from the WebSphere Application Server administrative console.
- 4) Verify that the user ID and password that is specified in the WebSphere Application Server administrative console under remote managed, is created in the `admin.passwd` file, using the `htpasswd` command.

- 5) If you are trying to connect securely, verify that you export the IBM HTTP Server administration server keydb personal certificate into the WebSphere Application Server key database as a signer certificate. This key database is specified by the com.ibm.ssl.trustStore directive in the sas.client.props file in the profile where your administrative console is running. This consideration is primarily for self-signed certificates.
 - 6) If you still have problems, check the IBM HTTP Server admin_error.log file and the WebSphere Application Server logs (trace.log file) to determine the cause of the problem.
22. Configure a Web server and a distributed application server profile on the same machine.
- The rest of these steps describe how to configure an application server that is federated into a deployment manager cell.

The following topology is considered a local distributed topology because it involves a cell:



This part of the procedure assumes that you have already installed the Network Deployment product on both machines. Also assumed is that you have already configured a deployment manager profile on Machine A and an application server profile on Machine B.

If you are planning to add the application server node into a deployment manager cell but have not done so yet, start the deployment manager and federate the node before installing the plug-in. You cannot add an application server with a Web server definition into the deployment manager cell.

A Web server definition on a federated application server is installed on the same managed node as the application server. There is one node, but with two server processes, the application server and the Web server definition.

If you are installing the plug-ins for use with a federated application server, start the deployment manager. Verify that the node agent process on the managed node is also running. Both the deployment manager and the node agent must be running to successfully configure a managed node.

23. Install IBM HTTP Server or another supported Web server on Machine B.
24. Launch the Plug-ins installation wizard on the machine with the Web server.
25. Clear the check box for the roadmap or select the check box to view the roadmap, then click **Next**.
26. Read the license agreement and accept the agreement if you agree to its terms, then click **Next**.
27. If your system does not pass the prerequisites check, stop the installation, correct any problems, and restart the installation. If your system passes the prerequisites check, click **Next**.
28. Select the type of Web server that you are configuring, then click **Next**.
29. Select **Application Server machine (local)** and click **Next**.
30. Accept the default location for the installation root directory for the plug-ins, then click **Next**.
31. Click **Browse** on the Application Server installation location panel to browse for the location of the Application Server profile, if necessary. Click **Next** when the installation root directory is correct.
32. Click **Browse** to select the configuration file for your Web server, verify that the Web server port is correct, and then click **Next**.
33. Specify a nickname for the Web server, then click **Next**.
34. Specify the location for the plugin-cfg.xml file and click **Next**.

This is a critical selection. A federated application server that does not have a Web server definition has the following path:

```
profile_root  
/config/cells/cell_name/nodes/  
node_name_of_AppServer/servers/  
web_server_name/plugin-cfg.xml
```

An application server that has an existing Web server definition has the following path:

```
plugins_root/config/  
web_server_name/plugin-cfg.xml
```

See “Plug-ins configuration” on page 16 for a description of the logic that determines what path is configured by default.

35. Click **Next** after verifying the characteristics of the plug-ins installation or click **Back** to make changes.

You can use the administrative console of the deployment manager to delete an existing Web server or to create new ones. Federated nodes can have more than one Web server definition.

36. Click **Next** on the pre-installation summary panel to begin the installation or click **Back** to change any characteristics of the installation.

The wizard begins installing the plug-ins and configuring the Web server and the application server.

The wizard shows an installation status panel as it installs the plug-ins.

The wizard displays the Installation summary panel at the completion of the installation.

37. Verify the success of the installation on the Installation summary panel and click **Finish** to exit the wizard.

38. Complete the installation by creating the Web server definition.

You can use the administrative console of the deployment manager to create the Web server definition on a federated node. Or, you can run the configuration script that the Plug-ins installation wizard created.

The script already contains all of the information that you must gather when using the administrative console option.

Select one of the following options:

- **Using the administrative console**

Click **Servers > Web servers > New** and use the Create new Web server entry wizard to create the Web server definition.

- **Running the configuration script**

Issue the appropriate command from a command window:

```
- AIX HP-UX Linux Solaris ./plugins_root/bin/configureweb_server_name.sh  
- Windows plugins_root\bin\configureweb_server_name.bat
```

If you have enabled security or changed the default JMX connector type, edit the script and include the appropriate parameters on the wsadmin command.

39. From the administrative console of the deployment manager, click **System administration > Save Changes to Master Repository > Synchronize changes with Nodes > Save**.

40. Source the Domino Web server script if necessary.

41. Start the snoop servlet.

See the snoop procedure for the stand-alone application server for the full procedure.

Results

The installation of the binary plug-in modules results in the creation of the Plugins directory and several subdirectories. The following directories are among those created on a Linux system, for example:

- `plugins_root/uninstall` contains the uninstaller program
- `plugins_root/bin` contains the binary plug-ins for all supported Web servers

- *plugins_root/logs* contains log files
- *plugins_root/properties* contains version information
- *plugins_root/roadmap* contains the roadmap for the Plug-ins installation wizard

The Plug-ins installation wizard creates a Web server definition within the application server profile unless one already exists.

The Plug-ins installation wizard configures the Web server to use the *profile_root/plugin-cfg.xml* file.

The application server regenerates the Web server plug-in configuration file, *plugin-cfg.xml* whenever an event occurs that affects the file. Such events include the addition or removal of an application, server, or virtual host. The stand-alone application server regenerates the file in the following location:

```
profile_root
  /config/cells/cell_name/nodes/
  web_server_name_node/servers/
  web_server_name/plugin-cfg.xml
```

On a federated node, the creation or removal of clusters and cluster members also causes file regeneration. The deployment manager regenerates the file for a federated application server in the following location:

```
profile_root
  /config/cells/cell_name/nodes/
  node_name_of_AppServer/servers/
  web_server_name/plugin-cfg.xml
```

What to do next

You can start a stand-alone application server and the Web server immediately after installing of the binary plug-in for the local Web server. Open the administrative console of the application server after you start the server and save the changed configuration.

After installing the binary plug-in for the local Web server, you can start a federated application server and the Web server after running the configuration script that completes the installation. Open the administrative console of the deployment manager. Wait for node synchronization to occur. Save the changed configuration that includes the new Web server definition.

See “Selecting a Web server topology diagram and roadmap” on page 9 for an overview of the installation procedure.

See “Plug-ins configuration” on page 16 for information about the location of the plug-in configuration file.

See “Web server configuration” on page 24 for information about the files involved in configuring a Web server.

See “Editing Web server configuration files” on page 81 for information about how the Plug-ins installation wizard configures supported Web servers.

See “Installing Web server plug-ins” on page 6 for information about other installation scenarios for installing Web server plug-ins.

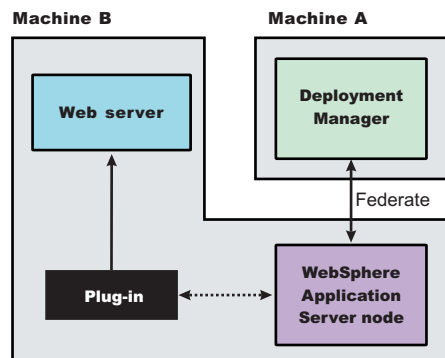
Configuring a Web server and a custom profile on the same machine

This procedure describes installing a Web server and its plug-in on a machine where the default profile is a custom profile.

Before you begin

When multiple profiles exist, the plug-ins installer configures only the default profile. See “Plug-ins configuration” on page 16 for a description of the flow of logic that determines how the installer selects the profile to configure.

This procedure configures the custom profile that is the default profile on the machine. This procedure assumes that you already have installed a deployment manager on Machine A.



The WebSphere Application Server node on Machine B is the custom node that you create in this procedure. This procedure starts the deployment manager and federates the custom node before installing the Web server plug-ins.

Start the deployment manager. The deployment manager must be running to successfully federate and configure the custom node.

About this task

Use the following procedure to install the Web server plug-in, configure the Web server, and create a Web server definition in the default custom profile (custom node).

1. Log on to the operating system. If you are installing as a non-root or non-administrative user, then there are certain limitations. See the documentation for non-root installation for more information.

AIX **HP-UX** **Linux** **Solaris** In addition, select a umask that allows the owner to read/write to the files, and allows others to access them according to the prevailing system policy. For root, a umask of 022 is recommended. For non-root users, a umask of 002 or 022 could be used, depending on whether or not the users share the group. To verify the umask setting, issue the following command:

```
umask
```

To set the umask setting to 022, issue the following command:

```
umask 022
```

Windows When installing as an administrative user on a Windows operating system, a Windows service is automatically created to autostart the application server. The installer user account must have the following advanced user rights:

- Act as part of the operating system
- Log on as a service

For example, on some Windows operating systems, click **Administrative Tools > Local Security Policy > User Rights Assignments** to set the advanced options. See your Windows operating system documentation for more information.

Windows

If you plan to run the application server as a Windows service, do not install from a user ID that contains spaces. A user ID with spaces cannot be validated. Such a user ID is not allowed to continue the installation. To work around this restriction, install with a user ID that does not contain spaces.

2. Install the WebSphere Application Server Network Deployment product.

Read the "Installing the product and additional software" topic for more information.

3. Create a custom profile as the first profile on the machine and federate the node as you create it.
4. Optional: Use the administrative console of the deployment manager to create an Application Server on the custom node.

Click **Servers > Applications servers > New** and follow the instructions to create a server. A server is not required for installing the plug-ins but it lets you verify the functionality of the Web server.

5. Optional: Install the DefaultApplication on the new server while you are in the administrative console of the deployment manager.

The DefaultApplication includes the snoop servlet. The verification step uses the snoop servlet.

6. Stop the application server if it is running.

Use the stopServer command or the administrative console of the deployment manager to stop the managed node.

7. Install the IBM HTTP Server or another supported Web server on Machine B.

8. Launch the Plug-ins installation wizard on the machine with the Web server.

Select the Plug-ins installation wizard from the launchpad or change directories to the plugin directory on the product disc or in the downloaded installation image and issue the install command.

9. Clear the check box for the roadmap or select the check box to view the roadmap, then click **Next**.

If you are unsure of which installation scenario to follow, display the roadmap instead. Print and keep the roadmap as a handy overview of the installation steps.

Press **Ctrl-P** to print the roadmap if the Web browser navigation controls and the menu bar are not present on the browser window that displays the Plug-ins roadmap. Press **Ctrl-W** to close the browser window if the navigation controls and the menu bar do not display. Or close the browser window with the window control in the title bar.

10. Read the license agreement and accept the agreement if you agree to its terms. Click **Next** when you are finished.

11. If your system does not pass the prerequisites check, stop the installation, correct any problems, and restart the installation. If your system passes the prerequisites check, click **Next**.

Look for the appropriate log file for information about missing prerequisites:

- If you stop the installation, see the temporaryPluginInstallLog.txt file in the temporary directory of the user who installed the plug-ins. For example, the /tmp/temporaryPluginInstallLog.txt file might exist if the root user installed the plug-ins on an operating system such as AIX or Linux.
- If you continue the installation in spite of warnings about missing prerequisites, see the *plugins_root/logs/install/log.txt* file after the installation is complete.

Read the "Troubleshooting installation" topic for more information about log files.

12. Select the type of Web server that you are configuring and click **Next**.

The Plug-ins installation wizard panel prompts you to identify the *Web servers* to configure. Actually you can select only one Web server each time you run the Plug-ins installation wizard.

Stop any Web server while you are configuring it. A step later in the procedure directs you to start the Web server as you begin the snoop servlet test.

If you select the Web server identification option labeled **None**, the Web server installs the binary plug-ins but does not configure the Web server.

13. Select **Application server machine (local)** and click **Next**.

14. Accept the default location for the installation root directory for the plug-ins. Click **Next**.

You can type another new directory or click **Browse** to select an empty directory. The fully qualified path identifies the plug-ins installation root directory.

The default location is shown in “Directory conventions,” on page 419.

Note: The installation directory cannot contain any unsupported characters. See “Object names: what the name string cannot contain” for more information.

A possibility exists that the Web server might run on a platform that WebSphere Application Server does not support.

15. Click **Browse** on the Application Server installation location panel to browse for the location of the managed node, if necessary. Click **Next** when the installation root directory is correct.

The fully qualified path identifies the installation root directory for the Network Deployment product core files.

16. Click **Browse** to select the configuration file for your Web server, verify that the Web server port is correct, and then click **Next** when you are finished.

Select the file and not just the directory of the file. Some Web servers have two configuration files and require you to browse for each file.

The following list shows configuration files for supported Web servers:

Apache HTTP Server

apache_root/config/httpd.conf

Domino Web Server

names.nsf and Notes.jar

The wizard prompts for the notes.jar file. The actual name is Notes.jar.

The Plug-ins installation wizard verifies that the files exist but the wizard does not validate either file.

IBM HTTP Server

Microsoft Internet Information Services (IIS)

The Plug-ins installation wizard can determine the correct files to edit.

Sun Java System Web Server (formerly Sun ONE Web Server and iPlanet Web Server) Version 6.0 and later

obj.conf and *magnus.conf*

The wizard displays a naming panel for the nickname of the Web server definition.

17. Specify a nick name for the Web server and click **Next**.

The wizard uses the value to name configuration folders in the plug-ins installation root directory. The wizard also uses the name in the configuration script for the application server to name the Web server definition.

18. Accept the location for the plugin-cfg.xml file and click **Next**.

See “Plug-ins configuration” on page 16 for a description of the logic that determines what path is configured by default. The wizard determines the characteristics of the managed application server node to determine the best path for the file:

A federated custom node has the following path:

```
profile_root
/config/cells/cell_name/nodes/
node_name_of_custom_profile/servers/
web_server_name/plugin-cfg.xml
```

```
profile_root
/config/cells/cell_name/nodes/
node_name_of_custom_profile/servers/
web_server_name/plugin-cfg.xml
```

Accept the default value.

19. Click **Next** after verifying the characteristics of the plug-ins installation or click **Back** to make changes.

You can use the administrative console of the deployment manager to delete an existing Web server or to create new ones. Federated nodes can have more than one Web server definition.

20. Click **Next** on the pre-installation summary panel to begin the installation or click **Back** to change any characteristics of the installation. The wizard begins installing the plug-ins and configuring the Web server and the managed custom node.

The wizard shows an installation status panel as it installs the plug-ins. The wizard displays the Installation summary panel at the completion of the installation.

21. After the wizard installs the code and creates the uninstaller program, examine the post-installation summary panel. Click **Next** when you are finished to display the Plug-ins installation roadmap.

The Plug-ins installation wizard installs the binary plug-in module. On a Linux system, for example, the installation creates the *plugins_root* directory. The *plugins_root/config/Web_server_name* directory contains the plugin-cfg.xml file.

The wizard displays the name and location of the configuration script and the plugin-cfg.xml file. The wizard also displays the type of Web server that is configured and the nickname of the Web server.

If a problem occurs and the installation is unsuccessful, examine the logs in the *plugins_root/logs* directory. Correct any problems and reinstall.

22. Close the road map and click **Finish** to exit the wizard.

Log files from the installation are in the *plugins_root/logs/install* directory.

23. Complete the installation by creating the Web server definition.

You can use the administrative console of the deployment manager to create the Web server definition on a federated node. Or, you can run the configuration script that the Plug-ins installation wizard created.

The script already contains all of the information that you must gather when using the administrative console option.

Select one of the following options:

- **Using the administrative console**

Click **Servers > Web servers > New** and use the Create new Web server entry wizard to create the Web server definition.

- **Running the configuration script**

Issue the appropriate command from a command window:

```
– AIX HP-UX Linux Solaris plugins_root/bin/configureWeb_server_name.sh  
– Windows plugins_root\bin\configureWeb_server_name.bat
```

If you have enabled security or changed the default JMX connector type, edit the script and include the appropriate parameters on the wsadmin command.

Shell scripts on some operating systems cannot contain double-byte characters or single-byte characters with pronunciation keys.

If the file path to the Web server includes double-byte characters or single-byte characters with pronunciation keys, such as o-umlaut (a diacritic mark over the o), c-cedilla (a mark is under the c), or characters with other keys, the script might not run correctly. Copy the content of such a script to the command line and run the wsadmin command directly.

24. **Domino Web server only:** Set the WAS_PLUGIN_CONFIG_FILE environment variable.

On platforms such as AIX or Linux, sourcing a script to the parent shell allows child processes to inherit the exported variables. On Windows systems, run the script as you would run any other command. Sourcing is automatic on Windows systems.

- a. Open a command window.
- b. Change directories to the plug-ins installation root directory.
- c. Issue the appropriate command for the *plugins_root/bin/setupPluginCfg.sh* script:

- **AIX** **HP-UX** **Solaris** `. plugins_root/bin/setupPluginCfg.sh` (Notice the space between the period and the installation root directory.)
- **Linux** `source plugins_root/bin/setupPluginCfg.sh`

The script is also in the *lotus_root/notesdata* directory on operating systems such as AIX or Linux. Issue the appropriate command for the script before starting the Domino Web Server.

25. Start the Snoop servlet to verify the ability of the Web server to retrieve an application from the Application Server.

Test your environment by starting your Application Server, your Web server, and using the snoop servlet with an IP address.

- a. Start the Application Server. In a Network Deployment environment, the Snoop servlet is available in the cell only if you included the DefaultApplication when adding the Application Server to the cell. The `-includeapps` option for the `addNode` command migrates the DefaultApplication to the cell. If the application is not present, skip this step.

Change directories to the *profile_root/bin* directory and run the `startServer` command:

- **AIX** **HP-UX** **Linux** **Solaris** `./startServer.sh server1`
- **Windows** `startServer server1`

- b. Start the IBM HTTP Server or the Web server that you are using.

To start the IBM HTTP Server from the command line:

Access the `apache` and `apachectl` commands in the *IBMHttpServer/bin* directory.

- **AIX** **HP-UX** **Linux** **Solaris** `./apachectl start`
- **Windows** `apache`

- c. Point your browser to `http://localhost:9080/snoop` to test the internal HTTP transport provided by the Application Server. Point your browser to `http://Host_name_of_Web_server_machine/snoop` to test the Web server plug-in.

The HTTP Transport port is 9080 by default and must be unique for every profile. The port is associated with a virtual host named `default_host`, which is configured to host the installed DefaultApplication and any installed Samples. The snoop servlet is part of the DefaultApplication. Change the port to match your actual HTTP Transport port.

- d. Verify that snoop is running.

Either Web address should display the Snoop Servlet - Request/Client Information page.

- e. **Remote IBM HTTP Server only:**

Verify that the automatic propagation function can work on a remote IBM HTTP Server by using the following steps. This procedure is not necessary for local Web servers.

- 1) Create a user=`adminUser`, password=`adminPassword` in the *IHS_root/conf/admin.passwd* file. For example: `c:\ws\ihs60\bin\htpasswd -cb c:\ws\ihs60\conf\admin.passwd adminUser adminPassword`
- 2) Use the administrative console of the deployment manager or the Application Server to enter the User ID and password information that you created for the administrative user of IBM HTTP Server. Go to **Servers > Web server > Web_server_definition > Remote Web server administration**. Set the following values: `admin Port=8008`, `User Id=adminUser`, `Password=adminPassword`.
- 3) Set the correct read/write permissions for the `httpd.conf` file and the `plugin-cfg.xml` file. See the *IHS_root/logs/admin_error.log* file for more information.

Automatic propagation of the plug-in configuration file requires the IBM HTTP administrative server to be up and running. If you are managing an IBM HTTP Server using the WebSphere Application Server administrative console, the following error might display:

"Could not connect to IHS Administration server error"

Perform the following procedure to correct the error:

- 1) Verify that the IBM HTTP Server administration server is running.
 - 2) Verify that the Web server host name and the port that is defined in the WebSphere Application Server administrative console matches the IBM HTTP Server administration host name and port.
 - 3) Verify that the fire wall is not preventing you from accessing the IBM HTTP Server administration server from the WebSphere Application Server administrative console.
 - 4) Verify that the user ID and password that is specified in the WebSphere Application Server administrative console under remote managed, is created in the admin.passwd file, using the htpasswd command.
 - 5) If you are trying to connect securely, verify that you export the IBM HTTP Server administration server keydb personal certificate into the WebSphere Application Server key database as a signer certificate. This key database is specified by the com.ibm.ssl.trustStore directive in the sas.client.props file in the profile where your administrative console is running. This consideration is primarily for self-signed certificates.
 - 6) If you still have problems, check the IBM HTTP Server admin_error.log file and the WebSphere Application Server logs (trace.log file) to determine the cause of the problem.
26. If the deployment manager does not have the DefaultApplication installed, you can test the functionality of the Web server and the custom node using an application of your own.
27. From the administrative console of the deployment manager, click **System administration > Save Changes to Master Repository > Synchronize changes with Nodes > Save**.
28. To create multiple Web server definitions for the managed node, use the Plug-ins installation wizard to configure each Web server.
- Identify the same managed node each time. Give each Web server a different nick name.

Results

This procedure results in the installation of the Web server plug-ins for WebSphere Application Server on a Web server machine. The Plug-ins installation wizard creates a Web server definition within the managed node.

The Plug-ins installation wizard configures the Web server to use the plugin-cfg.xml file that is within the managed custom node.

The deployment manager regenerates the Web server plug-in configuration file, plugin-cfg.xml whenever an event occurs that affects the file. Such events include the addition or removal of an application, server, or virtual host.

The creation or removal of clusters and cluster members also causes file regeneration. Automatic propagation through node synchronization copies the file after each regeneration to the following location on the custom node machine:

```
profile_root
/config/cells/cell_name/nodes/
node_name_of_custom_profile/servers/
web_server_name/plugin-cfg.xml
```

The installation of the binary plug-in modules results in the creation of the Plugins directory and several subdirectories. The following directories are among those created on a Linux system, for example:

- *plugins_root*/uninstall contains the uninstaller program
- *plugins_root*/bin contains the binary plug-ins for all supported Web servers
- *plugins_root*/logs contains log files
- *plugins_root*/properties contains version information
- *plugins_root*/roadmap contains the roadmap for the Plug-ins installation wizard

What to do next

After installing the binary plug-in for the local Web server, you can start the managed node and the Web server after running the configuration script that completes the installation.

See “Plug-ins configuration” on page 16 for information about the location of the plug-in configuration file.

See “Web server configuration” on page 24 for more information about the files involved in configuring a Web server.

See “Editing Web server configuration files” on page 81 for information about how the Plug-ins installation wizard configures supported Web servers.

See “Installing Web server plug-ins” on page 6 for information about other installation scenarios for installing Web server plug-ins.

Configuring a Web server and a deployment manager profile on the same machine

This procedure describes installing a Web server and its plug-in on a machine where the default profile is a deployment manager.

Before you begin

When multiple profiles exist, the plug-ins installer configures only the default profile. See “Plug-ins configuration” on page 16 for a description of the flow of logic that determines how the installer selects the profile to configure.

This procedure configures the deployment manager profile that is the default profile on the machine. A managed node must exist to define a Web server definition, which is always on a managed node. If the deployment manager is the default profile, the Plug-ins installation wizard looks for a managed custom node in the deployment manager configuration. If the deployment manager does not have a managed custom node, the Plug-ins installation wizard looks for a managed application server node. If the deployment manager does not have a managed node, then the Plug-ins installation wizard classifies the installation as a remote installation.

Start the deployment manager and the node agent for the managed node. The deployment manager and the node must be running to successfully change its configuration.

About this task

Use the following procedure to install the Web server plug-in, configure the Web server, and create a Web server definition in the default profile.

1. Log on to the operating system. If you are installing as a non-root or non-administrative user, then there are certain limitations. See the documentation for non-root installation for more information.

AIX **HP-UX** **Linux** **Solaris** In addition, select a umask that allows the owner to read/write to the files, and allows others to access them according to the prevailing system policy. For root, a umask of 022 is recommended. For non-root users, a umask of 002 or 022 could be used, depending on whether or not the users share the group. To verify the umask setting, issue the following command:

```
umask
```

To set the umask setting to 022, issue the following command:

```
umask 022
```

Windows When installing as an administrative user on a Windows operating system, a Windows service is automatically created to autostart the application server. The installer user account must have the following advanced user rights:

- Act as part of the operating system
- Log on as a service

For example, on some Windows operating systems, click **Administrative Tools > Local Security Policy > User Rights Assignments** to set the advanced options. See your Windows operating system documentation for more information.

Windows If you plan to run the application server as a Windows service, do not install from a user ID that contains spaces. A user ID with spaces cannot be validated. Such a user ID is not allowed to continue the installation. To work around this restriction, install with a user ID that does not contain spaces.

2. Install the WebSphere Application Server Network Deployment product.

Read the "Installing the product and additional software" topic for more information.

3. Create a deployment manager profile as the first profile on the machine.

4. Install the IBM HTTP Server or another supported Web server.

5. Launch the Plug-ins installation wizard on the machine with the Web server.

Select the Plug-ins installation wizard from the launchpad or change directories to the plugin directory on the product disc or in the downloaded installation image and issue the install command.

6. Clear the check box for the roadmap or select the check box to view the roadmap, then click **Next**.

If you are unsure of which installation scenario to follow, display the roadmap instead. Print and keep the roadmap as a handy overview of the installation steps.

Press **Ctrl-P** to print the roadmap if the Web browser navigation controls and the menu bar are not present on the browser window that displays the Plug-ins roadmap. Press **Ctrl-W** to close the browser window if the navigation controls and the menu bar do not display. Or close the browser window with the window control in the title bar.

7. Read the license agreement and accept the agreement if you agree to its terms. Click **Next** when you are finished.

8. If your system does not pass the prerequisites check, stop the installation, correct any problems, and restart the installation. If your system passes the prerequisites check, click **Next**.

Look for the appropriate log file for information about missing prerequisites:

- If you stop the installation, see the temporaryPluginInstallLog.txt file in the temporary directory of the user who installed the plug-ins. For example, the /tmp/temporaryPluginInstallLog.txt file might exist if the root user installed the plug-ins on an operating system such as AIX or Linux.
- If you continue the installation in spite of warnings about missing prerequisites, see the *plugins_root/logs/install/log.txt* file after the installation is complete.

Read the "Troubleshooting installation" topic for more information about log files.

9. Select the type of Web server that you are configuring and click **Next**.

The Plug-ins installation wizard panel prompts you to identify the *Web servers* to configure. Actually you can select only one Web server each time you run the Plug-ins installation wizard.

Stop any Web server while you are configuring it. A step later in the procedure directs you to start the Web server as you begin the snoop servlet test.

If you select the Web server identification option labeled **None**, the Web server installs the binary plug-ins but does not configure the Web server.

10. Select **Application Server machine (local)** and click **Next**.

11. Accept the default location for the installation root directory for the plug-ins. Click **Next**.

You can type another new directory or click **Browse** to select an empty directory. The fully qualified path identifies the plug-ins installation root directory.

The default location is shown in "Directory conventions," on page 419.

Note: The installation directory cannot contain any unsupported characters. See "Object names: what the name string cannot contain" for more information.

A possibility exists that the Web server might run on a platform that WebSphere Application Server does not support.

12. Click **Browse** on the Application Server installation location panel to browse for the location of the deployment manager, if necessary. Click **Next** when the installation root directory is correct.

The fully qualified path identifies the installation root directory for the Network Deployment product core files.

13. Click **Browse** to select the configuration file for your Web server, verify that the Web server port is correct, and then click **Next** when you are finished.

Select the file and not just the directory of the file. Some Web servers have two configuration files and require you to browse for each file.

The following list shows configuration files for supported Web servers:

Apache HTTP Server

apache_root/config/httpd.conf

Domino Web Server

names.nsf and Notes.jar

The wizard prompts for the notes.jar file. The actual name is Notes.jar.

The Plug-ins installation wizard verifies that the files exist but the wizard does not validate either file.

IBM HTTP Server

Microsoft Internet Information Services (IIS)

The Plug-ins installation wizard can determine the correct files to edit.

Sun Java System Web Server (formerly Sun ONE Web Server and iPlanet Web Server) Version 6.0 and later

obj.conf and magnus.conf

The wizard displays a naming panel for the nickname of the Web server definition.

14. Specify a nickname for the Web server and click **Next**.

The wizard uses the value to name configuration folders in the plug-ins installation root directory. The wizard also uses the name within the deployment manager as the name of the Web server definition.

15. Accept the location for the plugin-cfg.xml file and click **Next**.

See "Plug-ins configuration" on page 16 for a description of the logic that determines what path is configured by default. The wizard determines the characteristics of the deployment manager to determine the best path for the file.

When the deployment manager is the default profile, the path is:

```
plugins_root/config/  
web_server_name/plugin-cfg.xml
```

Accept the default value.

Note: If there is a managed custom node on the deployment manager machine, the Plug-ins installation wizard uses the following file path:

```
profile_root  
/config/cells/cell_name/nodes/  
node_name_of_custom_profile/servers/  
web_server_name/plugin-cfg.xml
```

In this case, accept the path and resume the procedure at this point in "Configuring a Web server and a custom profile on the same machine" on page 53.

16. Click **Next** after verifying the characteristics of the plug-ins installation or click **Back** to make changes.
You can use the administrative console of the deployment manager to delete an existing Web server or to create new ones. Federated nodes can have more than one Web server definition.
17. Click **Next** on the pre-installation summary panel to begin the installation or click **Back** to change any characteristics of the installation.
The wizard begins installing the plug-ins and configuring the Web server and the deployment manager.
The wizard shows an installation status panel as it installs the plug-ins.
The wizard displays the Installation summary panel at the completion of the installation.
18. After the wizard installs the code and creates the uninstaller program, examine the post-installation summary panel. Click **Next** when you are finished to display the Plug-ins installation roadmap.
The Plug-ins installation wizard installs the binary plug-in module. On a Linux system, for example, the installation creates the *plugins_root* directory. The *plugins_root/config/Web_server_name* directory contains the plugin-cfg.xml file.
The wizard displays the name and location of the configuration script and the plugin-cfg.xml file. The wizard also displays the type of Web server that is configured and the nickname of the Web server.
If a problem occurs and the installation is unsuccessful, examine the logs in the *plugins_root/logs* directory. Correct any problems and reinstall.
19. Close the road map and click **Finish** to exit the wizard.
Log files from the installation are in the *plugins_root/logs/install* directory.
20. Complete the installation by creating the Web server definition.
You must create an application server profile or a custom profile and federate the node before you can use the administrative console of the deployment manager to create a Web server definition. The same is true for running the configuration script that the Plug-ins installation wizard created. You must assign the Web server to a managed node when you create it.
The managed node must exist before running the Plug-ins installation wizard. Otherwise, the installation is considered a remote installation.
If you install the plug-in, save the script to run after you create a managed node. Otherwise an error occurs. Before starting the Web server, wait for these actions to occur:
 - The script runs successfully.
 - The script creates the Web server definition on the managed node.
 - Node synchronization occurs.
 Adding the node starts the nodeagent process. If the node agent is not running for some reason, start the node.

Note: If you want the Web server to handle requests for an application for multiple managed nodes, install the application on each managed node and on the Web server definition.
The script already contains all of the information that you must gather when using the administrative console option.
Select one of the following options:
 - **Using the administrative console**
Click **Servers > Web servers > New** and use the Create new Web server entry wizard to create the Web server definition.
 - **Running the configuration script**
If the node has only a deployment manager profile, then the plug-ins installer reverts to a remote plug-in configuration. You must manually copy the *plugins_root/bin/configureweb_server_name.sh* script or the *plugins_root/bin/configureweb_server_name.bat* script to the *app_server_root/bin* directory of the deployment manager to run the script.

Issue the appropriate command to configure the Web server.

- `AIX HP-UX Linux Solaris ./app_server_root/bin/configureweb_server_name.sh`
- `Windows app_server_root\bin\configureweb_server_name.bat`

If you have enabled security or changed the default JMX connector type, edit the script and include the appropriate parameters on the wsadmin command.

21. **Domino Web server only:** Set the WAS_PLUGIN_CONFIG_FILE environment variable.

On platforms such as AIX or Linux, sourcing a script to the parent shell allows child processes to inherit the exported variables. On Windows systems, run the script as you would run any other command. Sourcing is automatic on Windows systems.

a. Open a command window.

b. Change directories to the plug-ins installation root directory.

c. Issue the appropriate command for the `plugins_root/bin/setupPluginCfg.sh` script:

- `AIX HP-UX Solaris . plugins_root/bin/setupPluginCfg.sh` (Notice the space between the period and the installation root directory.)
- `Linux source plugins_root/bin/setupPluginCfg.sh`

The script is also in the `lotus_root/notesdata` directory on operating systems such as AIX or Linux.

Issue the appropriate command for the script before starting the Domino Web Server.

22. From the administrative console of the deployment manager, click **System administration > Save Changes to Master Repository > Synchronize changes with Nodes > Save**.

23. Start the Snoop servlet to verify the ability of the Web server to retrieve an application from the Application Server.

Test your environment by starting your Application Server, your Web server, and using the snoop servlet with an IP address.

- a. Start the Application Server. In a Network Deployment environment, the Snoop servlet is available in the cell only if you included the DefaultApplication when adding the Application Server to the cell. The `-includeapps` option for the `addNode` command migrates the DefaultApplication to the cell. If the application is not present, skip this step.

Change directories to the `profile_root/bin` directory and run the `startServer` command:

- `AIX HP-UX Linux Solaris ./startServer.sh server1`
- `Windows startServer server1`

- b. Start the IBM HTTP Server or the Web server that you are using.

To start the IBM HTTP Server from the command line:

Access the `apache` and `apachectl` commands in the `IBMHttpServer/bin` directory.

- `AIX HP-UX Linux Solaris ./apachectl start`
- `Windows apache`

- c. Point your browser to `http://localhost:9080/snoop` to test the internal HTTP transport provided by the Application Server. Point your browser to `http://Host_name_of_Web_server_machine/snoop` to test the Web server plug-in.

The HTTP Transport port is 9080 by default and must be unique for every profile. The port is associated with a virtual host named `default_host`, which is configured to host the installed DefaultApplication and any installed Samples. The snoop servlet is part of the DefaultApplication. Change the port to match your actual HTTP Transport port.

- d. Verify that snoop is running.

Either Web address should display the Snoop Servlet - Request/Client Information page.

- e. **Remote IBM HTTP Server only:**

Verify that the automatic propagation function can work on a remote IBM HTTP Server by using the following steps. This procedure is not necessary for local Web servers.

- 1) Create a user=adminUser, password=adminPassword in the *IHS_root/conf/admin.passwd* file. For example: `c:\ws\ihs60\bin\htpasswd -cb c:\ws\ihs60\conf\admin.passwd adminUser adminPassword`
- 2) Use the administrative console of the deployment manager or the Application Server to enter the User ID and password information that you created for the administrative user of IBM HTTP Server. Go to **Servers > Web server > Web_server_definition > Remote Web server administration**. Set the following values: admin Port=8008, User Id=adminUser, Password=adminPassword.
- 3) Set the correct read/write permissions for the *httpd.conf* file and the *plugin-cfg.xml* file. See the *IHS_root/logs/admin_error.log* file for more information.

Automatic propagation of the plug-in configuration file requires the IBM HTTP administrative server to be up and running. If you are managing an IBM HTTP Server using the WebSphere Application Server administrative console, the following error might display:

```
"Could not connect to IHS Administration server error"
```

Perform the following procedure to correct the error:

- 1) Verify that the IBM HTTP Server administration server is running.
- 2) Verify that the Web server host name and the port that is defined in the WebSphere Application Server administrative console matches the IBM HTTP Server administration host name and port.
- 3) Verify that the fire wall is not preventing you from accessing the IBM HTTP Server administration server from the WebSphere Application Server administrative console.
- 4) Verify that the user ID and password that is specified in the WebSphere Application Server administrative console under remote managed, is created in the *admin.passwd* file, using the *htpasswd* command.
- 5) If you are trying to connect securely, verify that you export the IBM HTTP Server administration server keydb personal certificate into the WebSphere Application Server key database as a signer certificate. This key database is specified by the *com.ibm.ssl.trustStore* directive in the *sas.client.props* file in the profile where your administrative console is running. This consideration is primarily for self-signed certificates.
- 6) If you still have problems, check the IBM HTTP Server *admin_error.log* file and the WebSphere Application Server logs (*trace.log* file) to determine the cause of the problem.

Results

The installation of the binary plug-in modules results in the creation of the *Plugins* directory and several subdirectories. The following directories are among those created on a Linux system, for example:

- *plugins_root/uninstall* contains the uninstaller program
- *plugins_root/bin* contains the binary plug-ins for all supported Web servers
- *plugins_root/logs* contains log files
- *plugins_root/properties* contains version information
- *plugins_root/roadmap* contains the roadmap for the Plug-ins installation wizard

The Plug-ins installation wizard configures the Web server to use the *plugins_root/plugin-cfg.xml* file.

What to do next

After installing the binary plug-in for the local Web server, you must create a managed node before you can successfully run the configuration script and use the Web server.

See “Plug-ins configuration” on page 16 for an overview of the installation procedure.

See “Web server configuration” on page 24 for more information about the files involved in configuring a Web server.

See “Editing Web server configuration files” on page 81 for information about how the Plug-ins installation wizard configures supported Web servers.

See “Installing Web server plug-ins” on page 6 for information about other installation scenarios for installing Web server plug-ins.

Troubleshooting Web server plug-ins installation and removal

This topic describes troubleshooting the installation and removal of the Web server plug-ins for WebSphere Application Server.

About this task

This procedure is divided into three parts:

- Troubleshooting plug-ins installation
- Troubleshooting plug-ins removal
- Miscellaneous messages, tips, and hints

Log files for Web server plug-ins for WebSphere Application Server

The following log files are in the *plugins_root/logs/install* directory:

Table 6. Plug-in log files

File name	Description
log.txt	Records all of the ISMP events that occur during the installation. The log also describes whether the installation was local or remote. Messages at the end of the file indicate whether manual configuration steps are required to complete the installation.
masterConfigurationLog.txt	Records all of the configuration events that occur during the installation.
installGSKit.log	Records events that occur during the installation of the GSKit code.
installWeb_serverPlugin.log	Records events that occur during the installation of a Web server plug-in, where <i>Web_server</i> is the Web server you are installing. <i>Web_server</i> can have one of the following values: <ul style="list-style-type: none"> • APACHE • IHS • IIS • SUNONE • DOMINO
configure_Web_server_webserver.log	Records events that occur during the configuration of a Web server plug-in. The name of the file varies to reflect the Web server.

• Troubleshooting plug-ins installation

1. Are you reinstalling with an INSTCONFSUCCESS message but no files are being installed?

If you do not use the Web server Plug-ins uninstaller program to uninstall the plug-ins, a reinstall can fail with an INSTCONFSUCCESS message.

The Web server Plug-ins installer is an Install Shield for Multiplatforms (ISMP) wizard. The wizard uses the *vpd.properties* file or the operating system registry to determine if the plug-ins are installed. If you do not use the uninstaller program for the plug-ins, neither the *vpd.properties* file nor the operating system registry is updated properly. You might see a *plugins_root/logs/install/log.txt* with content similar to the following example:

```
Plugin.Install, com.installshield.product.service.product.
  PureJavaProductServiceImpl$InstallProduct, msg1,
  Did not replace installed object (IBM WebSphere Application
  Server - Plugins) with object (IBM WebSphere Application
```



```

Server - Plugins)
Plugin.Install, ... msg1, Did not replace installed object
(WebServer Plugin Binaries and Configurations) with
object (WebServer Plugin Binaries and Configurations)
Plugin.Install... Did not replace installed object (GSKit) with object (GSKit)
Plugin.Install... Did not replace installed object (LAP Component) with
object (LAP Component)
Plugin.Install...Did not replace installed object (WebServer Plugin
Binaries) with object (WebServer Plugin Binaries)
Plugin.Install... Did not replace installed object (Additional Bytes
for non-HP) with object (Additional Bytes for non-HP)
Plugin.Install... Did not replace installed object (GSKit) with object (GSKit)
Plugin.Install... Did not replace installed object (Standalone JDK)
with object (Standalone JDK)

```

Use the uninstaller program in the *plugins_root/uninstPlugin* directory to uninstall the Web server Plug-ins. For example, do not delete the *plugins_root* directory to uninstall the plug-ins without running the installer program first.

If you have this problem, follow the uninstall procedure and the manual uninstall procedure to clean up your system before reinstalling the Web server Plug-ins.

2. Does the installation image have the proper directories?

The following directories are present on the installation image of the full product. The directories in bold are required for a successful installation of the plug-ins:

- *parent_directory*/WAS
- *parent_directory*/IHS
- ***parent_directory*/plugin**
- *parent_directory*/AppClient
- ***parent_directory*/JDK**
- ***parent_directory*/GSKit**

Note: The *parent_directory* variable is the directory where you can unpack the images. All of the directories in the list must have the same parent directory.

If the directories are not present, or if the directories are empty, download a new installation image or discuss the product CD that you are using with someone who is knowledgeable about its creation. The IBM product discs are certified.

Reinstall using the IBM product disc.

Symptoms that can occur when the JDK directory is missing or is not used: If the *plugins_root/logs/install/log.txt* log file records the following errors, the installer program did not use the correct Java 2 SDK, which is in the JDK directory on the installation image:

```

Plugin.Install, com.ibm.ws.install.ni.ismp.actions.
  ISMPComponentizedFileRepositoryDeployAction, msg1,
  Processing component: prereq.jdk
Plugin.Install, com.ibm.ws.install.ni.ismp.actions.
  ISMPComponentizedFileRepositoryDeployAction, err,
  Component not found: prereq.jdk
Plugin.Install, com.ibm.ws.install.ni.ismp.actions.
  ISMPComponentizedFileRepositoryDeployAction, err,
  ComponentNotFoundException.message

```

The installation program can pick up another Java 2 SDK 1.4 on the system when running from the command prompt in terminal window that already has set the Java 2 SDK for the environment. Another way to point the installation program to a particular SDK is with the *-is:javahome* option for the Install Shield for Multiplatforms (ISMP) wizard.

In either case, the JDK directory on the installation image is not being used. When the installation attempts to install the *prereq.jdk* component, the wizard cannot find the JDK directory and throws the error in the *plugins_root/logs/install/log.txt* file.

Verify the cause of the error by examining the CURRENT_WORKING_DIRECTORY value and the JAVA_INSTALL_PATH value in the log.txt file. The working directory value is typically the CD_root/plugin directory. An erroneous Java path might be non_CD_root/java/jre. The correct Java path is the CD_root/JDK/repository/prereq.jdk/java/jre directory.

In such a case:

- a. Verify that the JDK directory is on the product disc.
- b. Close the current command window.
- c. Open a new command window.
- d. Change directories to the plugin directory on the product disc.
- e. Restart the installation: ./install

Symptoms that occur when the GSKit directory is missing: If the JDK directory is present, but the GSKit directory is not present, the installation is only partially successful, as shown in the logs by the INSTCONFPARTIALSUCCESS indicator.

The *plugins_root/logs/install/masterConfigurationLog.txt* log file shows that the 99SGSKitInstall.ant script failed to run. Complete the full installation by manually installing GSKit. The *plugins_root/logs/install/installGSKit.log* file shows the command to use for the manual installation in the GSKIT 7 entry.

3. Is there any sign that the installation occurred?

If not, look for the temporaryPluginInstallLog.txt file in the temporary directory of the user who installed the plug-ins.

For example, the /tmp/temporaryPluginInstallLog.txt file might exist if the root user installed the plug-ins on a system such as AIX or Linux.

This log is of particular interest after a silent installation. Suppose that the silent installation responsefile.txt file has incorrect entries. The installation cannot succeed. The cause of the problem is recorded in the temporaryPluginInstallLog.txt file.

If the responsefile.txt does not pass validation, the failure is logged as an INSTCONFFAILED entry. The installation does not occur. Correct the failure and run the silent installation again.

You might have to start with a new copy of the file that is on the product disc if you cannot get your copy to work.

4. Does a local installation attempt result in an INSTCONFPARTIALSUCCESS status, reporting a **98SConfigureWebserverDefinition** failure in log.txt?

The plugin installation log file shows a code of INSTCONFPARTIALSUCCESS, and the following error message is found in *plugins_root/logs/install/log.txt*. This message states that the installation process was successful, but one of the configuration scripts responsible for configuring the plug-ins failed to complete. In this case, the name of the script which failed to complete is 98SConfigureWebserverDefinition.

```
Config action failed: 98SConfigureWebserverDefinition -
install_root\properties\version\nif\config\install\
98SConfigureWebserverDefinition.ant
Current install/uninstall process is successful. Process type is: install
```

This issue occurs under the following conditions:

- WebSphere Application Server is installed on the same system on which the plugin is being installed.
- The plug-ins are being installed in local configuration mode.
- The existing application server profile, specified in the installation options for the plug-ins, is installed to a location outside of the application server. In other words, the specified profile is not in the default profile location in *app_server_root/profiles/profile_name*. This might occur in IBM products which extend the application server as part of their installation, like WebSphere Portal Server for example.

Although the plug-ins installation is intact, the existing application server is not automatically configured to use the new plug-ins. There are two options to resolve the problem:

- Uninstall and reinstall the plug-ins in remote mode, then manually invoke the configuration script. Even though the application server is on the same machine as the plug-ins, you can still run the installation in remote mode. Read the remote configuration topic, “Configuring a Web server and an application server on separate machines (remote)” on page 28, for more information.
 - Keep the existing plug-ins, and manually invoke the configuration script as indicated on step 20 and afterwards in the remote configuration topic.
5. Does the installation result in an INSTCONFPARTIALSUCCESS status?
- a. Look for errors in the log.txt file.
 - b. Look for an entry in the log.txt file that shows the location of the masterConfigurationLog.txt file.
 - c. Edit the masterConfigurationLog.txt file.
 - d. Starting at the end of the file, scan towards the front of the file looking for configuration scripts that could not run.

For example, the following stanza shows a configuration script that could not run:

```
<record>
<date>2004-10-08T10:31:43</date>
<millis>1097245903200</millis>
<sequence>189</sequence>
<logger>com.ibm.ws.install.configmanager.ConfigManager</logger>
<level>INFO</level>
<class>com.ibm.ws.install.configmanager.ConfigManager</class>
<method>dumpNonFatalFailedActionsInfoToLogFile</method>
<thread>10</thread>
<message>This action failed to execute:
  C:\Plugins\properties\version\install\plugin\7.0.0.0\
  config\full\install\99SGSKitInstall.ant</message>
</record>
```

- e. A configuration script that fails to run is likely the cause of a partially successful installation status. To debug the example, look at the installGSKit.log, which is the log file for GSKit. Look for signs of a failed installation to determine if you can correct the problem.

Configuration script log files and recovery procedures: The following configuration scripts run during the configuration of supported Web servers.

Configuration script: 98SConfigureWebserverDefinition.ant

Log file to investigate for error	Recovery
<p>configureWeb_server_type_webserver.log</p> <p>For example, the file might be named:</p> <p>configure_IHS_webserver.log</p> <p>Logs the Web server definition creation.</p>	<ol style="list-style-type: none"> 1. If the Web server definition partially exists, delete the Web server definition. 2. Run the Web server definition script manually. The Web server definition is referenced in the script. The script is in the plugins_root/bin directory, with a name similar to the following convention: <ul style="list-style-type: none"> • AIX HP-UX Linux Solaris configureWeb_server_definition_name.sh • Windows configureWeb_server_definition_name.bat

Configuration script: 99SBootStrapPluginsSunOne.ant

Log file to investigate for error	Recovery
<p>installSunOnePlugin.log</p> <p>Investigate the log file to determine what failed:</p> <ol style="list-style-type: none">1. If the error is due to plugin-cfg.xml not existing, investigate why this file is not present.2. If the error is due to the Web server definition creation failing, address this as discussed under the 98SConfigureWebserverDefinition.ant description.3. If the Web server configuration files do not update, investigate the files to verify that they exist and that the file permissions are correct.	<ul style="list-style-type: none">• You can manually configure the Web server.• You can rerun the Plug-ins installation wizard. Select the existing plug-ins installation root directory. Doing so results in the Web server being configured to use the existing binaries.

Configuration script: 99SBootStrapPluginsDomino6.ant

Log file to investigate for error	Recovery
<p>installDomino6Plugin.log</p> <p>Investigate the log file to determine what failed:</p> <ol style="list-style-type: none">1. If the error is due to plugin-cfg.xml not existing, investigate why this file is not present.2. If the error is due to the Web server definition creation failing, address this as discussed under the 98SConfigureWebserverDefinition.ant description.3. If the Web server configuration files do not update, investigate the files to verify that they exist and that the file permissions are correct.	<ul style="list-style-type: none">• You can manually configure the Web server.• You can rerun the Plug-ins installation wizard. Select the existing plug-ins installation root directory. Doing so results in the Web server being configured to use the existing binaries.

Configuration script: 99SBootStrapPluginsDomino5.ant

Log file to investigate for error	Recovery
<p>installDomino5Plugin.log</p> <p>Investigate the log file to determine what failed:</p> <ol style="list-style-type: none">1. If the error is due to plugin-cfg.xml not existing, investigate why this file is not present.2. If the error is due to the Web server definition creation failing, address this as discussed under the 98SConfigureWebserverDefinition.ant description.3. If the Web server configuration files do not update, investigate the files to verify that they exist and that the file permissions are correct.	<ul style="list-style-type: none">• You can manually configure the Web server.• You can rerun the Plug-ins installation wizard. Select the existing plug-ins installation root directory. Doing so results in the Web server being configured to use the existing binaries.

Configuration script: 99SBootStrapPluginsIIS6.ant

Log file to investigate for error	Recovery
<p>installIIS6Plugin.log</p> <p>Investigate the log file to determine what failed:</p> <ol style="list-style-type: none">1. If the error is due to plugin-cfg.xml not existing, investigate why this file is not present.2. If the error is due to the Web server definition creation failing, address this as discussed under the 98SConfigureWebserverDefinition.ant description.3. Start the Internet Information Services (IIS) application and investigate the IIS configuration.	<ul style="list-style-type: none">• You can manually configure the Web server.• You can rerun the Plug-ins installation wizard. Select the existing plug-ins installation root directory. Doing so results in the Web server being configured to use the existing binaries.

Configuration script: 99SBootStrapPluginsIIS5.ant

Log file to investigate for error	Recovery
<p>installIIS5Plugin.log</p> <p>Investigate the log file to determine what failed:</p> <ol style="list-style-type: none">1. If the error is due to plugin-cfg.xml not existing, investigate why this file is not present.2. If the error is due to the Web server definition creation failing, address this as discussed under the 98SConfigureWebserverDefinition.ant description.3. Start the IIS application and investigate the IIS configuration.	<ul style="list-style-type: none">• You can manually configure the Web server.• You can rerun the Plug-ins installation wizard. Select the existing plug-ins installation root directory. Doing so results in the Web server being configured to use the existing binaries.

Configuration script: 99SBootStrapPluginsApache.ant

Log file to investigate for error	Recovery
<p>installApachePlugin.log</p> <p>Investigate the log file to determine what failed:</p> <ol style="list-style-type: none">1. If the error is due to plugin-cfg.xml not existing, investigate why this file is not present.2. If the error is due to the Web server definition creation failing, address this as discussed under the 98SConfigureWebserverDefinition.ant description.3. If the Web server configuration files do not update, investigate the files to verify that they exist and that the file permissions are correct.	<ul style="list-style-type: none">• You can manually configure the Web server.• You can rerun the Plug-ins installation wizard. Select the existing plug-ins installation root directory. Doing so results in the Web server being configured to use the existing binaries.

Configuration script: 99SBootStrapPluginsIHS.ant

Log file to investigate for error	Recovery
<p>installIHSPlugin.log</p> <p>Investigate the log file to determine what failed:</p> <ol style="list-style-type: none"> 1. If the error is due to plugin-cfg.xml not existing, investigate why this file is not present. 2. If the error is due to the Web server definition creation failing, address this as discussed under the 98SConfigureWebserverDefinition.ant description. 3. If the Web server configuration files do not update, investigate the files to verify that they exist and that the file permissions are correct. 	<ul style="list-style-type: none"> • You can manually configure the Web server. • You can rerun the Plug-ins installation wizard. Select the existing plug-ins installation root directory. Doing so results in the Web server being configured to use the existing binaries.

Configuration script: 99SGSKitInstall.ant

Log file to investigate for error	Recovery
<p>installGSKit.log</p> <p>Investigate the log file to determine what failed:</p> <ol style="list-style-type: none"> 1. In cases where the CD layout is not correct, then the gskit installer program cannot be found. GSKit fails to install. 2. In cases where GSKit is already installed, the installation might incorrectly report a failure. If the logs show that an installation already existed, you can safely ignore the error. 	<p>Manually install GSKit by running the installer program. The installGSKit.log file shows the installer program that runs to install GSKit.</p> <p>AIX HP-UX Linux Solaris Run the following command from the /cdrom/GSKit directory:</p> <ul style="list-style-type: none"> • /cdrom/GSKit/gskit.sh <p>Windows Run the following command from the "E:\GSKit\ directory, assuming the E drive is the CD-ROM drive:</p> <ul style="list-style-type: none"> • "E:\GSKit\Setup.exe" WASPLUGIN60_041008085014 "C:\Program Files" -s -z -f1"E:\GSKit\setup.iss"

Solaris Configuration script: 99SSolarisGSKitInstall4.ant

Log file to investigate for error	Recovery
<p>On Solaris, the installation is logged to the installGSKit.log file.</p> <p>Investigate the log file to determine what failed:</p> <ol style="list-style-type: none"> 1. In cases where the CD layout is not correct, then the gskit installer program cannot be found. GSKit fails to install. 2. In cases where GSKit is already installed, the installation might incorrectly report a failure. If the logs show that an installation already existed, you can safely ignore the error. 	<p>Manually install GSKit by running the installer program.</p> <p>Solaris Run the following command from the /cdrom/GSKit directory:</p> <ul style="list-style-type: none"> • /cdrom/GSKit/gskit4.sh

Log file to investigate for error	Recovery
<p>This script does not have an associated log file.</p> <p>The script creates the Windows registry entry for the Web server plug-ins for WebSphere Application Server.</p> <p>HKEY_LOCAL_MACHINE > Software > IBM > Web server Plug-ins for IBM WebSphere Application Server > 7.0.0.0</p>	<p>Add the HKEY_LOCAL_MACHINE > Software > IBM > Web server Plug-ins for IBM WebSphere Application Server > 7.0.0.0 key manually with the actual value of the installation location.</p> <p>Set the following registry values under this key:</p> <ul style="list-style-type: none"> • Name: <i>plugins_root</i> • Type: REG_SZ • Data: <i>plugins_root</i>

Configuration script: 99SModifySetupCmdLine.ant

Log file to investigate for error	Recovery
<p>This script does not have an associated log file.</p> <p>Investigate the <i>plugins_root/bin/setupCmdLine.sh</i> script to verify that the PLUGIN_HOME variable is set to the Web server plug-ins installation root directory.</p> <p>Verify that file permissions are 755 (rwxr-xr-x) on systems such as AIX or Linux.</p>	<ul style="list-style-type: none"> • You can edit the <i>setupCmdLine</i> file, replacing \$(PLUGIN_HOME) with the location to the plug-ins installation root directory. • Change the file permissions to 755. Issue the following command from the <i>plugins_root/bin</i> directory: <pre>chmod 755 setupCmdLine.sh</pre>

If you do not find the cause of the problem, open the *plugins_root/logs/web_server_name* directory. The directory is empty until the Web server loads the plug-in and errors occur. The plug-in then creates the *http_plugin.log* file to record the errors. If no errors occur, the directory is empty. Examine any relevant files for error entries. Correct any errors and reinstall.

6. Does the installation result in a INSTCONFFAILED status?
 - a. This is a serious failure of the installation.
 - b. Analyze the log files to determine the problem.
 - c. Uninstall the plug-ins.
 - d. Delete the plug-ins installation root directory.
 - e. Install the plug-ins again.
7. Do errors occur when you start the snoop application?

See "Start the Snoop servlet" in the installation troubleshooting topic.

If errors occur, look at the *plugins_root/logs/web_server_name/http_plugin.log* file for causes. Also investigate the WebSphere Application Server installation logs and the logs for your supported Web server.
8. AIX HP-UX Linux Solaris Do you have the correct file permissions for the *configureWeb_server_definition_name* script?

On operating systems such as AIX or Linux, copying the *configureWeb_server_definition_name.sh* script from one operating system, such as AIX, to another, such as HP-UX, can cause the file permissions of the script to be invalid for running the script.

Verify that file permissions are 755 (rwxr-xr-x) on systems such as AIX or Linux.
9. AIX HP-UX Linux Solaris Are you using the *configureWeb_server_definition_name* script when the Web server machine is running dynamic host configuration protocol (DHCP)?

The *configureweb_server_name.sh* script can contain null values for the name of the host machine of the Web server when using DHCP. Examine the script to see if the last few parameters include the word null. If so, you have this problem.

The `plugins.install` log file might also have an entry for the problem:

```
(MMM DD, YYYY HH:MM:SS AM|PM),  
Plugin.Install,  
com.ibm.ws.install.ni.ismp.actions.ISMPLogFileAction,  
msg1, WEB_SERVER_HOSTNAME : null
```

If the Network Deployment product cannot resolve the host name of the server, problems can occur with such situations as adding or administering nodes, or the node agent contacting the application server. To resolve the host name, the product opens a port, or queries for an IP address. The product then waits for the operating system to return the correct information. The operating system might go to multiple places to find the IP address, but the product does not care about the order in which the operating system does this, if the correct information is returned. If the host name of the server cannot be resolved, refer to your network administration documentation to resolve the problem. The following additional information might help you ensure that the host name is resolved.

- Some operating systems create an association between the host name of the machine and the loopback address of 127.0.0.1. Red Hat installations create the association by default. The association is in the hosts file.

If in the hosts file mappings exist from the 127.0.0.1 IP address to a host name other than `localhost`, remove the mappings. The following example illustrates what might happen if the mappings are not removed: When a node agent communicates with the deployment manager, it sends its IP address to the deployment manager. The node agent resolves the node agent host name to 127.0.0.1 if the operating system returns a mapping for the host name from the hosts file. This resolution prevents the deployment manager from sending a message to the node agent because the 127.0.0.1 IP address is also the IP address for the local machine of the deployment manager.

AIX **HP-UX** **Linux** **Solaris** The hosts file is located at `/etc/hosts`.

Windows The hosts file is located at `\WINDOWS\system32\drivers\etc\hosts`.

- **AIX** The default AIX installation checks the domain name server (DNS) first to return the information to a server so that the server host name of that server or another server can be resolved. If the host name cannot be resolved or cannot be resolved in a reasonable amount of time, you can add the following statement to the `/etc/netsvc.conf` file so that the AIX operating system checks the local hosts file first for the host name.

```
hosts=local,bind
```

Perform the following steps to successfully create the configuration script for creating and configuring the Web server definition in the application server node:

- Uninstall the Web server plug-ins.
- Perform the suggested edits based on your operating system.
- Install the Web server plug-ins again.

• Troubleshooting plug-ins removal

1. Does the `plugins_root/logs/uninstall` directory have any log files?
If log files exist, examine them, correct the errors, and reinstall.
2. Is there any sign that the removal occurred?
If not, look for the `temporaryPluginUninstallLog.txt` file in the temporary directory of the user who installed the plug-ins.
For example, the `/tmp/temporaryPluginUninstallLog.txt` file might exist if the root user uninstalled the plug-ins on a system such as AIX or Linux.
3. Does the installation result in a `INSTCONF PARTIAL SUCCESS` status?
 - a. Look for errors in the `log.txt` file.
 - b. Look for an entry in the `log.txt` file that shows the location of the `masterConfigurationLog.txt` file.
 - c. Edit the `masterConfigurationLog.txt` file.

- d. Starting at the end of the file, scan towards the front of the file looking for configuration scripts that could not run.

For example, the following stanza shows a configuration script that could not run:

```
<record>
<date>2004-10-08T10:31:43</date>
<millis>1097245903200</millis>
<sequence>189</sequence>
<logger>com.ibm.ws.install.configmanager.ConfigManager</logger>
<level>INFO</level>
<class>com.ibm.ws.install.configmanager.ConfigManager</class>
<method>dumpNonFatalFailedActionsInfoToLogFile</method>
<thread>10</thread>
<message>This action failed to execute:
  C:\Plugins\properties\version\install\plugin\7.0.0.0\
  config\full\uninstall\99SGSKitUnInstall.ant</message>
</record>
```

- e. A script that fails to run is likely the cause of a partially successful installation status. To debug our example, look at the installGSKit.log, which is the log file for GSKit. Look for signs of a failed removal to determine if you can correct the problem.

Configuration script	Log file to investigate for error
99SBootStrapPluginsSunOneUninstall <i>uniqueID</i> .ant	uninstallWeb_server_typePlugin.log
99SBootStrapPluginsDomino6Uninstall <i>uniqueID</i> .ant	For example, the file might be named:
99SBootStrapPluginsDomino5Uninstall <i>uniqueID</i> .ant	uninstallIHSPlugin.log
99SBootStrapPluginsIIS6Uninstall <i>uniqueID</i> .ant	Logs the Web server deconfiguration events for the Web server.
99SBootStrapPluginsIIS5Uninstall <i>uniqueID</i> .ant	
99SBootStrapPluginsApacheUninstall <i>uniqueID</i> .ant	
99SBootStrapPluginsIHSUninstall <i>uniqueID</i> .ant	
99SGSKitUnInstall.ant	uninstallGSKit.log On Windows only, the Web server plugins GSKit key will be unregistered. GSKit is uninstalled if no other product is registered to use GSKit. On operating systems such as AIX or Linux, it is your responsibility to uninstall GSKit when no other products are using it.
90SDeleteWinRegPlugin.ant	This script does not have an associated log file. The script deletes the Windows registry entry for the Web server plug-ins for WebSphere Application Server. HKEY_LOCAL_MACHINE > Software > IBM > Web server Plug-ins for IBM WebSphere Application Server > 7.0.0.0

4. Did the uninstall procedure fail?

If the uninstall failed in any way, follow the manual uninstall steps to verify the system is clean and if necessary, remove Web server plug-in entries.

- **Miscellaneous messages, tips and hints**

Option	Description
Unable to load mod_was_ap20_http...	<p>This error means that the following actions have occurred:</p> <ul style="list-style-type: none"> • You have the 32-bit version of the Apache Web Server installed on a 64-bit platform. • You used the 64-bit platform CD to install the plug-in for the 32-bit Apache Web Server • You started the Apache Web Server and it could not load the 64-bit plug-in because it requires the 32-bit plug-in. <p>To fix the problem, install the 32-bit plug-in from the CD for the 32-bit platform. Or manually configure the Apache Web server by changing the httpd.conf file to refer to the correct plug-in. Change the 64bit folder name in the directive to 32bit to fix the reference. For example, change /opt/IBM/WebSphere/Plugins/bin/64bit/mod_was_ap20_http.so to /opt/IBM/WebSphere/Plugins/bin/32bit/mod_was_ap20_http.so on a Linux system.</p>
SAFE_BROWSER_EXCEPTION_CAUGHT	<p>This SAFE_BROWSER_EXCEPTION_CAUGHT error means that:</p> <ul style="list-style-type: none"> • If you do not have a browser, this error is thrown and the roadmap is not launched when it is selected during the installation of the plug-ins. • AIX HP-UX Linux Solaris If you have a supported browser that is not the Konqueror browser, the roadmap is launched. This message is thrown with the supported browser when it is not the Konqueror browser. You can safely ignore this exception and open the roadmap in a supported browser.
Solaris number.tmp.sorted	Solaris A limitation in ISMP does not remove temporary working files in the / root directory on Solaris operating systems. You can safely ignore or delete the files that follow this naming convention.

Results

This procedure results in troubleshooting the installation or removal of the Web server plug-ins for WebSphere Application Server.

What to do next

For current information available from IBM Support on known problems and their resolution, see the IBM Support page.

IBM Support has documents that can save you time gathering the information that you need to resolve a problem. Before opening a PMR, see the IBM Support page.

Web server plug-in response file

This topic describes the response file for performing a silent installation of the Web server plug-ins for WebSphere Application Server.

Install the product silently using an options response file.

The responsefile.txt file has directives that set installation options. Comments in the file describe how to set the string value for each directive.

Use the options file to run the Plug-ins installation wizard in silent mode, which is referred to as installing silently. The wizard reads the options file to determine responses and does not display the graphical user interface.

Location of the response file

The sample options response file is named responsefile.txt. The file is in the plugin directory on the product disc or in the downloaded installation image.

Mode of use

The Plug-ins installation wizard can read an existing options response file and run silently without displaying the graphical user interface.

Installing silently

The options file supplies the values to the plug-ins installation wizard when installing silently. The wizard reads the options file to determine responses and does not display the graphical user interface. Use the following command to use a copy of the options file named myresponsefile.txt for a silent installation:

```
install -options "myresponsefile.txt" -silent
```

Creating an operational environment

The installation of the plug-ins is a three-step process:

1. Installing the binary plug-in modules for supported Web servers
2. Configuring the Web servers to use the binary module to communicate with the application server
3. Creating a Web server definition in the application server

As you install an application, you can install it on the Web server definition in addition to the application server. All applications on the Web server definition are listed in its plug-in configuration file. After propagation, the real Web server can access the applications.

The sample options response file, responsefile.txt, controls installing the binary plug-ins, configuring the Web server, and creates a script for creating the Web server definition on a remote application server machine. The script is customized according to values supplied in the responsefile.txt file. The script is generated to run on the application server machine to create the Web server definition.

If the Web server is on the same machine as a stand-alone application server, the responsefile.txt file can create the Web server definition directly without creating a script.

To edit and use the response file for installing the plug-ins and configuring the Web server and application server, perform the following procedure:

1. Copy the responsefile.txt file from the plugins directory on the product disc to a place that you can easily identify on your machine.
2. Edit the file to customize the values for your installation.
3. Save the file.
4. Start the installation. For example:

```
install -options /tmp/plugins/myresponsefile.txt -silent
```

Windows On non-Windows operating systems, silent installations run in a synchronous process. The process does not return until the silent installation finishes. For a silent installation to run synchronously on Windows, issue the first of the following commands:

- **Synchronous processing:** START /WAIT install.exe -options "C:\temp\myresponsefile.txt" -silent

- **Asynchronous processing:** install -options "C:\temp\plugins\myresponsefile.txt" -silent
5. After the installation, examine the logs for success.

Logging

If no installation logs exist, refer to temporary log file, log.txt in your <userhome>/plglogs directory. You can also cause ISMP to record status about a problem that is preventing the installation from occurring, as described in the following section.

For example, if you start the silent installation without accepting the license in the -OPT silentInstallLicenseAcceptance="false" directive, the installation does not occur. The fact that the license entry was not accepted is recorded in log.txt in the <userhome>/plglogs directory.

If all validations pass, the installation occurs. Then, the Plug-ins installation wizard records installation events in the following log files. The log files are in the *plugins_root/logs/install* directory:

log.txt Records all of the ISMP events that occur during the installation. The log also describes whether the installation was local or remote. Messages at the end of the file indicate whether manual configuration steps are required to complete the installation.

Key elements to look for in the installation record are:

Manual steps warning

When the wizard requires you to run a script to create the Web server definition, the wizard refers to the fact that manual steps are required.

If manual steps are required, the name and location of the script that you must run are written in the log file at the end of the installation record.

Web server type

The log has a record of the Web server type, such as IHS for the IBM HTTP Server, for example.

Location of the plug-in configuration file

The log has a record of the plugin-cfg.xml file location currently in the Web server configuration.

installconfig.log

Lists all of the configuration events that occur during the installation.

installGSKit.log

Lists events that occur during the installation of the GSKit code.

The command line for the installation is listed when the installation occurs. The GSKit 7 installation record is written after the GSKIT 7 : entry in the log.

Solaris The GSKit 4 installation record is written after the GSKIT 4: entry in the log.

installWeb_server_typePlugin.log

Records events that occur during the installation of a Web server plug-in. The name of the file varies to reflect the Web server:

- installAPACHEPlugin.log
- installIHSPlugin.log
- installIISPlugin.log
- installSunOnePlugin.log
- installDomino5Plugin.log
- installDomino6Plugin.log
- installDomino7Plugin.log

Each log lists the following critical information:

- The plug-in binary module that is currently installed
- The current location of the plug-in configuration file that is configured for the Web server

configure_Web_server_type_webserver.log

Lists events that occur during the configuration of a Web server plug-in. The name of the file varies to reflect the Web server:

- configure_APACHE_webserver.log
- configure_IHS_webserver.log
- configure_IIS_webserver.log
- configure_SUNJAVASYSTEM_webserver.log
- configure_DOMINO_webserver.log

The configure_Web_server_type_webserver.log file reports the actions that the Plug-ins installation wizard performs as it updates the Web server configuration file.

In a remote scenario, this log is not present because you must run the script to create the Web server definition manually.

In a federated scenario, the script is created and this log is not present.

Information that ISMP can log when it cannot start the Plug-ins installation wizard

Certain events can prevent the installer from starting the installation wizard. Such an event is not enough disk space to launch the installation wizard, for example. If your installation fails and there is no information in the installation logs, use the -log parameter to record entries for events that cause the installer program to fail to start the installation wizard. The syntax of the install command for logging such events is:

```
install -options fully_qualified_options_response_file_name
        -silent
        -log # !fully_qualified_log_file_name @ALL
```

• **AIX**

```
install -options "/usr/IBM/WebSphere/silentFiles/myresponsefile.txt"
        -silent -log # !/usr/IBM/WebSphere/myOptionFiles/log.txt @ALL
```

• **Linux**

HP-UX

Solaris

```
install -options "/opt/IBM/WebSphere/silentFiles/myresponsefile.txt"
        -silent -log # !/opt/IBM/WebSphere/myOptionFiles/log.txt @ALL
```

• **Windows**

```
install.exe -options "C:\IBM\WebSphere\silentFiles\myresponsefile.txt"
            -silent -log # !C:\IBM\WebSphere\silentFiles\log.txt @ALL
```

Verify or troubleshoot the installation if the *plugins_root/logs/install/log.txt* file does not contain a record of any problems, but problems exist.

If the error happens early in the installation, look for the logs in the system temporary directory. The installation program copies the logs from the system temporary directory to the logs directory at the end of the installation.

Read the "Troubleshooting installation" topic and the "Installation component troubleshooting tips" topic for more information.

Response file user entry validation

Validation of the response file has been coded into the installation. If response file validation does not pass, the failure is recorded in the temporaryPluginInstallLog.txt file.

Accepted license agreement

Default directive setting

-OPT silentInstallLicenseAcceptance="false"

Valid setting

You must set this directive to true to accept the license and install the plug-ins.

Error identifier in temporaryPluginInstallLog.txt

INSTCONFFAILED : LICENSE _ NOT _ ACCEPTED.

Valid install type**Default directive setting**

-OPT installType="local"

Valid setting

You must set this directive to remote or local. Any other value fails validation.

Error identifier in temporaryPluginInstallLog.txt

INSTCONFFAILED : INVALID _ PLUGIN _ INSTALL _ TYPE _ SCENARIO _ SELECTED.

Valid application server installation location**Default directive setting**

-OPT wasExistingLocation="C:\Program Files\IBM\WebSphere\AppServer"

The setting varies per operating system.

Valid setting

If you set the -OPT installType directive to local, this validation checks that the path is a valid WebSphere Application Server Version 7 directory. Any other path fails validation for a local installation type.

Error identifier in temporaryPluginInstallLog.txt

INSTCONFFAILED : NON _ WAS7 _ DIRECTORY

Valid Web server configuration file 1**Default directive setting**

-OPT webServerConfigFile1="C:\Program Files\IBM\HTTPServer\conf\httpd.conf"

The setting varies per operating system. Valid file names for each Web server are:

- IBM HTTP Server: httpd.conf
- Apache: httpd.conf
- Domino: Notes.jar
- Sun One Web Server: obj.conf

Valid setting

Validation checks that the file exists. A known problem in ISMP validates a directory specification. However, you must identify the file to establish a working configuration.

Error identifier in temporaryPluginInstallLog.txt

INSTCONFFAILED : NON _ EXISTENT _ WS _ CONFIG _ FILE.

Valid Web server configuration file 2**Default directive setting**

-OPT webServerConfigFile2=""

Valid file names for affected Web servers are:

- Domino 7: names.nsf
- Domino 8: names.nsf
- Sun One Web Server: magnus.conf

Valid setting

Validation checks that the file exists. A known problem in ISMP validates a directory specification. However, you must identify the file to establish a working configuration.

Error identifier in temporaryPluginInstallLog.txt

INSTCONFFAILED : NON _ EXISTENT _ WS _ CONFIG _ FILE2.

Valid Web server definition name**Default directive setting**

-OPT webServerDefinition="webserver1"

Valid setting

Validation verifies that spaces do not exist in the Web server definition name. If spaces exist, validation fails.

Error identifier in temporaryPluginInstallLog.txt

INSTCONFFAILED : INVALID _ WEB _ SERVER _ DEFINITION _ NAME.

Verify that application mapping is true or false**Default directive setting**

-OPT mapWebserverToApplications="true"

Valid setting

If you set the directive to a value other than true or false, no error is generated or recorded in the temporaryPluginInstallLog.txt file. The Plug-ins installation wizard defaults the value to true and continues the installation.

Error identifier in temporaryPluginInstallLog.txt

None.

Valid Web server**Default directive setting**

-OPT webServerSelected="none"

Valid setting

You must specify the correct Web server for your operating system. Valid values include:

- **AIX** none, ihs, apache, domino7, domino8, sunone
- none, ihs, apache, domino7, domino8, sunone
- **HP-UX** none, ihs, apache, domino7, sunone
- **Solaris** none, ihs, apache, domino7, domino8 (not supported on x86_64), sunone
- **Windows** none, ihs, apache, domino7, domino8, iis6, iis7, sunone

Error identifier in temporaryPluginInstallLog.txt

INSTCONFFAILED : INVALID _ WEBSERVER _ SELECTED.

Usage notes

- The file is not a read-only file.
- Edit this file directly with your flat file editor of choice.
- The file is updated when you specify the -options parameter when using the plug-ins installation wizard.
- The file must exist to perform a silent installation. The installation program reads this file to determine installation option values when you install silently. Provide the fully qualified file path.
- Save the file in a location that you can identify when you specify the fully qualified path as part of the installation command.

Editing Web server configuration files

Edit Web server configuration files to configure a Web server.

Before you begin

The Plug-ins installation wizard automatically configures supported Web servers during the installation of the binary plug-in modules. Use this topic to understand how the wizard configures the Web server configuration files.

If you must change a configuration for some reason, you can run the Plug-ins installation wizard again to reconfigure the Web server, or you can edit the files. The recommended approach is to always use the Plug-ins installation wizard to configure the Web server configuration file. However, sometimes you must edit the files. An example is when you are installing a Web server definition for a non-default profile.

To support a non-default profile, edit the configuration to point the Web server to the correct location of the plug-in configuration file (plugin-cfg.xml). In this case, you would change the profile path from the default profile to the secondary profile.

About this task

This task points you to information on editing Web server configuration files. Select a link appropriate for your Web server.

- Configure Apache HTTP Server 2.0. See “Configuring Apache HTTP Server V2.0.”
- Configure Apache HTTP Server 2.2. See “Configuring Apache HTTP Server V2.2” on page 85.
- Configure Lotus Domino Web Server Version 5 and Version 6.x. See “Configuring Lotus Domino” on page 87.
- Configure IBM HTTP Server powered by Apache 2.x. See “Configuring IBM HTTP Server powered by Apache 2.x” on page 90.
- Configure IBM HTTP Server Version 6.x. See “Configuring IBM HTTP Server Version 6.x” on page 92.
- Configure IBM HTTP Server Version 7.0. See “Configuring IBM HTTP Server Version 7.0” on page 94.
- Configure Microsoft® Internet Information Services (IIS). See “Configuring Microsoft Internet Information Services (IIS)” on page 95.
- Configure Sun Java System Web Server (formerly Sun ONE and iPlanet). See “Configuring the Sun Java System Web Server” on page 98.

Results

You can use the Plug-ins installation wizard to automatically configure supported Web servers. You can also configure a Web servers by editing its configuration.

Configuring Apache HTTP Server V2.0

This topic describes how to change configuration settings for Apache HTTP Server Version 2.0.

Before you begin

Apache HTTP Server v2.0 is different from IBM HTTP Server (powered by Apache). Apache HTTP Server is not supported on IBM i. For details on configuring IBM HTTP Server (Powered by Apache), see “Configuring IBM HTTP Server powered by Apache 2.x” on page 90.

When you install the Web server plug-ins for WebSphere Application Server on a non-IBM i system, the Plug-ins installation wizard configures the Web server.

This topic describes how to configure the Apache HTTP Server Version 2.0 Web server. Other procedures in “Editing Web server configuration files” on page 81 describe configuring other supported Web servers.

Note:

- If you are using an Apache HTTP Server that supports 64-bit addressing, you must use the 64-bit CD provided with the WebSphere Application Server product to install the Apache Web server plug-in binaries. If you use the 32-bit CD, you will receive an error message indicating that the plug-in binaries did not load.
- If you are using an Apache HTTP Server that supports 32-bit addressing, you must use the 32-bit CD provided with the WebSphere Application Server product to install the Apache Web server plug-in binaries. If you use the 64-bit CD, you will receive an error message indicating that the plug-in binaries did not load.

A sample error message follows:

```
httpd: Syntax error on line XXX of /home/apache/conf/httpd.conf: Cannot
load /home/apache/Plugins/mod_was_ap20_http.sl into server: Invalid argument
```

The plug-in was tested with the threaded worker multi-processing module (MPM) on all platforms except Windows. The plug-in was tested with the default threaded MPM on Windows.

The plug-in works with the Apache 2 prefork MPM but works best with the worker MPM. The plug-in maintains connection pools to backend WebSphere Application Servers and uses in-memory caching. These plug-in functions perform most efficiently when Apache 2.0 is configured to use a single child process with the ThreadsPerChild value equal to the MaxClients value. The plug-in can be used with the prefork MPM or the worker MPM that is configured with multiple child processes, but at reduced efficiency.

Compatibility Statement The plug-in works with versions of the Apache HTTP Server that claim full binary compatibility with Apache 2.0.47 and later, which are built with compilers and compiler options that are compatible with those used to build the plug-in.

About this task

Perform the step that configures Apache 2.0 for your operating system.

Examples and messages are shown on more than one line for clarity. Place each directive in a Web server configuration file on one line.

Local file path means a file path to the plugin-cfg.xml file on an Application Server that is on the same machine as the Web server. Remote file path means the file path to the plugin-cfg.xml file when the Application Server is on a remote machine. The Plug-ins installation wizard installs a dummy plugin-cfg.xml file during installation, but this file requires periodic propagation from the real file on the Application Server machine.

The *node_name* in the following Application Server local file paths is *web_server_name_node* for a stand-alone Application Server or *managed_node_name* for a managed node.

The name of the Web server definition in the following steps is *webserver1*.

- **AIX** Configure entries in the httpd.conf file.

Use the following examples of the LoadModule and the WebSpherePluginConfig directives as models for configuring your file:

```
LoadModule
    was_ap20_module /usr/IBM/WebSphere/Plugins/bin/mod_was_ap20_http.so
```

Local distributed example:

```
WebSpherePluginConfig
    profile_root/config/cells/
        dmgrcell/nodes/managednode/servers/webserver1/plugin-cfg.xml
```

Example:

```
WebSpherePluginConfig
    /usr/IBM/WebSphere/Plugins/config/webserver1/plugin-cfg.xml
```

- **Linux** **Solaris** Configure entries in the httpd.conf file.

Use the following examples of the LoadModule and the WebSpherePluginConfig directives as models for configuring your file:

```
LoadModule
    was_ap20_module /opt/IBM/WebSphere/Plugins/bin/mod_was_ap20_http.so
```

Local distributed example:

```
WebSpherePluginConfig
    profile_root/config/cells/
    dmgrcell/nodes/managednode/servers/webserver1/plugin-cfg.xml
```

Example:

```
WebSpherePluginConfig
    /opt/IBM/WebSphere/Plugins/config/webserver1/plugin-cfg.xml
```

Solaris On the Solaris SPARC 64-bit platform, the Plug-ins installation wizard installs both 32-bit and 64-bit versions of the plug-in for Apache 2.0, however it configures the Web server to use the 32-bit plug-in only. If the Web server is 64-bit, you need to configure the LoadModule directive in the httpd.conf file to use the 64-bit plug-in as follows:

```
LoadModule
    was_ap20_module /usr/IBM/WebSphere/Plugins/bin/64bits/mod_was_ap20_http.so
```

- **HP-UX** Configure entries in the httpd.conf file.

Use the following examples of the LoadModule and the WebSpherePluginConfig directives as models for configuring your file:

```
LoadModule
    was_ap20_module /opt/IBM/WebSphere/Plugins/bin/mod_was_ap20_http.sl
```

Local distributed example:

```
WebSpherePluginConfig
    profile_root/config/cells/
    dmgrcell/nodes/managednode/servers/webserver1/plugin-cfg.xml
```

Example:

```
WebSpherePluginConfig
    /opt/IBM/WebSphere/Plugins/config/webserver1/plugin-cfg.xml
```

- **Windows** Configure entries in the httpd.conf file.

Use the following examples of the LoadModule and the WebSpherePluginConfig directives as models for configuring your file:

```
LoadModule was_ap20_module
    drive:\IBM\WebSphere\Plugins\bin\mod_was_ap20_http.dll
```

Local distributed example:

```
WebSpherePluginConfig
    profile_root\config\cells\
    dmgrcell\nodes\managednode\servers\webserver1\plugin-cfg.xml
```

Example:

```
WebSpherePluginConfig
    C:\IBM\WebSphere\Plugins\config\webserver1\plugin-cfg.xml
```

Results

This procedure results in reconfiguring the Apache 2.0 Web server.

What to do next

The Plug-ins installation wizard installs the GSKIT SSL encryption library at the required level if it is not installed. If you manually copy the plug-in to a new machine, you might not have the required GSKIT libraries for encrypting back-end connections.

After configuring a Web server, you can install applications on it. See the Applications section of the information center for more information.

Configuring Apache HTTP Server V2.2

This topic describes how to change configuration settings for Apache HTTP Server Version 2.2.

Before you begin

Install Apache 2.2 and the latest version of the Web server plug-ins for WebSphere Application Server 7.0 using the UpdateInstaller.

Apache HTTP Server v2.2 is different from IBM HTTP Server (powered by Apache). Apache HTTP Server is not supported on IBM i.

When you install the Web server plug-ins for WebSphere Application Server on a non-IBM i system, the Plug-ins installation wizard configures the Web server.

This topic describes how to configure the Apache HTTP Server Version 2.2 Web server. Other procedures in “Editing Web server configuration files” on page 81 describe configuring other supported Web servers.

Note:

- If you are using an Apache HTTP Server that supports 64-bit addressing, you must use the 64-bit CD provided with the WebSphere Application Server product to install the Apache Web server plug-in binaries. If you use the 32-bit CD, you will receive an error message indicating that the plug-in binaries did not load.
- If you are using an Apache HTTP Server that supports 32-bit addressing, you must use the 32-bit CD provided with the WebSphere Application Server product to install the Apache Web server plug-in binaries. If you use the 64-bit CD, you will receive an error message indicating that the plug-in binaries did not load.

A sample error message follows:

```
httpd: Syntax error on line XXX of /home/apache/conf/httpd.conf: Cannot
load /home/apache/Plugins/mod_was_ap22_http.sl into server: Invalid argument
```

The plug-in was tested with the threaded worker multi-processing module (MPM) on all platforms except Windows. The plug-in was tested with the default threaded MPM on Windows.

The plug-in works with the Apache 2.2 prefork MPM but works best with the worker MPM. The plug-in maintains connection pools to backend WebSphere Application Servers and uses in-memory caching. These plug-in functions perform most efficiently when Apache is configured to use a single child process with the ThreadsPerChild value equal to the MaxClients value. The plug-in can be used with the prefork MPM or the worker MPM that is configured with multiple child processes, but at reduced efficiency.

Compatibility Statement The plug-in works with versions of the Apache HTTP Server that claim full binary compatibility with Apache 2.0.47 and later, which are built with compilers and compiler options that are compatible with those used to build the plug-in.

About this task

Perform the step that configures Apache 2.2 for your operating system.

Examples and messages are shown on more than one line for clarity. Place each directive in a Web server configuration file on one line.

Local file path means a file path to the plugin-cfg.xml file on an Application Server that is on the same machine as the Web server. Remote file path means the file path to the plugin-cfg.xml file when the Application Server is on a remote machine. The Plug-ins installation wizard installs a dummy plugin-cfg.xml file during installation, but this file requires periodic propagation from the real file on the Application Server machine.

The *node_name* in the following Application Server local file paths is *web_server_name_node* for a stand-alone Application Server or *managed_node_name* for a managed node.

The name of the Web server definition in the following steps is `webserver1`.

- **AIX** Configure entries in the `httpd.conf` file.

Use the following examples of the `LoadModule` and the `WebSpherePluginConfig` directives as models for configuring your file:

```
LoadModule
    was_ap22_module /usr/IBM/WebSphere/Plugins/bin/mod_was_ap22_http.so
```

Local distributed example:

```
WebSpherePluginConfig
    profile_root/config/cells/
        dmgrcell/nodes/managednode/servers/webserver1/plugin-cfg.xml
```

- **Linux** **Solaris** Configure entries in the `httpd.conf` file.

Use the following examples of the `LoadModule` and the `WebSpherePluginConfig` directives as models for configuring your file:

```
LoadModule
    was_ap22_module /opt/IBM/WebSphere/Plugins/bin/mod_was_ap22_http.so
```

Local distributed example:

```
WebSpherePluginConfig
    profile_root/config/cells/
        dmgrcell/nodes/managednode/servers/webserver1/plugin-cfg.xml
```

Solaris On the Solaris SPARC 64-bit platform, the Plug-ins installation wizard installs both 32-bit and 64-bit versions of the plug-in for Apache 2.2, however it configures the Web server to use the 32-bit plug-in only. If the Web server is 64-bit, you need to configure the `LoadModule` directive in the `httpd.conf` file to use the 64-bit plug-in as follows:

```
LoadModule
    was_ap22_module /usr/IBM/WebSphere/Plugins/bin/64bits/mod_was_ap22_http.so
```

- **HP-UX** Configure entries in the `httpd.conf` file.

Use the following examples of the `LoadModule` and the `WebSpherePluginConfig` directives as models for configuring your file:

```
LoadModule
    was_ap22_module /opt/IBM/WebSphere/Plugins/bin/mod_was_ap22_http.sl
```

Local distributed example:

```
WebSpherePluginConfig
    profile_root/config/cells/
        dmgrcell/nodes/managednode/servers/webserver1/plugin-cfg.xml
```

- **Windows** Configure entries in the `httpd.conf` file.

Use the following examples of the `LoadModule` and the `WebSpherePluginConfig` directives as models for configuring your file:

```
LoadModule was_ap22_module
    drive:\IBM\WebSphere\Plugins\bin\mod_was_ap22_http.dll
```

Local distributed example:

```
WebSpherePluginConfig
    profile_root\config\cells\
        dmgrcell\nodes/managednode\servers\webserver1\plugin-cfg.xml
```

Results

The Apache 2.2 Web server is reconfigured.

What to do next

The Plug-ins installation wizard installs the GSKIT SSL encryption library at the required level if it is not installed. If you manually copy the plug-in to a new machine, you might not have the required GSKIT libraries for encrypting back-end connections.

After configuring a Web server, you can install applications on it. See the Applications section of the information center for more information.

Configuring Lotus Domino

This task describes how to change configuration settings for Lotus® Domino.

Before you begin

When you install the Web server plug-ins for WebSphere Application Server on a non-IBM i system, the Plug-ins installation wizard configures the Web server.

Other procedures in “Editing Web server configuration files” on page 81 describe configuring other supported Web servers.

About this task

Use the following procedure to enable the Web server plug-in to work with Lotus Domino. Ensure that you install the plugin as root. Domino can only be installed as root, and the configuration files belong to root.

Note: If you install the plugin as local and the WebSphere Application Server as non-root, then Web server management on WebSphere Application Server, such as generation, propagation, deletion of Web server definitions and more, is unavailable because the plugin-cfg.xml file must be installed as root.

1. Start the Domino server.
2. Access the names.nsf file using your Web browser (for example, <http://tarheels2.raleigh.ibm.com/names.nsf>). The browser prompts you for a password.
3. Give the administrator name and password.
4. Click **Configuration** on the left side of the page.
5. Click **Servers** on the left side of the page.
6. Click **All Server Documents** on the left side of the page.
7. Click the server that you intend to have work with the product
8. Click **Edit Server** on the top-left of the center window.
9. Click **Internet Protocols** in the middle of the page.
10. Add the path to the Domino plug-in under the DSAPI Section in the middle-right of the page.
If specifications for filter files for the Domino Server Application Programming Interface (DSAPI) already exist, use a space to delimit the Web server plug-in for WebSphere Application Server.
11. Click **Save and Close** in the upper-left of the center window.
12. Define the location of the plugin-cfg.xml configuration file.

The location varies depending on how you have configured your system. If the Web server and the Application Server are on separate machines, you have a remote installation.

If the two servers are on the same machine, you have a local installation.

If the two servers are on the same machine and the Application Server node or the custom node is federated, you have a local distributed installation.

In the following examples, `webserver1` is the Web server definition name.

AIX

HP-UX

Linux

Solaris

Setting the path to the plug-in configuration file

Set the `WAS_PLUGIN_CONFIG_FILE` environment variable to the location of the plug-in configuration file using one of the following paths:

If the type of installation is:	Then use this command to set the environment variable:
Remote	<code>WAS_PLUGIN_CONFIG_FILE=/opt/IBM/WebSphere/Plugins/config/webserver1/plugin-cfg.xml</code>
Local standalone	<code>WAS_PLUGIN_CONFIG_FILE=profile_root/config/cells/sa_cell/nodes/webserver1_node/servers/webserver1/plugin-cfg.xml</code>
Local distributed	<code>WAS_PLUGIN_CONFIG_FILE=profile_root/config/cells/dmgrcell/nodes/managednode/servers/webserver1/plugin-cfg.xml</code>

During the installation process, the Plug-ins installation wizard creates the `setupPluginCfg.sh` file in two places:

- The `plugins_root/bin` directory
- The `lotus_root/notesdata` directory

You can run the script from either location to set the `WAS_PLUGIN_CONFIG_FILE` environment variable. However, if you are reconfiguring the Web server, you might want to set the path yourself by setting the value of the environment variable with a path from the preceding table.

The `setupPluginCfg.sh` script sets the file path value to the file path that the wizard configured originally. If you are reconfiguring the Web server to change the original file path, do not use this script.

Windows

Setting the path to the plug-in configuration file

Add the appropriate statement to your `lotus_domino_root\notes.ini` file:

Do not delimit any of the following file paths with quotation marks unless there is a space in the file path. Otherwise, the `plugin-cfg.xml` file might not load correctly.

If the type of installation is:	Then use this command to set the WebSpherePluginCfg variable:
Remote	<code>WebSpherePluginCfg=C:\Program Files\IBM\WebSphere\Plugins\config\webserver1\plugin-cfg.xml</code>
Local standalone	<code>WebSpherePluginCfg=profile_root\config\cells\sa_cell\nodes\webserver1_node\servers\webserver1\plugin-cfg.xml</code>
Local distributed	<code>WebSpherePluginCfg=profile_root\config\cells\dmgrcell\nodes\managednode\servers\webserver1\plugin-cfg.xml</code>

- Restart the Domino server. When the server starts, information similar to the following example is displayed:

```
01/21/2005 01:21:51 PM JVM: Java virtual machine initialized
WebSphere Application Server DSAPI filter loaded
01/21/2005 01:21:52 PM HTTP Web Server started
```

Results

This procedure results in reconfiguring Version 6.x of Lotus Domino.

What to do next

After configuring a Web server, you can install applications on it. See the Applications section of the information center for more information.

For more information on configuring Lotus Domino to work with WebSphere Application Server, search the Lotus Support Services Web site at <http://www-3.ibm.com/software/lotus/support/>. Enter the search term WebSphere in the keyword search field.

Lotus Domino file locations and troubleshooting tips

Lotus Domino Server is one of the Web servers that WebSphere Application Server supports. This topic describes configuration file locations and provides other tips related to using Lotus Domino.

Lotus Domino Server file locations

Lotus Domino server file locations are:

- **AIX**
 - /usr/lotus/notes/50091/ibmpow/Notes.jar
 - /usr/notesdata/names.nsf
- **Solaris**
 - /opt/lotus/notes/5080/sunspa/Notes.jar
 - /opt/notesdata/names.nsf
- **Windows**
 - c:\Program Files\lotus\notes\Notes.jar
 - c:\Program Files\lotus\notes\data\names.nsf

Solaris

The Domino Server plug-in might fail to configure on a Solaris platform

If this occurs during installation, a dsapi_stderr.txt file is created in the logs directory and you get the following error messages:

```
lotus.notes.NotesException: Could not load dll for system name SUNOS
    at lotus.notes.NotesThread.load(NotesThread.java:210)
    at lotus.notes.NotesThread.<clinit>(NotesThread.java:24)
java.lang.UnsatisfiedLinkError: NnotesInitThread
    at lotus.notes.NotesThread.NnotesInitThread(Native Method)
    at lotus.notes.NotesThread.initThread(NotesThread.java:99)
    at lotus.notes.NotesThread.run(NotesThread.java:133)
```

You can configure the WebSphere Application Server or Domino Server plug-in manually using the Domino Server Web administration tool. Use the following procedures:

1. Start the Domino Server.
2. Enter the URL for the Domino Server Web Administration site using a browser. For example, http://host_name/names.nsf. Enter the administrator user name and password.
3. Double-click **Server-Servers**.
4. Double-click **WebServer** to configure.
5. Double-click **Edit Server**.
6. Double-click **Internet Protocol**.
7. Add the WebSphere Application Server DSAPI plug-in to the **DSAPI** field. For example, `app_server_root/bin/libdomino5_http.so`
If there are already DSAPI filter files specified, use a space to delimit the WebSphere Application Server plug-in file.
8. Double-click **Save and Close**.
9. Restart the Domino Server.

Avoiding a DSAPI filter-loading error when the Lotus Domino Server starts

On operating systems such as AIX or Linux, if the Lotus Domino Web server starts using a non-root user, you are likely to generate a DSAPI filter-loading error when the Lotus Domino Server starts:

Error loading DSAPI filter.

Filter not loaded: `app_server_root/bin/libdomino6_http.a`

Manually change the WebSphere Application Server bin directory permissions from 750 to 755 to run Lotus Domino Server as a non-root user and not generate the error.

You must also change permissions on the WebSphere Application Server logs directory to 777 to allow Lotus Domino Server to write to the log. However, this change poses a security risk.

If the Lotus Domino Server is started as root, the problem does not occur.

Configuring IBM HTTP Server powered by Apache 2.x

This topic describes how to change configuration settings for IBM HTTP Server powered by Apache 2.x.

Before you begin

When you install the Web server plug-ins for the product on a non-iSeries system, the Plug-ins installation wizard configures the Web server.

This topic describes how to configure the IBM HTTP Server. Other procedures in “Editing Web server configuration files” on page 81 describe configuring other supported Web servers.

About this task

Perform the step that configures IBM HTTP Server version 2.x for your operating system. IBM HTTP Server powered by Apache 2.0 is supported on i5/OS V5R4. IBM HTTP Server powered by Apache 2.2 is supported on IBM i V6R1. On IBM i V6R1, the Web server plug-ins are included with the 5761-DG1 product.

Examples and messages are shown on more than one line for clarity. Place each directive in a Web server configuration file on one line.

Local file path means a file path to the `plugin-cfg.xml` file on an Application Server that is on the same machine as the Web server. Remote file path means the file path to the `plugin-cfg.xml` file when the Application Server is on a remote machine. The Plug-ins installation wizard installs a dummy `plugin-cfg.xml` file during installation, but this file requires periodic propagation from the real file on the Application Server machine.

The Web server definition name in the following examples is `webserver1`.

The node name `node_name` in the following Application Server local file paths is `web_server_name_node` for a stand-alone Application Server or `managed_node_name` for a managed node.

- **AIX** Configure entries in the `httpd.conf` file.

Use the following examples of the `LoadModule` and the `WebSpherePluginConfig` directives as models for configuring your file:

```
LoadModule
    was_ap20_module /usr/IBM/WebSphere/Plugins/bin/mod_was_ap20_http.so
```

Local distributed example:

```
WebSpherePluginConfig
  profile_root/config/cells/
    dmgrcell/nodes/managednode/servers/webserver1/plugin-cfg.xml
```

Example:

```
WebSpherePluginConfig
  /usr/IBM/WebSphere/Plugins/config/webserver1/plugin-cfg.xml
```

- **Linux** **Solaris** Configure entries in the httpd.conf file.

Use the following examples of the LoadModule and the WebSpherePluginConfig directives as models for configuring your file:

```
LoadModule
  was_ap20_module /opt/IBM/WebSphere/Plugins/bin/mod_was_ap20_http.so
```

Local distributed example:

```
WebSpherePluginConfig
  profile_root/config/cells/
    dmgrcell/nodes/managednode/servers/webserver1/plugin-cfg.xml
```

Example:

```
WebSpherePluginConfig
  /opt/IBM/WebSphere/Plugins/config/webserver1/plugin-cfg.xml
```

- **HP-UX** Configure entries in the httpd.conf file.

Use the following examples of the LoadModule and the WebSpherePluginConfig directives as models for configuring your file:

```
LoadModule
  was_ap20_module /opt/IBM/WebSphere/Plugins/bin/mod_was_ap20_http.sl
```

Local distributed example:

```
WebSpherePluginConfig
  profile_root/config/cells/
    dmgrcell/nodes/managednode/servers/webserver1/plugin-cfg.xml
```

Example:

```
WebSpherePluginConfig
  /opt/IBM/WebSphere/Plugins/config/webserver1/plugin-cfg.xml
```

- **Windows** Configure entries in the httpd.conf file.

Use the following examples of the LoadModule and the WebSpherePluginConfig directives as models for configuring your file:

```
LoadModule was_ap20_module
  drive:\IBM\WebSphere\Plugins\bin\mod_was_ap20_http.dll
```

Local distributed example:

```
WebSpherePluginConfig
  profile_root\config\cells\
    dmgrcell\nodes\managednode\servers\webserver1\plugin-cfg.xml
```

Example:

```
WebSpherePluginConfig
  C:\IBM\WebSphere\Plugins\config\webserver1\plugin-cfg.xml
```

Results

This procedure results in editing and reconfiguring IBM HTTP Server powered by Apache 2.x.

What to do next

If the IBM HTTP Server 1.3.2x directive, LoadModule ibm_app_server_http_module, is present in an IBM HTTP Server 2.x httpd.conf file, the IBM HTTP Server cannot start. You must comment or delete the directive to start the Version 2.x server.

The `mod_was_ap20_http` plug-in module requires the GSKIT Secure Sockets Layer (SSL) encryption library if the plug-in is configured to support encrypted connections to back-end application servers.

The Plug-ins installation wizard installs the GSKIT SSL encryption library at the required level if it is not installed. If you manually copy the plug-in to a new machine, you might not have the required GSKIT libraries for encrypting back-end connections.

After configuring a Web server, you can install applications on it. See the Applications section of the information center for more information.

Configuring IBM HTTP Server Version 6.x

This topic describes how to change configuration settings for IBM HTTP Server.

Before you begin

When you install the Web server plug-ins for WebSphere Application Server on a non-IBM system, the Plug-ins installation wizard configures the Web server.

See “Installing Web server plug-ins” on page 6.

This topic describes how to configure IBM HTTP Server, Version 6.x. Other procedures in “Editing Web server configuration files” on page 81 describe configuring other supported Web servers.

About this task

Perform the step that configures IBM HTTP Server for your operating system.

Examples and messages are shown on more than one line for clarity. Place each directive in a Web server configuration file on one line.

Local file path means a file path to the `plugin-cfg.xml` file on an Application Server that is on the same machine as the Web server. Remote file path means the file path to the `plugin-cfg.xml` file when the Application Server is on a remote machine. The Plug-ins installation wizard installs a dummy `plugin-cfg.xml` file during installation, but this file requires periodic propagation from the real file on the Application Server machine.

The `node_name` in the following Application Server local file paths is `web_server_name_node` for a stand-alone Application Server or `managed_node_name` for a managed node.

- **AIX** Configure entries in the `httpd.conf` file.

Use the following examples of the `LoadModule` and the `WebSpherePluginConfig` directives as models for configuring your file:

```
LoadModule
    was_ap20_module /usr/IBM/HTTPServer/Plugins/bin/mod_was_ap20_http.so
```

Local distributed example:

```
WebSpherePluginConfig
    profile_root/config/cells/
    dmgrcell/nodes/managednode/servers/webserver1/plugin-cfg.xml
```

Example:

```
WebSpherePluginConfig
    /usr/IBM/HTTPServer/Plugins/config/webserver1/plugin-cfg.xml
```

- **Linux** **Solaris** Configure entries in the `httpd.conf` file.

Use the following examples of the `LoadModule` and the `WebSpherePluginConfig` directives as models for configuring your file:

```
LoadModule
    was_ap20_module /opt/IBM/HTTPServer/Plugins/bin/mod_was_ap20_http.so
```

Local distributed example:

```
WebSpherePluginConfig
    profile_root/config/cells/
        dmgrcell/nodes/managednode/servers/webserver1/plugin-cfg.xml
```

Example:

```
WebSpherePluginConfig
    /opt/IBM/HTTPServer/Plugins/config/webserver1/plugin-cfg.xml
```

- **HP-UX** Configure entries in the httpd.conf file.

Use the following examples of the LoadModule and the WebSpherePluginConfig directives as models for configuring your file:

```
LoadModule
    was_ap20_module /opt/IBM/HTTPServer/Plugins/bin/mod_was_ap20_http.s1
```

Local distributed example:

```
WebSpherePluginConfig
    profile_root/config/cells/
        dmgrcell/nodes/managednode/servers/webserver1/plugin-cfg.xml
```

Example:

```
WebSpherePluginConfig
    /opt/IBM/HTTPServer/Plugins/config/webserver1/plugin-cfg.xml
```

- **Windows** Configure entries in the httpd.conf file.

Use the following examples of the LoadModule and the WebSpherePluginConfig directives as models for configuring your file:

```
LoadModule was_ap20_module
    drive:\Program Files\IBM\HTTPServer\Plugins\bin\mod_was_ap20_http.dll
```

Local distributed example:

```
WebSpherePluginConfig
    profile_root\config\cells\
        dmgrcell\nodes\managednode\servers\webserver1\plugin-cfg.xml
```

Example:

```
WebSpherePluginConfig
    C:\Program Files\IBM\HTTPServer\Plugins\config\webserver1\plugin-cfg.xml
```

Results

This procedure results in editing and reconfiguring IBM HTTP Server.

What to do next

If the IBM HTTP Server V1.3.x directive, LoadModule ibm_app_server_http_module, is present in an IBM HTTP Server Version 6.x and later httpd.conf file, IBM HTTP Server cannot start. You must comment or delete the directive to start the Version 6.x server.

The mod_was_ap20_http plug-in module requires the GSKIT Secure Sockets Layer (SSL) encryption library if the plug-in is configured to support encrypted connections to back-end WebSphere Application Servers.

The Plug-ins installation wizard installs the GSKIT SSL encryption library at the required level if it is not installed. If you manually copy the plug-in to a new machine, you might not have the required GSKIT libraries for encrypting back-end connections.

After configuring a Web server, you can install applications on it. See the Applications section of the information center for more information.

Configuring IBM HTTP Server Version 7.0

This topic describes how to change configuration settings for IBM HTTP Server.

Before you begin

When you install the Web server plug-ins for WebSphere Application Server on a non-IBM i system, the Plug-ins installation wizard configures the Web server.

See “Installing Web server plug-ins” on page 6.

This topic describes how to configure IBM HTTP Server, Version 7.0. Other procedures in “Editing Web server configuration files” on page 81 describe configuring other supported Web servers.

About this task

Perform the step that configures IBM HTTP Server for your operating system.

Examples and messages are shown on more than one line for clarity. Place each directive in a Web server configuration file on one line.

Local file path means a file path to the plugin-cfg.xml file on an Application Server that is on the same machine as the Web server. Remote file path means the file path to the plugin-cfg.xml file when the Application Server is on a remote machine. The Plug-ins installation wizard installs a dummy plugin-cfg.xml file during installation, but this file requires periodic propagation from the real file on the Application Server machine.

The *node_name* in the following Application Server local file paths is *web_server_name_node* for a stand-alone Application Server, or *managed_node_name* for a managed node.

- **AIX** Configure entries in the httpd.conf file.

Use the following examples of the LoadModule and the WebSpherePluginConfig directives as models for configuring your file:

```
LoadModule
    was_ap22_module /usr/IBM/HTTPServer/Plugins/bin/mod_was_ap22_http.so
```

Local distributed example:

```
WebSpherePluginConfig
    profile_root/config/cells/
    dmgrcell/nodes/managednode/servers/webserver1/plugin-cfg.xml
```

Example:

```
WebSpherePluginConfig
    /usr/IBM/HTTPServer/Plugins/config/webserver1/plugin-cfg.xml
```

- **Linux** **Solaris** Configure entries in the httpd.conf file.

Use the following examples of the LoadModule and the WebSpherePluginConfig directives as models for configuring your file:

```
LoadModule
    was_ap22_module /opt/IBM/HTTPServer/Plugins/bin/mod_was_ap22_http.so
```

Local distributed example:

```
WebSpherePluginConfig
    profile_root/config/cells/
    dmgrcell/nodes/managednode/servers/webserver1/plugin-cfg.xml
```

Example:

```
WebSpherePluginConfig
    /opt/IBM/HTTPServer/Plugins/config/webserver1/plugin-cfg.xml
```

- **HP-UX** Configure entries in the httpd.conf file.

Use the following examples of the `LoadModule` and the `WebSpherePluginConfig` directives as models for configuring your file:

```
LoadModule was_ap22_module /opt/IBM/HTTPServer/Plugins/bin/mod_was_ap22_http.s1
```

Local distributed example:

```
WebSpherePluginConfig
  profile_root/config/cells/
  dmgrcell/nodes/managednode/servers/webserver1/plugin-cfg.xml
```

Example:

```
WebSpherePluginConfig
  /opt/IBM/HTTPServer/Plugins/config/webserver1/plugin-cfg.xml
```

- **Windows** Configure entries in the `httpd.conf` file.

Use the following examples of the `LoadModule` and the `WebSpherePluginConfig` directives as models for configuring your file:

```
LoadModule was_ap22_module
  drive:\Program Files\IBM\HTTPServer\Plugins\bin\mod_was_ap22_http.dll
```

Local distributed example:

```
WebSpherePluginConfig
  profile_root\config\cells\
  dmgrcell\nodes\managednode\servers\webserver1\plugin-cfg.xml
```

Example:

```
WebSpherePluginConfig
  C:\Program Files\IBM\HTTPServer\Plugins\config\webserver1\plugin-cfg.xml
```

Results

This procedure results in editing and reconfiguring IBM HTTP Server.

What to do next

The `mod_was_ap22_http` plug-in module requires the GSKIT Secure Sockets Layer (SSL) encryption library if the plug-in is configured to support encrypted connections to back-end WebSphere Application Servers.

The Plug-ins installation wizard installs the GSKIT SSL encryption library at the required level if it is not installed. If you manually copy the plug-in to a new machine, you might not have the required GSKIT libraries for encrypting back-end connections.

After configuring a Web server, you can install applications on it. See the Applications section of the information center for more information.

Configuring Microsoft Internet Information Services (IIS)

This topic describes manual configuration settings for Internet Information Services (IIS).

Before you begin

When you install the Web server plug-ins for WebSphere Application Server, as described in “Installing Web server plug-ins” on page 6, the Plug-ins installation wizard configures the Web server. This topic describes how to configure the Internet Information Services (IIS) Web server. Other procedures in “Editing Web server configuration files” on page 81 describe configuring other supported Web servers.

You must have read/write access to the `plugins_root` directory to perform this task.

About this task

Use the following procedure to manually reproduce how the Installation wizard configures the Microsoft Internet Information Services Web server.

- Configure IIS Version 5.0.
 1. Start the IIS application and create a new virtual directory for the Web site instance that you intend to work with WebSphere Application Server. These instructions assume that you are using the Default Web Site.
 2. Expand the tree on the left until you see Default Web Site.
Right-click **Default Web Site**, then click **New > Virtual Directory** to create the directory with a default installation.
 3. Type **sePlugins** in the Alias to be used to Access Virtual Directory field.
 4. Browse to the *plugins_root\bin\IIS_Web_server_name* directory in the Enter the physical path of the directory containing the content you want to publish field.
 5. Select the appropriate **Execute** check box (such as ISAPI applications or CGI) in the What access permissions do you want to set for this directory field.
 6. Click **Next** to add the sePlugins virtual directory to your default Web site.
 7. Click **Finish**.
 8. Right-click **Default Web Site** in the navigation tree and click **Properties**.
Add the Internet Services Application Programming Interface (ISAPI) filter into the IIS configuration.
In the Properties dialog, perform the following steps:
 - a. Click the **Internet Information Services** tab.
 - b. Click **WWW Service** in the Master properties window.
 - c. Click **Edit** to open the WWW Service master properties window.
 - d. Click **ISAPI Filters > Add** to open the Filter properties window.
 - e. Type **iisWASPlugin** in the Filter Name field.
 - f. Click **Browse** in the Executable field.
 - g. Browse to the *plugins_root\bin\IIS_Web_server_name* directory.
 - h. Click the **iisWASPlugin_http.dll** file.
 - i. Click **OK** until all the open windows close.
- Configure IIS Version 6.0.
 1. Start the IIS application and create a new virtual directory for the Web site instance that you intend to work with WebSphere Application Server. These instructions assume that you are using the Default Web Site.
Click **Programs > Administrative Tools > Internet Information Services (IIS) Manager** on a Windows Server 2003 Standard Edition system, for example.
 2. Expand the tree on the left until you see **Default Web Site**.
Right-click **Default Web Site > New > Virtual Directory** to create the directory with a default installation.
 3. Type **sePlugins** in the **Alias** field in the Virtual Directory Alias panel of the Virtual Directory Creation Wizard, then click **Next**.
 4. Browse to the *plugins_root\bin\IIS_web_server_name* directory in the Path field of the Web Site Content Directory panel of the wizard, then click **Next**.
For example, select the C:\Program Files\IBM\WebSphere\Plugins\bin\IIS_webserver1 directory.
 5. Select the appropriate permission check boxes in the Virtual Directory Access Permissions panel of the wizard.

Select the **Read** check box and the **Execute (such as ISAPI applications or CGI)** check box, for example.

6. Click **Next** to add the sePlugins virtual directory to your default Web site.
7. Click **Finish** when the success message displays.
8. Copy the plug-in binaries to the *plugins_root\bin\IIS_Web_server_name* directory.
For example. copy the plug-in binary files to the C:\Program Files\IBM\WebSphere\Plugins\bin\IIS_webserver1 directory.
The plugin-cfg.loc file resides in this directory. The first line of the plugin-cfg.loc file identifies the location of the plugin-cfg.xml file.
9. Expand the **Web Sites** folder in the left pane navigation tree of the IIS Manager panel.
10. Right-click **Default Web Site** in the navigation tree and click **Properties**.

Add the Internet Services Application Programming Interface (ISAPI) filter into the IIS configuration.

In the Default Web Site Properties panel, perform the following steps:

- a. Click the **ISAPI Filters** tab.
 - b. Click **Add** to open the **Add/Edit Filter Properties** dialog window.
 - c. Type **iisWASPlugin** in the **Filter name** field.
 - d. Click **Browse** to select the C:\Program Files\IBM\WebSphere\Plugins\bin\IIS_webserver1\iisWASPlugin_http.dll file for the value of the **Executable** field.
Browse to your *plugins_root\bin\IIS_Web_server_name* directory to select the **iisWASPlugin_http.dll** file.
 - e. Click **OK** to close the **Add/Edit Filter Properties** dialog window.
 - f. Click **OK** to close the **Default Web Site Properties** window.
11. Set the value in the plugin-cfg.loc file to the location of the configuration file.
Set the location to the *plugins_root\config\webserver_name\plugin-cfg.xml* file, which might be C:\Program Files\IBM\WebSphere\Plugins\config\IIS_webserver1\plugin-cfg.xml file.
The location varies depending on how you have configured your system. If the Web server and the Application Server are on separate machines, you have a remote installation.
If the two servers are on the same machine, you have a local installation.
If the two servers are on the same machine and the application server is federated, you have a local distributed installation.

Local distributed example:

```
"C:\IBM\WebSphere\AppServer\profiles\custom01\config\cells\dmgrcell\nodes\managed_node\servers\webserver1\plugin-cfg.xml"
```

Example:

```
"C:\IBM\WebSphere\Plugins\config\webserver1\plugin-cfg.xml"
```

12. Configure the Web server to run WebSphere Application Server extensions:
 - a. Expand the left pane navigation tree and click on the **Web Service Extensions** folder in the IIS Manager panel.
 - b. Click **Add a new Web service extension** to open the **New Web Service Extension** dialog window.
 - c. In the **Extension name** field, type WASPlugin as the name of the new Web service extension.
 - d. Click **Add** to open the **Add file** dialog window.
 - e. In the **Path to file** field, type the path or click **Browse** to navigate to the correct iisWASPlugin_http.dll file that the new Web service extension requires, and click **OK**.
 - f. Select the **Set extension status to Allowed** check box to automatically set the status of the new Web service extension to Allowed and click **OK**.
- Optional: Configure multiple Web sites. Given:

- There are two Web sites defined: website1, website2.
 - The DLL files are already created as bin/website1/iisWASPlugin_http.dll and bin/website2/iisWebsite2/iisWASPlugin_http.dll.
 - The plugin-cfg.loc files are created in the same folder as the DLL files.
1. Run IIS in worker process isolation mode (default).
To enable worker process in isolation mode:
 - a. Open the IIS Manager console and expand the local computer by clicking the **plus sign**.
 - b. Expand the **Web Sites** folder, then right-click the **Default Web Sites** folder.
 - c. Click **Properties**, then click the **Service** tab.
 - d. Under Isolation mode, clear the Run Web service in IIS 5.0 isolation mode check box to enable worker process isolation mode.
 2. Define two application pools; one for website1 and the other for website2. Do not use the pre-defined application pool DefaultAppPool.
 3. Define the two Web sites, including the filter setting, virtual host setting, and extension settings.
 4. Assign an application pool for each Web site.
 - a. Under each Web site folder, right click on the **Web site name**.
 - b. Click **Property**, and select the **Home Directory** tab. 2.
 - c. At the bottom of the application settings, select the application pool you defined for Web site 1 from the drop-down list of application pools.
 - d. Click **OK**.
 - e. Repeat the previous steps for the second Web site and select the application pool you defined for Web site 2.
 5. Start the IIS service and start each Web site.

Results

This procedure results in reconfiguring the Internet Information Services (IIS) Web server.

Note: On some editions of the Windows operating system, the http_plugin.log file is not created automatically when the plug-in is installed and the IIS Web server is started. If the http_plugin.log file is not created after performing the procedure described above, take the following steps:

1. Open a Windows Explorer window.
2. Browse to the *plugins_root\logs\web_server_name* directory.
3. Share the folder and give full-control permission to everyone.

What to do next

You can now install applications on the configured Web server. See the Applications section of the information center for more information.

Configuring the Sun Java System Web Server

This topic describes how to change configuration settings for the Sun Java System Web Server (formerly Sun ONE Web Server and iPlanet Web Server), Version 6.0 and later.

Before you begin

When you install the Web server plug-ins for WebSphere Application Server, as described in “Installing Web server plug-ins” on page 6, the Plug-ins installation wizard configures the Web server. This topic describes how to configure the Sun Java System Web Server if you must change something in the

existing configuration. Other procedures in “Editing Web server configuration files” on page 81 describe configuring other supported Web servers.

About this task

Configure the Sun ONE Web Server 6.0 or Sun Java System Web Server, Version 6.1 and later.

Examples and messages are sometimes shown on more than one line for ease of presentation. Verify that each directive in a Web server configuration file is on one line.

1. Configure entries in the obj.conf configuration file and in the magnus.conf configuration file for Version 6.0 and later of Sun Java System Web Server.

- a. Add two directives to the obj.conf file after the <Object name=default> tag:

```
Service fn="as_handler"  
AddLog fn="as_term"
```

- b. Add two directives at the end of the magnus.conf file:

The location for the bootstrap.properties directive varies, depending on how you have configured your system. If the Web server and the application server are on separate machines, you have a remote installation.

If the two servers are on the same machine, you have a local installation.

If the two servers are on the same machine and the application server is a managed node, you have a local distributed installation.

- **AIX** **HP-UX** **Linux** **Solaris**

Local distributed example:

```
Init fn="load-modules"  
    funcs="as_init,as_handler,as_term"  
    shlib="/opt/IBM/WebSphere/Plugins/bin/libns41_http.so"  
Init fn="as_init"  
    bootstrap.properties="profile_root/config/  
        cells/dmgrcell/nodes/managed_node/servers/webserver1/plugin-cfg.xml"
```

Example:

```
Init fn="load-modules"  
    funcs="as_init,as_handler,as_term"  
    shlib="/opt/IBM/WebSphere/Plugins/bin/libns41_http.so"  
Init fn="as_init"  
    bootstrap.properties="/opt/IBM/WebSphere/Plugins/config/webserver1/plugin-cfg.xml"
```

- **Windows**

Local distributed example:

```
Init fn="load-modules"  
    funcs="as_init,as_handler,as_term"  
    shlib="C:\IBM\WebSphere\Plugins\bin\ns41_http.dll"  
Init fn="as_init"  
    bootstrap.properties=  
        "profile_root\config\cells\  
        dmgrcell\nodes\managed_node\servers\webserver1\plugin-cfg.xml"
```

Example:

```
Init fn="load-modules"  
    funcs="as_init,as_handler,as_term"  
    shlib="C:\IBM\WebSphere\Plugins\bin\ns41_http.dll"  
Init fn="as_init"  
    bootstrap.properties="C:\IBM\WebSphere\Plugins\config\webserver1\plugin-cfg.xml"
```

2. Set the shared library path on HP-UX machines. On some installations of Sun Java System Web Server on an HP-UX machine, it is necessary to manually set the SHLIB_PATH variable to /usr/lib before starting Sun Java System Web Server with a plug-in that is configured for Secured Sockets Layer (SSL). For example, in the korn shell, issue the following command before invoking the command to start the Sun Java System Web Server:

```
export SHLIB_PATH=/usr/lib:$SHLIB_PATH
```

3. Disable the feature of Sun Java System Web Server Version 6.1 that supports servlets and JavaServer Pages files by default. Disable this feature so that the WebSphere Application Server plug-in can handle the requests.

Perform the following steps to disable the feature:

- a. Remove or comment out the following two lines from the obj.conf configuration file:

```
NameTrans fn="ntrans-j2ee" name="j2ee"  
Error fn="error-j2ee"
```

- b. Remove or comment out the following line from the magnus.conf configuration file:

```
Init fn="load-modules"  
    shlib="C:/Sun/WebServer6.1/bin/https/bin/j2eeplugin.so"  
    shlib_flags="(global|now)"  
  
Init fn="load-modules"  
    shlib="C:\Sun\WebServer6.1\bin\https\bin\j2eeplugin.dll"  
    shlib_flags="(global|now)"
```

Results

This procedure results in editing and reconfiguring the Sun Java System Web Server.

What to do next

After configuring a Web server, you can install applications on it. See the Applications section of the information center for more information.

Installing Web server plug-in maintenance

Use the Update Installer program to install maintenance packages for Web server plug-ins. Stop the Web server before installing the maintenance package to allow the Update Installer to update the configuration files in the Web server.

Before you begin

Before installing a maintenance package for a Web server plug-in, you must first install the Web server plug-in from the WebSphere Application Server installation image. See "Installing Web server plug-ins" on page 6 for more information.

About this task

This topic describes installing a maintenance package for a Web server plug-in for WebSphere Application Server. WebSphere Application Server products supply a unique binary plug-in module for each supported Web server, which might require service updates that are included in a maintenance package. The plug-in configuration file that the WebSphere Application Server products supply is associated with the binary module and might also require maintenance. The Update Installer wizard installs any maintenance packages that you download from the Support site.

1. Go to the Recommended Updates for WebSphere Application Server Web page to determine which maintenance packages are available for the Web server plug-ins.

Install maintenance packages for Web server plug-ins in this order:

- a. Refresh packs
 - b. Fix packs
 - c. Interim fixes
2. Download the appropriate maintenance packages to a temporary location.
For example, click the ftp link for the download package labeled, Intel® Plug-ins, to download the maintenance package.

3. Follow the directions for using the Update Installer wizard to install the maintenance packages.
Verify that you stop the Web server process before installing a maintenance package for the Web server plug-ins.

Results

You can install a maintenance package for a Web server plug-in by following the procedures described in this topic.

What to do next

After installing maintenance packages, test the applications that the Web server provides to verify that the Web server is functioning correctly.

Uninstalling the Web server plug-ins for WebSphere Application Server

You can uninstall the Web server plug-ins for WebSphere Application Server using the uninstaller program.

Before you begin

You can uninstall Web server plug-ins for WebSphere Application Server without uninstalling the supported Web server or the application server.

If you used the IBM Key Management wizard to create SSL key files in the Web server Plug-ins home directory, back up the files to a directory outside of the Web server Plugins directory. After the uninstall procedure is complete, you can delete the SSL key files if they are no longer required.

About this task

To uninstall the Web server plug-ins, run the uninstaller program. The program deletes the installation root directory for the Web server plug-ins for WebSphere Application Server, which removes the binary plug-ins.

1. Stop the Web server to allow the uninstaller program to change the Web server configuration.
2. Open a command window.
3. Change directories to the *plugins_root/uninstall* directory.
4. Issue the uninstall command.

The Uninstall wizard displays a welcome panel.

The next few steps in this procedure describe using the wizard interactively. You can also issue the uninstall command with a silent parameter to use the wizard without the graphical user interface.

```
uninstall -silent
```

5. Click **Next** on the Welcome panel.

The Uninstaller wizard displays a confirmation panel for you to confirm what is to be uninstalled. Each feature that displays in the panel is a separate Web server installation. Each Web server installation is configured to use one of the binary plug-in modules that is being deleted.

6. Click **Next** on the confirmation panel to begin uninstalling the plug-ins.

The Uninstaller wizard displays a summary panel that provides status.

7. Click **Finish** to close the Uninstaller wizard.

Uninstalling the Web server plug-ins for WebSphere Application Server removes many files in the installation root directory, but leaves the following logs in the *plugins_root/log/uninstall* directory:

- ISMP uninstall log: log.txt
- Configuration uninstall log: masterConfigurationLog.txt

- Web server deconfiguration log: `uninstallweb_server_namePlugin.log`, such as the `uninstallApachePlugin.log`.
8. Uninstall the IBM Global Security Kit (GSKit).

The Plug-ins installation wizard does not uninstall GSKit 7 because there is no registry describing products that are enrolled to use the GSKit. Without this critical information, uninstalling the GSKit product might affect other products that use GSKit.

You can uninstall GSKit once it is no longer in use on your systems.

Solaris On Solaris systems, you must also uninstall GSKit 4 in addition to GSKit 7.

The Plug-ins uninstaller program unregisters the GSKit. The registry key is **HKEY_LOCAL_MACHINE > SOFTWARE > IBM > GSK7 > REGAPPS > WASPlugins60_unique_key**.

If the registry key `WASPlugins60_unique_key` is the last remaining key in the GSKit registry entry, the wizard also uninstalls the GSK7 product. If another product, such as IBM HTTP Server, is registered to use GSKit, the wizard does not uninstall the GSKit. The wizard always unregisters the registry key for the Web server plug-ins for WebSphere Application Server, which is `WASPlugins60_unique_key`.

The GSKit uninstall log is the `plugins_root/log/uninstall/uninstallGSKit.log` file.

Possible problem after using the Update Installer to install a maintenance package

When you apply a WebSphere Application Server plug-in refresh pack, the plug-in installer incorrectly adds an additional registry key that prevents the GSKit from uninstalling.

See “Manually uninstalling Web server plug-ins for WebSphere Application Server” on page 103 for information about manually uninstalling the GSKit.

9. Delete files from the system temporary directory.

Delete the following files:

- Install temporary log : `temporaryPluginInstallLog.txt`
- Uninstall temporary log : `temporaryPluginUnInstallLog.txt`

10. Delete the Web server definition in a standalone application server.

The uninstaller program for the Web server plug-ins for WebSphere Application Server does not delete Web server definitions. However, you can delete a Web server definition from admin console OR by using the following `wsadmin` commands:

```
$AdminTask deleteServer { -serverName webserver1 -nodeName WebserverHostName-node_node }
$AdminTask removeUnmanagedNode { -nodeName WebserverHostName-node_node }
$AdminConfig save
```

Results

After you exit from the `plugins_root/uninstPlugin` directory, the directory is removed.

The only remaining directory is the `plugins_root/logs` directory. The logs directory contains the install process log, the uninstall process log, and the Web server creation logs.

Note: The only other files that might exist are the SSL key files that you can create using the IBM Key Management Wizard. You can move these files to a safe location before using the manual uninstalling procedure, if necessary.

What to do next

To reinstall the Web server plug-ins for WebSphere Application Server, launch the installation procedure again.

To reinstall the Web server plug-ins for WebSphere Application Server into the original directory, delete the existing installation root directory for the plug-ins before reinstalling.

The default location is shown in “Directory conventions,” on page 419.

See “Installing Web server plug-ins” on page 6 for information about installation scenarios for reinstalling Web server plug-ins.

See “Manually uninstalling Web server plug-ins for WebSphere Application Server” for information about manually uninstalling Web server plug-ins.

Manually uninstalling Web server plug-ins for WebSphere Application Server

You can uninstall Web server plug-ins for WebSphere Application Server manually, instead of using the uninstaller program.

Before you begin

Try uninstalling Web server plug-ins for WebSphere Application Server using the uninstaller program. Use this manual procedure if the uninstaller is not available for some reason.

About this task

To manually uninstall the Web server plug-ins, delete the installation root directory for the Web server plug-ins for WebSphere Application Server. Deleting the directory removes the binary plug-ins.

1. Delete the installation root directory for Web server plug-ins for WebSphere Application Server.
2. Delete the Web server definition in the application server configuration.

You can delete a Web server definition from the admin console OR by using the following wsadmin commands:

```
$AdminTask deleteServer { -serverName webserver1 -nodeName WebserverHostName-node_node }  
$AdminTask removeUnmanagedNode { -nodeName WebserverHostName-node_node }  
$AdminConfig save
```

3. Manually delete GSKit.

Use the following procedure to verify that no other products are registered in GSKit before running the `isuninst` command.

- a. Click **Start > Run** and run the `regedit` command to edit the registry.
- b. Change directories to `HKEY_LOCAL_MACHINE\SOFTWARE\IBM\GSKx\CurrentVersion\REGAPP`, where `x` is the version of GSKit, such as `GSK7`.
- c. Look for other products that are registered within GSKit.

Look for entries other than the default entry and for products other than IBM HTTP Server and Web Server plug-ins for WebSphere Application Server. If the following entries are the only ones present, you can delete GSKit as described in the following step:

- (Default)
- WASPLUGIN60
- IHS60

- d. Run the following command to invoke the GSKit Uninstaller program:

```
isuninst -f"gskit_root\gsk7BUI.isu"
```

This command removes GSKit, regardless of whether or not other applications are registered as using GSKit.

4. Reconfigure any Web servers that were configured to use the binary plug-ins that you deleted.
See “Editing Web server configuration files” on page 81.
5. Delete the following registry keys using the platform-specific steps described in the “Uninstalling manually” topic.

Operating system	Registry keys
AIX, Linux, and Windows	WSPAA70, WSPAA70DefineglobalconstantsComponent, WSPAA70DefinelocalvariablesComponent, WSPAA70LicensingComponent, WSPAA70Webserverplugins, WSPAA70WebserverpluginsComponent, WSPAA70AddBytes, WSPAA70gskit, WSPAA70gskitComponent
HP-UX	WSPAA70, WSPAA70DGCC, WSPAA70DLVC, WSPAA70LC, WSPAA70WSPC, WSPAA70AddBytesHS, WSPAA70gskitHP, WSPAA70gskitHPC, WSPAA70jdkHP
Solaris	WSPAA70, WSPAA70AC, WSPAA70BC, WSPAA70CC, WSPAA70DC, WSPAA70FC, WSPAA70FB, WSPAA70GC, WSPAA70HC

Results

Deleting the installation root directory for the Web server plug-ins for WebSphere Application Server removes the binary plug-ins. Any Web servers that are configured to use the deleted binary modules do not work.

What to do next

After uninstalling the Web server plug-ins for WebSphere Application Server, you can reinstall the plug-ins. Reinstalling the Web server plug-ins for WebSphere Application Server and reconfiguring the Web servers restores their functionality.

See “Installing Web server plug-ins” on page 6 for information about other installation scenarios for reinstalling Web server plug-ins.

Allowing Web servers to access the administrative console

This topic describes how to add the virtual host that servers the administrative console to the plug-in configuration file so that you can access the administrative console through a Web server.

Before you begin

Note: If a Web server is in a configuration, its port must be higher than 1023 to use a non-root node agent. Otherwise the node agent must be running as root in order for the administrative console of the deployment manager to stop and start the Web server process in that managed node.

Install your Version 6 WebSphere Application Server product, a Web server, and the Web server plug-ins for WebSphere Application Server.

The Plug-ins installation wizard creates a Web server definition on the Application Server system, either directly when they are on the same machine, or by a script for remote scenarios.

After creating the Web server definition, the plug-in configuration file exists within the Web server definition.

The plugin-cfg.xml file can be overwritten by the deployment manager synchronization operation, the GenPluginCfg script or any other method that regenerates the file. If you make changes to the plugin-cfg.xml file, and want to keep those changes, it is recommended that you create a copy of the file in a separate location. Make your manual updates each time the file is automatically refreshed by another process.

About this task

This task gives you the option of configuring the `admin_host` so that Web servers can access the administrative console. When the Web server plug-in configuration file is generated, it does not include `admin_host` on the list of virtual hosts.

1. Use the administrative console to change the `admin_host` virtual host group to include the Web server port (80 by default).

- a. Click **Environment > Virtual Host > admin_host > Host Aliases > New**.

The default port that displays is 80, unless you specify a different port during profile creation.

- b. Specify the IP address, or the name of the machine that is hosting the HTTP server.

For example, if you installed a WebSphere Application Server product on a machine that is named `waslwaj.rtp.ibm.com`, specify the name in this field.

2. Click **Apply > Save**.

3. Stop and restart the application server.

For example, to access the administrative console of a stand-alone application server, stop and restart the `server1` process.

To stop `server1`, open a command window and navigate to the `profile_root/bin` directory. Then issue the following command:

```
./stopServer.sh server1
```

After receiving the following message, you can restart the application server:

```
Server server1 stop completed.
```

To start the application server, issue the following command:

```
./startServer.sh server1
```

When you receive a message that is similar to the following message, the `server1` process is running:

```
Server server1 open for e-business; process id is 1719
```

4. Stop and restart a deployment manager.

For example, to access the administrative console of a deployment manager, stop and restart the deployment manager.

To stop the deployment manager, open a command window and navigate to the `profile_root/bin` directory. Then issue this command:

```
./stopManager.sh
```

Then issue the following command to stop the deployment manager:

```
./stopManager.sh
```

After receiving the following message, you can restart the deployment manager:

```
Server dmgr stop completed.
```

To start the deployment manager, issue the following command:

```
./startManager.sh
```

When you receive a message that is similar to the following message, the deployment manager is running:

```
Server dmgr open for e-business; process id is 1720
```

5. Edit the `plugin-cfg.xml` file to include the following entries:

```
<VirtualHostGroup Name="admin_host">
  <VirtualHost Name="*:9060"/>
  <VirtualHost Name="*:80"/>
  <VirtualHost Name="*:9043"/>
</VirtualHostGroup>
...
...
...
<ServerCluster Name="server1_SERVER1HOSTserver1_Cluster">
  <Server LoadBalanceWeight="1" Name="SERVER1HOSTserver1_dmgr">
```

```

    <Transport Hostname="SERVER1HOST" Port="9060" Protocol="http"/>
  </Server>

  <PrimaryServers>
    <Server Name="SERVER1HOSTserver1_dmgr"/>
  </PrimaryServers>
</ServerCluster>
...
...
...
<UriGroup Name="admin_host_server1_SERVER1HOSTserver1_Cluster_URIs">
  <Uri AffinityCookie="JSESSIONID"
    AffinityURLIdentifier="jsessionid" Name="/ibm/console/*"/>
</UriGroup>
<Route ServerCluster="server1_SERVER1HOSTserver1_Cluster"
  UriGroup="admin_host_server1_SERVER1HOSTserver1_Cluster_URIs" VirtualHostGroup="admin_host"/>

```

If your HTTP server has an HTTP port other than 80, add an entry to the VirtualHostGroup:

```

<VirtualHost Name="*:port"/>

```

The *port* variable is your HTTP server port.

Results

You can configure your supported Web servers to access the administrative console application of a deployment manager or a stand-alone application server.

Web server plug-in properties

Use this page to view or change the settings of a Web server plug-in configuration file. The plug-in configuration file, `plugin_cfg.xml`, provides properties for establishing communication between the Web server and the Application Server.

To view this administrative console page, click **Servers > Server Types > Web servers > *web_server_name* > Plug-in Properties**.

On the **Configuration** tab, you can edit fields. On the **Runtime** tab, you can look at read-only information.

The **Runtime** tab is available only when this Web server has accessed applications running on application servers and there is an `http_plugin.log` file.

Ignore DNS failures during Web server startup

Specifies whether the plug-in ignores DNS failures within a configuration when starting.

This field corresponds to the `IgnoreDNSFailures` element in the `plugin-cfg.xml` file.

When set to **true**, the plug-in ignores DNS failures within a configuration and starts successfully if at least one server in each `ServerCluster` is able to resolve the host name. Any server for which the host name can not be resolved is marked **unavailable** for the life of the configuration. No attempts to resolve the host name are made later on during the routing of requests. If a DNS failure occurs, a log message is written to the plug-in log file and the plug-in initialization continues rather than causing the Web server not to start. When **false** is specified, DNS failures cause the Web server not to start.

Data type	String
Default	false

Refresh configuration interval

Specifies the time interval, in seconds, at which the plug-in should check the configuration file to see if updates or changes have occurred. The plug-in checks the file for any modifications that have occurred since the last time the plug-in configuration was loaded.

In a development environment in which changes are frequent, a lower setting than the default setting of 60 seconds is preferable. In production, a higher value than the default is preferable because updates to the configuration will not occur so often. If the plug-in reload fails for some reason, a message is written to the plug-in log file and the previous configuration is used until the plug-in configuration file successfully reloads. If you are not seeing the changes you made to your plug-in configuration, check the plug-in log file for indications of the problem.

Data type	Integer
Default	60 seconds.

Plug-in configuration file name

Specifies the file name of the configuration file for the plug-in. The Application Server generates the `plugin-cfg.xml` file by default. The configuration file identifies applications, Application Servers, clusters, and HTTP ports for the Web server. The Web server uses the file to access deployed applications on various Application Servers.

You can change the name of the plug-in configuration file. However, if you do change the file name, you must also change the Web server configuration to point to the new plug-in configuration file.

If you select a Web server plug-in during installation, the installer program configures the Web server to identify the location of the `plugin-cfg.xml` file, if possible. The plug-in configuration file, by default, is installed in the `plugins_root/config/web_server_name` directory.

The installer program adds a directive to the Web server configuration that specifies the location of the `plugin-cfg.xml` file.

For remote Web servers, you must copy the file from the local directory where the Application Server is installed to the remote machine. This is known as propagating the plug-in configuration file. If you are using IBM HTTP Server V6.1 or higher for your Web server, WebSphere Application Server can automatically propagate the plug-in configuration file for you to remote machines provided there is a working HTTP transport mechanism to propagate the file.

You can click **View** to display a copy of the current plug-in configuration file.

Data type	String
Default	<code>plugin-cfg.xml</code>

Automatically generate plug-in configuration file

To automatically generate a plug-in configuration file to a remote Web server:

- This field must be checked.
- The plug-in configuration service must be enabled

When the plug-in configuration service is enabled, a plug-in configuration file is automatically generated for a Web server whenever:

- The WebSphere Application Server administrator defines new Web server.
- An application is deployed to an Application Server.
- An application is uninstalled.

- A virtual host definition is updated and saved.

By default, this field is checked. Clear the check box if you want to manually generate a plug-in configuration file for this Web server.

Note: When the plug-in configuration file is generated, it does not include `admin_host` on the list of virtual hosts. The Information Center article "Allowing Web servers to access the administrative console" describes how to add `admin_host` to the list of virtual hosts.

Automatically propagate plug-in configuration file

Specifies whether or not you want the application server to automatically propagate a copy of a changed plug-in configuration file to a Web server:

- This field must be checked.
- The plug-in configuration service must be enabled
- In a Network Deployment environment, a WebSphere Application Server node agent must be on the node that hosts the Web server associated with the changed plug-in configuration file.

By default, this field is checked.

Note: The plug-in configuration file can only be automatically propagated to a remote Web server if that Web server is an IBM HTTP Server V6.1 or higher Web server and its administration server is running.

Because the plug-in configuration service runs in the background and is not tied to the administrative console, the administrative console cannot show the results of the automatic propagation.

For distributed platforms, you can check the related messages in the deployment manager `SystemOut.log` file to verify that the automatic propagation successfully completed.

Plug-in key store file name

Specifies the fully qualified directory path and file name of the database file containing your security key rings that the Web server plug-in uses for HTTPS requests. This file resides on the Web server that is associated with this Web server plug-in. After you specify the fully qualified directory path and file name of the database file, you can:

- Click **Manage keys and certificates** to update this file.
- Click **Copy to Web server key store directory** to add a copy of this file to the key store directory for the Web server.

Data type	String
Default	None

Plug-in configuration directory and file name

Specifies the fully qualified path of the Web server copy of the Web server plug-in configuration file. This path is the name of the file and its location on the machine where the Web server is running.

Plug-in key store directory and file name

Specifies the fully qualified path of the Web server copy of the database file that contains your security key rings. This path is the name of the file and its location on the machine where the Web server is running.

Plug-in logging

Specifies the location and name of the `http_plugin.log` file. Also specifies the scope of messages in the log.

The log describes the location and level of log messages that are written by the plug-in. If a log is not specified within the configuration file, then, in some cases, log messages are written to the Web server error log.

On a distributed platform, if the log file does not exist then it will be created. If the log file already exists, it will be opened in append mode and the previous plug-in log messages will remain.

Log file name - The fully qualified path to the log file to which the plug-in will write error messages.

Data type	String
Default	<code>plugins_root/logs/web_server_name/http_plugin.log</code>

Specify the file path of the `http_plugin.log` file.

Log level- The level of detail of the log messages that the plug-in should write to the log. You can specify one of the following values for this attribute:

- Trace. All of the steps in the request process are logged in detail.
- Stats. The server selected for each request and other load balancing information relating to request handling is logged.
- Warn. All warning and error messages resulting from abnormal request processing are logged.
- Error. Only error messages resulting from abnormal request processing are logged.
- Debug. All of the critical steps performed in processing requests are logged.
- Detail. All of the information about requests and responses are logged.

If a Log level is not specified, the default value **Error** is used.

Be careful when setting the level to **Trace**. A lot of messages are logged at this level which can cause the disk space/file system to fill up very quickly. A **Trace** setting should never be used in a normally functioning environment as it adversely affects performance.

Note: If the Web server and Web server plug-in are running on an AIX, HP-UX, Linux, or Solaris system, and you change the log level, in the `plugin-cfg.xml` file, this change is not picked up dynamically. You must restart the Web server to pick up the change. For example on Solaris, if you do not restart the Web server, the following error message appears in the `Plugin_Home/logs/http_plugin.log` file:

```
ERROR: ws_config_parser:handleLogEnd: Failed to open log file
'/opt/IBM/WebSphere/Plugin/logs/sunwebserver/http_plugin.log', OS
```

Data type	String
Default	Error

Web server plug-in request and response optimization properties

Use this page to view or change the request and response optimization properties for a Web server plug-in.

To view this administrative console page, click **Servers > Server Types > Web servers > web_server_name > Plug-in properties > Request and response**.

Maximum chunk size used when reading the HTTP response body

Specifies the maximum chunk size the plug-in can use when reading the response body.

This field corresponds to the `ResponseChunkSize` element in the `plugin-cfg.xml` file.

The plug-in reads the response body in 64K chunks until all of the response data is read. This approach causes a performance problem for requests whose response body contains large amounts of data.

If the content length of the response body is unknown, the values specified for this property is used as the size of the buffer that is allocated. The response body is then read in this size chunks, until the entire body is read. If the content length is known, then a buffer size of either the content length or the specified size (whichever is less) is used to read the response body.

Data type

Integer

Default

64 kilobytes

Specify the size in kilobytes (1024 byte blocks).

Enable Nagle algorithm for connections to the Application Server

When checked, the Nagle algorithm is enabled for connections between the plug-in and the Application Server.

This field corresponds to the ASDisableNagle element in the plugin-cfg.xml file.

The Nagle algorithm is named after engineer John Nagle, who invented this standard part of the transmission control protocol/internet protocol (TCP/IP). The algorithm reduces network overhead by adding a transmission delay (usually 20 milliseconds) to a small packet, which lets other small packets arrive and be included in the transmission. Because communications has an associated cost that is not as dependent on packet size as it is on frequency of transmission, this algorithm potentially reduces overhead with a more efficient number of transmissions.

By default, this field is not checked, and the Nagle algorithm is disabled. Select this field to enable the Nagle algorithm.

Enable Nagle Algorithm for the IIS Web Server

When checked, the Nagle algorithm is used for connections from the Microsoft Internet Informations Services (IIS) Web Server to the Application Server.

This field corresponds to the IHSDisableNagle element in the plugin-cfg.xml file. It only appears if you are using the Microsoft Internet Informations Services (IIS) Web server.

By default, this field is not checked, and the Nagle algorithm is disabled. Select this field to enable the Nagle algorithm for this connection.

Chunk HTTP response to the client

When checked, responses to the client are broken into chunks if a Transfer-Encoding : Chunked response header is present in the response.

This field corresponds to the ChunkedResponse element in the plugin-cfg.xml file. It only appears if you are using a Microsoft Internet Informations Services (IIS) Web Server, a Java System Web server, or a Domino Web server. The IBM HTTP Server automatically handles breaking the response into chunks to send to the client.

By default, this field is not checked, and responses are not broken into chunks. Select this field to enable responses to the client to be broken into chunks if a Transfer-Encoding : Chunked response header is present in the response.

Accept content for all requests

This field corresponds to the AcceptAllContent element in the plugin-cfg.xml file.

When selected, users can include content in POST, PUT, GET, and HEAD requests when a Content-Length or Transfer-encoding header is contained in the request header.

By default, this field is not checked. Select this field to enable users to include content in POST, PUT, GET, and HEAD requests when a Content-Length or Transfer-encoding header is contained in the request header.

Virtual host matching

When selected, virtual host mapping is performed by physically using the port number for which the request was received.

This field corresponds to the `VHostMatchingCompat` element in the `plugin-cfg.xml` file.

By default, this field is not checked, and matching is done logically using the port number contained in the host header. Select this field if you want virtual host mapping performed by physically using the port number for which the request was received.

Use the radio buttons to make your physical or logical port selection.

Application server port preference

Specifies which port number the Application Server should use to build URIs for a `sendRedirect`. This field is only applicable for a `sendRedirect` if you use relative URIs and does not affect absolute redirects. This field also specifies where to retrieve the value for `HttpServletRequest.getServerPort()`.

This field corresponds to the `AppServerPortPreference` element in the `plugin-cfg.xml` file.

Specify:

- `hostHeader` if the port number from the host header of the HTTP request coming in is to be used.
- `webserverPort` if the port number on which the Web server received the request is to be used.

The default is `hostHeader`.

Web server plug-in caching properties

Use this page to view or change the caching properties for a Web server plug-in.

To view this administrative console page, click **Servers > Server Types > Web servers > *web_server_name* > Plug-in properties > Caching Properties**.

Enable Edge Side Include (ESI) processing to cache the responses

Specifies whether to enable Edge Side Include processing to cache the responses.

This field corresponds to the `esiEnable` element in the `plugin-cfg.xml` file.

By default, this field is not checked. Select this field if you want Edge Side Include (ESI) processing used to cache responses. If ESI processing is disabled for the plug-in, the other ESI plug-in properties are ignored. Clear the checkbox to disable Edge Side Include processing.

Enable invalidation monitor to receive notifications

When checked, the ESI processor receives invalidations from the application server.

This field corresponds to the `ESIInvalidationMonitor` element in the `plugin-cfg.xml` file. It is ignored if Edge Side Include (ESI) processing is not enabled for the plug-in.

By default, this field is selected. Clear the check box if you do not want the application server to send invalidations to the ESI processor.

Maximum cache size

Specifies, in 1K byte units, the maximum size of the cache. The default maximum size of the cache is 1024K bytes (1 megabyte). If the cache is full, the first entry to be evicted from the cache is the entry that is closest its expiration time.

This field corresponds to the esiMaxCacheSize element in the plugin-cfg.xml file.

Data type	Integer
Default	1024 kilobytes
	Specify the size in kilobytes (1024 byte blocks).

Web server plug-in request routing properties

Use this page to view or change the request routing properties for a Web server plug-in.

To view this administrative console page, click **Servers > Server Types > Web servers > *web_server_name* > Plug-in properties > Request routing**.

Load balancing option

Specifies the load balancing option that the plug-in uses in sending requests to the various application servers associated with that Web server.

This field corresponds to the LoadBalanceWeight element in the plugin-cfg.xml file.

Select the appropriate load balancing option:

- Round robin
- Random

The Round Robin implementation has a random starting point. The first application server is picked randomly. Round Robin is then used to pick application servers from that point forward. This implementation ensures that in multiple process based Web servers, all of the processes don't start up by sending the first request to the same Application Server.

The default load balancing type is Round Robin.

Retry interval

Specifies the length of time, in seconds, that should elapse from the time an application server is marked down to the time that the plug-in retries a connection.

This field corresponds to the ServerWaitforContinue element in the plugin-cfg.xml file.

Data type	Integer
Default	60 seconds

Maximum size of request content

Select whether there is a limit on the size of request content. If limited, this field also specifies the maximum number of kilobytes of data a request can contain. When a limit is set, the plug-in fails any request that is received that contains more data than the specified limit.

This field corresponds to the PostSizeLimit element in the plugin-cfg.xml file.

Select whether to limit the size of request content:

- No limit
- Set limit

If you select **Set limit**, specify a limit size.

Data type

Integer

Default

Specify the size in kilobytes (1024 byte blocks).
-1, which indicates there is no limit for the post size.

Maximum buffer size used when reading HTTP request content

Specifies, in kilobytes, the maximum buffer size that is used when the content of an HTTP request is read. If the application server that initially receives a request cannot process that request, the data contained in this buffer is sent to another application server in an attempt to have that application server process the request.

This field corresponds to the `PostBufferSize` element in the `plugin-cfg.xml` file.

If **Set limit** is selected, specify a limit size.

Data type

Integer

Default

Specify the size in kilobytes (1024 byte blocks).
64

Remove special headers

When checked, the plug-in will remove any headers from incoming requests before adding the headers the plug-in is supposed to add before forwarding the request to an application server.

This field corresponds to the `RemoveSpecialHeaders` element in the `plugin-cfg.xml` file.

The plug-in adds special headers to the request before it is forwarded to the application server. These headers store information about the request that will need to be used by the application. Not removing the headers from incoming requests introduces a potential security exposure.

By default, the special headers are not retained. Clear the check box to retain special headers.

Clone separator change

When this option is selected, the plug-in expects the plus character (+) as the clone separator.

This field corresponds to the `ServerCloneID` element in the `plugin-cfg.xml` file.

Some pervasive devices cannot handle the colon character (:) used to separate clone IDs in conjunction with session affinity. If this field is checked, you must also change the configurations of the associated application servers such that the application servers separates clone IDs with the plus character as well.

By default, this option is selected. Clear the field if you want to use the colon character to separate clone IDs.

Web server plug-in configuration service property

Use this page to view or change the configuration settings for the Web server plug-in configuration service.

If you are using a stand-alone application server, click **Servers > Server Types > WebSphere application servers > *server_name* > Administration Services > Web server plug-in configuration service** to view this administrative console page.

For a WebSphere Application Server Network Deployment configuration, if you are using the deployment manager, click **System Administration > Deployment manager > Administration Services > Web server plug-in configuration service**.

For i5/OS and distributed platforms running in a Network Deployment environment, the deployment manager SystemOut log contains the status of the automatic plug-in generation and propagation.

For i5/OS and distributed platforms running in a Base or Express environment, the application server's SystemOut log contains the status of the automatic plug-in generation and propagation.

Enable automated Web server configuration processing

The Web server plug-in configuration service is selected by default. The service automatically generates the plug-in configuration file whenever the Web server environment changes, with a few exceptions. For example, the plug-in configuration file is regenerated whenever one of the following activities occurs:

- A new application is deployed on an associated application server
- The Web server definition is saved
- An application is removed from an associated application server
- A new virtual host is defined

In WebSphere Application Server Network Deployment configurations, the plug-in configuration file does not regenerate when:

- A cluster member is added to a cluster.
- TCP channel settings are updated for an application server.

By default, this option is selected. Clear the field to disable automated Web server configuration processing.

Application Server property settings for a Web server plug-in

Use this page to view or change application server settings for a Web server plug-in.

To view this administrative console page, click **Servers > Server Types > WebSphere application servers > server_name**, and then, in the Additional Properties section, click **Web server plug-in properties**.

Server Role

Specifies the role this application server is assigned.

Select **Primary** to add this application server to the list of primary application servers. The plug-in initially attempts to route requests to the application servers on this list.

Select **Backup** to add this application server to the list of backup application servers. The plug-in does not load balance across the backup application servers. A backup server is only used if a primary server is not available. When the plug-in determines that a backup application server is required, it goes through the list of backup servers, in order, until no servers are left in the list or until a request is successfully sent and a response received from one of the servers on this list.

Default setting

Primary

Connection timeout

Specifies the connection timeout settings.

This setting specifies whether or not there is a limited amount of time the application server will maintain a connection with the Web server. Check the **Use connection timeout** checkbox to set a connection timeout. If no timeout is selected, the plug-in performs nonblocking connections with the application server. If the checkbox is checked, you must specify a value in the seconds field. Specify a value of 0, if you want the plug-in to perform a blocking connection. Specify a value greater than 0, if you want the plug-in to wait the specified number for seconds to perform a successful connection. If a connection does not occur after that time interval, the plug-in marks the server unavailable and sends the request to another application server defined in the cluster.

This property enables the plug-in to perform non-blocking connections with the application server. Non-blocking connections are beneficial when the plug-in is unable to contact the destination to determine whether or not the port is available. If no value is specified for this property, the plug-in performs a blocking connect in which the plug-in sits until an operating system times out (which could be as long as 2 minutes depending on the platform) and allows the plug-in to mark the server unavailable.

A value of 0 causes the plug-in to perform a blocking connect. A value greater than 0 specifies the number of seconds you want the plug-in to wait for a successful connection. If a connection does not occur after that time interval, the plug-in marks the server unavailable and fails over to another application server defined for the requested application.

Data type	Integer
Default	5

Read/Write timeout

Specifies whether there is a time limit for how long the plug-in waits to send a request to or receive a response from the application server.

Check the **Use read/write timeout** checkbox to set a read/write timeout. If the checkbox is checked, you must specify the length of time, in seconds that the plug-in waits to send a request or to receive a response. When selecting a value to specify for this field, remember that it might take a couple of minutes for an application server to process a request. Setting the value too low might cause the plug-in to send a false server error response to the client. If the checkbox is not checked, the plug-in uses blocked I/O to write requests to and read responses from the application server until the TCP connection times out.

This field is ignored for a plug-in running on a Solaris platform.

Data type	Integer
Default	60 seconds

Maximum number of connections that can be handled by the Application Server

Specifies the maximum number of pending connections to an Application Server that can be flowing through a Web server process at any point in time.

This field corresponds to the ServerMaxConnections element in the plugin-cfg.xml file.

Check the **Use maximum number of connections** checkbox to set a maximum number of connections. If the checkbox is checked you, must specify the maximum number of connections that can exist between the Web server and the Application Server at any given point in time.

For example, assuming that:

- The application server is fronted by 5 nodes that are running an IHS Web server.
- Each node starts 2 processes.

- This property is set to 50.

In this example, the application server could potentially get up to 500 connections. (You take the number of nodes, 5, multiply it by the number of processes, 2, and then multiply that number by the number specified for this property, 50, for a total of 500 connections.)

If this attribute is set to either zero or -1, there is no limit to the number of pending connections to the Application Servers.

Data type	Integer
Default	0

Use extended handshake to check whether application server is running

When selected, the Web server plug-in will use an extended handshake to check whether or not the Application Server is running.

This field corresponds to the ServerExtendedHandshake element in the plugin-cfg.xml file.

Select this property if a proxy firewall is between the plug-in and the application server.

The plug-in marks a server as down when the connect() fails. However, when a proxy firewall is in between the plug-in and the application server, the connect() will succeed, even though the back end application server is down. This causes the plug-in to not failover correctly to other application servers.

If the plug-in performs some handshaking with the application server to ensure that it is started before it sends a request it can failover to another application server if it detects that the application server with which it is attempting to perform a handshake is down.

By default, this field is not checked. Select this field if you want to use extended handshake to check whether an application server is running.

Send the header "100 Continue" before sending the request content

This field corresponds to the WaitForContinue element in the plugin-cfg.xml file.

When selected, the Web server plug-in will send the header "100 Continue" to the application server before it sends the request content.

By default, this field is not checked. Select this field to enable this function.

Web server plug-in configuration properties

The following table indicates which panel in the administrative console you need to use to manually configure a Web server plug-in property.

Table 7. Web server plug-in configuration properties

Administrative console panel	Field name	Configuration property name
In the administrative console, click Servers > Server Types > Web servers > web_server_name > Plug-in properties	Refresh configuration interval	RefreshInterval

Table 7. Web server plug-in configuration properties (continued)

In the administrative console, click Servers > Server Types > Web servers > web_server_name > Plug-in properties	Plug-in log file name	Log->name
In the administrative console, click Servers > Server Types > Web servers > web_server_name > Plug-in properties	Plug-in logging	Log->LogLevel
In the administrative console, click Servers > Server Types > Web servers > web_server_name > Plug-in properties	Ignore DNS failures during Web server startup	IgnoreDNSFailures
In the administrative console, click Servers > Server Types > Web servers > web_server_name > Plug-in properties	KeyringLocation	Keyring
In the administrative console, click Servers > Server Types > Web servers > web_server_name > Plug-in properties	StashfileLocation	Stashfile
In the administrative console, click Servers > Server Types > Web servers > web_server_name > Plug-in properties > Custom properties > New	FIPSEnable	FIPSEnable
In the administrative console, click Servers > Server Types > Web servers > Web_server_name > Plug-in properties > Request routing	Load balancing option	LoadBalance
In the administrative console, click Servers > Server Types > Web servers > web_server_name > Plug-in properties > Request Routing	Clone separator change	CloneSeparatorChange
In the administrative console, click Servers > Server Types > Web servers > web_server_name > Plug-in properties > Request Routing	Retry interval	RetryInterval
In the administrative console, click Servers > Server Types > Web servers > web_server_name > Plug-in properties > Request routing	Maximum size of request content	PostSizeLimit
In the administrative console, click Servers > Server Types > Web servers > Web server_name > Plug-in properties > Request routing	Size of the buffer that is used to cache POST requests	PostBufferSize

Table 7. Web server plug-in configuration properties (continued)

In the administrative console, click Servers > Server Types > Web servers > Web_server_name > Plug-in properties > Request routing	Remove special headers	RemoveSpecialHeaders
In the administrative console, click Servers > Server Types > WebSphere application servers > server_name > Web server plug-in properties	Server role	PrimaryServers and BackupServers list
In the administrative console, click Servers > Server Types > WebSphere application servers > server_name > Web server plug-in properties	Connect timeout	Server ConnectTimeout
In the administrative console, click Servers > Server Types > Web servers > web_server_name > plug-in properties > Custom properties > New	The read and write timeouts for all the connections to the application server	ServerIOTimeout
In the administrative console, click Servers > Server Types > WebSphere application servers > server_name > Web server plug-in properties	Use extended handshake to check whether Application Server is running	Server Extended Handshake
In the administrative console, click Servers > Server Types > WebSphere application servers > server_name > Web server plug-in properties	Send the header "100 Continue" before sending the request content	WaitForContinue
In the administrative console, click Servers > Server Types > WebSphere application servers > server_name > Web server plug-in properties	Maximum number of connections that can be handled by the Application Server	Server MaxConnections
In the administrative console, click Servers > Server Types > Web servers > web_server_name > Plug-in properties > Request and Response	Application server port preference	AppServerPortPreference
In the administrative console, click Servers > Web Servers > Web_server_name > Plug-in properties > Request and Response	Enable Nagle algorithm for connections to the Application Server	ASDisableNagle
In the administrative console, click Servers > Server Types > Web servers > web_server_name > Plug-in properties > Request and Response	Enable Nagle Algorithm for the IIS Web Server	IISDisableNagle

Table 7. Web server plug-in configuration properties (continued)

In the administrative console, click Servers > Server Types > Web servers > web_server_name > Plug-in properties > Request and Response	Virtual host matching	VHostMatchingCompat
In the administrative console, click Servers > Server Types > Web servers > web_server_name > Plug-in properties > Request and Response	Maximum chunk size used when reading the response body	ResponseChunkSize
In the administrative console, click Servers > Server Types > Web servers > web_server_name > Plug-in properties > Request and Response	Accept content for all requests	AcceptAllContent
In the administrative console, click Servers > Server Types > Web servers > web_server_name > Plug-in properties > Request and Response	Chunk HTTP response to the client	ChunkedResponse
In the administrative console, click Servers > Server Types > Web servers > web_server_name > Plug-in properties > Request and Response	Priority used by the IIS Web server when loading the plug-in configuration file	IISPluginPriority
In the administrative console, click Servers > Server Types > Web servers > web_server_name > Plug-in properties > Caching	Enable Edge Side Include (ESI) processing to cache the responses	ESIEnable
In the administrative console, click Servers > Server Types > Web servers > web_server_name > Plug-in properties > Caching	Maximum cache size	ESIMaxCacheSize
In the administrative console, click Servers > Server Types > Web servers > web_server_name > Plug-in properties > Caching	Enable invalidation monitor to receive notifications	ESIInvalidationMonitor

Web server plug-in connections

The WebSphere Application Server Web server plug-ins are used to establish and maintain persistent HTTP and HTTPS connections to Application Servers .

When the plug-in is ready to send a request to the application server, it first checks its connection pool for existing connections. If an existing connection is available the plug-in checks its connection status. If the status is still good, the plug-in uses that connection to send the request. If a connection does not exist, the plug-in creates one. If a connection exists but has been closed by the application server, the plug-in closes that connection and opens a new one.

After a connection is established between a plug-in and an application server, it will not be closed unless the application server closes it for one of the following reasons:

- If the **Use Keep-Alive** property is selected and the time limit specified on the **Read timeout** or **Write timeout** property for the HTTP inbound channel has expired.

- The maximum number of persistent requests which can be processed on an HTTP inbound channel has been exceeded. This number is set using the **Maximum persistent requests** property that is specified for the HTTP inbound channel.
- The Application Server is shutting down.

Even if the application server closes a connection, the plug-in will not know that it has been closed until it tries to use it again. The connection will be closed if one of the following events occur:

- The plug-in receives a new HTTP request and tries to reuse the existing connection.
- The number of httpd processes drop because the Web server is not receiving any new HTTP requests. For the IBM HTTP Server, the number of httpd processes that are kept alive depends on the value specified on the Web server's MinSpareServers directive.
- The Web server is stopped and all httpd processes are terminated, and their corresponding sockets are closed.

Note: Sometimes, if a heavy request load is stopped or decreased abruptly on a particular application server, a lot of the plug-in's connections to that application server will be in CLOSE_WAIT state. Because these connections will be closed the first time the plug-in tries to reuse them, having a large number of connections in CLOSE-WAIT state should not affect performance

Web server plug-in remote user information processing

You can configure your Web server with a third-party authentication module and then configure the Web server plug-in to route requests to an application server.

If an application calls the getRemoteUser() method, it relies on a private HTTP header that contains the remote user information and is parsed by the plug-in. The plug-in sets the private HTTP header value whenever a Web server authentication module populates the remote user in the Web server data structure. If the private HTTP header value is not set, the application's call to getRemoteUser() returns a null value.

- In the case of an Apache Web server or the IBM HTTP Server, the plug-in builds the private header from the information contained in the associated request record.
- In the case of a Sun One Web server, the plug-in builds the private header from the information contained in the **auth_user** property associated with the request. The private header is usually set to the name of the local HTTP user of the Web browser, if HTTP access authorization is activated for the URL.
- In the case of a Domino Web server, the plug-in builds the private header from the information contained in the **REMOTE_USER** environment variable. The plug-in sets this variable to **anonymous** for users who have not logged in and to the *username* for users who are logged into the application.
- In the case of an Internet Information Services (IIS) Web server, the plug-in builds the private header from the information contained in the **REMOTE_USER** environment variable. The plug-in sets this variable to the name of the user as it is derived from the authorization header sent by the client.

Note: If the private header is not being set in the Sun One, IIS, or Domino Web server plug-in, make sure the request record includes information about the user requesting the data.

Note: If an application's call to getRemoteUser() returns a null value, or if the correct remote user information is not being added to the Web server plug-in's data structure, make sure the remote user parameter within the WebAgent is still set to **YES**. (Sometimes this parameter gets set to **NO** when service is applied.)

Web server plug-ins

Web server plug-ins enable the Web server to communicate requests for dynamic content, such as servlets, to the application server. A Web server plug-in is associated with each Web server definition. The configuration file (plugin-cfg.xml) that is generated for each plug-in is based on the applications that are routed through the associated Web server.

A Web server plug-in is used to forward HTTP requests from a supported Web server to an application server. Using a Web server plug-in to provide communication between a Web server and an application server has the following advantages:

- XML-based configuration file
- Standard protocol recognized by firewall products
- Security using HTTPS, replacing proprietary Open Servlet Engine (OSE) over Secure Sockets Layer (SSL)

Each of the supported Web server plug-ins runs on a number of operating systems. See the Supported Hardware and Software Web site for the product for the most current information about supported Web servers. This site is located at <http://www.ibm.com/support/docview.wss?rs=180&uid=swg27006921>.

Checking your IBM HTTP Server version

At times, you might need to determine the version of your IBM HTTP Server installation.

About this task

The following information describes how you can determine the IBM HTTP Server version and provides examples.

Note: You can also determine the IBM HTTP Server version using the versionInfo command.

1. Change the directory to the installation root of the Web server. For example, this is /opt/IBM/HTTPServer on a Solaris machine.
2. Find the subdirectory that contains the executable. The executable for IBM HTTP Server is:
 - **Windows** httpd.exe (the previous command, apache.exe is deprecated)
 - **Linux** httpd
 - **AIX** **HP-UX** **Solaris** apachectl
3. Issue the command with the -v option to display the version information.

Windows
httpd.exe -v

Linux
./httpd -v

AIX **HP-UX** **Solaris**
./apachectl -v

Results

The version is shown in the "Server version" field and will look something like the following:

```
IBM_HTTP_Server/7.0.0.0 (Windows)
Server built: Jul 31 2008 08:41:58
```

or

```
Server version: IBM_HTTP_Server/7.0.0.0 (Unix)
Server built: Jul 31 2008 08:41:58
```

Creating or updating a global Web server plug-in configuration file

If all of the application servers in a cell use the same Web server to route requests for dynamic content, such as servlets, from Web applications to application servers, you can create a global Web server plug-in configuration file for that cell. The resulting plugin-cfg.xml file is located in the %was_profile_home%/config/cells directory.

About this task

You must update the global Web server plug-in configuration file whenever you:

- Change the configuration settings for an application server, cluster, virtual host or Web container transport that is part of that cell.
- Add a new application server, cluster, virtual host or Web container transport to that cell.

To update the configuration settings for a global Web server plug-in, you can either use the Update global Web server plug-in configuration page in the administrative console, or issue the following command:

```
%was_profile_home%/config/cells/GenPluginCfg.sh|bat
```

Both methods for regenerating the global Web server plug-in configuration create a plugin-cfg.xml file in ASCII format.

To use the Update global Web server plug-in configuration page in the administrative console:

1. Click **Environment > Update global Web server plug-in configuration**.
2. Click **OK** to update the plugin-cfg.xml file.
- 3.
4. Click **View or download the current Web server plug-in configuration file** if you want to view or download the current version of this file. You can select this option if you want to:
 - View the current version of the file before you update it.
 - View the file after it is updated.
 - Download a copy of this file to a remote machine.

Results

Regenerating the configuration might take a while to complete. After it finishes, all objects in the administrative cell use their newest settings, which the Web server can access. Whether triggered manually or occurring automatically, plug-in regeneration requires about 30 to 60 seconds to complete when the Application Server is on the same physical machine (node) as the Web server. In other cases, it takes more time.

The delay is important because it determines how soon the new plug-in configuration takes effect. Suppose you add a new served path for a servlet, then regenerate the plug-in configurations. The regeneration requires 40 seconds, after which a user should be able to access the servlet by the new served path.

For an HTTP plug-in, the length of the delay is determined by the Refresh Interval attribute of the Config element in the plugin-cfg.xml file. The plug-in polls the disk, or file system, at this interval to see whether the configuration has changed. The default interval is 60 seconds. To regenerate the plug-in configuration requires twice the refresh interval.

In a development environment in which you are frequently changing settings in the administrative console, it is recommended that you set the refresh interval to 3 to 5 seconds.

In a production environment, set a longer refresh interval, perhaps as long as 30 minutes, depending on the frequency of changes.

What to do next

You might need to stop the application servers in the cell and then start the application servers again before the changes to the plug-in configuration go into effect.

If the Web server is running on a remote machine, click **View or download the current Web server plug-in configuration file** to download a copy of the plugin-cfg.xml file to a that machine.

When the deployment manager is installed on a machine that is remote from where the product is installed, one of the following solutions must be implemented in order for the plugin-cfg.xml file to retain the application server directory structures, and not assume those of the deployment manager after the plug-in is regenerated and a full synchronization occurs. The plugin-cfg.xml file is located in the application server /config/cells directory.

- **Command line:**

At a command prompt, enter the following command to change the DeploymentManager/bin directory and type on the machine where the deployment manager is installed. This command creates or updates the plugin-cfg.xml file, and changes all of the directories in the plugin-cfg.xml file to *app_server_root* directories.

```
GenPluginCfg -destination.root <app_server_root>
```

For example, issue the following command from the DeploymentManager/bin directory.

```
GenPluginCfg -destination.root "E:\WebSphere\AppServer"
```

- **plugin-cfg.xml file:**

Edit the plugin-cfg.xml file, located in the *app_server_root/DeploymentManager/config/cells* directory, to point to the correct directory structure for the log file, keyring, and stashfile.

Perform a full synchronization so the plugin-cfg.xml file is replicated in all the nodes. You can use scripting or the administrative console to synchronize the nodes in the cell.

The deployment manager plugin-cfg.xml file can point to the application server directories without any conflict.

Update the global Web server plug-in configuration setting

Use this page to create or update a global plug-in configuration file. The configuration settings this file contains are based on the topology of the cell that contains the applications servers that use this Web server plug-in. The Web server plug-in configuration file settings determine whether an application server or the Web server handles user requests.

A global Web server plug-in configuration file must be regenerated whenever:

- You change the configuration settings for an application server, cluster, Web container transport, or virtual host alias that is contained in the cell.
- You add a new application server, cluster, Web container transport, or virtual host alias to the cell.

The generated plugin-cfg.xml file is placed in the %was_profile_home%/config/cells directory. If your Web server is located on a remote machine, you must manually move this file to that machine.

To view this administrative console page, click **Environment > Update global Web server plug-in configuration**.

Click **OK** to update the global plugin-cfg.xml file.

Click **View or download the current Web server plug-in configuration file** if you want to:

- View the current version of the file before you update it.

- View the file after it is updated.
- Download a copy of this file to a remote machine.

Gskit install images files

The Global Security Kit (GSKit) installation image files for the WebSphere Web server plug-ins are packaged on the CD with the Web server plug-in files.

You can download the appropriate GSKIT file to the workstation on which your Web server is running. Use the following table to assist you in selecting the correct GSKIT installation image file.

Operating system	GSKit 7 Installation image file
Microsoft Windows	No image name
AIX	gskta.rte
HP-UX	gsk7bas
Solaris Operating Environment	gsk7bas
Linux	gsk7bas_7.0.3.1.i386.rpm
Linux390	gsk7bas-7.0.3.1.s390.rpm
LinuxPPC	gsk7bas-7.0.3.1.ppc.rpm

Plug-ins: Resources for learning

Use the following links to find relevant supplemental information about Web server plug-ins. The information resides on IBM and non-IBM Internet sites, whose sponsors control the technical accuracy of the information.

These links are provided for convenience. Often, the information is not specific to the IBM WebSphere Application Server product, but is useful all or in part for understanding the product. When possible, links are provided to technical papers and Redbooks that supplement the broad coverage of the release documentation with in-depth examinations of particular product areas.

View links to additional information about:

- Programming model and decisions
- Programming instructions and examples

Programming model and decisions

- Best Practice: WebSphere Plug-in Configuration Regeneration at http://www-128.ibm.com/developerworks/websphere/library/bestpractices/plug_in_configuration_regeneration.html

Programming instructions and examples

- IBM HTTP Server documentation at <http://www-3.ibm.com/software/webservers/httpservers/library.html>
- WebSphere Application Server education at <http://www-128.ibm.com/developerworks/search/searchResults.jsp?searchType=1&searchSite=dW&searchScope=dW&query=WebSphere+education>
- Listing of all IBM WebSphere Application Server Redbooks at <http://publib-b.boulder.ibm.com/Redbooks.nsf/Portals/WebSphere>

Web server plug-in tuning tips

Important tips for Web server plug-in tuning include how to balance workload and improve performance in a high stress environment. Balancing workloads among application servers in a network fronted by a Web server plug-in helps improve request response time.

Balancing workloads

During normal operation, the backlog of connections pending to an application server is bound to grow. Therefore, balancing workloads among application servers in a network fronted by a Web server plug-in helps improve request response time.

You can limit the number of connections that can be handled by an applications server. To do this:

1. Go to the **Servers > Server Types > WebSphere application servers > *server_name***.
2. In the Additional Properties section, click **Web Server Plug-in properties** .
3. Select **Set limit** for the Minimum number of connections that can be handled by the Application Server field.
4. Specify in the Connections field the maximum number of connections you want to allow.
5. Then click **Apply** and **Save**.

When this maximum number of connections is reached, the plug-in, when establishing connections, automatically skips that application server, and tries the next available application server. If no application servers are available, an HTTP 503 response code will be returned to the client. This code indicates that the server is currently unable to handle the request because it is experiencing a temporary overloading or because maintenance is being performed.

The capacity of the application servers in the network determines the value you specify for the maximum number of connections. The ideal scenario is for all of the application servers in the network to be optimally utilized. For example, if you have the following environment:

- There are 10 application servers in a cluster.
- All of these application servers host the same applications (that is, Application_1 and Application_2).
- This cluster of application servers is fronted by five IBM HTTP Servers.
- The IBM HTTP Servers get requests through a load balancer.
- Application_1 takes approximately 60 seconds to respond to a request
- Application_2 takes approximately 1 second to respond to a request.

Depending on the request arrival pattern, all requests to Application_1 might be forwarded to two of the application servers, say Appsvr_1 and Appsvr_2. If the arrival rate is faster than the processing rate, the number of pending requests to Appsvr_1 and Appsvr_2 can grow.

Eventually, Appsvr_1 and Appsvr_2 are busy and are not able to respond to future requests. It usually takes a long time to recover from this overloaded situation.

If you want to maintain 2500 connections, and optimally utilize the Application Servers in this example, set the number of maximum connections allowed to 50. (This value is arrived at by dividing the number of connections by the result of multiplying the number of Application Servers by the number of Web servers; in this example, $2500/(10 \times 5) = 50$.)

Limiting the number of connections that can be established with an application server works best for Web servers that follow use a single, multithreaded process for serving requests.

Windows IBM HTTP Server uses a single, multithreaded process for serving requests. No configuration changes are required.

UNIX IBM HTTP Server typically uses multiple multithreaded processes for serving requests. Specify the following values for the properties in the Web server configuration file (httpd.conf) to prevent the IBM HTTP Server from using more than one process for serving requests.

ServerLimit	1
ThreadLimit	1024
StartServers	1
MaxClients	1024
MinSpareThreads	1
MaxSpareThreads	1024
ThreadsPerChild	1024
MaxRequestsPerChild	0

Improving performance in a high stress environment

Windows If you use the default settings for a Microsoft Windows operating system, you might encounter Web server plug-in performance problems if you are running in a high stress environment. To avoid these problems, consider tuning the TCP/IP setting for this operating system. Two of the keys setting to tune are TcpTimedWaitDelay and MaxUserPort.

To tune the TcpTimedWaitDelay setting, change the value of the tcp_time_wait_interval parameter from the default value of 240 seconds, to 30 seconds:

1. Locate in the Windows Registry:

```
HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\tcpip\Parameters\TcpTimedWaitDelay
```

If this entry does not exist in your Windows Registry, create it by editing this entry as a new DWORD item.

2. Specify, in seconds, a value between 30 and 300 inclusive for this entry. (It is recommended that you specify a value of 30.)

To tune the MaxUserPort setting:

1. Locate in the Windows Registry:

```
HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\tcpip\Parameters\MaxUserPort
```

If this entry does not exist in your Windows Registry, create it by editing this entry as a new DWORD item.

2. Set the maximum number of ports to a value between 5000 and 65534 ports, inclusive. (It is recommended that you specify a value of 65534,)

See the Microsoft Web site for more information about these settings.

Private headers

A Web server plug-in can use private headers to forward requests for dynamic content, such as servlets, to the application server.

After you configure a Web server plug-in, in addition to regular plug-in functions, you can use private headers as a mechanism for forwarding proxy information from the plug-in to an application server. This information is not normally included in HTTP requests.

Private headers are implemented as a set of HTTP request header name and value pairs that the plug-in adds to the HTTP request header before the request is forwarded to an application server. The application server's Web container removes this information from the header and then processes this information.

Private headers can include such information as the remote (client) user, the remote (client) host name, or an SSL client certificate. They conform to a naming standard so that there is no namespace collision with the architected HTTP header fields.

For example, authentication information, such as a client certificate, is normally requested by the Web server once during the establishment of an HTTP session. It is not required again for individual requests

within that session. However, a client certificate must accompany each request forwarded to the application server. The application server can then use the certificate as needed.

Similarly, the Web server examines TCP/IP socket connections for information about the host address of the client. The application server cannot perform this examination because its socket connection is with the plug-in and not with the actual client. Therefore, one of the private headers is the host address of the actual client.

plugin-cfg.xml file

The plugin-cfg.xml file includes the following elements and attributes. Unless indicated otherwise, each element and attribute can only be specified once within the plugin-cfg.xml file.

Note: Use the administrative console to set these properties for a given Web server definition. Any manual changes you make to the plug-in configuration file for a given Web server are overridden whenever the file is regenerated.

Config (required)

This element starts the WebSphere HTTP plug-in configuration file. It can include one or more of the following elements and attributes.

IgnoreDNSFailures

Specifies whether the plug-in ignores DNS failures within a configuration when starting. When set to true, the plug-in ignores DNS failures within a configuration and starts successfully if at least one server in each ServerCluster is able to resolve the host name. Any server for which the host name can not be resolved is marked *unavailable* for the life of the configuration. No attempts to resolve the host name are made later on during the routing of requests. If a DNS failure occurs, a log message is written to the plug-in log file and the plug-in initialization continues rather than causing the Web server not to start. The default value is false, meaning DNS failures cause the Web server not to start.

RefreshInterval

The time interval (in seconds) at which the plug-in should check the configuration file to see if updates or changes have occurred. The plug-in checks the file for any modifications that have occurred since the last time the plug-in configuration was loaded.

In a development environment in which changes are frequent, a lower setting than the default setting of 60 is preferable. In production, a higher value than the default is preferable because updates to the configuration will not occur so often. If the plug-in reload fails for some reason, a message is written to the plug-in log file and the previous configuration is used until the plug-in configuration file successfully reloads. If you are not seeing the changes you made to your plug-in configuration, check the plug-in log file for indications of the problem.

ASDisableNagle

Specifies whether the user wants to disable nagle algorithm for the connection between the plug-in and the application server. By default, nagle algorithm is enabled.

The value can be true or false.

IISDisableNagle

Specifies whether the user wants to disable nagle algorithm on Microsoft Internet Informations Services (IIS). By default, nagle algorithm is enabled.

The value can be true or false.

AppServerPortPreference

This attribute is used to specify which port number the Application Server should use to build URI's for a sendRedirect. The following values can be specified:

- hostHeader if the port number from the host header of the HTTP request coming in is to be used.
- webserverPort if the port number on which the Web server received the request is to be used.

The default is hostHeader.

ResponseChunkSize

The plug-in reads the response body in 64k chunks until all of the response data is read. This approach causes a performance problem for requests whose response body contains large amounts of data.

The ResponseChunkSize attribute lets you specify the maximum chunk size to use when reading the response body. For example, Config ResponseChunkSize="N">, where N equals the chunk size in kilobytes.

If the content length of the response body is unknown, a buffer size of N kilobytes is allocated and the body is read in N kilobyte size chunks, until the entire body is read. If the content length is known, then a buffer size of either content length or N (whichever is less) is used to read the response body.

The default chunk size is 64k.

AcceptAllContent

Specifies whether or not users can include content in POST, PUT, GET, and HEAD requests when a Content-Length or Transfer-encoding header is contained in the request header. You can specify one of the following values for this attribute:

- True if content is to be expected and read for all requests
- False if content only is only to be expected and read for POST and PUT requests.

False is the default.

ChunkedResponse

Specifies whether the plug-in should chunk the response to the client when a Transfer-Encoding : Chunked response header is present in the response.

This attribute only applies to the IIS, IPlanet, and Domino Web servers. The IBM HTTP Server automatically handles the chunking of the response to the client.

You can specify one of the following values for this attribute:

- true if the plug-in is to chunk the response to the client when a Transfer-Encoding : Chunked response header is present in the response.
- false if the response is not to be chunked.

false is the default.

IISPluginPriority

Specifies the priority in which the IIS Web server loads the WebSphere Web server plug-in. You can specify one of the following values for this attribute:

- High
- Medium
- Low

The default value is High.

NOTES:

- The IIS Web server uses this value during startup. Therefore, the Web server must be restarted before this change will take effect.
- The default value of High ensures that all requests are handled by the WebSphere Web server plug-in before they are handled by any other filter/extensions. If problems occur while using a priority of Medium or Low, you will have to rearrange the order or change the priority of the interfering filter/extension.

Log

The log describes the location and level of log messages that are written by the plug-in. If a log is not specified within the configuration file, then, in some cases, log messages are written to the Web server error log.

For example, you might specify the following:

```
<Log LogLevel="Error" Name="/opt/WebSphere/AppServer60/logs/http_plugin.log"/>
```

Name (exactly one attribute for each Log)

The fully qualified path to the log file to which the plug-in will write error messages.

If the file does not exist then it will be created. If the file already exists then it will be opened in append mode and the previous plug-in log messages will remain.

LogLevel (zero or one attribute for each Log)

The level of detail of the log messages that the plug-in should write to the log. You can specify one of the following values for this attribute:

- Trace. All of the steps in the request process are logged in detail.
- Stats. The server selected for each request and other load balancing information relating to request handling is logged.
- Warn. All warning and error messages resulting from abnormal request processing are logged.
- Error. Only error messages resulting from abnormal request processing are logged.
- Debug. All of the critical steps performed in processing requests are logged.
- Detail. All of the information about requests and responses are logged.

If a LogLevel is not specified for the Log element, the default value Error is used.

Be careful when setting the level to Trace. A lot of messages are logged at this level which can cause the disk space to fill up very quickly. A Trace setting should never be used in a normally functioning environment as it adversely affects performance.

Property Name="esiEnable" Value="true/false"

Used to enable or disable the Edge Side Include (ESI) processor. If the ESI processor is disabled, the other ESI elements in this file are ignored.

Value can be set to true or false. By default, the ESI processor is enabled (set to true).

Property Name="esiMaxCacheSize" Value="interger"

An integer specifying, in 1K byte units, the maximum size of the cache. The default maximum size of the cache is 1024K bytes (1 megabyte). If the cache is full, the first entry to be evicted from the cache is the entry that is closest its expiration time.

Property Name="ESIInvalidationMonitor" Value="true/false"

Used to indicate whether or not the ESI processor should receive invalidations from the Application Server.

Value can be set to true or false. By default, this property is set to false.

Property Name="FIPSEnable" Value="true/false"

Used to indicate whether or not the Federal Information Processing Standard (FIPS) is enabled for making secure (SSL) connections to the Application Server. This property should be set to true, if FIPS is enabled on the Application Server..

Value can be set to true or false. By default, this property is set to false.

ServerCluster (one or more elements for each Config)

A group of servers that are generally configured to service the same types of requests.

In the simplest case, the cluster contains only one server definition. In the case in which more than one server is defined, the plug-in will load balance across the defined servers using either a Round Robin or a Random algorithm. The default is Round Robin.

Following is an example of a ServerCluster element

```
<ServerCluster Name="Servers">
<ClusterAddress Name="ClusterAddr">
<Transport Hostname="192.168.1.2" Port="9080" Protocol="HTTP"/>
<Transport Hostname="192.168.1.2" Port="9443" Protocol="HTTPS">
<Property Name="Keyring" value="c:/WebSphere/AppServer/keys/keyring.kdb"/>
<Property Name="Stashfile" value="c:/WebSphere/AppServer/keys/keyring.sth"/>
</Transport>
</ClusterAddress>
<Server Name="Server1">
<Transport Hostname="192.168.1.3" Port="9080" Protocol="HTTP"/>
<Transport Hostname="192.168.1.3" Port="9443" Protocol="HTTPS">
<Property Name="Keyring" value="c:/WebSphere/AppServer/keys/keyring.kdb"/>
<Property Name="Stashfile" value="c:/WebSphere/AppServer/keys/keyring.sth"/>
</Transport>
</Server>
```



```

<Server Name="Server2">
<Transport Hostname="192.168.1.4" Port="9080" Protocol="HTTP"/>
<Transport Hostname="192.168.1.4" Port="9443" Protocol="HTTPS">
<Property Name="Keyring" value="c:/WebSphere/AppServer/keys/keyring.kdb"/>
<Property Name="Stashfile" value="c:/WebSphere/AppServer/keys/keyring.sth"/>
</Transport>
</Server>
<Server Name="Server3">
<Transport Hostname="192.168.1.5" Port="9080" Protocol="HTTP"/>
<Transport Hostname="192.168.1.5" Port="9443" Protocol="HTTPS">
<Property Name="Keyring" value="c:/WebSphere/AppServer/keys/keyring.kdb"/>
<Property Name="Stashfile" value="c:/WebSphere/AppServer/keys/keyring.sth"/>
</Transport>
</Server>
<PrimaryServers>
<Server Name="Server1"/>
<Server Name="Server2"/>
</PrimaryServers>
<BackupServers>
<Server Name="Server3"/>
</BackupServers>
</ServerCluster>

```

Name (exactly one attribute for each ServerCluster)

The logical or administrative name to be used for this group of servers.

LoadBalance (zero or one attribute for each ServerCluster)

The default load balancing type is Round Robin.

The Round Robin implementation has a random starting point. The first server will be picked randomly. Round Robin will be used to pick servers from that point forward. This implementation ensures that in multiple process based Web servers, all of the processes don't start up by sending the first request to the same Application Server.

IgnoreAffinityRequests (zero or one attribute for each ServerCluster)

Specifies whether the plug-in ignores the number of affinity requests made to a server when selecting servers based on the Round Robin algorithm. The value can be true or false. If the value is set to false, the number of affinity requests made is also taken into account in the server selection process.

The default value is true, which means the number of affinity requests made are not used in the Round Robin algorithm.

RetryInterval (zero or one attribute for each ServerCluster)

An integer specifying the length of time that should elapse from the time that a server is marked down to the time that the plug-in will retry a connection. The default is 60 seconds.

RemoveSpecialHeaders (zero or one attribute for each ServerCluster)

The plug-in adds special headers to the request before it is forwarded to the application server. These headers store information about the request that will need to be used by the application. By default the plug-in will remove these headers from incoming requests before adding the headers it is supposed to add.

The value can be true or false. Setting the attribute to false introduces a potential security exposure by not removing headers from incoming requests.

CloneSeparatorChange (zero or one attribute for each ServerCluster)

Some pervasive devices cannot handle the colon character (:) used to separate clone IDs in conjunction with session affinity. This attribute for the server group tells the plug-in to expect the plus character (+) as the clone separator. You must change application server configurations so that an application server separates clone IDs with the plus character as well.

The value can be true or false.

PostSizeLimit (zero or one attribute for each ServerCluster)

The maximum number of bytes of request content allowed in order for the plug-in to

attempt to send the request to an application server. If a request is received that is greater than this size, the plug-in fails the request. The default value is -1 bytes, which indicates that there is no limit for the post size.

Server (one or more elements for each ServerCluster)

A WebSphere Application Server instance that is configured to handle requests routed to it given the routing rules of the plug-in configuration. The Server should correspond to an application server running on either the local machine or a remote machine.

Name (exactly one attribute for each Server)

The administrative or logical name for the server.

CloneID (zero or one attribute for each Server)

If this unique ID is present in the HTTP cookie header of a request (or the URL if using URL rewriting), the plug-in routes the request to this particular server, provided all other routing rules are met. If a CloneID is not specified in the Server, then session affinity is not enabled for this server.

This attribute is used in conjunction with session affinity. When this attribute is set, the plug-in checks the incoming cookie header or URL for **JSESSIONID**. If **JSESSIONID** is found then the plug-in looks for one or more clone IDs. If clone IDs are found, and a match is made to the value specified for this attribute, then the request is sent to this server rather than load balanced across the cluster.

If you are not using session affinity then it is best to remove these clone IDs from the configuration because there is added request processing in the plug-in when these are set. If clone IDs are not in the plug-in then it is assumed that session affinity is not on and the request is load balanced across the cluster.

WaitForContinue (zero or one attribute for each Server)

Specifies whether to use the HTTP 1.1 100 Continue support before sending the request content to the application server. Possible attribute values are true or false. The default value is false; the plug-in does not wait for the 100 Continue response from the application server before sending the request content because it is a performance hit.

This property will be ignored for POST requests in order to prevent a failure from occurring if the Application server closes a connection because of a keep alive time-out.

Enable this function (set to true) when configuring the plug-in to work with certain types of proxy firewalls.

LoadBalanceWeight (zero or one attribute for each Server)

Specifies the weight associated with this server when the plug-in does weighted Round Robin load balancing. The starting value for a server can be any integer between 0 and 20. However, zero should be specified only for a server that is shut down.

The LoadBalanceWeight for each server is decremented for each request processed by that server. After the weight for a particular server in a server cluster reaches zero, only requests with session affinity are routed to that server. When all servers in the cluster reach a weight of zero, the weights for all servers in the cluster are reset, and the algorithm starts over.

When a server is shut down, it is recommended that you set the weight for that server to zero. The plug-in can then reset the weights of the servers that are still running, and maintain proper load balancing.

ConnectTimeout (zero or one attribute for each Server)

The ConnectTimeout attribute of a Server element enables the plug-in to perform non-blocking connections with the application server. Non-blocking connections are beneficial when the plug-in is unable to contact the destination to determine if the port is available or unavailable.

If no ConnectTimeout value is specified, the plug-in performs a blocking connect in which the plug-in sits until an operating system times out (as long as two minutes depending on the platform) and allows the plug-in to mark the server *unavailable*. A value of 0 causes the plug-in to perform a blocking connect. A value greater than 0 specifies the number of seconds you want the plug-in to wait for a successful connection. If a connection does not occur after that time interval, the plug-in marks the server *unavailable* and fails over to one of the other servers defined in the cluster. The default value is 5 seconds.

ExtendedHandshake (zero or one attribute for each Server)

The ExtendedHandshake attribute is used when a proxy firewall is between the plug-in and the application server. In such a case, the plug-in is not failing over, as expected.

The plug-in marks a server as down when the connect() fails. However, when a proxy firewall is in between the plug-in and the application server, the connect() will succeed, even though the back end application server is down. This causes the plug-in to not failover correctly to other application servers.

The plug-in performs some handshaking with the application server to ensure that it is started before sending the request. This enables the plug-in to failover in the event the application server is down.

The value can be true or false.

MaxConnections (one element for each Server)

The MaxConnections attribute is used to specify the maximum number of pending connections to an Application Server that can be flowing through a Web server process at any point in time.

For example, assuming that:

- The application server is fronted by 5 nodes that are running an IBM HTTP Server.
- Each node starts 2 processes.
- The MaxConnections attribute is set to 50.

In this example, the application server could potentially get up to 500 connections. (You take the number of nodes, 5, multiply it by the number of processes, 2, and then multiply that number by the number specified for the MaxConnections attribute, 50, for a total of 500 connections.)

By default, MaxConnections is set to -1. If this attribute is set to either zero or -1, there is no limit to the number of pending connections to the Application Servers.

Transport (one or more elements for each Server)

The transport for reading and writing requests to a particular WebSphere application server instance. The transport provides the information needed to determine the location of the application server to which the request will be sent. If the Server has multiple transports defined to use the same protocol, the first one will be used.

It is possible to configure the Server to have one non-secure transport and one that uses SSL. In this configuration, a match of the incoming request protocol will be performed to determine the appropriate transport to use to send the request to the application server.

Hostname (exactly one attribute for each Transport)

The host name or IP address of the machine on which the WebSphere application server instance is running.

Port (exactly one attribute for each Transport)

The port on which the WebSphere application server instance is listening.

Protocol (exactly one attribute for each Transport)

The protocol to use when communicating over this transport -- either HTTP or HTTPS.

Property (zero, one, or more elements for each Transport)

When the Protocol of the Transport is set to HTTPS, use this element to supply the various initialization parameters, such as password, keyring and stashfile. for example, the portion of the plugin_cfg.xml file containing these elements might look like the following:

```
<Transport Hostname="192.168.1.2" Port="9443" Protocol="HTTPS">
<Property Name="keyring" value="c:/WebSphere/AppServer/keys/keyring.kdb"/>
<Property Name="stashfile" value="c:/WebSphere/AppServer/keys/keyring.sth"/>
<Property Name="password" value="WebAS"/>
```

Note: The default password for viewing the plugin-key.kdb file using iKeyMan is WebAS.

Name (exactly one attribute for each Property)

The name of the Property being defined. Supported names recognized by the transport are keyring, stashfile, and password.

Note: password is the only name that can be specified for the WebSphere HTTP Plug-in for z/OS. keyring, and stashfile, if specified, will be ignored.

Value (exactly one attribute for each Property)

The value of the Property being defined.

ServerIOTimeout

The ServerIOTimeout attribute of a server element enables the plug-in to set a time out value, in seconds, for sending requests to and reading responses from the application server. If a value is not set for the ServerIOTimeout attribute, the plug-in, by default, uses blocked I/O to write request to and read response from the application server until the TCP connection times out. For example, if you specify:

```
<Server Name="server1" ServerIOTimeout=300>
```

In this case, if an application server stops responding to requests, the plug-in waits 300 seconds (5 minutes) before timing out the TCP connection. Setting the ServerIOTimeout attribute to a reasonable value enables the plug-in to time out the connection sooner, and transfer requests to another application server when possible.

When selecting a value for this attribute, remember that sometimes it might take a couple of minutes for an application server to process a request. Setting the value of the ServerIOTimeout attribute too low could cause the plug-in to send a false server error response to the client. The default value is 60 seconds.

ClusterAddress (zero or one element for each ServerCluster)

A ClusterAddress is like a Server element in that you can specify the same attributes and elements as for a Server element. The difference is that you can only define one of them within a ServerCluster. Use a ClusterAddress when you do not want the plug-in to perform any type of load balancing because you already have some type of load balancer in between the plug-in and the application server.

Note: If you include a ClusterAddress tag, you must include the Name attribute on that tag. The plug-in uses the name attribute to associate the cluster address with the correct host and port. If you do not specify the Name attribute, the plug-in assigns the cluster address the name that is specified for the server that is using the same host and port.

```

<ClusterAddress Name="MyClusterAddr">
<Transport Hostname="192.168.1.2" Port="9080" Protocol="HTTP"/>
<Transport Hostname="192.168.1.2" Port="9443" Protocol="HTTPS">
</ClusterAddress>

```

If a request comes in that does not have affinity established, the plug-in routes it to the cluster address, if defined. If affinity has been established, then the plug-in routes the request directly to the clone, bypassing the cluster address entirely. If no cluster address is defined for the server cluster, then the plug-in load balances across the servers in the primary servers list.

PrimaryServers (zero or one element for each server cluster)

Specifies a list of servers to which the plug-in routes requests for this cluster. If a list of primary servers is not specified, the plug-in routes requests to servers defined for the server cluster.

BackupServers (zero or one element for each server cluster)

Specifies a list of servers to which requests should be sent to if all servers specified in the primary servers list are unavailable. The plug-in does not load balance across the backup servers, but traverses the list in order until no servers are left in the list or until a request is successfully sent and a response received from an application server.

VirtualHostGroup

A group of virtual host names that will be specified in the HTTP Host header. Enables you to group virtual host definitions together that are configured to handle similar types of requests.

Following is an example of a VirtualHost Group element and associated elements and attributes

```

<VirtualHostGroup Name="Hosts">
<VirtualHost Name="www.x.com"/>
<VirtualHost Name="www.x.com:443"/>
<VirtualHost Name="*:8080"/>
<VirtualHost Name="www.x.com:*/>
<VirtualHost Name="*:*/>
</VirtualHostGroup>

```

Name (exactly one attribute for each VirtualHostGroup)

The logical or administrative name to be used for this group of virtual hosts.

VirtualHost (one or more elements for each VirtualHostGroup)

The name used for a virtual or real machine used to determine if incoming requests should be handled by WebSphere Application Server or not. Use this element to specify host names that will be in the HTTP Host header which should be seen for requests that need to be handled by the application server. You can specify specific host names and ports that incoming requests will have or specify an asterisk (*) for either the host name, port, or both.

Name (exactly one attribute for each VirtualHost)

The actual name that should be specified in the HTTP Host header in order to match successfully with this VirtualHost.

The value is a host name or IP address and port combination, separated by a colon.

You can configure the plug-in to route requests to the application server based on the incoming HTTP Host header and port for the request. The Name attribute specifies what those combinations are.

You can use a wildcard for this attribute. The only acceptable solutions are either an asterisk (*) for the host name, an asterisk for the port, or an asterisk for both. An asterisk for both means that any request will match this rule. If no port is specified in the definition the default HTTP port of 80 is assumed.

UriGroup

A group of URIs that will be specified on the HTTP request line. The same application server must be able to handle the URIs. The route will compare the incoming URI with the URIs in the group to determine if the application server will handle the request.

Following is an example of a UriGroup element and associated elements and attributes:

```

<UriGroup Name="Uris">
<Uri Name="/servlet/snoop"/>
<Uri Name="/webapp/*"/>
<Uri Name="*.jsp"/>
</UriGroup>

```

Name (exactly one attribute for each UriGroup)

The logical or administrative name for this group of URIs.

Uri (one or more elements for each UriGroup)

The virtual path to the resource that will be serviced by WebSphere Application Server. Each URI specifies the incoming URLs that need to be handled by the application server. You can use a wildcard in these definitions.

Name (exactly one attribute for each Uri)

The actual string that should be specified in the HTTP request line in order to match successfully with this URI. You can use a wildcard within the URI definition. You can specify rules such as *.jsp or /servlet/* to be handled by WebSphere Application Server. When you assemble your application, if you specify **File Serving Enabled** then only a wildcard URI is generated for the Web application, regardless of any explicit servlet mappings. If you specify **Serve servlets by classname** then a URI having <Uri Name="Web_application_URI/servlet/*"> is generated.

AffinityCookie (zero or one attribute for each Uri)

The name of the cookie the plug-in should use when trying to determine if the inbound request has session affinity. The default value is **JSESSIONID**.

See the description of the CloneID attribute for additional session affinity information.

AffinityURLIdentifier (zero or one attribute for each Uri)

The name of the identifier the plug-in should use when trying to determine if the inbound request has affinity specified in the URL to a particular clone. The default value is **jsessionid**.

See the description of the CloneID attribute for additional session affinity information.

Route A request routing rule by which the plug-in will determine if an incoming request should be handled by a WebSphere application server.

The route definition is the central element of the plug-in configuration. It specifies how the plug-in will handle requests based on certain characteristics of the request. The route definition contains the other main elements: a required ServerCluster, and either a VirtualHostGroup, UriGroup, or both.

Using the information that is defined in the VirtualHostGroup and the UriGroup for the route, the plug-in determines if the incoming request to the Web server should be sent on to the ServerCluster defined in this route.

Following is an example of this element:

```
<Route VirtualHostGroup="Hosts" UriGroup="Uris" ServerCluster="servers/>
```

VirtualHostGroup (zero or one attribute for each Route)

The group of virtual hosts that should be used in route determination. The incoming host header and server port are matched to determine if this request should be handled by the application server.

It is possible to omit this from the route definition. If it is not present then every request will match during the virtual host match portion of route determination.

UriGroup (zero or one attribute for each Route)

The group of URIs to use for determining the route. The incoming URI for the request is matched to the defined URIs in this group to determine if this request should be handled by the application server.

It is possible to omit this from the route definition. If it is not present than every request will match during the URI match portion of route determination.

ServerCluster (exactly one attribute for each Route)

The cluster to which to send request that successfully match the route.

The cluster that should be used to handle this request. If both the URI and the virtual host matching is successful for this route then the request is sent to one of the servers defined within this cluster.

RequestMetrics

This element is used to determine if request metrics is enabled, and how to filter the requests based on the Internet protocol (IP) and Uniform Resource Identifiers (URI) filters when request metrics is enabled.

Following is an example of this element:

```
<RequestMetrics armEnabled="false" loggingEnabled="true"
  rmEnabled="false" traceLevel="PERF_DEBUG">
```

armEnabled (zero or one attribute for RequestMetrics)

This attribute indicates whether the ARM 4 agent is enabled in the plug-in. When it is set to true, the ARM 4 agent will be called.

Note: For the SunOne (iPlanet) Web server the following directive must be included in the obj.conf file to enable ARM 4 support:

```
AddLog fn="as_term"
```

If this directive is not included, the arm_stop procedure will never be called.

loggingEnabled (exactly one attribute for RequestMetrics)

This attribute indicates whether request metrics logging is enabled in the plug-in. When it is set to true and the traceLevel is not set to NONE, the request response time (and other request information) is logged. When it is set to false, there is no request logging. The value of loggingEnabled depends on the value specified for the system property com.ibm.websphere.pmi.reqmetrics.loggingEnabled. When this system property is not present, loggingEnabled is set to true.

rmEnabled (exactly one attribute for RequestMetrics)

This attribute indicates whether or not the request metrics is enabled in the plug-in. When it is set to true, the plug-in request metrics will look at the filters and log the request trace record in the plug-in log file. This action is performed if a request passes the filters. When this attribute is set to false, the rest of the request metrics attributes will be ignored..

traceLevel (exactly one attribute for RequestMetrics)

When rmEnabled is true, this attribute indicates how much information is logged. When this attribute is set to NONE, no request logging is performed. When this attribute is not set to NONE, and loggingEnabled is set to true, the request response time (and other request information) is logged when the request is done.

filters (zero, one, or two attributes for RequestMetrics)

When rmEnabled is true, the filters control which requests are traced.

enable (exactly one attribute for each filter)

When enable is true, the type of filter is on and requests must pass the filter.

type (exactly one attribute for each filter)

There are two types of filters: SOURCE_IP (for example, client IP address) and URI. For the SOURCE_IP filter type, requests are filtered based on a known IP address. You can specify a mask for an IP address using the asterisk (*). If the asterisk is used, the asterisk must always be the last character of the mask, for example 127.0.0.*, 127.0.*, 127*. For performance reasons, the pattern matches character by character, until either an asterisk is found in the filter, a mismatch occurs, or the filters are found as an exact match.

For the URI filter type, requests are filtered based on the URI of the incoming HTTP request. The rules for pattern matching are the same as matching SOURCE_IP address filters.

If both URI and client IP address filters are enabled, Request Metrics requires a match for both filter types. If neither is enabled, all requests are considered a match.

filterValues (one or multiple attribute for each filter)

The filterValues show the detailed filter information.

value (exactly one attribute for each filterValue)

Specifies the filter value for the corresponding filter type. This could be either a client IP address or a URI.

enableESIToPassCookies

Specifies whether to allow forwarding of session cookies to WebSphere Application Server when processing ESI include requests. If the value is set to `true`, this custom property is enabled. If the value is set to `false`, the custom property is disabled. By default, the value is set to `false`.

GetDWLMTable

Specifies whether to allow a newly spawned plug-in process to proactively request a partition table from WebSphere Application Server before it handles any HTTP requests. This custom property is used only when memory-to-memory session management is configured. If the value is set to `true`, this custom property is enabled. If the value is set to `false`, the custom property is disabled. By default, the value is set to `false`.

Setting up a local Web server

This topic describes how to install the Web server and the Web server plug-in on the machine where you installed WebSphere Application Server.

About this task

You can define a locally-installed Web server on an unmanaged or managed node. If the Web server is defined on an unmanaged node, the administrative functions are handled through the IBM HTTP Server administration server. If the Web server is defined on a managed node, the administrative functions of the Web server are handled through the WebSphere Application Server node agent, which is beneficial.

Note: Web servers that are *not* provided with the WebSphere Application Server product do not provide an administration server. Web servers that do not provide an administration server must reside on a managed node to facilitate plug-in administration functions and generation and propagation of the `plugin-cfg.xml` file.

The following steps create a Web server definition in the default profile.

1. Install your WebSphere Application Server product.
2. Install IBM HTTP Server or another supported Web server.
3. Install the binary plug-in module using the Plug-ins installation wizard.

The Web server definition is automatically created and configured during the installation of the plug-ins.

4. Complete the setup by creating the Web server definition using the WebSphere Application Server administrative console, or run the plug-in configuration script. The creation of this object is exclusive of the Web server installation.

Select one of the following options:

- **Using the administrative console.**

Create a Web server definition on an existing application server or unmanaged node using the wizard:

- a. Click **Servers > Server Types > Web servers > New** and use the **Create new Web server entry** wizard to create the Web server definition.
- b. Select the appropriate node.
- c. Select a template. Select a system template or a user-defined template for the Web server you want to create.

- d. Enter the Web server properties:
 - Type: The Web server vendor type
 - Port: The existing Web server port (default: 80)
 - Installation path: The Web server installation path. This field is required for IBM HTTP Server only.
 - Service name (Windows operating systems): The Windows operating system service name of the Web server.
The default is IBMHTTPServer7.0.
 - Use secure protocol: Use the HTTPS protocol to communicate with the Web server. The default is HTTP.
 - Plug-in installation location: The directory path in which the plug-in is installed.

e. Confirm the creation of the new Web server and click **Finish**.

After creating the Web server, complete the following steps to verify that the plugin-key.kdb file is generated and to configure the Web server plug-in with SSL:

- a. Click **Security > SSL certificate and key management**.
- b. Under Configuration settings, click **Manage endpoint security configurations**.
- c. Under Inbound or Outbound, expand **cell_name > nodes > Web_server_node_name > servers** and click **server_name**.
- d. Under Related Items, click **Key stores and certificates**. The administrative console displays the CMSKeyStore configuration with the path to the plugin-key.kdb file.
- e. Export the default certificate from key.p12, and add it as a signer certificate to the plugin-key.kdb.

- **Running the plug-in configuration script.**

If you install the plug-in, save the plug-in configuration script to run after you create a managed node, otherwise an error occurs. Wait until the script runs successfully and creates the Web server definition on the managed node and node synchronization occurs before starting the Web server. Adding the node starts the node agent process. If the node agent is not running, start the node.

Note: If you want the Web server to handle requests for an application for multiple managed nodes, install the application on each managed node and on the Web server definition. The script already contains all of the information that you must gather when using the administrative console option.

See the *Using the administrative clients* PDF for more information.

What to do next

You can configure non-IBM HTTP Server Web servers as a remote Web server on unmanaged nodes, or as a local Web server on managed nodes. For a non-IBM HTTP Server Web server on a managed node, the following functions are supported:

- Generation of the plug-in configuration, based on WebSphere Application Server repository changes.
- Propagation of the plugin-cfg.xml file, based on using node synchronization with the WebSphere Application Server node. Node synchronization is necessary in order to propagate configuration changes to the affected node or nodes.

The plugin-cfg.xml file is propagated to the application server node repository tree from the deployment manager repository.

Note: The plugin-cfg.xml file is propagated to the application server node repository tree. This is not the default plugin-cfg.xml file installation location. Changes may have to be made to non-IBM HTTP Server Web server configuration files to update the location of the plugin-cfg.xml file that is read by the plug-in module.

For example, Internet Information Services (IIS) has a file name called plugin-cfg.loc, which is read by

the IIS plug-in modules to determine the location of the `plugin-cfg.xml` file. The `plugin-cfg.loc` file has to be updated to reflect the `plugin-cfg.xml` file location in the application server node repository. Other non-IBM HTTP Server Web servers have different methods to specify the location of the `plugin-cfg.xml` file for the plug-in module. However, in order for propagation to work, update the location to reflect the location in the application server node repository.

The following functions are not supported on a managed node for a non-IBM Web server.

- Starting and stopping the Web server.
- Viewing and editing the configuration file.
- Viewing the Web server logs.

For a non-IBM HTTP Server Web Server on an unmanaged node, you can generate plug-in configuration, based on WebSphere Application Server repository changes. The following functions are not supported on an unmanaged node for a non-IBM HTTP Server Web server:

- Starting and stopping the Web server.
- Viewing and editing the configuration file.
- Viewing the Web server logs.
- Propagation of the Web server `plugin-cfg.xml` file.

Setting up a remote Web server

You can create a Web server definition in the administrative console when the Web server and the Web server plug-in for WebSphere Application Server are on the same machine and the application server is on a different machine. This allows you to run an application server on one platform and a web server on another platform.

Before you begin

With a remote Web server installation, WebSphere Application Server can facilitate plug-in administration functions and generation and propagation of the `plugin-cfg.xml` file for IBM HTTP Server for WebSphere Application Server, but not for other Web servers.

Web servers that are not IBM HTTP Server for WebSphere Application Server must reside on the same machine as the WebSphere Application Server (as a managed node) to facilitate plug-in administration functions and generation and propagation of the `plugin-cfg.xml` file.

About this task

You can choose a remote Web server installation if you want the Web server on the outside of a firewall and WebSphere Application Server on the inside of a firewall. You can create a remote Web server on an unmanaged node. Unmanaged nodes are nodes without node agents. Because there is no WebSphere Application Server or node agent on the machine that the node represents, there is no way to administer a Web server on that unmanaged node unless the Web server is IBM HTTP Server for WebSphere Application Server. With IBM HTTP Server, there is an administration server that will facilitate administrative requests such as start and stop, view logs, and view and edit the `httpd.conf` file.

Note: The administration server is not provided with IBM HTTP Server for WebSphere Application Server which runs on z/OS® platforms. So, administration using the administrative console is not supported for IBM HTTP Server for z/OS on an unmanaged node.

The following steps will create a Web server definition in the default profile. This procedure does not apply when setting up a remote Web server for an i5/OS® Web server. For information about setting up an i5/OS Web server, see the topic entitled *Selecting a Web server topology diagram and roadmap*.

1. Install your WebSphere Application Server product.

2. Install IBM HTTP Server or another supported Web server.
3. Install the binary plug-in module using the Plug-ins installation wizard.
4. Complete the setup by creating the Web server definition. You can use the WebSphere Application Server administrative console or run the Plug-in configuration script:
 - **Using the administrative console:**
 - a. Click **System Administration > Nodes > Add Node** to create an unmanaged node in which to define a Web server in the topology.
 - b. Click **Servers > Server Types > Web servers > New** to launch the **Create new Web server entry** wizard. You will create the new Web server definition using this wizard. The wizard values are as follows:
 - 1) Select appropriate node
 - 2) Enter Web server properties:
 - **Type:** The Web server vendor type.
 - **Port:** The existing Web server port. The default is 80.
 - **Installation Path:** The Web server installation path. This field is required field for IBM HTTP Server only.
 - **WINDOWS Service Name:** The Windows operating system service name of the Web server. The default is IBMHTTPServer7.0.
 - **Use secure protocol:** Use the HTTPS protocol to communicate with the Web server. The default is HTTP.
 - **Plug-in installation location:** The directory path where the plug-in is installed.
 - 3) Enter the remote Web server properties. The properties for the IBM HTTP Server administration server follow:
 - **Port:** The administration server port. The default is 8008.
 - **User ID:** The user ID that is created using the htpasswd script.
 - **Password:** The password that corresponds to the user ID created with the htpasswd script.
 - **Use secure protocol:** Use the HTTPS protocol to communicate with the administration server. The default is HTTP.
 - 4) Select a Web server template. Select a system template or a user-defined template for the Web server you want to create.
 - 5) Confirmation of Web server creation.
 - **Run the Plug-in configuration script.**
5. **For AIX, HP-UX, Linux or Solaris operating system:** On the remote Web server, run the setupadm script. The administration server requires read and write access to configuration files and authentication files to perform Web server configuration data administration. You can find the setupadm script in the `<IHS_install_root>/bin` directory. The administration server has to execute `adminctl restart` as root to perform successful restarts of IBM HTTP Server. In addition to the Web server files, you must manually change the permissions to the targeted plug-in configuration files.

The setupadm script prompts you for the following input:

 - User ID - The user ID that you use to log on to the administration server. The script creates this user ID.
 - Group name - The administration server accesses the configuration files and authentication files through group file permissions. The script creates the specified group through this script.
 - Directory - The directory where you can find configuration files and authentication files.
 - File name - The following file groups and file permissions change:
 - Single file name
 - File name with wildcard

- All (default) - All of the files in the specific directory
- Processing - The setupadm script changes the group and file permissions of the configuration files and authentication files.

In addition to the Web server files, you must change the permissions to the targeted plug-in configuration files. See Setting permissions manually for instructions.

6. **For AIX, HP-UX, Linux, Solaris, or Windows operating system:** On the remote Web server, run the `htpasswd` script. The administration server is installed with authentication enabled and a blank `admin.passwd` password file. The administration server will not accept a connection without a valid user ID and password. This is done to protect the IBM HTTP Server configuration file from unauthorized access.

Launch the **htpasswd** utility that is shipped with the administration server. This utility creates and updates the files used to store user names and password for basic authentication. Locate **htpasswd** in the `bin` directory.

- On Windows operating systems: `htpasswd -cm <install_dir>\conf\admin.passwd [login name]`
- On AIX, HP-UX, Linux, and Solaris platforms: `./htpasswd -cm <install_dir>/conf/admin.passwd [login name]`

where `<install_dir>` is the IBM HTTP Server installation directory and `[login name]` is the user ID that you use to log into the administration server. The `[login name]` is the user ID that you entered in the user ID field for the remote Web server properties in the administrative console.

7. Start IBM HTTP Server. Refer to Starting the IBM HTTP administration server for instructions.

What to do next

For a non-IBM HTTP Server Web Server on an unmanaged node, you can generate a plug-in configuration, based on WebSphere Application server repository changes. However, the following functions are not supported on an unmanaged node for a non-IBM HTTP Server Web server:

- Starting and stopping the Web server.
- Viewing and editing the configuration file.
- Viewing the Web server logs.
- Propagation of the Web server `plugin-cfg.xml` file.

You can configure non-IBM HTTP Server Web servers as a local Web server on a managed node. For a non-IBM HTTP Server Web server on a managed node, the following functions are supported:

- Generation of the plug-in configuration, based on WebSphere Application Server repository changes.
- Propagation of the `plugin-cfg.xml` file, based on using node synchronization with the WebSphere Application Server node. Node synchronization is necessary in order to propagate configuration changes to the affected node or nodes.

Note: When WebSphere Application Server is installed using a standalone profile on one machine and IBM HTTP Server is installed on a different machine as root user using the administrative server, to ensure that propagation functions correctly, the root user must manually change the permissions of the `plugin-cfg.xml` file to the nonroot user running IBM HTTP Server from the administrative server. The username and group needed to start the administrative server are located in the `HTTPServer/config/admin.conf` file.

The `plugin-cfg.xml` file is propagated to the application server node repository tree from the deployment manager repository.

Note: The `plugin-cfg.xml` file is propagated to the application server node repository tree. This is not the default `plugin-cfg.xml` file installation location. Changes may have to be made to non-IBM HTTP Server Web server configuration files to update the location of the `plugin-cfg.xml` file that is read by the plug-in module.

For example, Internet Information Services (IIS) has a file name called `plugin-cfg.loc`, which is read by

the IIS plug-in modules to determine the location of the `plugin-cfg.xml` file. The `plugin-cfg.loc` file has to be updated to reflect the `plugin-cfg.xml` file location in the application server node repository. Other non-IBM HTTP Server Web servers have different methods to specify the location of the `plugin-cfg.xml` file for the plug-in module. However, in order for propagation to work, update the location to reflect the location in the application server node repository.

For a non-IBM HTTP Server Web server that is configured as a local Web server on a managed node, the following functions are not supported:

- Starting and stopping the Web server.
- Viewing and editing the configuration file.
- Viewing the Web server logs.

Web server definition

To administer or manage a Web server using the administrative console, you must create a Web server definition or object in the WebSphere Application Server repository.

The creation of this object is exclusive of the actual installation of a Web server. The Web server object in the WebSphere Application Server repository represents the Web server for administering and managing the Web server from the administrative console.

The Web server object contains the following Web server properties:

- installation root
- port
- configuration file paths
- log file paths

In addition to Web server properties, the Web server contains a plug-in object. The plug-in object contains properties that define the `plugin-cfg.xml` file.

The definitions of the Web server object are made using the **wsadmin** command or the administrative console. You can also define a Web server object in the WebSphere Application Server repository using the profile create script during installation, a `.jac1` script, and by using the administrative console wizard.

There are three types of WebSphere Application Server nodes upon which you can create a Web server. The type depends on the version of WebSphere Application Server, as follows:

- **Managed node.** A node that contains a node agent. This node can exist only in a deployment manager environment. The importance of defining a Web server on a managed node is that the administration and configuration of the Web server is handled through the node agent from the administrative console. Support for administration and configuration through the administrative console is limited to IBM HTTP Server only. Non-IBM HTTP Server Web servers must be on a managed node to handle plug-in administrative functions and the generation and propagation of the `plugin-cfg.xml` file.
- **Stand-alone node.** A node that does not contain a node agent. This node usually exists in a base or Express WebSphere Application Server environment. A stand-alone node can become a managed node in a deployment manager environment after the node is federated. A stand-alone node does not contain a node agent, so to administer and manage IBM HTTP Server, there must be an IBM HTTP Server administration server installed and running on the stand-alone machine that the node represents. IBM HTTP Server ships with the IBM HTTP Server administration server and is installed by default. Support for administration and configuration through the administrative console is limited to IBM HTTP Server only.
- **Unmanaged node.** A node that is not associated with a WebSphere Application Server node agent. This node cannot be federated. Typically, the unmanaged node represents a remote machine that does not have WebSphere Application Server installed. However, you can define an unmanaged node on a machine where WebSphere Application Server is installed. This node can exist in a WebSphere

Application Server – Express, base, or deployment manager environment. An unmanaged node does not contain a node agent, so to administer and manage IBM HTTP Server, an IBM HTTP Server administration server must be installed and running on the stand-alone machine that the node represents. Support for administration and configuration through the administrative console is limited to IBM HTTP Server only.

Web servers, which are not IBM HTTP Servers for WebSphere Application Server, are not fully administered from the WebSphere Application Server administrative console. The administration functions for Web servers, which are not IBM HTTP Servers for WebSphere Application Server, are:

- On managed nodes:
 - Web server status in the Web server collection panel or `serverStatus.sh`
 - Generation of the `plugin-cfg.xml`
 - Propagation of the `plugin-cfg.xml`
- On unmanaged nodes:
 - Web server status in the Web server collection panel or `serverStatus.sh`
 - Generation of the `plugin-cfg.xml`

Special consideration for IBM HTTP Server for z/OS, powered by Apache: To support remote administration and configuration of IBM HTTP Server for z/OS, a Web server type IHSZOS must be defined in the WebSphere Application Server repository.

Note: During profile creation, using the z/OS Profile Management Tool, if you select **Advanced profile creation** and **Create a Web server definition**, you will not be permitted to select the combination of Web server type IBM HTTP Server and Web server operating system of z/OS. However, you can create the Web server type for IBM HTTP Server on z/OS through the administrative console wizard, `createWebServerDefintion.jacl`, or the `wsadmin` command after the profile create, using the z/OS Profile Management Tool.

Editing the Web server type

This topic provides information on how to change the type of Web server.

About this task

If you install a Web server that is different from the one that is currently installed, you can modify the Web server type from IBM HTTP Server to a non-IBM HTTP Server and vice versa, rather than delete the Web server and create a new Web server definition. If you change the Web server type from IBM HTTP Server to non-IBM HTTP Server Web server, the administration capabilities are lost accordingly.

1. From the WebSphere Application Server administrative console, click **Servers > Server Types > Web servers**.
2. Select the server that you want to modify.
3. On the Web server configuration panel, change your Web server by selecting an option from the Type drop-down menu. If you are changing from a non-IBM HTTP Server to an IBM HTTP Server, you are also prompted for information such as IBM HTTP Server administration server port, user ID, and IBM HTTP Server administration server password.
4. Click **Apply**.

Results

You can verify your changes on the Web servers collection panel. The Web server type displays in the Web Server Type column.

Web server collection

Use this page to view configure, manage, and view information about your Web servers.

Web servers

To view this administrative console page click **Servers > Server Types > Web Servers**.

To create a new Web server, click **New** to launch the Create new Web server entry wizard. To manage an installed Web server, select the check box beside the application name in the list and click a button:

Button	Resulting Action
Generate Plug-in	When the plug-in configuration service is enabled, a plug-in configuration file is automatically generated for a Web server whenever: <ul style="list-style-type: none">• The WebSphere Application Server administrator defines new Web server.• An application is deployed to an Application Server.• An application is uninstalled.• A virtual host definition is updated and saved.
Propagate Plug-in	Choosing this action will copy the plugin-cfg.xml file from the local directory where the Application Server is installed to the remote machine. If you are using IBM HTTP Server V6 or higher for your Web server, WebSphere Application Server can automatically propagate the plug-in configuration file to remote machines provided there is a working HTTP transport mechanism to propagate the file.
New	Launches the wizard to create a new Web server entry.
Delete	Deletes one or more of the selected Web server entries.
Templates ...	Opens the Web server templates list panel. From this panel you can create a new template or delete existing templates.
Start	Starts one or more of the selected Web servers.
Stop	Stops one or more of the selected Web servers.
Terminate	Terminates one or more of the selected Web servers.

Name

Specifies a logical name for the Web server. This can be the host name of the machine, or any name you choose.

Web server type

Indicates the type of Web Server you are using.

Node

Specifies a logical name for the Web server. This can be the host name of the machine, or any name you choose.

Specifies the name of the node on which the Web server is defined. This column only applies for the WebSphere Application Server Network Deployment product.

Version

Specifies the version of the WebSphere Application Server on which the Web server is defined.

Status

Indicates whether the Web server is started, stopped, or unavailable.

If IBM HTTP Server is defined on an unmanaged node, you must start the IBM HTTP Server administration server before you can start and stop IBM HTTP Server.

If the status is unavailable, the node agent or IBM HTTP Server administration server is not running in that node, and you must start the node agent before you can start the Web server.

	Started	The Web server is running.
	Stopped	The Web server is not running.
	Unknown	Status cannot be determined. A Web server with an unknown status might, in fact, be running but has an unknown status because the application server that is running the administrative console cannot communicate with this Web server.

Web server configuration

Use this page to configure Web server properties.

Web servers

To view this administrative console page click **Servers > Server Types > Web Servers > Web_server_name**.

Web server name

Specifies a logical name for the Web server.

Type

Specifies the vendor of the Web server. The default value is IBM HTTP Server.

The options for the type of Web servers are:

- IHS
- APACHE
- IIS
- SUNJAVASYSTEM
- DOMINO

Port

The port from which to ping the status of the Web server. This field is required.

You can use the WebSphere Application Server administrative console to check if the Web server is started by sending a ping to attempt to connect to the Web server port that is defined. In most cases the port is 80. If you have a firewall between the Web servers and application servers, you will not use port 80 on the firewall between the two systems. In most cases, your port will be different, such as 9080 or 9443. You should set the alternate ports for the Web server using the WebSphere Application Server administrative console, then set that port to the Web server to listen on, in addition to the typical port 80 and 443.

Installation path

Enter the fully qualified path where the Web server is installed. This field is required if you are using IBM HTTP Server. For all other Web Servers, this field is not required. If you enable any administrative function for non-IBM HTTP Server Web servers, the installation path will be necessary.

Configuration file name

There are two ways to view or modify the contents of the configuration file:

1. Click **Edit** to view the configuration file. You will be able to make modifications from this view. This is valid for IBM HTTP Server only.
2. Click **Configuration file** under Additional properties. You will be able to make modifications from this view. This is valid for IBM HTTP Server only.

Service name - Microsoft Windows operating systems only

Specifies the Microsoft Windows operating system name for the Web server. The name is the service name and you can find it by opening the **General** properties tab of the Web server service name.

Web server log file

Use this page to view the log file for your Web server.

To view this administrative console page, click **Servers > Server Types > Web Servers > *Web_server_name* > Log file.**

Web server log file configuration

Access log file name

Any request that is made to the Web server displays in this file.

Error log file name

Any error that occurs in the Web server displays in this file.

Web server log file runtime

Access log file name

Click **View** to display the contents of this file.

Error log file name

Click **View** to display the contents of this file.

Web server custom properties

Use this page to view and manage arbitrary name-value pairs of data, where the name is a property key and the value is a string value that can be used to set internal system configuration properties.

The administrative console contains several Custom Properties pages that work similarly. To view one of these administrative pages, click a Custom Properties link.

Web servers

To view this administrative console page click **Servers > Server Types > Web Servers > *Web_server_name* > Custom properties.**

Name

Specifies the name (or key) for the property.

Value

Specifies the value paired with the specified name.

Description

Provides information about the name-value pair.

Remote Web server management

Use this page to configure a remote IBM HTTP Server Web server.

To view the administrative console page for Remote Web server management, click **Servers > Server Types > Web Servers > *Web_server_name* > Remote Web server management**.

Note: For a WebSphere Application Server Network Deployment configuration, click **System Administration > Nodes > Add node** to create a new, unmanaged node. Then click **Servers > Web Servers > New** to create a Web server using the newly-created unmanaged node.

Web servers

Port	Indicates the port to access the administration server (default is 8008).
Use SSL	Specifies if the port is secure.
User ID	Specifies a user ID in the <code><install_dir>/conf/admin.passwd</code> file. Create this with the <code>htpasswd</code> script file, located in the <code><install_dir>/bin</code> directory.
Password	Specifies a password in the <code><install_dir>/conf/admin.passwd</code> file. Create this with the <code>htpasswd</code> script file, located in the <code><install_dir>/bin</code> directory.

Web server configuration file

Use this page to view or modify the contents of the Web server configuration file in your Web browser.

Web servers

If you have made changes to the configuration file you will need to restart your Web server, in order for the changes to take effect.

Global directives

Use this page to configure the global directives for your Web server.

To view this administrative console page, click **Servers > Server Types > Web Servers > *Web_server_name* > Global directives**.

Security enabled

Specifies if security is enabled in your Web server.

Key store certificate alias

If the **Security enabled** box is checked, specify the key store certificate alias.

Server name

Specifies the hostname that the Web server uses to identify itself.

Listen ports

Specifies the port on which your Web server will listen for requests.

Document root

The directory where the Web server will serve files.

Key store name

Specifies the name you have assigned to your keystore. A button is also provided to **Manage keys and certificates**.

Target key store directory and file name

Specifies the target directory and file name of your keystore on the machine where the Web server is installed. A button is also provided to **Copy to Web server key store directory**.

SSL Version 2 timeout

Specifies the SSL Version 2 timeout.

SSL Version 3 timeout

Specifies the SSL Version 3 timeout.

Web server virtual hosts collection

Use this page create or edit virtual hosts for your Web server.

To view this administrative console page, click **Servers > Server Types > Web Servers > Web_server_name > Configuration settings > Virtual hosts**.

To create a new virtual host, click the **New** button to launch the virtual host configuration panel. To delete an existing virtual host, select the check box beside the virtual host in the list and click **Delete**.

IP address:Port

The IP address and port number of your virtual host for the specified Web server.

Server name

Specifies the name of your virtual host for the specified Web server.

Security enabled

Specifies whether or not security is enabled for your virtual host for the specified Web server. The values are **true** or **false**.

Web server virtual hosts detail

Use this page create or edit virtual hosts for your Web server.

To view this administrative console page, click **Servers > Server Types > Web Servers > Web_server_name > Configuration settings > Virtual hosts > New**.

Security enabled

Specifies whether or not security is enabled for your virtual host for the specified Web server. Check the box to enable security

IP address

The IP address of your virtual host for the specified Web server.

Port

The port number of your virtual host for the specified Web server.

Server name

Specifies the name of your virtual host for the specified Web server.

Document root

Specifies the location of the htdocs directory for your Web server.

Keystore filename

Specifies the name you have assigned to your keystore.

Keystore directory

Specifies the target directory for the key store file on the machine where your Web Server is installed.

Keystore certificate label

Specifies the keystore certificate label. The certificate label specified here is the certificate that used in secure communication for this virtual host.

Chapter 3. Using the DataPower appliance manager

The DataPower® appliance manager automatically starts if you issue a request to the DataPower appliance manager and it is not already started. You can initiate a request using the wsadmin tool, or by selecting any of the administrative console pages that enable you to view or change settings for DataPower appliances, firmware, or managed sets, or the administrative console page that is used to monitor DataPower appliance manager tasks. The appliance manager also automatically starts when the deployment manager starts if there are any DataPower appliances configured in the appliance manager.

Before you begin

The first time that you use the DataPower appliance manager, you must add at least one appliance to the appliance manager. Before adding an appliance to the appliance manager, you must verify that:

- The appliance that you are adding is at a Version 3.6.0.4 or higher firmware level. The appliance manager cannot manage an appliances that is not at a Version 3.6.0.4 or higher firmware level.
- The appliance manager can communicate with the port that is used for the XML Management interface AMP endpoint on the appliance. The appliance manager uses this port to send commands to an appliance.
- The appliance can communicate with the port that is used for the DataPowerMgr_inbound_secure endpoint on the deployment manager. The appliance uses this port to send events to the appliance manager.
- The Appliance Management Protocol (AMP) endpoint is enabled for each appliance. If the XML Management interface AMP endpoint was disabled during installation, use the DataPower WebGUI to enable the AMP endpoint.
- There is not a firewall between the deployment manager and the appliances that will be part of a managed set. If there is a firewall between the deployment manager and the appliances, that firewall might prevent the appliance manager from communicating with the appliances in a managed set.

Note: Remember that the XML management interface port, which defaults to port number 5550, is different from the Web Management Service port, which defaults to port number 9090. The DataPower appliance manager uses the XML management interface port to manage the DataPower appliances. You use the Web Management Service port to access the WebGUI on the DataPower appliance. You can use the same user ID and password to access both the XML management interface and the WebGUI.

About this task

When the DataPower appliance manager starts, it automatically starts the channel chain which causes a bind to the port that is defined for the DataPower appliance manager.

To access the DataPower appliance manager, in the administrative console, click **Servers > DataPower**, and then perform one or more of the following actions. As previously stated, if this is the first time that you are using the DataPower appliance manager, you must add at least one appliance to the appliance manager before completing any of the other actions.

If you need to shut down the DataPower appliance manager, in the administrative console, click **Servers > DataPower > Appliance manager**, and then click **Shutdown Manager**.

- Add an appliance to the DataPower appliance manager.
- Add a new firmware version to the DataPower appliance manager.
- Create a new managed set.
- Monitor the tasks that are queued for the DataPower appliance manager
- Modify appliance manager settings.

- Manage the DataPower appliance domains.
- View the managed domains history.
- Manage versions of sharable appliance settings.

What to do next

You can configure Secure Sockets Layer (SSL) for the DataPower appliance manager.

WebSphere DataPower appliance manager overview

WebSphere DataPower appliance manager provides a set of capabilities for managing sets of appliances. DataPower appliance manager can be used to manage appliances with a 3.6.0.4 or higher level of firmware.

IBM® WebSphere® DataPower SOA Appliances are purpose-built, easy-to-deploy network devices that simplify, help secure, and accelerate your XML and Web services deployments.

The first time you use DataPower appliance manager, you must add appliances, firmware versions, and managed sets. Verify that each appliance that you want to add has a 3.6.0.4 or higher level of firmware. Also verify that the Appliance Management Protocol (AMP) endpoint is enabled for each appliance. If the XML Management interface AMP endpoint was disabled during installation, use the DataPower WebGUI to enable the AMP endpoint.

Note: The DataPower WebGUI is different from the WebSphere Application Server administrative console that you use to administer the DataPower appliance manager. The DataPower WebGUI is a separate user interface on the DataPower appliance that is used to configure the appliance.

Managed sets

A managed set is a collection of appliances that share the same hardware type, model type, and feature license set. A managed set synchronizes sharable appliance settings, managed domains, and firmware across multiple appliances.

A managed set can contain one or more appliances. An appliance is not actively managed unless it is a member of a managed set. You must first add an appliance to the DataPower appliance manager, and then add the appliance to a managed set.

Sharable appliance settings

Sharable appliance settings are the global attributes for an appliance that can be shared with other appliances. These attributes are not included in the domain or properties. For example, NTP configuration and SNMP configuration are sharable appliance settings, but appliance-specific settings, such as IP address and role-based management attributes are not sharable appliance settings,

Sharable appliance settings are not managed until an appliance is added to a managed set. After you add an appliance to a managed set, any changes that you make to the sharable appliance settings, using the DataPower WebGUI or command line interface, are synchronized from the master appliance to all of the subordinate appliances in the managed set.

Master appliances

The master appliance is the appliance in the managed set that is used to synchronize sharable appliance settings and managed domains for all appliances within the managed set. Each managed set must have at least one master appliance. Each managed set might also have subordinate appliances.

All subordinate appliances are synchronized with the master appliance, and have the same sharable appliance settings and managed domains as the master appliance. You use the DataPower WebGUI or command line interface to change the sharable appliance settings, or a managed domain on a master appliance. The DataPower command line interface is a command line user interface on the DataPower appliance that can be used to configure the appliance.

Sharable appliance settings and managed domains on subordinate appliances are automatically overwritten whenever a change is made to the master appliance. If you use the DataPower WebGUI or the DataPower command line interface to change the sharable appliance settings, or a managed domain on a master appliance, the appliance manager detects the change, and propagates the changes to the remaining appliances in the managed set. Therefore, if the sharable appliance settings or a managed domain is changed on a subordinate appliance, making the sharable appliance settings or a managed domain different from what is on the master appliance, the appliance manager automatically overwrites the changes on the subordinate appliance with the sharable appliance settings, or managed domain values that are on the master appliance.

Note: Ensure that any changes that you make to the shareable appliance settings or a managed domain on a master appliance can be used for all of the appliances in the managed set.

Managed domains

DataPower supports the use of application domains to partition configuration information into self contained units that are easier to manage. Because an application domain consists of resources that are configured to provide and support one or more services, you can use domains to group configuration information on a appliance. For example, you might set up a domain for a set of business applications because you want to keep their DataPower appliance configuration separate from the DataPower appliance configuration for the other applications on that appliance.

A managed domain is a domain on the master appliance that has been added to a managed set in the DataPower appliance manager. The DataPower appliance manager uses the managed domain to synchronize configuration changes to the subordinate appliances that are part of the managed set.

Both master appliances and subordinate appliances can also have unmanaged domains. The DataPower appliance manager does not make configuration changes to unmanaged domains.

Note: The DataPower appliance manager synchronizes managed domains from the master appliance to the subordinate appliances in the managed set. However, it is possible that the managed domain might not be completely functional on all of the subordinate appliances. For example, the managed domain might not be completely functional on a subordinate appliance if a service object, such as an XML firewall, in the managed domain has a listening port conflict on that subordinate appliance.

Versions of sharable appliance settings

Whenever the appliance manager detects that you have used the DataPower WebGUI or DataPower command line interface to change the sharable appliance settings for a master appliance, the appliance manager automatically creates a new version of the sharable appliance settings. This new version of the sharable appliance settings is called a settings version. The newest settings version is, by default, the active version for the managed set. This new settings version is automatically copied to all of the appliances in the managed set.

You can deploy any version of the sharable appliance settings to a managed set. Whenever you deploy a settings version, the deployed version becomes the active version until the sharable appliance settings are changed, or you deploy a different settings version. If you have more than one version of sharable appliance settings for a managed set, you can complete these tasks.

Note: Changes to sharable appliance settings only apply for appliances that are members of the same managed set. Changes are not propagated to appliances that are members of a different managed set.

- Copy a version of sharable appliance settings to another managed set. The sharable appliance settings are applied to all appliances in this other managed set.

Note: After the initial copy of the sharable appliance settings, the two managed sets are managed independently. Therefore, future changes to the sharable appliance settings in one managed set are not reflected in the other managed set.

- Delete an inactive version of sharable appliance settings. You cannot delete an active version. You can also specify the maximum number of versions that you want to keep.
- Deploy a version of the sharable appliance settings. You deploy a version of the sharable appliance settings to make a different version active. When the different version becomes the active version, that version is deployed to all of the members of the managed set.

Versions of managed domains

When you change a managed domain on a master appliance, the appliance manager automatically detects the change and creates a new version of the managed domain. The newest version of the managed domain is, by default, the active version for the managed set. This new version of the domain is automatically copied to all appliances in the managed set. You can deploy any version of a managed domain to a managed set, and that deployed version automatically becomes the active managed domain for that managed set.

When a managed domain is deleted from a master appliance, the domain is automatically recreated on the master appliance. To delete a managed domain, you must convert the managed domain to an unmanaged domain.

When you have multiple versions of a managed domain, you can perform the following tasks:

- Copy a version of the managed domain to another managed set. The domain is applied to all of the appliances in the managed set.

Note: After the initial copy of the managed domain, the two managed sets are managed independently. Therefore, future changes to the sharable appliance settings in one managed set are not reflected in the other managed set.

- Delete an inactive version of the managed domain. You cannot delete an active version. You can also specify the maximum number of versions that you want to keep.
- Deploy a version of the managed domain. You deploy a version of a managed domain to make that version the active version. The active version is then deployed to all members of the managed set.

Firmware

Firmware version files must be obtained from the IBM support Web site and are specific to appliance types, model types, and licensed features. When a firmware version is loaded to an appliance, it must be compatible with the appliance type, model type, and licensed features. DataPower appliance manager manages appliances with a 3.6.0.4 or higher level of firmware. A firmware file is typically in a scrypt2 format.

Versions of firmware

The appliance manager automatically determines the firmware version, intended model type, appliance type, and licensed features provided by libraries in the firmware. The appliance manager allows the firmware types to be deployed only to matching appliances.

A firmware version must exist in the DataPower appliance manager before that version can be deployed to appliances. If the firmware version that is running on an appliance is not in this file, a managed set that includes that appliance can only contain that single appliance, because the appliance manager cannot deploy that firmware version to any other appliance.

When you deploy a particular version of firmware, that version becomes the active version. When you have more than one version of firmware, you can perform the following tasks:

- Deploy a version firmware to the managed set. You deploy a version of firmware to roll back, or upgrade the firmware on the appliances to a specific version. Whenever a new version is deployed, that version becomes the active version for the managed set, and is deployed to all of the appliances in that managed set.

The firmware versions, that are in the DataPower appliance manager, can be used with multiple managed sets if the appliance type and model type are the same and the licensed features are compatible.

- Delete a version of firmware. You cannot delete an active version of firmware. As an alternative to deleting firmware versions, you can configure the maximum number of versions of any one object that you want to keep.

Setting up and administering a managed set

If you want to create at least one managed set, you must complete the following tasks. These tasks make it possible for the DataPower appliance manager to manage the appliances in a managed set:

- Add one or more DataPower appliances to the appliance manager.
- Create the firmware version in the appliance manager that you want used on all of the appliances in the managed set. You can have different firmware versions for different managed sets, or you can share the firmware versions between managed sets.
- Create a managed set for all of the appliances that are intended to share the same firmware version, shared appliance settings and managed domains.

After at least one managed set is created, you can complete the following tasks in any order:

- Manage versions of the firmware, sharable appliance settings, and managed domains with roll back capability.
- Monitor appliance synchronization and operation status.

You can also use the administrative console to manage long running tasks for the DataPower appliance manager, view the status of these tasks, or delete one or more of these task. However, you cannot delete a task to stop the task from being completed. The only way to interrupt a running task, or prevent the appliance manager from running a task, is to shutdown the appliance manager. Shutting down the appliance manager terminates all running and queued tasks.

Propagating sharable appliance settings and managed domains from master to non-master appliances

If there are multiple appliances in the managed set, then the changes made to the active version of the sharable appliance settings are propagated to the subordinate appliances in the managed set. Likewise, changes made to the managed domains of master appliances are propagated to the subordinate appliances in the managed set.

The appliance manager also detects when subordinate appliances are available. For example, if sharable appliance settings are changed for the master appliance, but the subordinate appliances are not available, then the master appliance and the subordinate appliances cannot be synchronized. When the subordinate appliances are available, the appliance manager detects the change in status and initiates synchronization from the master appliance to the subordinate appliances in the managed set.

Adding DataPower appliances to the DataPower appliance manager

You can use the DataPower appliance manager that is provided with the product to administer a DataPower appliance. After you add an appliance to the DataPower appliance manager, you can make it part of a managed set of appliances if you want the DataPower appliance manager to keep the shared appliance settings for this appliance synchronized with the shared appliance settings of the other appliances that are part of that managed set.

Before you begin

- Verify that the appliance that you are adding is at a Version 3.6.0.4 or higher firmware level. The appliance manager cannot manage an appliances that is not at a Version 3.6.0.4 or higher firmware level.
- Verify that the appliance manager can communicate with the port that is used for the XML Management interface AMP endpoint on the appliance. The appliance manager uses this port to send commands to an appliance.
- Verify that the appliance can communicate with the port that is used for the DataPowerMgr_inbound_secure endpoint on the deployment manager. The appliance uses this port to send events to the appliance manager.
- Verify that the Appliance Management Protocol (AMP) endpoint is enabled for each appliance. If the XML Management interface AMP endpoint was disabled during installation, use the DataPower WebGUI to enable the AMP endpoint.
- Verify that there is not a firewall between the deployment manager and the appliances that will be part of a managed set. If there is a firewall between the deployment manager and the appliances, that firewall might prevent the appliance manager from communicating with the appliances in a managed set.

About this task

You can use the administrative console to add a new DataPower appliance or to access the DataPower WebGUI. If you want to access the DataPower WebGUI, in the administrative console, click **Servers > DataPower > Appliances**, and then click **Launch**. Complete the following steps if you want to add a add a new DataPower appliance.

Note: Remember that the XML management interface port, which defaults to port number 5550, is different from the Web Management Service port, which defaults to port number 9090. The DataPower appliance manager uses the XML management interface port to manage the DataPower appliances. You use the Web Management Service port to access the WebGUI on the DataPower appliance. You can use the same user ID and password to access both the XML management interface and the WebGUI.

1. In the administrative console, click **Servers > DataPower > Appliances > New**.
2. Specify a unique name for the appliance in the **Name** field. An appliance name must be unique and cannot contain an invalid character. The name field cannot contain the characters # \$ @ \ / , ; " * ? < > | = + & % or '.
3. Specify an IP address or fully qualified host name in the **Host name** field.
4. Specify the XML management interface port that the DataPower appliance uses in the **Administrative port** field.
5. Specify the ID that you want to use to connect to the DataPower appliance in the **User ID** field.
6. Specify the password that you want to associate with the user ID in the **Password** and **Verify password** fields.
7. Click **OK** to add the new appliance.

Results

The DataPower appliance manager is managing this appliance.

What to do next

You can change appliance settings, add the appliance to a managed set, or remove the appliance from the DataPower appliance manager.

Modifying DataPower appliance settings

You can use the administrative console to modify the shared appliance settings for an appliance.

Before you begin

Verify that the DataPower appliance manager is managing this appliance. You cannot use the administrative console to change the sharable appliance settings for an appliance that has not been added to the DataPower appliance manager.

About this task

Complete the one or more of the following actions to make modifications to the settings for an appliance.

- From the administrative console, click **Servers > DataPower > Appliances > *appliance_name***.
- Specify a different IP address or fully qualified host name in the **Host name** field.
- Specify a different XML management interface port for the DataPower appliance to use in the **Administrative port** field.

Note: Remember that the XML management interface port, which defaults to port number 5550, is different from the Web Management Service port, which defaults to port number 9090. The DataPower appliance manager uses the XML management interface port to manage the DataPower appliances. You use the Web Management Service port to access the WebGUI on the DataPower appliance. You can use the same user ID and password to access both the XML management interface and the WebGUI.

- Specify a different user ID in the **User ID** field.
- Specify a new password to associated with the user ID in the **Password** and **Verify password** fields.
- Select an appliance type from the list of appliance types in the **Appliance type** field.
- In the Domains section, expand **Managed Domains**, or **Unmanaged Domains** to view a list of managed and unmanaged domains that are associated with this appliance.

If you want to change one or more managed domains to unmanaged domains, click **Servers > DataPower > Managed sets > *managedset_name***, select the domains that you want to change, and then click **Unmanage**. Similarly, if you want to change one or more unmanaged domains to managed domains, select the domains that you want to change, and then click **Manage**.

Clicking either **Manage** or **Unmanage**, creates an appliance manager task. If you want to know when this task completes, you can click **Servers > DataPower > Tasks** to check the status of this task.

- Click **OK** to save a copy of your changed settings as a new settings version.

Removing a DataPower appliance

You can use the administrative console to remove an appliance from the DataPower appliance manager.

About this task

If you no longer want the DataPower appliance manager to manage one of you appliances, complete the following steps to remove that appliance from the DataPower appliance manager.

1. In the administrative console, click **Servers > DataPower > Appliances**.

2. Verify that the appliance that you want to remove is not part of a managed set.

You cannot remove an appliance from the DataPower appliance manager if that appliance is part of a managed set. If the appliance that you want to remove has the value unmanaged in the Managed Set column, the appliance is not part of a managed set.

If the name of a managed set is specified in the Managed Set column, you must remove the appliance from the managed set before you remove it from the DataPower appliance manager.

- If the appliance that you want to remove from a managed set is the last appliance in that managed set, you must delete the managed set before you can remove the appliance from the DataPower appliance manager. When you delete the managed set, the appliance automatically becomes an unmanaged appliance and can be removed from the DataPower appliance manager.
- If the appliance that you want to remove from a managed set is not the last appliance in that managed set, complete the following actions to remove the appliance from that managed set.
 - a. Click **Managed sets**, and then click the name of the managed set that includes the appliance that you want to remove.
 - b. Click **Edit Membership**, select the appliance that you want to remove from the DataPower appliance manager, and then click **Remove**.
 - c. Click **Save** to save your changes.

3. Select the appliance that you want to remove from the DataPower appliance manager.

If you want to remove multiple appliances from the DataPower appliance manager, you can select all of the appliances that you want to remove.

4. Click **Remove**.

Appliance collection

Use this page to add, change, or remove a DataPower appliance in the DataPower appliance manager. You can also use this page to view information about the DataPower appliances that are in the appliance manager. Adding appliances to a DataPower appliance manager enables you to automatically keep synchronized the settings for all of the appliances that are in a DataPower appliance manager managed set.

To view this administrative console page, click **Servers > DataPower > Appliances**.

To view the values specified for a DataPower appliance, click the name of that appliance name in the list of available appliances. The displayed appliance settings page shows the values specified.

To initiate an operation on a DataPower appliance, select the name of the appliance that you want to manage, and click the operation that you want performed on that appliance. Click **Launch** to access the DataPower WebGUI in the default domain on the selected appliance. Pop-ups must be enabled for your browser.

Name

Specifies the symbolic name of the DataPower appliance.

An appliance name must be unique and cannot contain an invalid character. The name field cannot contain the following characters: # \$ @ \ / , ; " * ? < > | = + & % '.

Host Name

Specifies the host name or IP address of the DataPower appliance.

Managed Set

Specifies the name of the managed set that the appliance is assigned to. Appliances that are assigned to a managed set must be removed from the managed set before they can be removed from the appliance manager.

Master

Specifies the master appliance for the managed set to which this appliance is a member. All appliances in the managed set are synchronized using the settings of the master appliance.

Operational Status

Specifies the operational status of the appliance. The status is always unknown if the appliance is not part of a managed set or if the managed set has no managed domains. If the appliance is part of a managed set, the status reflects the aggregated operational status of all of the managed domains on this appliance.

This field is read-only. The following values might appear in this field:

up Indicates that all of the service objects in all the managed domains on the appliance are enabled.

partial Indicates that the service objects in all the managed domains on the appliance are a mix of enabled, disabled and unknown.

down Indicates that all of the service objects in all the managed domains on the appliance are disabled.

unknown

Indicates that the state of all of the managed domains on the appliance could not be determined. It is possible that the state has yet to be retrieved, or that communication to the appliances has been lost. An appliance always has a status of unknown if the appliance is not part of a managed set, or if the appliance is part of a managed set that has no managed domains.

Synchronization Status

Specifies whether the appliances are synchronized with other appliances in the managed set.

This field is read-only. The following values might appear in this field:

synced

Indicates that the firmware, sharable appliance settings and managed domains for the appliance are synchronized with the active firmware, sharable appliance and domain versions for the managed set.

changes pending

Indicates that at least one firmware synchronization, sharable appliance settings synchronization, or managed domains synchronization is queued for the appliance.

in progress

Indicates that the synchronization is currently being processed. The appliance has at least one firmware synchronization, settings synchronization, or managed domain synchronization active. Other firmware synchronizations, settings synchronizations, or managed domain synchronizations might also be pending for this appliance.

unknown

Indicates that the synchronization status is not known for at least one firmware synchronization, settings synchronization, or managed domain synchronization. Typically, this status means that the appliance manager has not yet contacted the appliance to get the management status. Firmware synchronizations, settings synchronizations, or managed domain synchronizations might also be pending, or in progress, for this appliance.

error

Indicates that an attempt to process a firmware synchronization, settings synchronization, or managed domain synchronization for this appliance has failed because of an error on the appliance. The appliance did respond, but responded with an error. This appliance might also have a management status of unknown, in progress or changes pending for a firmware synchronization, settings synchronization, or managed domain synchronization. See the error log for the deployment manager for more information.

unreachable

Indicates that an attempt to synchronize the appliance has failed because the appliance is not responding. This appliance might also have a management status of unknown, in progress,

changes pending or error for a firmware synchronization, settings synchronization, or managed domain synchronization. See the error log for the deployment manager for more information.

New appliance settings

Use this page to specify the settings for an appliance that you are adding to the DataPower appliance manager.

To view this administrative console page, click **Servers > DataPower > Appliances > New**.

Name

Specifies a user-defined symbolic name for a DataPower appliance.

An appliance name must be unique, and cannot contain the following invalid characters: # \$ @ \ / , ; " * ? < > | = + & % '.

Host name

Specifies an IP address or the fully qualified host name of a DataPower appliance.

Administrative port

Specifies the XML management interface port used by the appliance.

User ID

Specifies the user ID used by DataPower appliance manager to log into an appliance.

Password

Specifies the password for the user ID provided.

Verify password

Also specifies the password for the user ID provided. If you change the password value, you must respecify the new value in this field to ensure that you entered it correctly.

Appliance settings

Use this page to change the settings for a DataPower appliance. Appliances are added to the DataPower appliance manager to synchronize the firmware, shareable appliance settings and managed domains of a group of appliances called a managed set.

To view this administrative console page, click **Servers > DataPower > Appliances > *appliance_name***.

In the Domains section, you can expand **Managed domains** to view a list of the managed domains that are on the appliance, and the status of each of these domains. You can also expand **Unmanaged domains** to view the unmanaged domains that are on the appliance. Domains, by default, are not managed. Unmanaged domains are not copied to subordinate appliances in the managed set. Managed domains are copied to subordinate appliances in the managed set.

Name

Specifies a user-defined symbolic name for a DataPower appliance.

An appliance name must be unique, and cannot contain the following invalid characters: # \$ @ \ / , ; " * ? < > | = + & % '.

Host name

Specifies an IP address or the fully qualified host name of a DataPower appliance.

Administrative port

Specifies the XML management interface port used by the appliance.

User ID

Specifies the user ID used by DataPower appliance manager to login to an appliance.

Password

Specifies the password for the user ID provided.

Verify password

Also specifies the password for the user ID provided. If you change the password value, you must respecify the new value in this field to ensure that you entered it correctly.

Serial number

Specifies the serial number for the appliance. This field is read-only.

Appliance type

Specifies the appliance type, such as XA35, XS40, or XI50.

Licensed features

Specifies a comma-separated list of the licensed features that are on this appliance. All of the appliances in a managed set must have the same appliance type, model type and licensed features. Firmware has a list of the licensed features that are contained in the firmware. Any firmware that is loaded onto an appliance must have a list of licensed features that is compatible with the licensed features that are on that appliance.

Because managed sets contain appliances that all must be of the same appliance type, model type and licensed features, you cannot use a firmware with a managed set that has licensed features that are incompatible with the appliances in that managed set. If you attempt to load a firmware that has a list of features that is incompatible with the licensed features of the appliance, the firmware does not load, and an error message is issued.

Firmware level

Specifies the numeric level of the firmware that is currently on the appliance. Each firmware version in the DataPower appliance manager is for a specific appliance type, model type, list of licensed features, and level.

It is possible that a firmware version might already exist in the DataPower appliance manager that has the same appliance type, model type, list of licensed features, and level as the firmware that is currently on this appliance.

Firmware synchronization status

Specifies whether the firmware version for an appliance is synchronized with the active firmware version for the managed set. The status is unknown if the appliance is not part of a managed set. If the appliance is part of a managed set, the status indicates whether the firmware that is associated with the appliance is synchronized with the designated firmware for the managed set.

This field is read-only. The following values might appear in this field:

synched

Indicates that the firmware version for this appliance is the same as the firmware version for the managed set.

changes pending

Indicates that the firmware is being synchronized.

in progress

Indicates that the synchronization for the firmware on this appliance is currently being processed.

unknown

Indicates that the synchronization status for the firmware on this appliance is not known. Typically,

this status indicates that the appliance manager has not yet contacted the appliance to get the synchronization status, or the appliance is not a member of a managed set.

error Indicates that an attempt to synchronize the firmware that is associated with the managed set has failed because of an error on the appliance. The appliance did respond, but responded with an error. See the error log for the deployment manager for more information.

unreachable

Indicates that an attempt to synchronize the firmware on this appliance failed because the appliance is not responding. See the error log for the deployment manager for more information.

Settings synchronization status

Specifies whether the shareable appliance settings for an appliance are synchronized with the active settings version for the managed set. The status is unknown if the appliance is not part of a managed set. If the appliance is part of a managed set, the status indicates whether or not the sharable settings for the appliance are synchronized with the designated sharable appliance settings version for the managed set.

This field is read-only. The following values might appear in this field:

synched

Indicates that the sharable appliance settings for this appliance are the same as the sharable appliance settings version for the managed set.

changes pending

Indicates that the synchronization for the sharable appliance settings on this appliance is queued.

in progress

Indicates that the shareable appliance settings are being synchronized.

unknown

Indicates that the synchronization status for sharable appliance settings on this appliance is not known. Typically, this status indicates that the appliance manager has not yet contacted the appliance to get the synchronization status, or the appliance is not a member of a managed set, or the appliance is not a member of a managed set.

error Indicates that an attempt to synchronize the sharable appliance settings on this appliance failed because of an error. The appliance did respond, but responded with an error. See the error log for the deployment manager for more information.

unreachable

Indicates that an attempt to synchronize the sharable appliance settings on this appliance failed because the appliance is not responding. See the error log for the deployment manager for more information.

Model type

Specifies a numeric value for the appliance model. Some model types include 9001, 9002, and 9003. Each appliance is assigned a model type.

Synchronization status

Specifies the combined status of the firmware synchronization, the sharable settings synchronization, and the managed domains synchronization for the appliance.

This field is read-only. The following values might appear in this field:

synched

Indicates that the firmware, sharable appliance settings and managed domains for the appliance are synchronized with the active firmware, sharable appliance and domain versions for the managed set.

changes pending

Indicates that at least one firmware synchronization, sharable appliance settings synchronization, or managed domains synchronization is queued.

in progress

Indicates that the synchronization is currently being processed. The appliance has at least one firmware synchronization, settings synchronization, or managed domain synchronization active. Other firmware synchronizations, settings synchronizations, or managed domain synchronizations might also be pending for this appliance.

unknown

Indicates that the synchronization status is not known for at least one firmware synchronization, settings synchronization, or managed domain synchronization. Typically, this status means that the appliance manager has not yet contacted the appliance to get the management status. It is possible that other firmware synchronization, settings synchronization, or managed domain synchronizations are also pending or in progress for the appliance.

error Indicates that an attempt to process a firmware synchronization, settings synchronization, or managed domain synchronization for this appliance has failed because of an error. The appliance did respond, but responded with an error on the appliance. This appliance might also have a management status of unknown, in progress or changes pending for a firmware synchronization, settings synchronization, or managed domain synchronization. See the error log for the deployment manager for more information.

unreachable

Indicates that an attempt to synchronize the appliance has failed because the appliance is not responding. This appliance might also have a management status of unknown, in progress, changes pending or error for a firmware synchronization, settings synchronization, or managed domain synchronization. See the error log for the deployment manager for more information.

Name (in the Managed domains section)

Specifies the name of the managed domain.

Operational Status

Specifies the operational status of the managed domain. Each service inside of a DataPower domain on an appliance has an operational state of either enabled or disabled. The operational state of the domain in the DataPower appliance manager represents the aggregated operational status of all the service objects in the domain.

This field is read-only. The following values might appear in this field:

up Indicates that all of the service objects in the managed domain on the appliance are enabled.

partial Indicates that the service objects in the managed domain on the appliance are a mix of enabled, disabled and unknown.

down Indicates that all of the service objects in the managed domain on the appliance are disabled

unknown

Indicates that the state of the managed domain on the appliance cannot be determined. It is possible that the state has yet to be retrieved, or that communication to the appliance has been lost.

Synchronization status (in the Managed domains section)

Specifies the synchronization status of the managed domain on the appliance.

This field is read-only. The following values might appear in this field:

synched

Indicates that the managed domain on this appliance is synchronized with the domain version for the managed set.

changes pending

Indicates that the synchronization is queued.

in progress

Indicates that the synchronization is currently being processed.

unknown

Indicates that the synchronization status is not known. The managed domain synchronization state for the appliance is unknown. Typically, this status means that the appliance manager has not yet contacted the appliance to get the management status.

error Indicates that an attempt to synchronize the managed domain on the appliance has failed because of an error. The appliance did respond, but responded with an error. See the error log for the deployment manager for more information.

unreachable

Indicates that an attempt to synchronize the managed domain on the appliance has failed because the appliance is not responding. See the error log for the deployment manager for more information.

Name (in the Unmanaged domains section)

Specifies the name of any unmanaged domain on the device.

Adding new firmware versions to the DataPower appliance manager

You can use the DataPower appliance manager to add a new firmware version to the DataPower appliance manager. Appliances that the DataPower appliance manager manages must have a 3.6.0.4 or higher firmware level.

Before you begin

Download one or more firmware versions from the IBM support site.

About this task

Complete the following steps if you want to use the administrative console to add a new firmware version.

1. From the administrative console, click **Servers > DataPower > Firmware > New**.

2. Select either **Upload from local system**, or **Upload from remote system**.

If you select **Upload from local system**, the firmware version is uploaded to the DataPower appliance manager, from your local file system.

If you select **Upload from remote system**, the firmware version is uploaded to the DataPower appliance manager, from a remote file system.

3. Specify the fully qualified name of the file that you are uploading.

If you do not know the fully qualified name of the file that you are uploading, you can use the browse function to search for the correct file.

4. Optional: Add a description of this firmware version in the **Comments** field.

You might want to use this field to provide information that differentiates this firmware version from other firmware versions that are stored on the local or remote system.

After you enter your comment, click **Apply**.

5. Click **Submit**.

What to do next

After a new firmware version is added, you can use the DataPower appliance manager to modify the settings for the new firmware version, add the new firmware version to a managed set, or delete the new firmware version when it is no longer needed.

Modifying settings for firmware versions

You can use the administrative console to add new firmware versions to the DataPower appliance manager, or to delete existing firmware versions. You can also use the administrative console to view the settings for the firmware versions that are on the DataPower appliance manager.

About this task

When you add a new firmware version to the DataPower appliance manager, you are not creating a new version. You are actually uploading a file that contains an existing firmware version, from either a remote or local file system, to the DataPower appliance manager. If you need to create a new firmware version, you must use the DataPower WebGUI. To access the DataPower WebGUI, in the administrative console, click **Servers > DataPower > Appliance**, and then click **Launch**.

To use the administrative console to view the settings for the current firmware version or to add or delete a firmware version, perform one or more of the following actions.

1. To view the settings for the firmware versions on the DataPower appliance manager, click **Servers > DataPower > Firmware**.
2. To add a new firmware version to the DataPower appliance manager, click **New**. After you select the file to upload, click **Submit**.
3. To delete firmware versions from the DataPower appliance manager, select the versions to delete, and then click **Delete**.
4. To add a comment about a firmware version that is on the DataPower appliance manager, click the name of that firmware version and add the comment in the **Comment** field.

After you enter your comment, click **Apply**, and then click **Submit**.

Deleting firmware versions

You can delete a firmware version from the DataPower appliance manager, unless a managed set is using that firmware version. You cannot delete a firmware version that a managed set is using.

About this task

You can use the administrative console to delete firmware versions. Complete the following actions.

1. In the administrative console, click **Servers > DataPower > Firmware**.
2. Select the firmware versions that you want to delete.
3. Click **Delete**.

Results

The firmware version is deleted from the DataPower appliance manager.

Firmware collection

Use this page to add a new firmware version to the appliance manager, view existing firmware versions, or delete a firmware version from the appliance manager. A firmware version must exist in the appliance manager before that version can be designated as the active firmware version for a managed set.

This page lists the firmware versions that can be used for appliances managed by DataPower appliance manager.

To view this administrative console page, click **Servers > DataPower > Firmware**.

To view the settings for a specific firmware version, click that firmware version in the list of available versions. The displayed firmware settings page shows the values that are specified for that firmware version. On the settings page, you can change the settings for the selected firmware version.

To add a new firmware version, click **New**.

To delete an existing firmware version, select the version that you want to delete and click **Delete**.

Version

Specifies the numeric version, or level, of the firmware. When combined with an appliance type, model type, and licensed features, the numeric version, or level, uniquely identifies a firmware version.

Appliance Type

Specifies the type of appliance that the firmware version is used for. Some appliance types include XA35, XS40, and XI50. The appliance type is based on the purpose for using the appliance.

Model Type

Specifies a numeric value for the appliance model. Some model types include 9001, 9002, and 9003.

Licensed Features

Specifies a comma-separated list of the licensed features that are on this appliance. All of the appliances in a managed set must have the same appliance type, model type and licensed features. Firmware has a list of the licensed features that are contained in the firmware. Any firmware that is loaded onto an appliance must have a list of licensed features that is compatible with the licensed features that are on that appliance.

Because managed sets contain appliances that all must be of the same appliance type, model type and licensed features, you cannot use a firmware with a managed set that has licensed features that are incompatible with the appliances in that managed set. If you attempt to load a firmware that has a list of features that is incompatible with the licensed features of the appliance, the firmware does not load, and an error message is issued.

Release Date

Specifies the date when the firmware was released by the manufacturer.

Firmware settings

Use this page to specify firmware settings for a firmware version in the DataPower appliance manager. A firmware version is a firmware image of a specific level that is used with a specific appliance type, model type, and set of licensed features.

To view this administrative console page, click **Servers > DataPower > Firmware > *firmware_version***.

Version

Specifies the numeric version, or level, of the firmware. This field is read-only.

Release date

Specifies the date when the firmware was released by the manufacturer. This field is read-only.

Appliance type

Specifies the type of appliance that the firmware version is used for. Some appliance types include XA35, XS40, and XI50. This field is read-only.

Model type

Specifies a numeric value for the appliance model. Some model types include 9001, 9002, and 9003. This field is read-only.

Licensed features

Specifies a comma-separated list of the licensed features that are contained in this firmware. The firmware that is loaded onto an appliance must have a list of licensed features that is compatible with the licensed features of the appliance. This field is read-only.

All of the appliances that are part of a managed set must have the same appliance type, model type and licensed features. Therefore, you cannot use a firmware with a managed set if the firmware includes a list of licensed features that is not compatible with the appliances in the managed set. If you attempt to load firmware that includes a list of licensed features that is not compatible with the licensed features of the appliances in the managed set, the firmware does not load, and an error message is issued.

Comments

Specifies user-defined information about a firmware version.

Managed sets

Specifies a list of managed sets that use this firmware version. The firmware version must have the same appliance type, model type and licensed features as the appliances that are in the managed set. This field is read-only.

Managed set firmware settings

Use this page to change the firmware on a managed set. You can upload a file from a local or remote file system. The firmware for a managed set, must have the same value for appliance type, model type and licensed features as the devices in the managed set.

To view this administrative console page, click **Servers > DataPower > Managed sets > *managed_set_name* > Change firmware**.

From a version already in the DataPower appliance manager

Select this option if you want to use a firmware version that already exists in the appliance manager, and then select the version that you want to use from the list of available versions.

Upload from local system

Select this option if you want to upload a firmware version from a local file system. You can either type a fully qualified path name in the **Location** field, or you can select a file from the local file system.

Upload from remote system

Select this option if you want to upload a firmware version from a remote file system. You can either type a fully qualified path name in the **Location** field, or you can select a file from the remote file system.

Comments

Specifies user-defined information about a firmware version.

New firmware version settings

Use this page to upload a new firmware version from either a local or remote file system.

To view this administrative console page, click **Servers > DataPower > Managed sets > New**.

Upload from local system

Select this option if you want to upload a firmware version from a local file system. You can either type a fully qualified path name in the **Location** field, or you can select a file from the local file system.

Upload from remote system

Select this option if you want to upload a firmware version from a remote file system. You can either type a fully qualified path name in the **Location** field, or you can select a file from the remote file system.

Comments

Specifies user-defined information about a firmware version.

Adding a new managed set

Add a new managed set to synchronize settings for multiple appliances. A managed set is a grouping of appliances that share the same shareable appliance settings, managed domains and firmware version. Shareable appliance settings and managed domains are propagated to the subordinate appliances from the master appliance.

Before you begin

Before you begin, ensure that you add appliances and firmware versions.

About this task

Complete these steps to add a new managed set from the administrative console. You can click **Launch** to access the DataPower WebGUI to configure an appliance before you add the appliance to a managed set.

1. From the administrative console, click **Servers > DataPower > Managed sets > New**. This starts the New managed set wizard.
2. Enter a unique name for the new managed set in the **Name** field. A managed set name must be unique and cannot contain an invalid character. The name field cannot contain the characters # \$ @ \ / , ; " * ? < > | = + & % or '.
3. Choose a master appliance, and then click **Next**.
4. Select appliances to add to the managed set, and click **Next**.
5. View the summary of actions, and then click **Finish**.

What to do next

Creating a new managed set might take several minutes to complete. When a new managed set is added, the DataPower appliance manager automatically creates a version of the sharable appliance settings from the master appliance and copies it to any subordinate appliances in the managed set. The firmware version for the managed set might also be automatically deployed to all of the appliances in the managed set that do not already have this firmware version.

Modifying a managed set

You can change the members of a managed set, manually synchronize the sharable appliance settings, administer the versions of sharable appliance settings that are available for a managed set, change firmware versions, administer domains, or deploy a domain or sharable appliance settings version.

About this task

To modify a managed set, in the administrative console, click **Servers > DataPower > Managed sets > *managedset_name***, and then complete one or more of the following actions.

- To add appliances to the managed set, click **Edit Membership**, and then select the appliances that you want to add to this managed set.
- To remove appliances from the managed set, click **Edit Membership**, and then select the appliances that you want to remove from this managed set.

- To choose a different appliance as the master appliance, click **Edit Membership**, and then select a different appliance as the master appliance.
- Click **Synchronize** to manually synchronize the firmware, shared appliance settings and managed domains throughout the managed set.
The appliance manager always attempts to keep the firmware, shared appliance settings and managed domains synchronized across all of the appliances in a managed set. However, you can click **Synchronize** if you need to manually force a synchronization to occur.
- To access the DataPower WebGUI to configure an appliance or a firmware version, click **Launch**.
- To modify which firmware version is the active firmware version for the managed set, in the Firmware section, click **Change firmware**.
- To view a list of settings versions that have been used for the managed set, in the Settings section, click **Version history**. You can then click **Detail**, if you want to change the version that is being used for this managed set, or if you want to delete a version.
- To view a list of appliances assigned to the managed set, click **Appliances**.
- To view a list of all of the managed domains on the master appliance, in the Managed domains section, expand the **Managed domains** field. Within this field, you can change a domain from managed to unmanaged, or access the DataPower WebGUI to configure a domain.
 1. To change a domain from managed to unmanaged, select the domain, and then click **Unmanage**.
 2. To access the DataPower WebGUI to configure a domain, click **Launch** at the top of the Managed Domains table.
Clicking **Launch** launches the DataPower WebGUI in the managed domain on the master appliance.
- To view a list of all of the managed domains for this managed set, in the Managed domains section, expand the **Unmanaged domains on the master appliance** field. Within this field, you can change a domain from unmanaged to managed, or click **Launch** to access the DataPower WebGUI, and configure a domain.
 1. To change a domain from unmanaged to managed, select the domain, and then click **Manage**
 2. To access the DataPower WebGUI to configure a domain, click **Launch** at the top of the Managed Domains table.

What to do next

When you add appliances to or remove appliances from a managed set, perform a manual synchronization, change firmware versions, shareable settings or domain versions, an appliance manager task is created. To monitor the progress of these tasks, in the administrative console, click **Servers > DataPower > Tasks**.

Removing a managed set from the DataPower appliance manager

You can remove a managed set from the DataPower appliance manager if that managed set is not being used to manage appliances.

Before you begin

Verify that there are no appliances in the managed set that you want to remove from the DataPower appliance manager. If the managed set contains appliances, you must remove those appliances from that managed set before you try to remove the managed set from the appliance manager.

About this task

Complete the following actions to remove a managed set from the DataPower appliance manager.

1. From the administrative console, click **Servers > DataPower > Managed sets**.
2. Select the managed set that you want to remove.

You can select multiple managed sets if there are multiple managed sets that you want to remove.

3. Click **Remove**.

Managed set collection

Use this page to add, view, or delete a managed set. A managed set is a group of appliances whose firmware, shareable appliance settings and managed domains are all kept synchronized.

To view this administrative console page, click **Servers > DataPower > Managed sets**.

You can then:

- Click the name of one of the managed sets to view the values specified for that managed set configuration. The settings page displays and shows the current settings for the selected managed set.
- Click **New** if you want to create a new managed set.
- Select one of the managed sets in the list, and then click **Delete** if you want to delete that managed set.
- Click **Launch** to access the DataPower WebGUI in the default domain on the master appliance in the selected managed set. Pop-ups must be enabled. See the documentation for the DataPower WebGUI for information about the tasks that you can perform from the DataPower WebGUI.

Name

Specifies a user-defined symbolic name for a managed set.

A managed set name must be unique and cannot contain an invalid character. The name field cannot contain the following characters: # \$ @ \ / , ; " * ? < > | = + & % '.

Operational Status

Specifies the aggregated operational status of all the managed domains on all of the appliances in the managed set.

This field is read-only. The following values might appear in this field:

up Indicates that all of the service objects in all the managed domains on all of the appliances in the managed set are enabled.

partial Indicates that the service objects in all the managed domains on all of the appliances in the managed set are a mix of enabled, disabled and unknown.

down Indicates that all of the service objects in all the managed domains on all of the appliances in the managed set are disabled.

unknown

Indicates that the state of all of the all the managed domains on all of the appliances in the managed set cannot be determined. It is possible that there are no managed domains, or the state of the managed domains has yet to be retrieved, or that communication has been lost to the appliances.

Synchronization Status

Specifies whether the designated firmware version, sharable appliance settings, and managed domain versions for the managed set are synchronized across all of the appliances in the managed set. This synchronization status combines the firmware synchronization status, the settings synchronization status, and the synchronization status of all the managed domains on all the appliances of the managed set.

This field is read-only. The following values might appear in this field:

synced

Indicates that the managed set is synchronized.

changes pending

Indicates that the synchronization is queued. At least one appliance in the managed set has a firmware synchronization, settings synchronization, or managed domain synchronization queued.

in progress

Indicates that the synchronization is currently being synchronized. At least one appliance in the managed set has a firmware synchronization, settings synchronization, or managed domain synchronization in progress. Other firmware synchronizations, settings synchronizations, or managed domain synchronizations might also be pending for appliances in the managed set.

unknown

Indicates that the synchronization status is not known. At least one appliance in the managed set has a firmware synchronization, settings synchronization, or managed domain synchronization for which the synchronization state is unknown. Typically, this means that the appliance manager has not yet contacted the appliance to get the management status. Other firmware synchronizations, settings synchronizations, or managed domain synchronizations might also have changes pending, or in progress for appliances in the managed set.

error

Indicates that an attempt to synchronize the managed set has failed because of an error on an appliance in the managed set. The appliance did respond, but responded with an error. Appliances in the managed set might also have a management status of unknown, in progress or changes pending for a firmware synchronization, settings synchronization, or managed domain synchronization. See the error log for the deployment manager for more information.

unreachable

Indicates that the appliance is unreachable. An attempt to synchronize the managed set has failed because an appliance is not responding. Appliances in the managed set might also have a management status of unknown, in progress, changes pending or error for a firmware synchronization, settings synchronization, or managed domain synchronization. See the error log for the deployment manager for more information.

Managed set settings

Use this page to view the general properties for the managed set, and a list of the appliances that are part of this managed set. You can also use this page to view lists of the shareable appliance settings versions, and managed and unmanaged domains that can be used with this managed set.

To view this administrative console page, click **Servers > DataPower > Managed sets > *managedset_name***.

In addition to specifying values for a managed set, you can use the following buttons to edit appliance membership, synchronize appliances or access the DataPower WebGUI to configure an appliance.

Button	Resulting action
Edit Membership	Accesses a list of DataPower appliances that are members of the managed set. You can use this list to add appliances, remove appliances, or mark an appliance as the master appliance.
Synchronization	Forces a synchronization of the appliances in the managed set. The appliance manager attempts to maintain the synchronization automatically, but this option allows you to request that a synchronization be performed.

Button	Resulting action
Launch Note: You can also use the Launch button that is located in the Unmanaged domains section to launch the DataPower WebGUI in the default domain on the master appliance. The Launch button that is located in the Managed domains section provides a similar function. However, if you click that button, the DataPower WebGUI is launched in the managed domain on the master appliance instead of in the default domain.	Launches the DataPower WebGUI in the default domain on the master appliance in the managed set using the credentials that are defined in the DataPower appliance manager. Whenever you modify shareable appliance settings or a managed domain on the master appliance, the changes are distributed to every appliance in the managed set. Verify that pop-ups are enabled on your browser before you launch the DataPower WebGUI.

In the Firmware section, click **Change firmware** to change the firmware for the managed set. The firmware must have the same appliance type, model type and licensed features as the appliances in the managed set.

In the Settings section, click **Version history** to view the versions of shareable appliance settings that are available for this managed set

In the Appliance section, click **Appliance** to display a table of all of the appliances that are assigned to the managed set.

Name

Specifies a user-defined symbolic name for a managed set. This field is read-only

Firmware version

Specifies the firmware version that is to be synchronized to all of the appliances in the managed set.

If you have a single appliance managed set, and if there is not a firmware version in the DataPower appliance manager that matches the firmware that is used by the master appliance for that managed set, this field might be blank. In this situation, you can either add a firmware to the DataPower appliance manager that has the same version number, appliance type, model type and compatible licensed features as the single appliance in the managed set, or you can click **Change firmware** in the Firmware section, and select a firmware version for the managed set. This new firmware automatically becomes the firmware for the single appliance in this managed set, and must have the same appliance type, model type and licensed features as that appliance.

Operational status

Specifies the aggregated operational status of all the managed domains on all of the appliances in the managed set.

The status is unknown if the managed set does not have any managed domains. If the managed set has managed domains, the status reflects the aggregated operational status of all of the managed domains on all of the appliances in this managed set.

This field is read-only. The following values might appear in this field:

up Indicates that all of the service objects in all the managed domains on all of the appliances in the managed set are enabled.

partial Indicates that the service objects in all the managed domains on all of the appliances in the managed set are a mix of enabled, disabled and unknown.

down Indicates that all of the service objects in all the managed domains on all of the appliances in the managed set are disabled.

unknown

Indicates that the state of all of the managed domains on all of the appliances in the managed set

cannot be determined. This status might mean that the state has not yet been retrieved, or communication to the appliance has been lost. A managed set that does not have any managed domains always has a status of unknown.

Synchronization status

Specifies whether the designated firmware version, sharable appliance settings, and managed domain versions for the managed set are synchronized across all of the appliances in the managed set. This synchronization status combines the firmware synchronization status, the settings synchronization status, and the synchronization status of all the managed domains on all the appliances of the managed set.

This field is read-only. The following values might appear in this field:

synced

Indicates that the managed set is synchronized.

changes pending

Indicates that the synchronization is queued. The appliance has at least one firmware synchronization, settings synchronization, or managed domain synchronization queued.

in progress

Indicates that the synchronization is currently being processed. At least one appliance in the managed set has a firmware synchronization, settings synchronization, or managed domain synchronization in progress. Other firmware synchronizations, settings synchronizations, or managed domain synchronizations might also be pending for appliances in the managed set.

unknown

Indicates that the synchronization status is not known. At least one appliance in the managed set has a firmware synchronization, settings synchronization, or managed domain synchronization for which the synchronization state is unknown. Typically, this means that the appliance manager has not yet contacted the appliance to get the management status. Firmware synchronizations, settings synchronizations, or managed domain synchronizations might also be pending or in progress for appliances in the managed set.

error Indicates that an attempt to synchronize the managed set has failed because of an error on an appliance in the managed set. The appliance did respond, but responded with an error. Appliances in the managed set might also have a management status of unknown, in progress or changes pending for a firmware synchronization, settings synchronization, or managed domain synchronization. See the error log for the deployment manager for more information.

unreachable

Indicates that the appliance is unreachable. An attempt to synchronize the managed set has failed because an appliance is not responding. Appliances in the managed set might also have a management status of unknown, in progress, changes pending or error for a firmware synchronization, settings synchronization, or managed domain synchronization. See the error log for the deployment manager for more information.

Managed domains

Specifies a list of managed domains. You can use this list to change a managed domain to an unmanaged domain.

The managed domains of a master appliance are copied to all appliances in the managed set. Expand this section to view a list of the managed domains. You can then select one or more managed domain, and click **Unmanage** if you want to turn the selected domains into unmanaged domains. When a managed domain becomes an unmanaged domain, it displays in the Unmanaged domains table instead of in the Managed domains table.

If you need to modify a managed domain on the master appliance, select that managed domain, and then click **Launch** to launch the DataPower WebGUI on the selected managed domain on the master appliance

in the managed set. A new domain version is automatically created whenever you save changes in the managed domain on the master appliance. This new version becomes the current version and the changes are synchronized across the managed set.

Unmanaged domains on the master appliance

Specifies a list of unmanaged domains. You can use this list to manage these unmanaged domain.

The managed domains of a master appliance are copied to all appliances in the managed set. By default, domains are not managed. Unmanaged domains are not copied to subordinate appliances in the managed set.

Expand this section to view a list of the unmanaged domains. You can then select one or more unmanaged domain and click **Manage** to turn the selected domains into managed domains. When a unmanaged domain becomes a managed domain, it displays in the Managed domains table instead of in the Unmanaged domains table.

If you need to modify the configuration of the master DataPower appliance that is contains in an unmanaged domain, select that domain, and then click **Launch** to launch the DataPower WebGUI on the default domain of the master appliance in the managed set. You can then navigate to the unmanaged domain in the DataPower WebGUI. Clicking this **Launch** button is equivalent to clicking the **Launch** button at the top of this administrative console page.

Edit membership for a managed set

Use this page to add appliances to the managed set, remove appliances from the managed set, or mark an appliance as the master appliance for the managed set. All of the appliances in a managed set must have the same appliance type, model type and licensed features.

This page lists appliances that are members of this managed set.

To view this administrative console page, click **Servers > DataPower > Managed sets > *managedset_name* > Edit membership**.

To mark an appliance as the master appliance for the managed set, select the appliance that you want to make the master appliance, click **Mark as master**, and then click **Save**. After you click **Save**, the selected appliance becomes the master appliance.

To remove appliances from the managed set, select the appliances that you want to remove, click **Remove**, and then click **Save**. After you click **Save**, all the appliances that you selected are removed from the managed set.

Add appliances to the Managed Set.

Specifies a list of DataPower appliances that you can add to a managed set. Because all of the appliances in a managed set must have the same appliance type, model type and licensed features, only the appliances that meet these conditions are included in this list.

To add appliances to a managed set, select the appliances that you want to add from the list of appliances, and then click **Apply > Save**. After you click **Save**, all of the appliances that you selected are added to the managed set.

Appliance

Specifies the user-defined appliance name for a DataPower appliance. This field is read-only

Host name

Specifies an IP address or the fully qualified host name of a DataPower appliance. This field is read-only

Appliance type

Specifies the appliance type, such as XA35, XS40, and XI50. This field is read-only

Master

Specifies whether or not an appliance is the master appliance for the managed set. All appliances in the managed set are synchronized using the firmware version, settings, and managed domains of the master appliance. This field is read-only

Modifying DataPower appliance manager settings

You can change the global settings that apply to the DataPower appliance manager.

About this task

Complete the following actions to change the global settings that apply to the DataPower appliance manager.

1. From the administrative console, click **Servers > DataPower > Appliance manager settings**.
2. To change the maximum number of domain and settings versions that can be stored in the DataPower appliance manager, specify a new value in the **Maximum number of versions to store** field.

When the value specified in this field is exceeded, the DataPower appliance manager does not allow you to create any new versions until you delete one or more of the existing versions.

Note: If you decrease the value for this setting, you might lose versions if the current number of versions stored exceeds the new maximum number of versions stored.

3. To change the directory where firmware versions, shareable appliance settings and managed domain versions are saved, specify that new directory in the **Versions directory** field.

Note: If you change the versions directory, you must move all of the current versions from the current directory to the new directory. This process might take a considerable amount of time, depending on how many versions you have, and could keep the appliance manager from performing other operations

4. View the status for the appliance manager.
The DataPower appliance manager automatically starts when the deployment manager starts, if the appliance manager has appliances to manage. The appliance manager also automatically starts when a user submits a request to the appliance manager. For example, displaying the list of appliances that are defined in the appliance manager causes the appliance manager to start if it is not already running.
5. In the Operations section, click **Export** to export a copy of the DataPower appliance manager configuration to a file. This process is useful if you want to create backup copies of the appliance manager configuration.
6. In the Operations section, click **Import** to import the DataPower appliance manager configuration from a file. The file that you import from must reside on the same system as the deployment manager.
7. If you changed the maximum number of versions that you want stored, or you changed the version directory, click **Apply** to apply your changes.
8. Click **Shutdown Manager** if you need to stop the appliance manager.

Appliance manager settings

Use this page to specify global settings for the DataPower appliance manager. You can also use this page to shutdown the appliance manager.

To view this administrative console page, click **Servers > DataPower > Appliance manager settings**.

You can then click:

- **Export**, if you want to start the wizard that is used to export your DataPower appliance manager configuration to a file. After the wizard starts, you must specify the fully qualified name of the file into which you want to export the DataPower appliance manager configuration. The Export wizard exports the specified file onto the file system of the deployment manager.
- **Import** to start the wizard that is used to import a previously exported copy of your DataPower appliance manager configuration from a file. After the wizard starts, you must select either **Upload from local system** or **Upload from remote system**. Then specify the fully qualified file path of the file from which you want to import your DataPower appliance manager configuration. If you do not know the fully qualified name of the file, you can use the browse function to locate the file.
- **Shutdown Manager** if you want to shutdown the appliance manager.

Maximum versions stored

Specifies the maximum number of domain and settings versions that can be stored in the DataPower appliance manager. For example, if you specify a value of 3, for each managed set, you can have 3 sharable settings versions, and 3 versions of each managed domain.

You can view the shareable appliance settings version history, or the domain versions history to determine which versions are available for specific managed sets.

To view the sharable settings version history for a managed set, click **Servers > DataPower > Managed sets > *managed_set_name* > Settings Version history**

To view the version history for a managed domain for a managed set, click **Servers > DataPower > Managed sets > *managed_set_name* > Managed Domains > *.managed_domain_name* > Version history**

Versions directory

Specifies the directory into which you want the versions stored. Whenever the DataPower appliance manager detects a change in the sharable appliance settings, or a managed domain on the master appliance for a managed set, a new sharable appliance settings, or managed domain version is automatically created.

Status

Specifies whether the appliance manager is started or stopped. Click **Shutdown Manager** to stop the appliance manager.

DataPower appliance manager automatically starts when the deployment manager starts if there are appliances for the appliance manager to control. The appliance manager also automatically starts when requests are sent to the appliance manager.

If the status is stopped, the DataPower appliance manager automatically restarts the next time that you send a request to the appliance manager.

Importing an exported DataPower appliance manager configuration

You can import an exported DataPower appliance manager configuration if you want to use that configuration, instead of the current configuration, as the configuration for the appliance manager. For example, if the current configuration is corrupted, you might want to import a backup copy of that configuration that you have stored in either a local or remote file system.

About this task

Like a file that contains a firmware version, a file that contains an exported DataPower[®] appliance manager configuration can be stored on the system on which the browser is running, or it can be stored on the system on which the deployment manager is running. A file that is on stored on the system on

which the browser is running is considered stored on a local system. A file that is stored on the system on which the deployment manager is running is considered stored on a remote system.

Complete these steps to use the administrative console to import a repository file from a remote system.

1. In the administrative console, click **Servers > DataPower > Appliance manager settings**.
2. In the Operations section, click **Import** to import an exported DataPower appliance manager configuration.
3. Select **Upload from local system** or **Upload from remote system**.
4. In the **Fully qualified file name** field, enter the fully qualified file name for the file that you want to import. The fully qualified name of a file includes the full directory path to that file, and the file name. If you don't know the fully qualified name of the file, use the browse function to locate the file.
5. Click **Next**.
6. View the summary information, and then click **Finish**.

Exporting the DataPower appliance manager configuration

You can export the DataPower appliance manager configuration to a file on the system where the deployment manager is running.

About this task

Complete these steps if you want to use the administrative console to export the DataPower appliance manager configuration to a file on the system where the deployment manager is running.

1. From the administrative console, click **Servers > DataPower > Appliance manager settings**.
2. In the Operations section, click **Export** to export the file to the system where the deployment manager is running.
3. In the **Fully qualified file name** field, enter the fully qualified file name for the file that you want to export. The fully qualified name of a file includes the full directory path to that file, and the file name.
4. Click **Next**.
5. View the summary information, and if the summary information is correct, click **Finish**.

Monitoring tasks that DataPower appliance manager is handling

Use this page to view the status of long running requests, or tasks, that are queued for the DataPower appliance manager to complete.

About this task

There are additional tasks that the appliance manager automatically runs that are used to maintain the state of the managed sets. These tasks are not included in the list of long running tasks. However, if an error occurs while one of these tasks are running, the error is recorded in the log to help you diagnose the problem.

Complete the following steps if you want to use the administrative console to view the long-running tasks that are in queue for the DataPower appliance manager.

1. From the administrative console, click **Servers > DataPower > Tasks**.
2. From the Tasks collection page, view information about the tasks.
3. Optional: Select the Task ID of the task that you want to delete, and then click **Remove**.
When you click **Remove**, the task is removed from your view, but the appliance manager still runs the task as soon as it is able to do so.

Note: You cannot remove a task to stop the task from being completed. The only way to interrupt a running task, or prevent the appliance manager from running a task, is to shutdown the appliance manager. Shutting down the appliance manager terminates all running and queued tasks.

Tasks collection

Use this page to view the status of a task. A task is a long running request that you have asked the DataPower appliance manager to process. You can remove a task from the list of pending tasks. However, removing a task from that list does not prevent that task from running.

This page lists the long running tasks that need to be completed for an appliance. If you notice that an error has occurred during the processing of a task, you can find additional information about that error in the deployment manager log. This additional information might help you to diagnose the problem.

There are additional tasks that the appliance manager automatically runs that are used to maintain the state of the managed sets. These tasks are not included in the list of long running tasks. However, if an error occurs while one of these tasks are running, the error is also recorded in the deployment manager log.

To view this administrative console page, click **Servers > DataPower > Tasks**.

To remove a task, select the task identifier (ID) in the list of tasks, and click **Remove**.

Note: Removing a task does not stop the task from being completed, nor does it delete the task. Even though you are no longer able to view the task, the appliance manager still executes the task when it gets to that task in the queue of tasks to execute. The only way to interrupt a running task, or prevent the appliance manager from running a task, is to shutdown the appliance manager. Shutting down the appliance manager terminates all running and queued tasks.

Task ID

Specifies an automatically generated ID number for a task.

Creation Date

Specifies the date on which the task was created

Description

Specifies information about the task.

Result

Specifies the result for the completed task, such as the serial number for an added appliance.

Created By

Specifies the username of the user who issued the request that resulted in the task being created. This field is blank if security is not enabled for the deployment manager.

Status

Specifies whether the task is completed, or is still in progress. Valid states include: Completed, Error, In progress, and Queued.

Administering managed domain versions

You can change the version of a managed domain that the DataPower appliance manager uses for a managed set, or you can copy a version of a managed domain to another managed set.

About this task

Each time domain settings are changed, DataPower appliance manager automatically creates a copy of the previous domain settings as a new domain version. You can use the administrative console to view a history of domain versions, change a different domain version to the active version, or copy a domain version to another managed set. Complete one or more of the following steps to administer managed domains.

1. From the administrative console, click **Servers > DataPower > Managed sets > *managedset_name***.
2. In the Managed domains section, expand Managed domains, and then click the name of the managed domain whose history you want to view.
3. Click **Versions > Version history > Details** to view a list of the versions of that managed domain. From the Domains history settings page, you can perform the following actions:
 - a. Select a version, and click **Change Domain Version** to change the selected version to be the new active version
 - b. Click a version number to view information about the version, or change the description for the version.
 - c. In the Additional Properties section, click **Copy to another managed set** to copy the domain version to another managed set.

Domain history collection

Use this page to view the domain history for DataPower appliance manager. A domain version is an automatically generated copy of a managed domain.

To view this page, in the administrative console, click **Servers > DataPower > Managed sets > *managed_set_name* > Managed domains > *domain_name* > Versions > Version history > Details**.

The newest version of the domain is, by default, the active version. To change from the current domain version to a domain version in the list, select the version number in the list of available versions, and click **Change Domain Version**.

Version

Specifies a numeric value that represents a domain version. When you change the settings for a domain on the master appliance in a managed set, the appliance manager automatically detects the change and creates a copy of the change as a new domain version.

Date

Specifies the date when the domain version was created and stored in the DataPower appliance manager. Each time a domain is changed, a new version is created and stored in the DataPower appliance manager.

Description

Specifies information about the domain version. When a domain version is automatically generated, a default description is provided.

In Use

Specifies which version is active. The active version is automatically synchronized to all of the appliances in the managed set. Only one version can be active at a time.

Domain history collection

Use this page to view detailed information about a specific domain version for a managed domain in a managed set. A domain version is an automatically generated copy of an existing domain.

You can also use this page to specify a description for this domain version, or copy this domain version to a managed set.

To view this administrative console page, click **Servers > DataPower > Managed sets > *managed_set_name* > Managed domains > *domain_name* > Versions > Version history > Details > *version_number***.

Version

Specifies the automatically generated version number of this domain version. This field is read-only.

Whenever you change a managed domain, the appliance manager automatically detects the change and creates a copy of the changed domain as a new domain version. Because the version number is automatically generated, you cannot change the value that appears in this field.

Date stored

Specifies the date when the domain version was created and stored in the DataPower appliance manager. This field is read-only.

Whenever a domain is changed, a new version is created and stored in the DataPower appliance manager. Because the date is automatically generated, you cannot change the value that appears in this field.

Description

Specifies information about the domain version. When a domain version is automatically generated, a default description is provided.

Managing versions of sharable appliance settings

Every time that sharable appliance settings are changed, the DataPower appliance manager automatically creates a copy of the new shareable appliance settings as a new settings version. You can view a list of all of the available settings versions that are available for an managed set, you can change which version is the active version, or you can copy a version to another managed set.

About this task

To manage the versions of sharable appliance settings for a managed set, in the administrative console, click **Servers > DataPower > Managed sets > *managedset_name***, and then, in the Settings section, click **Version history**. Then complete one or more of the following actions if you want to display a list of all of the versions of sharable appliance settings that are available for a managed set, display information about a specific version, deploy a previous version, or associate a version with another managed set.

- View the list of available versions.
- To change which sharable appliance settings version is the active version, select a version, and then click **Change Settings Version**. The DataPower appliance manager automatically synchronizes all of the appliances in the managed set to use the new active version.
- To view information about a specific version, click the name of that
- To change the description that is specified for a specific version, click the name of that version and update the **Description** field.
- To associate the selected version to another managed set, click the name of that version, and then, in the Additional Properties section, click **Copy to another managed set**.
 1. Using the wizard, choose a managed set from the list of managed sets, and then click **Next**
 2. View the summary information to verify that you have selected the correct managed set, and then click **Finish**

The Copy to another Managed Set wizard starts and enables you to copy the selected version to a different managed set. The sharable appliance settings are applied to all appliances in this other managed set.

Note: After the initial copy of the sharable appliance settings, the two managed sets are managed independently. Therefore, future changes to the sharable appliance settings in one managed set are not reflected in the other managed set.

If you click **Servers > DataPower > Managed sets**, click the name of the managed set that you selected in the wizard, and then, in the Settings section, click **Version history**, you should see a new version as the active sharable appliance settings version for that managed set. The new active version is the version that the Copy to another Managed Set wizard created during the copy process.

What to do next

Changing the active settings version or copying the settings to another managed set causes a task to be created. If you want to monitor the progress of this task, in the administrative console, click **Servers > DataPower > Tasks**

Settings history collection

Use this page to view the settings history for a DataPower appliance manager. A settings version is an automatically generated copy of the sharable appliance settings for the master appliance in a managed set.

To view this page, in the administrative console, click **Servers > DataPower appliances > Managed sets > *managedset_name* > Settings > Version history > Details**.

The newest version of the settings is, by default, the active version. To change from the current settings version to a settings version in the list, select the appropriate version number in the list, and click **Change Settings Version**.

Version

Specifies a numeric value that represents a settings version. When you change the shareable appliance settings for a master appliance in a managed set, the appliance manager automatically detects the change, and creates a copy of the change as a new settings version.

In Use

Specifies which version is active. The active version is automatically synchronized to all of the appliances in the managed set. Only one version can be active at a time.

Date stored

Specifies the date when the settings version was created and stored in the DataPower appliance manager.

Description

Specifies user-defined information about the settings version.

Settings version Collection

Use this page to view detailed information about a specific settings version for the appliances in a managed set. A settings version is an automatically generated copy of the shareable appliance settings for the appliances in a managed set.

You can also use this page to specify a description for this settings version, or you can click **Copy settings version to another managed set** to copy this settings version to another managed set. For example, if you have a managed set on a test system, and a different managed set on a production system, you can use this copy capability to ensure that you are using the same sharable appliance settings version on both managed sets.

When you copy a settings version to a different managed set, the copied settings version becomes the active version for the destination managed set.

To view this administrative console page, click **Servers > DataPower > Managed sets > *managedset_name***, and then, in the Settings section, click **Version history**.

Version

Specifies a numeric value that represents a settings version. When you change the sharable appliance settings for the master appliance in a managed set, the appliance manager automatically detects the change, and creates a copy of the change as a new settings version. This is a read-only field.

Date stored

Specifies the date when the settings version was created and stored in the DataPower appliance manager. This is a read-only field.

Description

Specifies information about the settings version. When a settings version is automatically generated, a default description is provided.

Administering DataPower appliance domains

A DataPower appliance domain is a group of configuration information for an appliance. By default, these domains are unmanaged. You can use the administrative console to change an unmanaged domain to a managed domain, or to change a managed domain to an unmanaged domain. However, you cannot use the administrative console to configure a domain. You must use the DataPower WebGUI to configure a domain. See the DataPower WebGUI documentation for information about configuring a domain.

About this task

You can use the administrative console to change an unmanaged domain to a managed domain. However, you cannot use the administrative console to configure a domain. Use the DataPower WebGUI to configure a domain. See the DataPower WebGUI documentation for information about configuring a domain.

Note: The DataPower appliance manager synchronizes managed domains from the master appliance to the subordinate appliances in the managed set. However, it is possible that the managed domain might not be completely functional on all of the subordinate appliances. An example of this situation is when a service object, such as an XML firewall, in the managed domain has a listening port conflict on one of the subordinate appliances.

Complete one or more of the following steps to administer domains.

1. From the administrative console, click **Servers > DataPower > Managed sets > *managedset_name***.
2. To change an unmanaged domain to a managed domain, complete the following actions.
 - a. In the Managed Domains section, expand Unmanaged domains.
 - b. Select the unmanaged domains that you want to convert to managed domains, and then click **Manage**.
3. To change an managed domain to an unmanaged domain, complete the following actions.
 - a. In the Managed Domains section, expand Managed domains.
 - b. Select the managed domains that you want to convert to unmanaged domains, and then click **Unmanage**.
4. To access the DataPower WebGUI and re-configure a managed or unmanaged domain, select the domain that you want to re-configure, and click **Launch**.
5. To delete a managed domain, select the domain that you want to delete, and click **Delete**.

Results

If you changed any managed domains to unmanaged, those domains are now listed under Unmanaged domains. Similarly if you changed any unmanaged domains to managed, those domains are now listed under Managed domains.

Managed domain settings

Use this page to view the settings for a managed domain. You can also use this page to view the version history for this managed domain, or to view the managed domains that exist for a specific appliance.

DataPower appliances support the use of application domains. An application domain consists of resources that are configured to provide and support one or more services.

Domains are used to partition configuration information on the appliance into self contained units that make the information easier to manage. A managed domain is a domain on the master appliance that has been added to a managed set in the DataPower appliance manager. The DataPower appliance manager synchronizes the managed domain across all of the appliances in the managed set.

To view this administrative console page, click **Servers > DataPower > Managed sets > *managedset_name* > Managed domains > *domain_name***.

Click **Launch** to access the DataPower WebGUI in the domain on the master appliance in the managed set. Pop-ups must be enabled to launch the DataPower WebGUI.

Click **Appliance** to display a table of all of the appliances that are assigned to the managed set and are synchronized for this managed domain. The table provides the following information for each appliance:

- The Name column specifies the name of the appliance in the appliance manager.
- The Master column indicates whether the appliance is the master in the managed set that contains this managed domain.
- The Operational Status column indicates the operational status of the managed domain in the appliance. The following values might appear in this column:

up Indicates that all of the service objects in the managed domain on the appliance are enabled.

partial Indicates that the service objects in the managed domain on the appliance are a mix of enabled, disabled and unknown.

down Indicates that all of the service objects in the managed domain on the appliance are disabled

unknown

Indicates the state of all of the in the managed domain on the appliance could not be determined. Possible causes are that the state has yet to be retrieved or communication has been lost to the appliance.

- The Synchronization Status column indicates whether the managed domain on the appliance is the same as the specified domain version for the managed set. The following values might appear in this column:

synced

Indicates that the managed domain on this appliance is the same as the specified domain version for the managed set.

changes pending

Indicates that the synchronization is queued for the managed domain on the appliance.

in progress

Indicates that the synchronization is currently being processed for the managed domain on the appliance.

unknown

Indicates that the synchronization status is not known. Typically, this means that the appliance manager has not yet contacted the appliance to get the synchronization status.

error Indicates that an attempt to synchronize the managed domain on the appliance has failed because of an error on an appliance. The appliance did respond, but responded with an error. See the log for the deployment manager for more information.

unreachable

Indicates that the appliance is unreachable. An attempt to synchronize the managed domain has failed because the appliance is not responding. See the log for deployment manager for more information.

Click **Version History** to display a table of the versions for the managed domain. The table provides the following information for each version:

- The Version column specifies the version of the managed domain.
- The Date column indicates the date that the domain version was created.
- The In Use column indicates which domain version is active. The active domain version is the version this is synchronized across all of the appliances in the managed set.
- The Description column provides a description of the domain version.

Name

Specifies the name for the managed domain.

Secure Socket Layer communication with DataPower

Based on the default installations of the application server and the DataPower appliance manager, secure sockets layer (SSL) communication is used to send commands and receive events. The default SSL configuration used by the DataPower appliance manager can be strengthened by customizing the SSL connection. Modifying the default SSL configuration is optional and only needs to be done if the default configuration is not sufficient for your requirements.

SSL is used to send commands to each known appliance manager. In this scenario, the application server and the DataPower appliance manager behave as the SSL client and the DataPower appliances are acting as the SSL servers. This SSL connection uses the `ibmPKIX` trustmanager to do some verification of the DataPower appliance. Neither the certificate chain nor the revocation list for the certificate of the DataPower appliance are checked. The default configuration also does not do any SSL client validation for this scenario.

SSL is also used for the events received by the application server and the DataPower appliance manager from each DataPower appliance being managed. In this scenario, the application server and the DataPower appliance manager is the SSL server and the DataPower appliances are the SSL client. SSL client validation is also not performed in this scenario by default.

Most customizations that are available for SSL connections in WebSphere Application Server can be applied to the connections used by the DataPower appliance manager. To customize the SSL connections used for communication between the DataPower appliance manager and the DataPower appliances, each change made to the SSL connection on the DataPower appliance manager must also be accompanied by a complimentary change on each of the DataPower appliances that it manages. The DataPower appliance manager uses the **DataPowerMgr_sslConfig SSL** profile to connect with the DataPower appliances to send the appliances commands. You may make changes to this profile to influence the SSL connection that is used to send commands to the appliances. The DataPower appliance manager uses the **DataPowerMgr_inbound_secure** inbound endpoint on the Dmgr to receive events from the appliances it manages. You may make changes to the profile used by this endpoint to influence the SSL connection used to send events from the managed appliances.

Note: For instructions on how to modify the AMP XML Management Interface SSL configuration on a DataPower appliance, refer to the DataPower appliance WebGUI Guide section on the XML Management Interface and how to create a custom SSL Proxy profile.

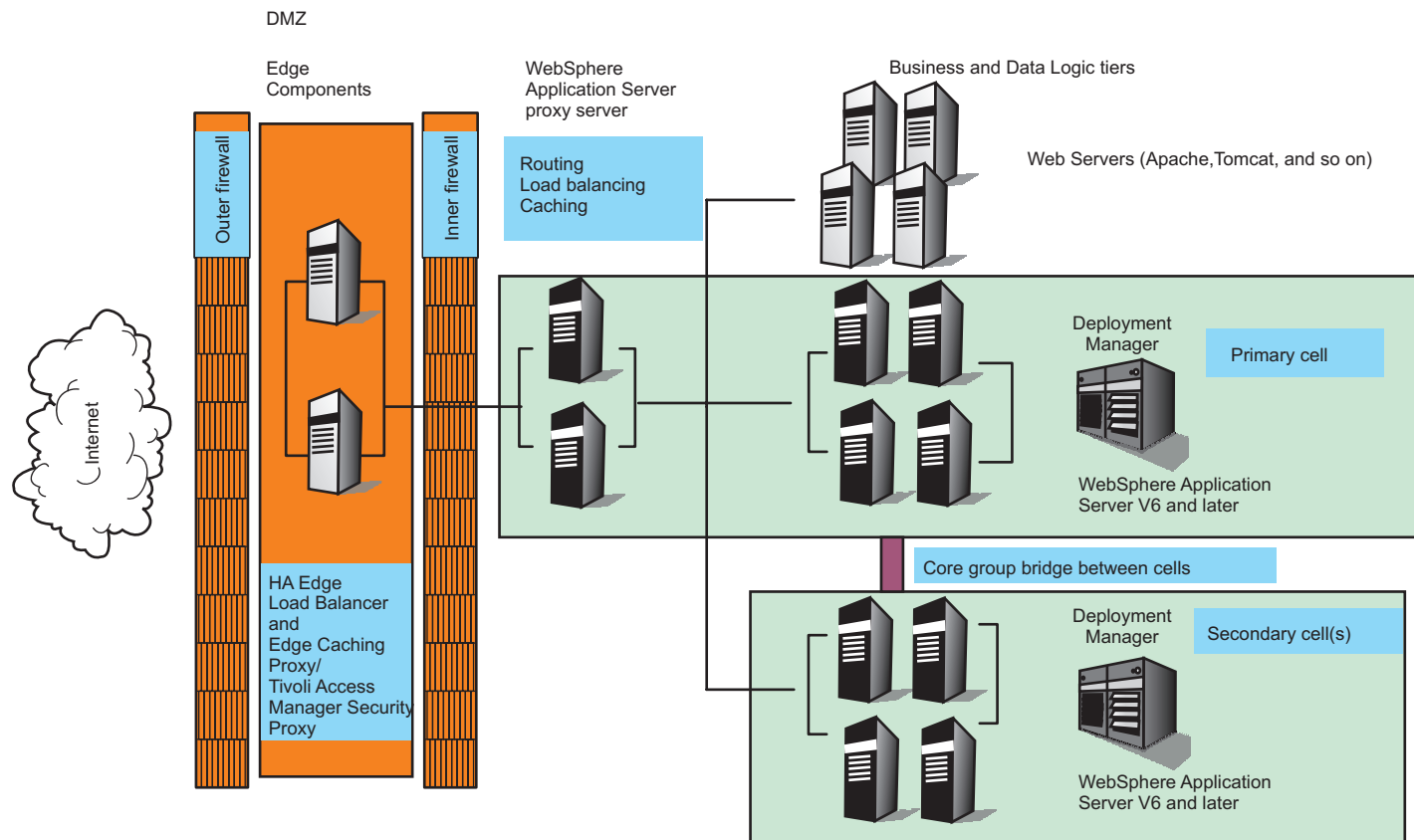
Chapter 4. Setting up the proxy server

The *proxy server* is a specific type of application server that routes HTTP requests to content servers that perform the work. The proxy server is the initial point of entry, after the firewall, for requests into the enterprise.

About this task

The proxy server acts as a surrogate for content servers within the enterprise. As a surrogate, you can configure the proxy server with rules to route to and load balance the clusters of content servers. The proxy server is also capable of securing the transport, using Secure Sockets Layer (SSL), and the content using various authentication and authorization schemes. Another important feature is its capability to protect the identity of the content servers from the Web clients by using response transformations (URL rewriting). The proxy server can also improve performance by caching content locally and by protecting the content servers from surges in traffic.

A proxy server configuration provides settings that control how a proxy server can provide services for the enterprise applications and their components. This section describes how to create and configure proxy servers in an existing application server environment.



In Version 6.0.2 of the product, you had to augment the deployment manager profile to manage the proxy server. For Version 6.1 and higher, the proxy server is managed from the administrative console without initial augmentation.

- Create an HTTP or Session Initiation Protocol proxy server to route requests to application server nodes.
- Install a Session Initiation Protocol proxy server.
- Migrate profiles for the proxy server.

- Modify or add HTTP endpoints

What to do next

Next, create proxy servers in the cell.

Creating a proxy server

This topic provides information to create and configure a proxy server. When creating a proxy server, only the application server profile can be used as the target node.

About this task

The proxy server routes requests to application server nodes. The proxy server can dynamically route requests to all on-demand configuration (ODC) enabled application servers without additional configuration.

When you install the product, two profiles are created: a stand-alone application server profile, which is the default profile, and a deployment manager profile, which is called dmgr. When creating a proxy server, do not choose the deployment manager profile. Only the application server profile can be used as the target node.

1. Create a proxy server in the administrative console by clicking **Servers > Server Types > WebSphere proxy servers > New**.
2. Select the node on which you want the proxy server to reside.
3. Enter a name for the new proxy server and click **Next**.
4. The supported protocols HTTP and SIP are selected for you. Select HTTP if your proxy server will route requests to and from a Web container. Select Session Initiation Protocol (SIP) if your proxy server will route requests to and from a SIP container. Determine whether or not to generate unique HTTP ports by selecting or clearing Generate unique HTTP ports. Click **Next**.

Note: If you create multiple proxy servers on the same node for vertical scaling, then you might select the option to generate unique ports to avoid port conflicts. Certain advanced scenarios pertain to port mapping that might require unique ports. For example, a load balancer can load balance requests to the proxy servers within the same node, assuming that each proxy server is listening on a unique HTTP port. For the proxy server to accept requests for a specific virtual host, it is necessary to add the unique HTTP ports that are generated to the host alias of the virtual host. It might also be necessary to modify the port values that the wizard generates, if these ports conflict with other local servers on the same node.

5. Select a proxy server template on which to base your proxy server. Click **Next**. You can select a default template, or you can choose to map to an existing proxy server.

Note:

Mapping to a pre-existing proxy server is a time-saving technique. You can build one proxy server and apply all of the specific configurations your environment needs, and then use that proxy server as a template.

6. Review the summary panel and click **Finish**.

Results

You now have a functional proxy server that automatically routes HTTP requests to the cell that the proxy server belongs to or SIP requests to and from a SIP container. To enable routing to another cell, configure your cell to communicate with other cells.

Note: If the proxy server fails to start when attempting to start it as a non-privileged user on UNIX®-based operating systems, change the ports of the PROXY_HTTP_ADDRESS and PROXY_HTTPS_ADDRESS transport chains to values greater than 1024.

Proxy server collection

This topic lists the proxy servers in the cell. A proxy server resides within a node.

A proxy server is used to classify, prioritize, and route HTTP and SIP requests to servers in the enterprise as well as cache content from servers. You can use this page to create, delete, or modify a proxy server.

To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers > .**

To configure the proxy server to route work to Version 6.0 WebSphere Application Servers in another cell, use core group bridge settings (**Servers > Core Groups > Core group bridge settings**), which sets up communication between cells.

Currently, configuring the proxy server to route work to a V6 WebSphere Application Server - Express cell requires advanced configuration.

To configure the proxy server to route work to an application server which is not a WebSphere application server or is an application server that resides in a pre-Version 6.0 cell, the following advanced configuration is required:

1. Define a generic server cluster. From the administrative console, click **Servers > Clusters > Generic server clusters**.
2. Define a URI group. From the administrative console, click **Environment > URI groups**.
3. Create routing rules. From the administrative console, click **Servers > Server Types > WebSphere proxy servers > proxy_server_name > Proxy Server Properties > Routing rules**.

Both generic server clusters and URI groups are also accessible in the administrative console under Related Items for the proxy server.

Name

Specifies a logical name for the proxy server. For WebSphere Application Server, server names must be unique within a node.

If you have multiple nodes in a cluster, the server names must also be unique within the cluster. You cannot use the same server name within two nodes that are part of the same cluster. WebSphere Application Server uses the server name for administrative actions, such as referencing the server in scripting.

Node

The name of the node where the proxy server resides.

Version

Indicates the WebSphere Application Server version you are running.

Security level

The current overall security level of the proxy server.

The overall security level is determined based on the custom security settings. The possible values for Security level are High, Medium, Low, and Not applicable. The overall security level is equal to the security level of the setting that is considered the least secure. For example, to have an overall security level of High, all settings must be configured to the values associated with a HIGH level of security. If any of the settings are configured with a less secure value, the overall security level is the value of that setting.

Protocol




Indicates the protocol or protocols that the proxy server is configured to handle. This information is based on the types of transport channels that are included in the transport chains that are configured for the proxy server.

For example, if a transport chain includes an HTTP channel, HTTP displays in this field. If a transport chain includes both a SIP and an HTTP channel, SIP,HTTP displays in this field.

Status

Indicates whether the proxy server is started, stopped, or unavailable.

If the status is unavailable, the node agent is not running in that node and you must restart the node agent before you can start the server.

	Started	The server is running.
	Stopped	The server is not running.
	Unknown	Status cannot be determined. A server with an unknown status might, in fact, be running but has an unknown status because the application server that is running the administrative console cannot communicate with this server.

Proxy server configuration

You can modify an existing proxy server to perform advanced routing options, such as routing requests to a non-WebSphere Application Server cell, and to perform caching. The options to configure the proxy server from this panel are under Proxy server properties.

To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers > proxy_server_name**

Name

Indicates a logical name for the proxy server. Proxy server names must be unique within a node.

Run in development mode

Enabling this option can reduce the startup time of a proxy server. This time can include Java Virtual Machine (JVM) settings, such as disabling bytecode verification and reducing just-in-time (JIT) compilation costs. Do not enable this setting on production servers.

Specify this option if you want to use the -Xverify and -Xquickstart JVM settings on startup. After selecting this option, save the configuration and restart the proxy server to activate development mode.

The default setting for this option is `false`, which indicates that the proxy server is not started in development mode. Setting this option to `true` specifies that the proxy server is started in development mode with settings that speed server startup time.

Data type Boolean

Default false

Parallel start

Select this field to start the proxy server on multiple threads. This option might shorten the startup time.

Specify this option if you want the proxy server components, services, and applications to start in parallel, rather than sequentially.

The default setting for this option is `true`, which indicates that the proxy server is started using multiple threads. Setting this option to `false` specifies that the server does not start using multiple threads which might lengthen startup time.

The order in which the applications start depends on the weights that you assigned to each of them. Applications that have the same weight are started in parallel. You set the weight of an application with the **Starting weight** option on the **Applications > Application Types > WebSphere enterprise applications > *application_name*** page of the administrative console.

Data type	Boolean
Default	true

Start components as needed

Select this check box if you want the server components started as they are needed by an application that is running on this server.

When this check box is selected, server components are dynamically started as they are needed. When this check box is not selected, all of the server components are started during the server startup process. Therefore, selecting this option can improve startup time, and reduce the memory footprint of the server, because fewer components are started during the startup process.

Starting components as they are needed is most effective if all of the applications, that are deployed on the server, are of the same type. For example, using this option works better if all of your applications are Web applications that use servlets, and JavaServer Pages (JSP). This option works less effectively if your applications use servlets, JavaServer Pages files and Enterprise JavaBeans™ (EJB).

Current Security Level

The current overall security level of the proxy server.

The overall security level is determined by evaluating the security settings. The possible values for Current Security Level are High, Medium, Low, and Not applicable. The overall security level is equal to the security level of the setting that is considered the least secure. For example, to have an overall security level of High, all settings must be configured to the values associated with a HIGH level of security. If any of the settings are configured with a less secure value, the overall security level is the value of that setting.

Proxy server settings

Use this topic to perform advanced configuration on a proxy server. Proxy settings enable the system administrator to fine tune the behavior of the proxy server. In particular, you can configure the connections and requests to the application server, enable caching, configure the requests that must be rejected, define how error responses are handled, and specify the location of the proxy logs.

The proxy server, upon creation, auto-senses the environment and is capable of routing requests to the product. Additional configuration can be applied to the proxy server to meet the needs of a particular environment.

To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers > *proxy_server_name* > HTTP Proxy Server Settings > Proxy settings**.

You can edit configurable field settings for the proxy server on the Configuration tab.

Enable Web services support

Specifies whether to enable the proxy server to route Web services traffic.

Data type Boolean
Default True

Static routing file directory

Specifies the directory where the static file is located.

Data type String
Default \${USER_INSTALL_ROOT}/staticRoutes

HTTP methods disabled

Specifies a list of HTTP methods that are disabled for the proxy server. Select the checkbox to enable this setting. Click **New** or **Delete** to add or remove HTTP methods from the list.

Data type String
Default Blank

Outbound connection settings

Specifies basic HTTP connection parameters between the proxy server and content servers.

Outbound request timeout

Specifies the default number of seconds the proxy server waits for a response before timing out a request to a content server. Consider this option carefully when changing the value.

Outbound connect timeout

Specifies the number of milliseconds that the proxy server waits to connect to a server. If this time expires, the proxy server attempts to connect to a different server. If no other available servers exist, the request times out. A value of 0 indicates that the proxy server should use the operating system kernel timeout value.

Pool connections to content server

Specifies the option to pool connections to the server is an optimization feature. Pooling prevents the need to frequently create and destroy socket connections to the server, by allowing the proxy server to pool these connections and reuse them.

Maximum connections per server

Specifies the maximum number of connections that will be pooled to any single content server.

Local outbound TCP address

Specifies the local outbound Transmission Control Protocol (TCP) address for data that enters and exits the SIP container. The value for this setting is the hostname or IP address to use for all communications between the SIP proxy and the SIP containers when the network is segmented.

Data type String
Default *
Range IP address or valid host name

The following proxy custom properties are available to adjust the outbound connections.

- key=http.maxTargetReconnects: Maximum number of reconnects to the same target content server for each request. The default is 5.
- key=http.maxTargetRetries: Maximum number of times the proxy will attempt to select a new target content server for each request. The default is 5.
- key=http.routing.sendReverseProxyNameInHost: Determines whether or not the host header is rewritten for content that is not on a WebSphere Application Server content server. The options are true or false

and are not case sensitive. If the value of this property is false, which is the default setting, then the host header is rewritten as the host of the target server. If the value for this property is true, then the host header is not rewritten.

- `key=http.compliance.disable`: Determines whether HTTP V1.1 compliance is enforced on proxy content server connections. The options are `true` or `false` and are not case sensitive. The default is `false`.
- `key=http.compliance.via`: The value of the `via` header that is appended to requests and responses for HTTP compliance. If the value is null, a `via` header will not be appended. If the value is true, a default `via` value is appended. Otherwise, the specified string `via` value is appended. The default is null.

Inbound connection SSL configuration

Specifies the SSL configuration from one of several sources.

Centrally managed

When selected, specifies to use the SSL configuration that is scoped for this endpoint.

Specific to this endpoint

When selected, enables the **Select SSL Configuration** list.

Select SSL Configuration

Specifies a predefined SSL configuration.

Data type	String
Default	None
Range	NONE, CellDefaultSSLSettings, or NodeDefaultSSLSettings

Caching

Specifies whether to enable the proxy server to cache the content of servers.

When **Enable caching** is selected, static content caching is enabled for the proxy server, as defined by HTTP 1.1 specifications. By default, caching content is enabled.

The properties that follow apply only if caching is enabled:

Cache instance name

Specifies the dynamic cache object cache instance that is configured in **Resources > Cache instances > Object cache instances**, which is used to cache all static and dynamic content responses. This object cache instance must be configured to support new I/O (NIO) application program interfaces (APIs).

Cache SSL content

Determines whether client proxy server SSL connections that are terminated by the proxy server should have their responses cached.

Cache aggressively

Enables caching of HTTP responses that would not normally be cached. Caching rules that are defined by HTTP 1.1 may be broken in order to gain caching optimizations.

Cache dynamic content

Specifies whether dynamic content that is generated by WebSphere Application Servers V6.02 or later is cached. Caching dynamic content generated by content servers prior to WebSphere Application Server V6.02 is not supported.

Limit memory cache entry size

When selected, the setting **Memory cache entry size** is enabled.

Memory cache entry size

Specifies the maximum size of an individual cached response in MB. Any cached response larger than this will not be cached.

Logging

The proxy server has logs that are generated for proxy and stored cache requests. When **Enable access logging** is selected, you can specify the size and location of the access logs.

Access log maximum size

Specify the maximum size, in megabytes, for an access log.

Data type	Integer
Units	Megabytes
Default	500

Proxy access log

Specifies a directory location for a proxy access log.

Data type	String
Default	\${SERVER_LOG_ROOT}/proxy.log

Cache access log

Specifies a directory location for a cache access log.

Data type	String
Default	\${SERVER_LOG_ROOT}/cache.log

Local access log

Specifies a directory location for a local access log.

Data type	String
Default	\${SERVER_LOG_ROOT}/local.log

Note: There is a log called `${SERVER_LOG_ROOT}/local.log` that logs locally served proxy content. This content is not in the proxy cache.

HTTP requests are logged in one of three logs: proxy, cache, and local. Local log configuration is not currently available in the administrative console, but it is available at `${SERVER_LOG_ROOT}/local.log`. Specify the location of this log by setting the `http.log.localFileName` custom property to the file location. The content of each log is formatted using National Center for Supercomputing Applications (NCSA) common log format.

- Proxy access log: Logs responses that are received from remote servers.
- Cache access log: Logs responses that are served from the local cache.
- Local access log: Logs all non-cache local responses, for example, redirects and internal errors.

Proxy custom properties that can be used to tweak logging are as follows:

- `key=http.log.disableAll`: This property disables all logging. A value of `true` stops proxy, cache, and local logging.
- `key=http.log.maxSize`: The maximum log size in megabytes (MB). A value of `UNLIMITED` indicates unlimited. 25 MB is the default.
- `key=http.log.localFileName`: Contains the name of the local log. A value of `NULL` indicates that the default `${SERVER_LOG_ROOT}/local.log` is used.

Security

Use this section to set up security options.

Use a proxy-masking server header

When selected, specifies to forward the content server's name to the client.

Use the backend server header

When selected, specifies the default server name is sent as the content server name.

Specify a server header value

When selected, the **Server header** setting is enabled.

Server header

Specifies the server name that is used in HTTP responses.

Trusted security proxies

Specifies intermediaries other than the proxy server to handle requests. This setting identifies which proxies can be trusted; for example, Web servers read incoming requests to verify which proxy they are routed to. Use an IP or fully qualified host name in this field.

Select the checkbox to enable **Security proxy**. Click **New** or **Delete** to add or remove proxies from the list.

Note: An empty list of trusted security proxies indicates that all WebSphere Application Server plug-in clients are trusted.

Data type	String
Default	Blank
Range	IP address or valid host name

Proxy plug-in configuration policy

Use this section to configure proxy plug-ins.

Generate plug-in configuration

Specifies the generation of a proxy plug-in configuration file that you can use on a Web server that is deployed in front of the proxy server. The plug-in can determine the URI that the proxy is handling on behalf of the application server. The plug-in can determine the endpoint, or boundaries of the proxy so that it can properly route requests that it receives to the proxy.

The options available to generate the plug-in are described in the following table:

Scope	Description
None	No scope.
All	The proxy server generates a plug-in configuration that includes all of the URIs that are handled by proxy servers in the local cell and all cells that are connected by a core group bridge.
Cell	The proxy server generates a plug-in configuration that includes all of the URIs that are handled by all the proxy servers in the cell.
Node	Includes all of the URIs that are configured for the node.
Server	The proxy server generates a plug-in configuration file only for the proxy server that is currently configured.

Plug-in config change script

Specifies the path to a script that is run after the WebSphere Application Server plug-in configuration is generated.

Custom error page policy

Use this section to configure settings for error pages when errors occur during the processing of a request.

The default is for no customized error pages to be generated.

Error page generation application URI

Specifies that if a valid uniform resource locator (URI) to an installed application is provided, the custom error page policy is enabled. If a valid URI to an installed application is not provided, the custom error page policy does not handle requests.

Handle remote errors

When selected, specifies HTTP response error status codes generated by the proxy server and HTTP response error status codes generated elsewhere after the proxy on the proxy content server connection error responses are handled. When not selected, only HTTP response error status codes generated by the proxy server are handled. A best practice is to configure an error page application on the same physical machine as the proxy server.

Headers to forward to error page application

Specifies additional header values from the client request to forward to the error page application as query parameters. The responseCode and URI query parameters are always sent to the error page application, in addition to the ones that are configured. The responseCode parameter is the HTTP status code that generates internally or is returned by the content server. The URI parameter is the request URI for the client.

Example - The error page URI is /ErrorPageApp/ErrorMessage, the headers to forward contain Host, and a client sends the following request:

```
GET /house/rooms/kitchen.jpg HTTP/1.1
Host: homeserver.companyx.com
```

The request results in a HTTP 404 response (local or remote), and the request URI to the error page application would be:

```
/ErrorPageApp/ErrorMessage?responseCode=404&uri=/house/rooms/kitchen.jpg&Host= homeserver.companyx.com
```

HTTP status codes that are to be recognized as errors

Specifies the status codes that the error page policy provides a response for. If a status code is not specified, the original content of responses with that status code are returned. If no HTTP status codes are specified, the defaults, 404 and 5XX, are used. Instead of specifying status codes individually, the following method is recommended to represent a range:

- 5XX: 500-599
- 4XX: 400-499
- 3XX: 300-399
- 2XX: 200-299

Proxy custom property to use when tweaking the custom error page:

key=http.statuscode.errorMessageRedirect. This custom property determines whether error page generation is done using the redirect, instead of using the proxy error page application. The values are true or false. The default is false.

Static file serving

Specifies the values needed for the proxy server to perform static file serving.

Static file document root

Specifies the location on the file system where the static document files are located.

Data type	String
Default	\${USER_INSTALL_ROOT}/staticContent

Content mappings

Specifies the content type mapping for a particular file extension. Specify a value for the following settings.

Extension	The subject file extension to map to a context type
-----------	---

Header	The header name to send to the client
Value	The value of the header to send to the client in the context-type header
Weight	A float value used to calculate the rank of files with this extension

Workload management

Specifies the values needed for the proxy server to perform workload management.

High availability monitor timeout

Specifies the amount of time, in seconds, before a high availability monitor timeout.

Data type	String
Units	Seconds
Default	300

Advisor URI

Specifies the uniform resource identifier (URI) for an advisor.

Data type	String
Default	/

Load balancing algorithm

Specifies the algorithm for the load balancer.

Data type	String
Default	Blank

Generic server clusters collection

Use this page to create, delete or modify a generic server cluster. Creating a generic server cluster is the next step towards generating the ability to route requests to a non-IBM WebSphere Application Server or a pre-Version 6.0 cell, after creating the proxy server.

To view this administrative console page, click **Servers > Clusters > Generic server clusters > .**

The system administrator can use the Generic Server Cluster panel to configure external servers, which are non-IBM WebSphere Application Server or a pre-V6 WebSphere Application Server, to create a logical cluster the proxy server can route work to. A generic server cluster defines the server endpoints that URI groups that are mapped to.

After you have created a generic server cluster, you want to create cluster members and define the cluster endpoints. Select the generic server cluster you created and click **Ports**.

Name

Specifies a logical name for the cluster. This is a user-defined field.

The name field cannot contain the following characters: # \ / , : ; " * ? < > | = + & % ' .

The name that defined must be unique among generic server clusters and cannot begin with a period or a space. A space does not generate an error, but leading and trailing spaces are automatically deleted.

Generic server clusters configuration

Use this topic to configure a generic server cluster. Creating a generic server cluster is the next step, after creating the proxy server, towards generating the ability to route requests to a non-IBM WebSphere Application Server or a pre-Version 6.0 WebSphere Application Server cell.

To view this administrative console page, click **Servers > Clusters > Generic server clusters > *cluster_name***.

Now that you have configured a generic server cluster, you can create cluster members and define the cluster endpoints. Click **Additional properties > Ports**.

You can edit generic server cluster configurable field settings on the Configuration tab.

Name

Specifies the user-defined name for the cluster.

Protocol

The protocol field determines whether or not secure communication is used when connecting to members of the cluster.

The choices are HTTP and HTTPS.

Generic server cluster ports collection

After defining the generic server cluster name, use this page to create, delete, and configure the members of the cluster.

To view this administrative console page, click **Servers > Clusters > Generic server clusters > *cluster_name*Ports**.

Host

The host name is either an IP address or the qualified host name of the cluster member. Specify a valid host name.

Port

Specify a valid port number to associate with the host.

Generic server cluster members

After defining the generic server cluster name, use this page to define the members of the cluster.

To view this administrative console page, click **Servers > Clusters > Generic server clusters > *cluster_name* > ports > *host_name***.

After you complete this task, you can create a URI group and then define routing rules.

You can edit generic server cluster member field settings on the Configuration tab.

Host

The host name is either an IP address or the qualified host name of the cluster member.

Specify a valid host name.

Port

Enter the port on which the host name is listening. This entry ensures that the proxy server can communicate with the cluster member.

Specify a valid port number.

Weight

The load balancing weight is used to determine how frequently the cluster member is routed, relative to the other members in the group. The higher the weight the more frequently the cluster member is routed to.

The recommended weight value is between 1 and 20. A value of 0 will result in a cluster member that will never be routed to.

URI groups

A group of URI patterns, which you define, that can be mapped back to generic server clusters. When creating a URI group, verify that you are planning for URIs that form a logical collection. From this topic, you can create, delete, or modify a URI group.

To view this administrative console page, click **Environment > URI Groups**.

Name

The URI group name is a user-specified name.

The name field cannot contain the following characters: # \ / , : ; " * ? < > | = + & % ' ,

The name that is defined must be unique among URI groups and cannot begin with a period or a space. A space does not generate an error, but leading and trailing spaces are automatically deleted.

URI group configuration

Use this topic to configure a URI group that can be mapped back to generic server clusters. When creating a URI group, ensure that you are planning for URIs that form a logical collection.

After you create a generic server cluster and a URI group, you are ready to define the routing rules that map the URI group to the generic server cluster. The routing rules ensure that the requests for specific URIs go to the proper generic server cluster.

To view this administrative console page, click **Environment > URI Groups > URI_group_name**.

You can edit URI groups configuration fields on the Configuration tab.

Name

The name field is required and is a user-defined field.

The name field cannot contain the following characters: # \ / , : ; " * ? < > | = + & % ' ,

The name that is defined must be unique among URI groups and cannot begin with a period or a space. A space does not generate an error, but leading and trailing spaces are automatically deleted.

URI pattern

Use this field to define URI patterns that constitute the URI group.

The patterns can be any valid URI and can contain one or more wildcard characters.

Routing rules

Use this topic to set the advanced configuration routing rules to ensure work requests arrive at the proper generic server cluster. From this topic you can create, delete, or modify a routing rule.

To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers > proxy_server_name > HTTP Proxy Server Settings > Routing Rules**.

Before you create routing rules, which are used to route requests to servers, you must define a generic server cluster (**Servers > Clusters > Generic server clusters**), a URI group (**Environment > > URI groups**), and optionally appropriate virtual hosts (**Environment > Virtual hosts > default host**).

Routing rules are used to assist the routing of work requests to non-IBM WebSphere Application Server nodes. In addition, using routing rules, a system administrator can reroute work without heavily impacting the environment. This capability is useful when nodes are taken down for maintenance.

For example, the system administrator can set up a routing rule to route `/images/*` to the ImageServerCluster generic server cluster. If the **ImageServerCluster** cluster has to come down, the administrator can then route `/images/*` to another cluster with similar capability, or use a redirect rule. This situation explains why the URI group can be defined independently of the generic server cluster. If the generic server cluster must come down, the URI group can be rerouted elsewhere. When you create the generic server cluster by providing a name, you can configure the cluster by using the ports link to create the actual cluster members.

Routing rules function by using the configured virtual hosts and URIs as matching criteria. The proxy server scans all incoming requests and compares the URI and host header from it and matches it against the virtual host and URIs that are configured in the rule. You must create the URI group for a routing rule before creating the routing rule. If you are routing to a generic server cluster, you must also create the cluster before defining the routing rule. You can create the URI group by completing the following tasks:

1. Create the routing rule name.
2. Determine if you want to enable this rule. You can create routing rules and not enable them. This capability is useful when planning for the maintenance of nodes or for emergency planning.
3. Select the virtual host name from the drop-down menu. The virtual host name field is a selectable field that is preconfigured with the defined virtual hosts in the cell. If you do not see the virtual host that you want in the menu, click **Environment > > Virtual hosts** and define the host there.
4. Select the URI group for the routing rule. The URI group field is populated with all the preconfigured URI groups in the cell. If you do not see the URI group that you are looking for, click **Environment > > URI groups** and create one.
5. Select and define a routing rule. This option specifies how to route a request that matches the defined virtual host and URI group. The three options for this field are:
 - Generic Server Cluster: Routes requests to a preconfigured generic server cluster. Use the drop-down box to select the generic server cluster.
 - Fail: Rejects requests by returning the specified HTTP status code.
 - Redirect: Redirects a client to the specified URL. This option can be used to ensure a request is routed through Secure Sockets Layer (SSL).

Name

The name field is required and is a user-defined field.

The name field cannot contain the following characters: `# \ / , : ; " * ? < > | = + & % ' ``

The name that is defined must be unique among routing rules and cannot begin with a period or a space. A space does not generate an error, but leading and trailing spaces are automatically deleted.

Routing rules configuration

Use this topic to create the advanced configuration routing rules to ensure that work requests arrive at the proper generic server cluster.

To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers > proxy_server_name > Proxy Server Properties > Routing Rules > rule_name**.

You can edit routing rules configurable field settings on the Configuration tab.

Name

The name field is required and is a user-defined field.

The name field cannot contain the following characters: # \ / , : ; " * ? < > | = + & % ' ,

The name that is defined must be unique among routing rules and cannot begin with a period or a space. A space does not generate an error, but leading and trailing spaces are automatically deleted.

Enable this rule

You can create routing rules and then not enable them. This capability is useful when planning for the maintenance of nodes or emergencies.

By default, the rule is enabled.

Virtual host name

The virtual host name field is a selectable field that is preconfigured with the defined virtual hosts in the cell.

If you do not see the virtual host that you are looking for in the menu, click **Environment > Virtual Hosts** from the administrative console and define the virtual host there.

URI group

The URI group field is populated with all the preconfigured URI groups in the cell.

If you do not see the URI group that you are looking for, click **Environment > URI Groups** and create one.

Routing action

This option specifies to the proxy server how to route a request that matches the given criteria (virtual host and URI group) that you defined. You have three options for this field.

Generic Server Cluster: If you want only the proxy server to seek a preconfigured generic server cluster, select that option and use the drop-down box to select the generic server cluster.

Failure Status Code: If you want to reject the requests that match the specific criteria, you use the failure status code and provide an HTTP status code to use in the response to the sender.

Redirect URL: Use this option to send a redirect to the client. If you select this option, enter a fully qualified URL like `http://abc.xyz.com`. Usually, the URL is somewhere within the enterprise, sometimes right back to the proxy on a different port. You can use this option to ensure a request is routed through protocols like Secure Sockets Layer (SSL).

Local route: Use this option to route requests to the root directory on the file system where static files are located.

Rewriting rules collection

Use this page to define how to rewrite URLs in a request or response.

To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers > proxy_server_name > HTTP proxy server settings > Rewriting rules**.

From URL pattern

Specifies the original URL pattern in the 302 response header from the target server.

To URL pattern

Specifies how that URL should be modified so that the client is redirected to the proxy server or an appropriate public URL.

Rewriting rules configuration

Use this topic to rewrite redirected URLs.

Rewriting rules define how the proxy server will rewrite URLs. The proxy server currently only supports rewriting redirected responses. Responses that have been redirected by target servers typically return a 302 status code with a location header that defines the URL that the client should be redirected to. Rewriting this URL is necessary if the target server is not aware of the proxy servers. The redirected URL is modified to correctly point clients to the proxy server instead of directly to a target server that may not be visible to clients.

To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers > proxy_server_name > HTTP proxy server settings > Rewriting rules > New**.

From URL Pattern

Specifies the original URL pattern in the 302 response header from the target server.

The pattern can include the following wild card symbol: *

To URL Pattern

Specifies how that URL should be modified so that the client is redirected to the proxy server or an appropriate public URL.

The pattern can include the following wild card symbol: *

A rule with a from URL pattern of `http://internalserver/*` and to url pattern of `http://publicserver/*` would cause a redirected response with the original location header of `http://internalserver/secure/page.html` to be rewritten as `http://publicserver/secure/page.html`.

HTTP proxy inbound channel settings

Use this page to view and configure an HTTP proxy inbound channel. This type of transport channel provides the HTTP proxy capabilities.

To view this administrative console page, click **Servers > Proxy Servers > server_name > HTTP Proxy Server Settings > Proxy server transports > HTTP_PROXY_CHAIN > HTTP_proxy_inbound_channel_name**.

Transport channel name

Specifies the name of the HTTP proxy inbound channel.

The name field cannot contain the following characters: # \ / , : ; " * ? < > | = + & % ' .

This name must be unique across all channels in a WebSphere Application Server environment. For example, an HTTP proxy inbound channel and a TCP transport channel cannot have the same name if they reside within the same system.

Data type

String

Discrimination weight

Specifies the priority this channel has in relation to the other channels in this transport chain. This property is only used when port sharing is enabled and the channel chain includes multiple channels to which it might forward data. The channel in the chain with the lowest discrimination weight is the first one given the opportunity to look at incoming data and determine whether or not it owns that data.

Data type	Positive integer
Default	0

Starting a proxy server

Starting a proxy server starts a new server process based on the process definition settings of the current proxy server configuration.

Before you begin

Before you start a proxy server, verify that all of the resources that the proxy requires are available. You must also start all prerequisite subsystems.

About this task

This procedure for starting a server also applies to restarting a server. However, if a server fails and you want the recovery functions to complete their processing prior to new processes being started on the server, you must restart the server in recovery mode.

Use one of the following options to start a proxy server.

- For the z/OS and distributed platforms, except AIX, you can issue the startServer command from the command line to start a single proxy server.

You can issue the startServer command from the C:\WebSphere\AppServer\profiles\AppSrv02\bin directory.

```
# .\startServer.sh proxyserver1
```

AIX You can issue the startServer command from the /usr/WebSphere/AppServer/bin directory.

```
# ./startServer.sh proxyserver1
```

- You can use the administrative console to start a proxy server.
 1. In the administrative console, click **Servers > Server Types > WebSphere proxy servers > .**
 2. Select the proxy server that you want to start, and click **Start**.
 3. Confirm that you want to start the proxy server.
 4. View the **Status** value and any messages or logs to see whether the proxy server was started.

Results

The specified proxy server starts. To verify that the proxy server was started, in the administrative console, click **Servers > Server Types > WebSphere proxy servers > .**

What to do next

If you need to start the proxy server with standard Java debugging enabled, then complete these steps:

1. From the administrative console, expand **Servers > Server Types > WebSphere proxy servers > proxy_server_name**.
2. From Server Infrastructure, click **Java and process management > Process definition**.
3. Click **Java virtual machine**.

4. Select **Debug mode** to enable the standard Java debugger. Set **Debug mode** arguments, if needed, and click **OK**.
5. Save the changes to a configuration file.
6. Stop the proxy server, and restart the proxy server.

Stopping a proxy server

Stopping a proxy server ends a server process based on the process definition settings in the current application server configuration.

Before you begin

Ensure that you understand how stopping a particular server affects your ability to handle work requests, especially if you need to maintain a highly available environment.

About this task

There are times you need to stop a proxy server. For example, you might want to upgrade the operating system, or you might want to change a configuration setting for the proxy server. You can use one of the following options to stop a proxy server.

Note: To perform a proxy quiesce for your Session Initiation Protocol (SIP) proxy server, you must shut down the SIP proxy server by issuing the stopServer command from the command line. If you attempt to shut down the proxy server from the administrative console, then the server shuts down immediately and the proxy quiesce is not completed.

- You can issue the stopServer command from the command line to stop a single proxy server. You can issue the stopServer command from the C:\WebSphere\AppServer\profiles\AppSrv02\bin directory.

```
# .\stopServer.sh proxysvr1
```

AIX You can issue the stopServer command from the /usr/WebSphere/AppServer/bin directory.

```
# ./stopServer.sh proxysvr1
```

- You can use the administrative console to stop a proxy server.
 1. From the administrative console, click **Servers > Server Types > WebSphere proxy servers**.
 2. Select the proxy server, and click **Stop**.
 3. Confirm that you want to stop the selected proxy server.
 4. View the **Status** value and any messages or logs to see whether the proxy server stops.

Results

The specified proxy server stops as soon as requests assigned to that server finish processing. To verify that the proxy server is in the stop state, in the administrative console, click **Servers > Server Types > WebSphere proxy servers**.

Note: If the stopServer command is issued from the command line, then the server delays shutdown for a period of time until new inbound messages to route are no longer being received. The quiesce feature notifies the load balancer to discontinue routing inbound messages by sending error responses to the advisor messages.

What to do next

If errors occur when you shut down a server, then read the *Troubleshooting and support* PDF.

By default, the SIP proxy server stops the flow of messages between the load balancer and the back-end containers to prevent calls from being lost when the proxy server shuts down. This process is called a proxy quiesce.

During proxy quiesce, the SIP proxy server notifies the load balancer and the back-end containers that the server is shutting down. After the devices stop forwarding messages through the proxy server, the server shuts down.

The default quiesce timeout period is three minutes. The SIP proxy server also waits a minimum of 20 seconds to allow the quiesce process to complete. The SIP proxy server continues to forward messages to the back-end containers while it responds to advisor messages from the load balancer with an error response. During a quiesce, the SIP proxy server also notifies the back-end containers that the proxy server is no longer a member of the cluster. After the initial 20 seconds, the SIP proxy server shuts down based on the specified amount of time configured for the proxy quiesce, which ranges from one second to a maximum of three minutes.

Complete these steps if you want to change the timeout period for proxy quiesce.

1. From the administrative console, expand **Servers > Server Types > WebSphere proxy servers > proxy_server_name**.
2. From Server infrastructure, click **Java and Process management > Process definition**.
3. Click **Java virtual machine**.
4. Set the Generic JVM argument to `-Dcom.ibm.ejs.sm.server.quiesceTimeout=120`.
5. Define these SIP proxy custom properties to set the SIP proxy server to be fronted by a load balancer running a SIP advisor: **LBIPADDR** and **SIPAdvisorMethodName**.

HTTP proxy server custom properties

You can add the following custom properties to the configuration settings for an HTTP proxy server.

To specify custom properties for a specific HTTP proxy server, navigate to the custom properties page, and specify a value for the custom property.

1. In the administrative console, expand **Servers > Server Types > WebSphere proxy servers > proxy_server_name** to open the configuration tab for the server.
2. From **HTTP proxy server settings**, expand **HTTP proxy settings**, and click **Custom properties**.
3. From **Additional properties**, select **Custom properties** → **New**.
4. On the settings page, type the custom property to configure in the **Name** field, and type the value of the custom property in the **Value** field.
5. Click **Apply** or **OK**.
6. Click **Save** on the console task bar to save your configuration changes.
7. Restart the server.

The following list of HTTP proxy server custom properties is provided with the product. These properties are not shown on the settings page for an HTTP proxy server.

localOutboundTCPAddress

Specifies the host interface that is dedicated to HTTP traffic. This property determines the interface that is used to make outbound HTTP connections to the HTTP container.

Data type	String
Default	*

http.clientInfoFromTrustedIntermediary

Specifies whether the proxy should extract the IP address from a WAS private header in a request. When the plug-in is deployed in front the proxy, the proxy server extracts the client IP address from the channels instead of from private headers forwarded from the plug-in. If this property is set to true, the proxy server extracts client information from private HTTP headers sent from the trusted plug-in instead of the channels.

Data type	String
Default	false

http.connectionPoolUseForPOST

Specifies whether the proxy server uses connection pooling for POST requests. POST requests, by default, are neither pooled, nor persistent requests. Therefore, if excessive POST requests are sent through the proxy server, port exhaustion might occur, resulting in bind exceptions. If this property is set to true, connection pooling is used for POST requests.

Data type	boolean
Default	false

http.odcUpdateTimeout

Specifies the amount of time, in seconds, for the HTTP proxy server to wait during server startup before routing information. The proxy server waits for the specified number of seconds before binding its ports.

This custom property can be used to configure a delay in startup (before the HTTP/HTTPS ports are bound) for a period of time to allow for the propagation of the routing information. If you set the value to 300, the proxy server waits for 300 seconds to allow time for the routing information to propagate to the proxy server. If the routing information is propagated to the proxy server before 300 seconds, server startup resumes.

Data type	String
Default	150

http.routing.sendReverseProxyNameInHost

Determines whether or not the host header is rewritten for content that is not on a WebSphere Application Server content server.

The options for this property are true or false and are not case sensitive. If the value of this property is false, then the host header is rewritten as the host of the target server. If the value for this property is true, then the host header is not rewritten.

Data type	Boolean
Default	false

SIP proxy server custom properties

You can add the following custom properties to the configuration settings for the Session Initiation Protocol (SIP) proxy server.

To specify custom properties for a specific SIP proxy server, navigate to the custom properties page, and specify a value for the custom property.

Note: The custom properties are supported as the primary method of configuration. Therefore, if a custom property is set and then you set the corresponding setting in the administrative console, the custom property value is used.

1. In the administrative console, expand **Servers > Server Types > WebSphere application servers > *server_name*** to open the configuration tab for the server.
2. From **SIP proxy server settings**, expand **SIP proxy settings**, and click **Custom properties**.
3. From **Additional properties**, select **Custom properties** → **New**.
4. On the settings page, type the custom property to configure in the **Name** field, and type the value of the custom property in the **Value** field.
5. Click **Apply** or **OK**.
6. Click **Save** on the console task bar to save your configuration changes.
7. Restart the server.

The SIP proxy custom properties listed here are provided with the product. These properties are not shown on the settings page for a proxy server.

Overload custom properties

Several of the SIP custom properties allow you to apply proxy-managed overload protection (PMOP). The PMOP overload settings allow for real-time protection against container overload.

- burstResetFactor
- deflatorRatio
- dropOverloadPackets
- inDialogAveragingPeriod
- maxThroughputFactor
- outDialogAveragingPeriod
- perSecondBurstFactor
- proxyTransitionPeriod
- sipProxyStartupDelay
- overloadResponseCode
- overloadResponseReasonPhrase

For more information on overload controls, refer to the Information Center topic *Session Initiation Protocol overload protection*.

burstResetFactor

Specifies the percentage of bursts during a given period of time. This custom property controls the amount of bursts that occur during the averaging period.

Data type	String
Default	100

deflatorRatio

Specifies a static ratio. This custom property is only used during the transition period when a transition period is specified.

Data type	String
Default	0

dropOverloadPackets

Specifies whether or not to drop packets when the SIP container is in an overloaded state. When this value is set to `False`, the proxy server responds with a 503 error when overloaded, otherwise the packet is dropped.

Data type	String
------------------	--------

Default False

inDialogAveragingPeriod

Specifies the period of time, in seconds, during which in-dialog messages are averaged.

Data type String
Default 180

maxThroughputFactor

Specifies the percentage of the maximum number of messages per averaging period set. If this value is set to 0, then the maximum throughput feature is disabled. This custom property is used to calculate the maximum number of messages allowed per second before the proxy server begins to reject requests for new sessions. This custom property should be set to the same value for each proxy server.

Data type String
Default 0

outDialogAveragingPeriod

Specifies the period of time, in seconds, during which out-dialog messages are averaged.

Data type String
Default 360

perSecondBurstFactor

Specifies the percentage of bursts allowed through periodically (BTF).

Data type String
Default 150

proxyTransitionPeriod

Specifies the period of time, in seconds, to lock the deflator after the container shuts down.

Data type String
Default 0

sipProxyStartupDelay

Specifies the period of time, in seconds, before the proxy server restarts to allow the proxy to become stable in the cluster and avoid an erroneous overloaded state.

Data type String
Default 0

overloadResponseCode

Specifies the value for the response code returned from the proxy when overload occurs and SIP requests from the container are rejected. When the proxy is configured for overload protection, the SIP proxy can be configured to detect overload status. The proxy monitors the amount of traffic processed at the proxy and limits the number of new requests. If a container is overloaded, the proxy rejects requests with a 503 response code. If you prefer to use a different response code for overload protection, you can configure this custom property to return a different response code.

Data type String
Default 503

overloadResponseReasonPhrase

Specifies the value for the response reason phrase provided from the proxy when overload occurs and SIP requests from the container are rejected. When the proxy is configured for overload protection, the SIP proxy can be configured to detect overload status. The proxy monitors the amount of traffic processed at the proxy and limits the number of new requests. If a container is overloaded, the proxy rejects requests with a Service Unavailable response phrase. If you prefer to use a different response phrase, you can configure this custom property to return a different response phrase.

Data type	String
Default	Service Unavailable

General custom properties

The following SIP custom properties allow you to apply a variety of settings.

isSipComplianceEnabled

Specifies whether or not compliance checking is enabled. Compliance checking might be disabled if a client is not compliant with a specification, such as RFC 3261, and you do not want to log an interoperability event.

Data type	String
Default	True

keepAliveInterval

Specifies the interval, in milliseconds, at which keepalive messages are sent to the SIP containers. A keepalive message is sent at the specified interval. If the specified number of **keepAliveFailures** messages is received from the SIP container, the proxy considers the container to be down. The proxy then routes data to a back-up SIP container until the connection between the proxy and the primary container is restored.

The first keepalive message contains the interval of time between the keep alive messages and the number of failures that are required before the container is considered down. The starting values should be specified based on the high availability (HA) heartbeat configuration.

Data type	String
Default	0

keepAliveFailures

Specifies the number of keepalive messages that must be missed before the proxy determines that the connection with the SIP container is down.

The proxy sends a keepalive message to the container at each **keepAliveInterval**. If the proxy does not receive a response to the message, it considers the lack of response as a failure. If the proxy receives a specific number of consecutive failures, it considers the container down and begins forwarding messages to a different SIP container.

Data type	String
Default	0

LBIPAddr

Specifies the IP address, such as 192.101.1.5, of the load balancer used to load balance the SIP proxy. Multiple load balancer addresses can be configured by separating each IP address using a semicolon (;).

When SIP messages with the method configured as `SIPAdvisorMethodName` are received by the SIP proxy from the specified IP address, the SIP proxy responds with a success message if the SIP proxy can forward the messages to the SIP container. The SIP proxy responds with a failure message if messages cannot be forwarded to the SIP container. The load balancer then determines if the messages should be routed to the SIP proxy.

Data type String
Default null

localOutboundTCPAddress

Specifies the source interface to which the proxy binds when establishing connections to back-end SIP containers. This property is used when your proxy server is multihomed and needs to be configured to use a specific interface to send SIP traffic to the SIP containers. This property applies to both Transmission Control Protocol (TCP) and Transport Layer Security (TLS) connections.

Data type String
Default *

maxWriteQueueEntries

Specifies the number of messages that can be queued when a connection is slow or cannot be established. If the value is a large number, then more memory is consumed. A small number causes packets to be lost if the endpoint clears.

Data type String
Default 100000

receiveBufferSizeChannel

Specifies the value, in bytes, for the maximum size of an incoming UDP packet, which is the size of the receive buffer that is allocated in the proxy server-side UDP connection.

Data type String
Default 65535

receiveBufferSizeSocket

Specifies the value, in bytes, for the lower-level datagram buffers, which is the size of the `DatagramSocket` receive buffer (`SO_RCVBUF`) in the proxy server-side User Datagram Protocol (UDP) connection.

Use this property to buffer multiple packets in the `DatagramSocket` layer. If the value of the property is too small, then packets might be lost if the server is overloaded. If the value is too large, then the packets might be delayed.

Data type String
Default 1024000

retryAfterValue

Specifies the amount of time, in seconds, before the client tries again to route a request to the proxy server. This custom property value is returned to the client in the error response if the SIP container is overloaded or if the SIP proxy cannot locate a server to which to route a request.

Data type String
Default 5

sendBufferSizeSocket

Specifies the value, in bytes, for the lower-level datagram buffers, which is the size of the DatagramSocket send buffer (SO_SNDBUF) in the proxy server-side UDP connection.

Use this property to buffer multiple packets in the DatagramSocket layer. If the value of the property is too small, then packets might be lost if the server is overloaded. If the value is too large, then the packets might be delayed.

Data type	String
Default	1024000

serverUDPInterface

Specifies the host name or IP address that is used for all communications between the SIP proxy and the SIP containers when the network is segmented. This interface is the specific network interface for all UDP data that enters or exits the SIP containers. You must use this property with the serverUDPPort property.

Data type	String
Default	*

serverUDPPort

Specifies the UDP port that is used for SIP container communications. When the firewall is between the SIP proxy and the SIP container, you might set this value if a specific interface is needed to communicate with the SIP containers or if a specific port is required to pass through the firewall.

Data type	String
Default	dynamic

SIPAdvisorMethodName

Specifies a string value for the method sent by the load balancer to the SIP proxy for health checks.

The format is OPTIONS or INFO. This property is used with the LBIPAddr property.

Data type	String
Default	null

SIP container custom properties

You can add any of the following custom properties to the configuration settings for a Session Initiation Protocol (SIP) container.

To specify custom properties for a specific SIP container, navigate to the custom properties page, and then specify a value for the custom property.

Note: The custom properties are supported as the primary method of configuration. Therefore, if a custom property is set and then you set the corresponding setting in the administrative console, the custom property value is used.

1. In the administrative console, expand **Servers > Server Types > WebSphere application servers > *server_name*** to open the configuration tab for the server.
2. From **Container settings**, expand **SIP Container settings**, and click **SIP container**.
3. From **Additional properties**, select **Custom Properties** → **New**.
4. On the settings page, type the custom property to configure in the **Name** field, and then type the value of the custom property in the **Value** field.
5. Click **Apply** or **OK**.

6. Click **Save** on the console task bar to save your configuration changes.
7. Restart the server.

The following list of SIP container custom properties is provided with the product. These properties are not shown on the settings page for the container.

enable.system.headers.modify

Specifies whether the application has access to headers that are otherwise restricted.

Data type	String
Default	False

replicate.with.confirmed.dialog.only

Specifies whether to replicate the application session, even when no dialogs are confirmed. If the value is set to `false`, then the application session is replicated immediately after the session is created. Otherwise, the application session is only replicated when an associated dialog is confirmed.

Data type	String
Default	False

com.ibm.sip.sm.lnm.size

Specifies the number of logical names in the application server. Each SIP object that can be replicated, such as a SIP session, is associated with a logical name. All objects with the same logical name are replicated to the same back-up container. The proxy can route messages to the correct container using the logical name found in the message. The value must be greater than 1.

Data type	String
Default	10

auth.int.enable

Specifies the **auth-int** quality of protection (QOP) for digest authentication. Digest authentication defines two types of QOP: **auth** and **auth-int**. By default, **auth** is used. When this custom property is set to `True`, the highest level of protection is used, which is the **auth-int** QOP.

Data type	String
Default	False

DigestPasswordServerClass

Specifies the Java class name that implements the PasswordServer interface.

Data type	String
Default	LdapPasswordServer

com.ibm.websphere.sip.security.tai.usercachecleanperiod

Specifies the clean security subject cache period in minutes.

Data type	String
Default	15

com.ibm.ws.sip.tai.DisableSIPBasicAuth

Specifies whether to allow basic authentication for SIP.

Data type	String
------------------	--------

Default False

com.ibm.websphere.sip.security.digest.ldap.cachecleanperiod

Specifies the clean Lightweight Directory Access Protocol (LDAP) cache period in minutes.

Data type String

Default 120

pws_atr_name

Specifies the LDAP attribute name that stores the user password.

Data type String

Default userpassword

javax.sip.transaction.invite.auto100

Specifies whether to automatically reply to invite requests with a 100 Trying response. Disabling this property might increase the number of invite retransmissions.

Data type String

Default True

Note: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core group that contains a mixture of Version 7.0 and Version 6.x processes.

Note: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use this custom property.

javax.sip.force.connection.reuse

Specifies whether to force reuse of inbound connections for outbound requests. This custom property is only relevant for stream transports, such as Transmission Control Protocol (TCP) and Transport Layer Security (TLS). Disabling this property causes the container to create a separate connection for outbound requests, even if an existing connection is already established to the same peer address. The connection is automatically reused if the top Via header in the inbound request contains an alias parameter. (<http://www.ietf.org/internet-drafts/draft-ietf-sip-connect-reuse-07.txt>)

Data type String

Default TrueFalse

Note: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core group that contains a mixture of Version 7.0 and Version 6.x processes.

Note: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use this custom property.

javax.sip.hide.message.body

Specifies to hide message content in logs. Set the value of this property to true to remove the message body text from SIP messages printed in the log files. This property only affects the representation of the messages in log files.

Data type String

Default False

Note: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core group that contains a mixture of Version 7.0 and Version 6.x processes.

Note: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use this custom property.

javax.sip.PATH_MTU

Specifies the maximum transmission unit, in bytes, for outbound User Datagram Protocol (UDP) requests. The SIP stack measures the size of a request before sending it out on the UDP channel. If the request is larger than the value specified for **PATH_MTU-200** (1300 bytes by default), then the transport is switched from UDP to TCP before transmission. Increase this value to send larger requests over the UDP channel; however, messages might be truncated or dropped. See the RFC 3261-18.1.1 specification for details.

Data type	String
Default	1500

Note: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core group that contains a mixture of Version 7.0 and Version 6.x processes.

Note: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use this custom property.

javax.sip.OUTBOUND_PROXY

Specifies the fixed address for routing all outbound SIP messages. The format is *address:port/transport*, such as *1.2.3.4:5065/tcp*.

Note: Do not use this property if the container is fronted by an application server SIP proxy.

Data type	String
Default	null

Note: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core group that contains a mixture of Version 7.0 and Version 6.x processes.

Note: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use this custom property.

javax.sip.trace.msg.in

Specifies whether to print incoming messages to a system.out file.

Data type	String
Default	False

javax.sip.trace.msg.out

Specifies whether to print outbound messages to a system.out file.

Data type	String
Default	False

javax.sip.stat.report.interval

Specifies the amount of time, in milliseconds, for reporting dispatch and timer statistics to a system.out file. A value of zero indicates no report.

Data type	String
Default	0

javax.sip.detect.pre.escaped.params

Specifies whether to prevent the container from re-escaping Uniform Resource Identifier (URI) parameters that were pre-escaped by the application.

Enabling this property provides the application with more control over escaping URI parameters, when calling the **javax.servlet.sip.SipFactory.createURI()** and the **javax.servlet.sip.SipURI.setParameter()** parameters.

By default, the container only escapes characters that it must encode according to the RFC 3261 25.1 specification. In some cases, however, escaping additional characters might be required. Due to a limitation in the JSR 116 (5.2.1) specification, the application cannot perform its own escaping. Because of this limitation, attempts by the application to encode URI parameters causes the container to re-encode the percent sign. If the value of this property is set to true, the container cannot re-encode the percent sign.

Setting the value to true is not in compliance with the JSR 116 (5.2.1) specification, but provides the application with greater control over URI parameter escaping. APAR PK37192 describes the problem and the workaround.

Data type	String
Default	False

DigestPasswordServerClass

Specifies types of user registries that are supported, except LDAP. To configure DigestTAI without the LDAP user registry, complete the following steps.

1. Create a class that implements this interface: `com.ibm.ws.sip.security.digest.DigestPasswordServer` 1
2. Add these properties to the `SlpDigestTAI` custom property.
 name =
 DigestPasswordServerClass value =
3. Ensure that all users declared by the **impl** class are declared in the user registry configured for the product security.

Data type	String
Default	impl

javax.sip.bind.retries

Specifies the amount of time, in milliseconds, between attempts to start the SIP channel if the SIP port is busy with another process during server startup.

Data type	String
Default	60

javax.sip.bind.retry.delay

Specifies the delay, in milliseconds, between attempts to start the SIP channel if the SIP port is busy with another process during server startup.

Data type	String
Default	5000

javax.sip.transaction.timer.t1

Specifies the amount of time, in milliseconds, for a network round trip delay for timer T1 for the RFC 3261 specification. The value is used as a base for calculating some timers and is relevant for all types of transactions, such as client, server, invite, and non-invite transactions.

Data type	String
Default	500

Note: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core group that contains a mixture of Version 7.0 and Version 6.x processes.

Note: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use this custom property.

javax.sip.transaction.timer.t2

Specifies the maximum time in milliseconds before retransmitting non-invite requests and invite responses for timer T2 for the RFC 3261 specification.

Data type	String
Default	4000

Note: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core group that contains a mixture of Version 7.0 and Version 6.x processes.

Note: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use this custom property.

javax.sip.transaction.timer.t4

Specifies the maximum amount of time, in milliseconds, for a message to remain in the network. This value is used as a base for calculating other timers for timer T4 for the RFC 3261 specification.

Data type	String
Default	5000

Note: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core group that contains a mixture of Version 7.0 and Version 6.x processes.

Note: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use this custom property.

javax.sip.transaction.timer.a

Specifies, for UDP only, the amount of time, in milliseconds, before retransmitting invite requests for timer A for the RFC 3261 specification. This property is relevant for the invite client transaction.

Data type	String
Default	javax.sip.transaction.timer.t1

Note: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core group that contains a mixture of Version 7.0 and Version 6.x processes.

Note: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use this custom property.

javax.sip.transaction.timer.b

Specifies the amount of time, in milliseconds, for the invite client transaction timeout timer (timer B) for the RFC 3261 specification.

Data type	String
Default	64*javax.sip.transaction.timer.t1

Note: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core group that contains a mixture of Version 7.0 and Version 6.x processes.

Note: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use this custom property.

javax.sip.transaction.timer.d

Specifies the wait time, in milliseconds, before retransmission of the invite response for timer D for the RFC 3261 specification. This property is relevant for the invite client transaction.

Data type	String
Default	32000

Note: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core group that contains a mixture of Version 7.0 and Version 6.x processes.

Note: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use this custom property.

javax.sip.transaction.timer.e

Specifies, for UDP only, the amount of time, in milliseconds, before the retransmission of the initial non-invite request for timer E for the RFC 3261 specification. This property is relevant for the non-invite client transaction.

Data type	String
Default	javax.sip.transaction.timer.t1

Note: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core group that contains a mixture of Version 7.0 and Version 6.x processes.

Note: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use this custom property.

javax.sip.transaction.timer.f

Specifies the amount of time, in milliseconds, for the non-invite transaction timeout timer (timer F) for the RFC 3261 specification. This property is relevant for the non-invite client transaction.

Data type String
Default 64*javax.sip.transaction.timer.t1

Note: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core group that contains a mixture of Version 7.0 and Version 6.x processes.

Note: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use this custom property.

javax.sip.transaction.timer.g

Specifies the amount of time, in milliseconds, before retransmission of an initial invite response for timer G for the RFC 3261 specification. This property is relevant for the invite server transaction.

Data type String
Default javax.sip.transaction.timer.t1

Note: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core group that contains a mixture of Version 7.0 and Version 6.x processes.

Note: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use this custom property.

javax.sip.transaction.timer.h

Specifies the amount of time, in milliseconds, to wait for an acknowledgement (ACK) receipt for timer H for the RFC 3261 specification. This property is relevant for the invite server transaction.

Data type String
Default 64*javax.sip.transaction.timer.t1

Note: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core group that contains a mixture of Version 7.0 and Version 6.x processes.

Note: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use this custom property.

javax.sip.transaction.timer.i

Specifies the amount of time in milliseconds to wait for an ACK retransmission for timer I for the RFC 3261 specification. This property is relevant for the invite server transaction.

Data type String
Default javax.sip.transaction.timer.t4

Note: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core group that contains a mixture of Version 7.0 and Version 6.x processes.

Note: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use this custom property.

javax.sip.transaction.timer.j

Specifies the amount of time in milliseconds to wait for non-invite request retransmission for timer J for the RFC 3261 specification. This property is relevant for the non-invite server transaction.

Data type	String
Default	64*javax.sip.transaction.timer.t1

Note: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core group that contains a mixture of Version 7.0 and Version 6.x processes.

Note: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use this custom property.

javax.sip.transaction.timer.k

Specifies the amount of time, in milliseconds, to wait for non-INVITE response retransmissions for timer K for the RFC 3261 specification. This property is relevant for the non-invite client transaction.

Data type	String
Default	javax.sip.transaction.timer.t4

Note: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core group that contains a mixture of Version 7.0 and Version 6.x processes.

Note: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use this custom property.

javax.sip.transaction.timer.non.invite.server

Specifies the amount of time, in milliseconds, for an Application Programming Interface (API) timer for the application to respond to a non-invite request. This property is relevant for non-invite server transactions.

This timer is not defined in the RFC specification. This property is needed to stop the transaction if the application does not generate a final response to the request. The timer starts when the request arrives in the stack and stops when a response is generated by the application. If no response is generated before the timer stops, then the transaction completes.

Data type	String
Default	34000

javax.sip.transaction.timer.invite.server

Specifies the amount of time, in milliseconds, for the timer to keep the invite server transaction in the complete state. This timer is not defined in the RFC specification.

To avoid creating a new server transaction when a client retransmits an invite request, keep the completed server transaction for a period of time before removing invite retransmissions. This timer is started when the transaction changes to the terminated state. When the timer completes, the transaction is removed.

Data type	String
Default	32000

javax.sip.transaction.timer.cancel

Specifies the amount of timer, in milliseconds, for the timer to keep the cancelled client transaction in the proceeding state before completing the cancelled transaction for the RFC 3261 9.1 specification. This property is relevant for the invite client transaction.

Data type	String
Default	64*javax.sip.transaction.timer.t1

SIP_RFC3263_nameserver

Specifies whether to allow a SIP URI to be resolved through Domain Name System (DNS) into the IP address, port, and transport protocol of the next hop.

The value of the property is a string containing one or two address and port tuples, where two tuples are separated by a space. The following examples specify a one address and port tuple or a two address and port tuple.

dottedDecimalAddress@.port
hostname.domain@port
IPV6address@port

The following example values represent a single tuple.

- 1.2.3.4@53
- example.com@53
- a:b:c::d@53

The following example values represent two tuples separated by a space.

- 1.2.3.4@53 example.com@53
- a:b:c::d@53 9.32.211.14@53

Data type	String
Default	null

Note: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core group that contains a mixture of Version 7.0 and Version 6.x processes.

Note: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use this custom property.

thread.message.queue.max.size

Specifies the maximum number of events allowed in the container threads queue. When this number is exceeded, the proxy server is notified that the container is overloaded and requests for new sessions are not accepted. Instead, the container returns an error message that indicates that the container is temporarily unavailable.

This value represents the total number of messages for all queues and reflects the state of the CPU. When the CPU approaches 100%, the maximum value for this custom property is reached quickly. Configure your system to limit the queue size and prevent the queue from reaching this threshold.

Data type	String
Default	1000

weight.overload.watermark

Specifies the threshold value for the internal weight calculated by the container. When the container calculates the internal weight to be higher than the value specified, an overloaded container becomes available for service again.

This custom property represents a percentage of the maximum internal weight, such as 30 percent when the default value is set. When the high-water mark, or maximum threshold, is exceeded, the container waits until the weight drops below the maximum weight. This value cannot exceed 10.

Data type	String
Default	3

sip.container.heartbeat.enabled

Specifies whether or not SIP network outage detection is enabled for the SIP container. SIP network outage detection allows the SIP proxy to send keepalive messages to the SIP container if the value of this property is set to `true`.

If the value is set to `false` for the SIP container, then this property has no effect on the SIP proxy. However, if the value is set to `true` for the SIP container, the value should also be set to `true` for the SIP proxy to ensure that keepalive messages are answered at the SIP container and not presented to the application.

Data type	String
Default	true

Creating a proxy server cluster

A proxy server cluster consists of a group of proxy servers. You can create a cluster of proxy servers that can route requests to applications in a cell.

Before you begin

Version 7 of the product must be installed on the nodes that will belong to the proxy server cluster. You must also know whether DNS, Load Balancer, or proxy will be used to route requests to the proxy server cluster.

About this task

To create a proxy server cluster in the administrative console, click **Servers > Clusters > Proxy server clusters > New** to start the Create a new proxy server cluster wizard.

Results

You have successfully created a proxy server cluster.

What to do next

Click **Servers > Clusters > Proxy server clusters**, select the newly created cluster, and then click **Start** to start the cluster, or click on the name of the new cluster to change its configuration settings, or add additional members to the cluster.

Proxy server cluster collection

Use this page to view information about and change configuration settings for a proxy server cluster. A proxy cluster consists of a group of proxy servers.

To view this administrative console page, click **Servers > Clusters > Proxy server clusters** .

To define a new proxy server cluster, click **New** to start the Create a new proxy server cluster wizard.

Name

Specifies a logical name for the proxy cluster. The name must be unique among proxy clusters within the containing cell.

Supported Protocols






Indicates the protocol or protocols that the proxy server is configured to handle.

This information is based on the types of transport channels that are included in the transport chains that are configured for the proxy server. For example, if a transport chain includes an HTTP channel, HTTP displays in this field. If a transport chain includes both a SIP and an HTTP channel, SIP,HTTP displays in this field.

Status

This field indicates whether the proxy cluster is partially started, started, partially stopped, stopped, or unavailable. If the status is unavailable, the node agent is not running in that node and you must restart the node agent before you can start the proxy server.

After you click **Start** or **Ripplestart** to start a proxy cluster, each server that is a member of that proxy cluster launches if it is not already running. When the first member launches, the status changes to partially started. The status remains partially started until all proxy cluster members are running. When all proxy cluster members are running, the state changes to running and the status changes to started. Similarly, when you click **Stop** or **ImmediateStop** to stop a proxy cluster, the status changes to partially stopped when the first member stops, then changes to stopped when all proxy cluster members are not running.

	Started	All of the proxy cluster members are running.
	Partially started	At least one of the proxy cluster members is running.
	Stopped	All of the proxy cluster members have stopped running.
	Partially stopped	At least one of the proxy cluster members has stopped running.
	Unavailable	Status cannot be determined. All of the members of a proxy cluster with an unavailable status might, in fact, be running but the proxy cluster has an unavailable status because the proxy server that is running the administrative console cannot communicate with all of the proxy cluster members. The node agent should be started on all proxy nodes to ensure that the correct status is shown.

Proxy server cluster settings

Use this page to view a list of the proxy server clusters that are defined for your system, and the status of each of these clusters. You can also use this page to view or change the configuration settings for a specific proxy server cluster or to view the local topology of a specific proxy server cluster.

To view a list of the proxy server clusters that are defined for your system, and the status of each of these clusters, in the administrative console click **Servers > Clusters > Proxy server clusters**.

To display the topology of your proxy server clusters, click **Servers > Clusters > Proxy server clusters > proxy_server_cluster_name**, and then click the **Local Topology** tab.

To view or change the configuration settings of a proxy server cluster, in the administrative console click **Servers > Clusters > Proxy server clusters > *proxy_server_cluster_name***.

Cluster name

Specifies a logical name for the proxy cluster. The name must be unique among proxy clusters within the containing cell.

Bounding node group name

Specifies the node group that forms the boundaries for this proxy cluster.

A node group is a collection of proxy server nodes. A node is a logical grouping of managed proxy servers, usually on a system that has a distinct IP host address. All proxy servers that are members of a proxy cluster must be on nodes that are members of the same node group. Nodes that are organized into a node group need enough capabilities in common to ensure that proxy clusters formed across the nodes in the node group can host the same application in each proxy cluster member. A node must be a member of at least one node group and can be a member of more than one node group.

Create and manage node groups by clicking **System administration > Node groups** in the administrative console.

Local Topology tab

Use this page to display, in a tree format, a list of all of the application server proxy clusters defined for your environment. The list shows all of the nodes and proxy cluster members that are included in each proxy cluster contained in a cell.

Proxy server cluster member collection

Use this page to view and manage proxy servers that belong to a proxy cluster.

To view this administrative console page, click **Servers > Clusters > Proxy server clusters > *proxy_cluster_name* > Cluster members**.

Proxy servers that are part of a proxy cluster are referred to as proxy cluster members. A copy of the first proxy cluster member that you create is stored as part of the proxy cluster data and becomes the template for all additional proxy cluster members that you create.

You can use this page to perform the following actions:

- Create a New proxy cluster member. To create a new member, click **New**.

A copy of the first proxy cluster member that you create is stored as part of the proxy cluster data and becomes the template for all additional proxy cluster members that you create. Any individual configuration change that you make to a proxy cluster member does not affect the configuration settings of the proxy cluster member template.

You can use wsadmin commands to modify the proxy cluster member template, or you can click **Servers > Clusters > Proxy server clusters > *proxy_cluster_name* > Cluster members**, and then click **Templates**. Any change that you make to the template does not affect existing proxy cluster members.

- Delete a proxy cluster member. To delete a proxy cluster member, select the proxy cluster member you want to delete, then click **Delete**.
- Create or edit proxy cluster templates. To work with proxy cluster templates, click **Templates**.
- Start a proxy cluster member. To start a proxy cluster member, select the server that you want to start, then click **Start**.
- Stop a proxy cluster member. To stop a proxy cluster member, select the server that you want to stop, then click **Stop**.

Member name

Specifies the name of the server in the proxy cluster. On most platforms, the name of the server is the process name. The name must match the (object) name of the proxy server.

Node





Specifies the name of the node for the proxy cluster member.

Version

Specifies the version of the product on which the proxy cluster member runs.

Status

This field indicates whether the proxy cluster member is started, stopped, partially stopped, or unavailable. If the status is unavailable, the node agent is not running in that node and you must restart the node agent before you can start the server.

	Started	The proxy cluster member is running.
	Partially stopped	The proxy cluster member is in the process of changing from a started state to a stopped state.
	Stopped	The proxy cluster member is not running.
	Unavailable	Status cannot be determined. A proxy cluster member with an unavailable status might, in fact, be running but has an unavailable status because the proxy cluster member that is running the administrative console cannot communicate with this proxy cluster member.

Proxy cluster member settings

Use this page to manage the members of a proxy cluster. A proxy cluster of proxy servers is managed together and participate in workload management.

To view this administrative console page, click **Servers > Clusters > Proxy server clusters > proxy_cluster_name > Cluster members > cluster_member_name**.

Member name

Specifies the name of the proxy server in the proxy cluster. On most platforms, the name of the server is the process name. The member name must match the name of one of the servers that are listed on the proxy servers page.

Run in development mode

Enabling this option may reduce the startup time of an proxy server. This might include Java virtual machine (JVM) settings, such as disabling bytecode verification and reducing Just in Time (JIT) compiler compilation costs. Do not enable this setting on production servers.

Specifies that you want to use the JVM settings, **-Xverify** and **-Xquickstart**, on startup. After selecting this option, save the configuration and restart the server to activate development mode.

The default setting for this option is `false`, which indicates that the server is not started in development mode. Setting this option to `true` specifies that the server is started in development mode, using settings that decrease server startup time.

Data type

Boolean

Default

`false`

Parallel start

Specifies whether to start the server on multiple threads. When you start the server on multiple threads, the server components, services, and applications start in parallel rather than sequentially, which might shorten the startup time.

The default setting for this option is `true`, which indicates that the server uses multiple threads when it starts. Setting this option to `false` specifies that the server uses a single thread when it starts, which might lengthen startup time.

Data type	Boolean
Default	true

Start components as needed

Select this property if you want the proxy server components started as they are needed by an application that is running on this proxy server.

When this property is selected, proxy server components are dynamically started as they are needed. When this property is not selected, all of the proxy server components are started during the startup process. Therefore, selecting this property usually results in improved startup time because fewer components are started during the startup process.

Note: If you are running other WebSphere products on top of the Application Server, make sure that those other products support this functionality before you select this property.

Proxy cluster member templates collection

Use this page to view the list of proxy cluster member templates that exist for this proxy cluster.

To view the proxy cluster member templates that are available for creating a new member of a proxy cluster, in the administrative console, click **Servers > Clusters > Proxy server clusters > *proxy_cluster_name* > Cluster members > Templates**.

To edit the attributes of an existing proxy cluster member template, click the name of the template.

Usually, only one template exists that you can use to create additional members for a proxy cluster. However, if a proxy cluster includes nodes that are at different versions of the product, a different template exists for each of these versions. For example, if a proxy cluster has proxy cluster members residing on both a Version 6.1 node and a Version 7.0 node, the proxy cluster has two templates. The Version 6.1 template is used when you create an additional proxy cluster member on the Version 6.1 node, and the Version 7.0 template is used when you create an additional proxy cluster member on the Version 7.0 node.

If you modify a template, all new proxy cluster members are created with the server property settings of the modified template. However, the property settings of existing proxy cluster members do not change. If you make any change to a proxy cluster member template, you should make the same change to all of the existing proxy cluster members.

Name

Specifies the name of the proxy cluster member template.

Platform

Specifies the operating system platform to which this template applies.

Version

Specifies the version of the product to which the template applies.

Description

Specifies a description of this proxy cluster member template. This field is optional and might be blank.

Proxy cluster member template settings

Use this page to modify proxy cluster member template attributes.

To view this administrative console page, click **Servers > Clusters > Proxy server clusters > proxy_cluster_name > Cluster members > Templates > cluster_member_template_name**. The following attributes are displayed:

Name

Displays the name of the proxy cluster member template.

Run in development mode

Enabling this option may reduce the startup time of a proxy server. This might include Java virtual machine (JVM) settings, such as disabling bytecode verification and reducing Just in Time (JIT) compiler compilation costs. Do not enable this setting on production servers.

Specifies that you want to use the JVM settings, **-Xverify** and **-Xquickstart**, on startup. After selecting this option, save the configuration and restart the server to activate development mode.

The default setting for this option is `false`, which indicates that the server is not started in development mode. Setting this option to `true` specifies that the server is started in development mode, using settings that decrease server startup time.

Data type	Boolean
Default	false

Parallel start

Specifies whether to start the proxy server on multiple threads. When you start the proxy server on multiple threads, the proxy server components, services, and applications start in parallel rather than sequentially, which might shorten the startup time.

The default setting for this option is `true`, which indicates that the proxy server uses multiple threads when it starts. Setting this option to `false` specifies that the proxy server uses a single thread when it starts, which might lengthen startup time.

Data type	Boolean
Default	true

Start components as needed

Select this property if you want the proxy server components started as they are needed by an application that is running on this proxy server.

Note: When this property is selected, proxy server components are dynamically started as they are needed. When this property is not selected, all of the proxy server components are started during the startup process. Therefore, selecting this property usually results in improved startup time because fewer components are started during the startup process.

Avoid trouble: If you are running other WebSphere products on top of the Application Server, make sure that those other products support this functionality before you select this property.

Creating a proxy cluster: Basic proxy cluster settings

Use this page to enter the basic settings for a proxy cluster.

To view this administrative console page, click **Servers > Clusters > Proxy server clusters > New**.

Proxy cluster name

Specifies the name of the proxy cluster. The proxy cluster name must be unique within the cell.

Supported Protocols

Indicates the protocol or protocols that the proxy server is configured to handle.

This information is based on the types of transport channels that are included in the transport chains that are configured for the proxy server. For example, if a transport chain includes an HTTP channel, HTTP displays in this field. If a transport chain includes both a SIP and an HTTP channel, SIP, HTTP displays in this field.

Creating a proxy cluster: Create first proxy cluster member

Use this page to specify settings for the first proxy cluster member.

There are two ways to create the first member of a proxy cluster:

- You can create the first member when you create a new proxy cluster.
To create a new proxy cluster, in the administrative console, click **Servers > Clusters > Proxy server clusters > New**.
- You can create an empty proxy cluster and then add a first member after you finish creating the proxy cluster.
To create a proxy cluster member for an existing proxy cluster, in the administrative console, click **Servers > Clusters > Proxy server clusters > *proxy_cluster_name* > Cluster members > New**.

When you create the first proxy cluster member, a copy of that member is stored as part of the proxy cluster data and becomes the template for all additional proxy cluster members that you create.

When adding servers to a proxy cluster, note that you can either remove a proxy server from a proxy cluster by deleting the proxy server from the list of proxy cluster members, or by deleting the server from the Proxy Servers collection.

Member name

Specifies the name of the proxy server that is created for the proxy cluster.

The member name must be unique on the selected node.

Select node

Specifies the node on which the proxy server resides.

Generate unique HTTP ports

Specifies that a unique HTTP port is generated for the proxy server. By generating unique HTTP ports for the proxy server, you avoid potential port collisions and configurations that are not valid.

Select basis for first proxy cluster member:

Specifies the basis you want to use for the first proxy cluster member.

- If you select **Create the member using a proxy server template**, the settings for the new proxy server are identical to the settings of the proxy server template you select from the list of available templates.
- If you select **Create the member using an existing proxy server as a template**, the settings for the new proxy server are identical to the settings of the proxy server you select from the list of existing proxy servers.

- If you select **Create the member by converting an existing proxy server**, the proxy server you select from the list of available proxy servers becomes a member of this proxy cluster.
- If you select **None. Create an empty proxy cluster**, a new proxy cluster is created but it does not contain any proxy cluster members.

Note: The basis options are available only for the first proxy cluster member. All other members of a proxy cluster are based on the proxy cluster member template, which is created from the first proxy cluster member.

Creating a proxy cluster: Create additional proxy cluster members

Use this page to create additional members for a proxy cluster. You can add a member to a proxy cluster when you create the proxy cluster or after you create the proxy cluster. A copy of the first proxy cluster member that you create is stored as part of the proxy cluster data and becomes the template for all additional proxy cluster members that you create.

To add members to a proxy cluster, in the administrative console, click **Servers > Clusters > Proxy server clusters > New > *proxy_cluster_name* > Cluster members > New**. After you enter the required information about the new proxy cluster member, click **Add Member** to add this member to the proxy cluster member list.

After adding a proxy cluster member, you might need to change one or more of the property settings for this proxy cluster member, or another proxy cluster member that you just added. To change one or more of the editable property settings (Member name, Node, and Generate unique ports) for any proxy cluster member that you just added, other than the first proxy cluster member, select that proxy cluster member, and then click **Edit**. When you finish changing the property settings, click **Update Member** to save your changes.

If you decide not to create a particular proxy cluster member, select the member and then click **Delete**.

You cannot edit or delete the first proxy cluster member or an already existing proxy cluster member.

If you create additional proxy cluster members immediately after you create the first proxy cluster member, the list of proxy cluster members includes a checklist in front of the names of these additional proxy cluster members. However, a check box does not appear in front of the name of the first proxy cluster member because you cannot delete this member or edit its settings. To modify the first proxy cluster member, click **Previous**.

Similarly, if you are adding proxy cluster members to a proxy cluster that already has existing members, the existing members appear in the list of proxy cluster members but a check box does not appear in front of the names of these proxy cluster members. To delete one of these existing members or to change the settings of one of these proxy cluster members, in the administrative console click **Servers > Clusters > Proxy server clusters > *proxy_cluster_name* > Cluster members**, and then select the member that you want to delete or whose configuration settings you want to change.

Member name

Specifies the name of the proxy server that is created for the proxy cluster.

The member name must be unique on the selected node.

Select node

Specifies the node on which the proxy server resides.

In a mixed cell environment, you can use any server from within the node group to create a new proxy cluster member. For example, if the node group to which the proxy cluster belongs consists of a V7.0 node, and a V6.1 node, you can use a server from either the V6.1 or the V7.0 node to create a new proxy cluster member.

Generate unique HTTP ports

Specifies that a unique HTTP port is generated for the proxy server. By generating unique HTTP ports for the proxy server, you avoid potential port collisions and configurations that are not valid.

Creating a proxy cluster: Summary settings

Use this administrative console page to view and save settings when you create a proxy cluster or proxy cluster member.

You can view this administrative console page whenever you create a new proxy cluster or a new proxy cluster member. This summary page displays your configuration changes before you commit the changes and the new proxy cluster or proxy cluster member is created.

To create a cluster, in the administrative console, click **Servers > Clusters > Proxy server clusters > New**.

To create a proxy cluster member for an existing proxy cluster, in the administrative console, click **Servers > Clusters > Proxy server clusters > New > *proxy_cluster_name* > Cluster members > New**. After you enter the required information about the new proxy cluster member, click **Add Member** to add this member to the proxy cluster member list.

The bounding node group of the proxy cluster is based on the first application server that is added as a member of the proxy cluster. The settings for this first member become the settings for the proxy cluster member template that is then used to create additional proxy cluster members.

To select a different bounding node group, in the administrative console, click **Servers > Clusters > Proxy server clusters > New > *proxy_cluster_name* > Cluster members > New**. Then select the appropriate node group for the new proxy cluster member. When you select a different node group, another proxy cluster member template is automatically created for that node group, if one does not already exist.

Review the changes to your configuration. Click Finish to complete and save your work.

Managing a proxy server cluster

You can manage a proxy server cluster using either the administrative console or the wsadmin tool.

About this task

You can use either the administrative console or wsadmin to manage a proxy server cluster. To use the administrative console to manage your proxy server clusters, in the administrative console, click **Servers > Clusters > Proxy server clusters**. You have the following options: New, Delete, Start, Stop, Ripplestart, and ImmediateStop.

- To start one or more proxy server clusters, select the proxy server clusters and click either **Start** or **RippleStart**.
 - Start launches the server process of each member of the cluster by calling the node agent for each server to start the servers. After all servers are running, the state of the cluster changes to running. If the call to a node agent for a server fails, the server does not start.
 - RippleStart combines stopping and starting operations. It first stops and then restarts each member of the cluster. For example, your cluster contains 3 cluster members named server_1, server_2 and server_3. When you click RippleStart, server_1 stops and restarts, then server_2 stops and restarts,

and finally `server_3` stops and restarts. Use the `RippleStart` option instead of manually stopping and then starting all of the application servers in the cluster.

- To stop one or more proxy server clusters, select the proxy server clusters and click either **Stop** or **ImmediateStop**.
 - Stop halts each server in a manner that allows the server to finish existing requests and allows failover to another member of the cluster. When the stop operation begins, the cluster state changes to partially stopped. After all servers stop, the cluster state becomes stopped.
 - ImmediateStop stops the server quickly without regard to existing requests. The server ignores any current or pending tasks. When the stop operation begins, the cluster state changes to partially stopped. After all servers stop, the cluster state becomes stopped.
- To change the configuration settings, click the name of the proxy server cluster to access the Configuration tab.
- To create a new proxy server cluster, click **New**. Type the name for the cluster in the **Cluster name** field.
- To delete an existing proxy server cluster, click **Delete** and then click **OK**.
- To add new members to a proxy server cluster, click the name of an existing proxy server cluster, and click **Cluster members > New**. Type the name for the proxy server in the **Member name** field.

Results

You have successfully modified a proxy server cluster.

Migrating profiles for the proxy server

This topic describes how Version 6.0.2 profiles contain the proxy server feature when migrating to Version 7.0 profiles.

About this task

Complete the following steps to migrate from a Version 6.0.2 profile to a Version 7.0 profile.

1. Run the **WASPreUpgrade.bat** or the **WASPreUpgrade.sh** command from the `install_root/bin` directory and point it to the Version 6.0.2 release.

This action makes a copy of all the required Version 6.0.2 files that are needed for the **WASPostUpgrade.bat** or **WASPostUpgrade.sh** command. You can delete these Version 6.0.2 files, but this action is not recommended.
2. Create your corresponding profiles in Version 7.0, if you have not already done so.
3. Run the **WASPostUpgrade.bat** or the **WASPostUpgrade.sh** command from the `install_root/bin` directory and point the commands to the WASPreUpgrade backup directory that you created in step one.

Customizing routing to applications

This topic provides information on disabling routing through the proxy server to applications that are on on-demand configuration (ODC)-compliant application servers.

About this task

Complete these steps to disable routing to ODC-compliant application servers.

1. From the administrative console, select **Applications > Enterprise Applications > application_name**.
2. In the Modules section, click **Manage modules**.
3. Click a module name.
4. In the Additional Properties section, click **Web Module Proxy Configuration**.

5. In the General Properties section , deselect **Enable proxy** to disable routing using the proxy server.
6. Optional: Choose a protocol from the **Web Module Transport Protocol** list. A transport protocol, such as HTTP, can be used for the protocol between the proxy server and the application server instead of the protocol that the client uses to communicate with the proxy server.
For example, in order to stop Secure Sockets Layer (SSL), also known as SSL offload, for HTTPS traffic for the Web module, select **HTTP** for the transport protocol.
7. Click **Apply**.
8. Click **OK**.

Web module proxy server configuration settings

Use this page to specify proxy server configuration settings for the Web module.

To view this administrative console page, click **Applications** → **Application Types** → **WebSphere enterprise applications** → *application_name* → **Manage Modules** → *Web_module_name* → **Web Module Proxy Configuration**.

Name

Specifies the name of the proxy server.

Enable Proxy

Select **Enable Proxy** to enable proxy server to route requests to this Web module. Deselect **Enable Proxy** to disable routing using the proxy server.

Web Module Transport Protocol

Specifies the protocol that the proxy server uses when communicating with this Web module.

Choose the protocol in the Web module transport protocol field if you want to use a specific transport protocol, such as HTTP, for the protocol between the proxy server and the application server. This is an alternative to using the protocol that is used by the client to communicate with the proxy server. For example, in order to perform Secure Sockets Layer (SSL) termination, which is also known as SSL offload, for HTTPS traffic for the Web module, select HTTP for the transport protocol. HTTPS protocol is for advanced configurations and is not commonly used.

Custom Properties

Specifies additional custom properties for this runtime component. Some components use custom configuration properties that are defined by this option.

Routing requests to ODC-compliant application servers in other cells

On Demand Configuration (ODC) enables product components, such as the proxy server, to build application deployment, availability, and accessibility information in order to route requests for these applications without any additional configuration at the proxy server. If you want to isolate proxy servers with firewalls, place them in a cell that is separate from the applications and configure the core group bridge service.

About this task

ODC uses the high availability capabilities of the product to publish and subscribe updates. The deployment manager within a cell and the application servers that have the ODC capabilities typically publish the updates that are subscribed to by intermediaries such as the proxy server.

You can configure the proxy server to route requests to applications that are hosted in other cells. The core group bridge service provides communication between a cell with a proxy server and a cell with a target application. Once the cells are linked with core group bridges, the proxy server will be able to route

to the application without additional configuration. To view your core group bridge settings, in the administrative console, click **Servers > Core groups > Core group bridge settings**.

Note: Cross-cell failover is not supported. If the same application is installed on both the local cell and in one or more remote cells, and the application or application servers in the local cell go down, then the ones in the remote cell will not be used. The proxy server will always route to the local cell even if the resource is available in a remote cell.

The basic steps to configure cross-cell routing are configuring and enabling core group bridges in the cell that the proxy server belongs to, and in each of the other cells that are being routed to. The core group bridges need to be configured with the same access point group name.

1. Create bridge interfaces, peer access groups and peer ports for cells. The peer access point group name must be the same in the cells. If you want to use `CGB_ENABLE_602_FEATURES`, enable it on all of the cells involved.
2. Restart all processes that are now bridge interfaces.

What to do next

If security is enabled, it must be enabled in all cells for the core group bridge service.

Configuring rules to route requests to Web servers

This section provides information to configure rules to route requests to web servers or application servers that are not on-demand configuration (ODC)-compliant.

About this task

The proxy server permits explicit configuration of routing rules in order for client requests to route to Web servers or application servers that are not ODC-compliant. This involves completing following tasks.

1. Create a URI group.
To define the URI patterns that are associated with the Web content that is hosted on these servers, in the administrative console click **Environment > URI groups**.
2. Create a generic server cluster definition.
To define the endpoints that define the cluster membership, in the administrative console click **Servers > Clusters > Generic server clusters**.
3. Create routing rules from the detail view of the proxy server panel that associates the inbound virtual host and a defined URI group to a defined generic server cluster.

Modifying the HTTP endpoints that the proxy server listens on

By default, the proxy server listens on the following named endpoints: `PROXY_HTTP_ADDRESS` and `PROXY_HTTPS_ADDRESS`. You can modify the ports and hosts that are associated with these endpoints in the administrative console.

About this task

Modify the ports and hosts that are associated with the proxy server endpoints as follows:

1. In the administrative console, click **Servers > Server Types > WebSphere proxy servers > *proxy_server_name* > Communications > Ports**.
2. Select the port you want to modify.
The host and port that are associated with the specified endpoints can be modified to suit the requirements of the deployment. The administrator must ensure for correct routing, the virtual hosts

that are associated with applications to be routed to are correctly updated with the appropriate host aliases (host aliases with the modified host and port combinations).

3. Click **Apply**.

What to do next

Restart the server. The server must be restarted for this modification to go into effect.

Adding a new HTTP endpoint for the proxy server

An administrator can use the proxy server to create additional endpoints to listen for HTTP and HTTPS requests.

About this task

The following steps will create a new transport chain:

1. Click **Servers > Server Types > WebSphere proxy servers > proxy_server_name > HTTP proxy server settings > Proxy server transports** in the administrative console.
2. Click **New** to launch the Create new transport chain wizard.
3. Determine a name for the transport chain.
4. Choose a template based on whether the endpoint should accept HTTP (proxy template) or HTTPS (proxy-secure template) requests. The wizard prompts for the host and port for the endpoint. Creating a new endpoint for the proxy server requires a restart of the proxy server to be effective.

Setting up caching in the proxy server

This topic provides information on caching static and dynamic content in the proxy server.

About this task

Complete the following steps to configure a proxy server such that it can cache static and dynamic content.

1. Configure the object cache instance for size, disk offload location, and other such capabilities, in the administrative console. Click **Servers > Server Types > WebSphere proxy servers > proxy_server_name > HTTP proxy server settings > Proxy cache instance config**. Repeat these steps on any nodes that have a proxy server.
2. Select the proxy cache store instance and enable configuration attributes such as cache size, disk offload, and cache replication. For disk offload, it is recommended that the location be set to a dedicated disk partition.
3. Enable caching at the proxy server, in the administrative console. Click **Servers > Server Types > WebSphere proxy servers > proxy_server_name > HTTP proxy server settings > Proxy settings** page in the administrative console.
4. Select **Enable caching** and choose a cache instance from the drop-down box.
 - a. To enable dynamic content to be cacheable with the proxy server, in the administrative console, click **Servers > Server Types > WebSphere proxy servers > proxy_server_name > HTTP proxy server settings > Proxy settings**, and then select **Cache dynamic content**. You enable cacheability and invalidation of dynamic content when you enable servlet caching on the application server, and specifying the cache criteria in a cachespec.xml file that is associated with that application. Invalidation is received by connecting to the cache update URI that is associated with the invalidation servlet hosted on the application server cluster.

Dynamic content is content that an application, that is hosted on an application server, generates. A proxy server caches dynamic content only if the content is identified as edge cacheable in the cachespec.xml file for the application. All of the information that describes the cache, such as the

ID to use for the cache, dependency identifiers for invalidation, and expiration times, is also defined in the cachespec.xml file. Proxy Server uses the ESI protocol to obtain this information from the file.

See the *Administering applications and their environment* PDF for more information on how to set up a cachespec.xml file for an application.

Cached dynamic content can be invalidated by events in the application server. The ESI Invalidation Servlet, that is contained in the DynacacheEsi.ear application, propagates these invalidation events from the application server to the proxy server. The DynacacheEsi.ear is shipped with the product, and must be deployed in the cluster with the application that is generating the dynamic content for dynamic caching at the proxy server to function properly.

- b. Static caching is enabled by default when caching is enabled for the proxy server. *Static content* is Web content that is public and accompanied by HTTP response headers, such as EXPIRES and LAST_MODIFIED_TIME, that describe how long the response can be cached. The proxy server uses the HTTP 1.1 RFC (2616), which specifies how content should be treated and includes capabilities such as VARY header support for caching variants of the same resource Uniform Resource Identifier (URI).

Static cache rules collection

This topic lists the static cache rules for a proxy server. From this topic you can create, delete, or modify a static cache rule.

To view this administrative console topic, click **Servers > Server Types > WebSphere proxy servers > proxy_server_name > HTTP Proxy Server Settings > Static cache rules**.

The security proxy, or the intermediary that is the entry point into the enterprise, is responsible for terminating SSL connections with the client, authenticating the request, propagating the connection characteristics for the client, and any other credentials to the application server in the enterprise.

The proxy server enables security proxies to be identified so that private headers that are set in the request are propagated as they are to the application servers. Identify the list of security proxies that are trusted by the proxy server using the trusted security proxies field. To access this administrative console field, click **Servers > Server Types > WebSphere proxy servers > proxy_server_name > HTTP proxy server settings > Proxy settings**. You can find trusted security proxies in the **Security** section of this administrative console page.

URI Groups

The URI group name is a user-specified name.

The name field cannot contain the following characters: # \ / , : ; " * ? < > | = + & % ' .

The name that is defined must be unique among URI groups and cannot begin with a period or a space. A space does not generate an error, but leading and trailing spaces are automatically deleted.

Disable caching for this URI group

Specifies whether or not caching is disabled.

The default is false, which indicates that caching is enabled for the URI group.

Default expiration

The default expiration that you set for the cached response for the URI that is associated with this cache rule.

The default expiration value is in seconds.

Last modified factor

Use this field to derive the cache expiration value for a response if it does not have HTTP expiration headers and when it has a LastModifiedTime header in the response.

Name of the virtual host

A virtual host that is configured using the virtual host service. This virtual host is associated with the proxy server. This attribute is one of the elements in a request that is matched by the proxy server to determine if this rule is activated.

Static cache rule settings

Use this topic to configure a cache rule that is associated to a URI group for the proxy server. HTTP 1.1 defines a set of rules for proxy servers to cache content. Static cache rules enable these default rules to be overridden for a given address space. Before the rules have any meaning, you must enable caching on the **Servers > Server Types > WebSphere proxy servers > proxy_server_name > HTTP Proxy Server Settings > Proxy settings** administrative console page.

To view this administrative console page, click **Servers > Proxy Servers > server_name > HTTP Proxy Server Settings > Static cache rules > URI_group**.

You can edit proxy server setting fields on the Configuration tab.

URI groups

URI groups, along with the virtual host, define the scope of the address space to have cache customizations performed.

The name field cannot contain the following characters: # \ / , : ; " * ? < > | = + & % ' .

The name that is defined must be unique among URI groups and cannot begin with a period or a space. A space does not generate an error, but leading and trailing spaces are automatically deleted.

Disable caching for this URI group

Disables caching for this address space. A user may wish to disable caching for a set of content servers that are known to contain sensitive or highly-personalized information.

The default is `false`, which indicates that caching is enabled for the URI group.

Default expiration

The default expiration value, in seconds, that is used to determine the validity of a cached response when all other HTTP 1.1 caching-related response headers do not give guidance. The default value is sufficient in most environments.

The default expiration value is in seconds.

Last modified factor

The percentage of a last-modified header for a response that determines the validity of a cached response when the response does not have explicit HTTP expiration headers. The default value is sufficient in most environments.

The default for this field is `0.0`.

Name of the virtual host

A virtual host that is configured using the virtual host service. This virtual host is associated with the proxy server. This attribute is one of the elements in a request that is matched by the proxy server to determine if this rule is activated.

The default for this field is `none`.

Routing requests from a plug-in to a proxy server

This topic provides information on setting up a Web server plug-in to route requests to a proxy server.

About this task

An administrator may choose to set up a Web server, such as IBM HTTP Server, with the Web server plug-in as a front-end to the proxy server. The plug-in configuration file for such a topology cannot use the traditional plug-in configuration generation mechanism if the requests are routed through the proxy server.

To generate the `plugin-cfg.xml` file to use with the Web server plug-in to route through the proxy server, complete the following steps:

1. From the administrative console, click **Servers > Server Types > WebSphere proxy servers > *proxy_server_name* > HTTP proxy server settings > Proxy settings**.
2. In the **Generate plug-in configuration** drop-down menu, select the appropriate scope.
3. Optional: If you have a script that manually copies the `plugin-cfg.xml` file from the node to the plug-in installation location, enter the path to the script in the **Plugin config change script** field.
4. In the Trusted Security Proxy field, add the hostname or IP address of the node for the plug-in that serves as the trusted intermediary for the proxy server.
5. Click **OK**.
6. Disable the automatic propagation of the plug-in if you are using IBM HTTP Server with remote administration. From the administrative console, click **Servers > Server Types > Web Servers > *web_server_name* > Plug-in properties**. Deselect **Automatically propagate plugin configuration file**. This will prevent WebSphere Application Server from copying the traditional `plugin-cfg.xml` file over the proxy server `plugin-cfg.xml` file.
7. Save your changes.
8. Stop and restart the proxy server. The `plugin-cfg.xml` file will be in the `<install_root>/<profile_dir>/etc/` directory unless your plugin was generated at server scope. If you generated the plugin for the server scope, the `plugin-cfg.xml` file will be in the `<install_root>/<profile_dir>/etc/<server_name>` directory. If you do not have a script in the **Plugin config change script** field, manually copy the `plugin-cfg.xml`. If you do not have a script in the **Plugin config change script** field, manually copy the `plugin-cfg.xml` file to the plug-in.

What to do next

To verify that the proxy server trusts the Web server, add the host name or address of the Web server to the Trusted security proxies section on the Proxy Settings page in the administrative console. To reach this page, click **Servers > Server Types > WebSphere proxy servers > *proxy_server_name* > HTTP proxy server settings > Proxy settings**. This enables the proxy server to honor the private headers that are set by the fronting intermediary server.

Creating a proxy server cluster using the `wsadmin` command

This topic describes how to create a cluster of proxy servers, using the `wsadmin` command, that can route requests to applications in a cell.

Before you begin

The cluster includes the machines and nodes that are going to belong to the proxy server cluster. Consider how requests are routed to the proxy server cluster. For example, are requests routed using the domain name server (DNS), the Load Balancer, or the proxy server.

Before you create a cluster, start the deployment manager.

About this task

Create a proxy server cluster, as follows:

1. Start the wsadmin utility.
2. Create an empty cluster with no members, by typing:

```
$AdminTask createCluster {-clusterConfig {-clusterName <name_of_cluster> -clusterType PROXY_SERVER}}
```

Or, to create a cluster and add a specified proxy server to that cluster, type:

```
$AdminTask createCluster { -clusterConfig {{<name_of_cluster> true PROXY_SERVER}} -convertServer  
  {{<node_name> <name_of_proxy_server> "" "" ""}}
```

This proxy server serves as the template for all subsequent members that are added to the cluster.

3. Add one member at a time to the cluster, as follows:

```
$AdminTask createClusterMember {-clusterName <name_of_cluster> -memberConfig  
  {-memberNode <node_name> -memberName <name_of_proxy_server>}}
```

If no members exist in the cluster, the first member that is added serves as the template for subsequent members that are added to the cluster.

When a proxy server is added to a cluster, proxy-specific configuration settings for it can only be configured using the wsadmin scripting client.

4. Save the configuration changes, as follows:

```
$AdminConfig save
```

5. Start the proxy server cluster so that request routing is enabled, as described in Starting clusters using scripting.
6. Configure requests to route to the proxy server. For DNS-based routing, associate the logical name of the site with the IP addresses of the proxy server cluster members in DNS.

For Load balancer routing, configure the IP addresses of the cluster members as the target of the virtual cluster.

For Edge proxy or IBM HTTP Server with WebSphere Application Server plug-in-based routing, generate the plug-in configuration file for the proxy server cluster, and configure the Edge proxy or the WebSphere Application Server plug-in with this information.

Results

The proxy server cluster is created with the members and is enabled for routing traffic.

What to do next

Monitor the traffic. See “Monitoring the proxy server with PMI” on page 238 for more information.

Related tasks

Starting clusters using scripting

Use the wsadmin tool to start application server, generic server, and proxy server clusters in the application server runtime.

“Monitoring the proxy server with PMI”

You can monitor the traffic of a proxy server using the Performance Monitoring Infrastructure (PMI) function.

Related information

“Generic server clusters collection” on page 197

Use this page to create, delete or modify a generic server cluster. Creating a generic server cluster is the next step towards generating the ability to route requests to a non-IBM WebSphere Application Server or a pre-Version 6.0 cell, after creating the proxy server.

“Generic server clusters configuration” on page 198

Use this topic to configure a generic server cluster. Creating a generic server cluster is the next step, after creating the proxy server, towards generating the ability to route requests to a non-IBM WebSphere Application Server or a pre-Version 6.0 WebSphere Application Server cell.

“Generic server cluster ports collection” on page 198

After defining the generic server cluster name, use this page to create, delete, and configure the members of the cluster.

“Generic server cluster members” on page 198

After defining the generic server cluster name, use this page to define the members of the cluster.

“URI groups” on page 199

A group of URI patterns, which you define, that can be mapped back to generic server clusters. When creating a URI group, verify that you are planning for URIs that form a logical collection. From this topic, you can create, delete, or modify a URI group.

“URI group configuration” on page 199

Use this topic to configure a URI group that can be mapped back to generic server clusters. When creating a URI group, ensure that you are planning for URIs that form a logical collection.

Monitoring the proxy server with PMI

You can monitor the traffic of a proxy server using the Performance Monitoring Infrastructure (PMI) function.

Before you begin

The proxy server must be running before performing these steps.

1. In the administrative console, click **Monitoring and Tuning > Performance Monitoring Infrastructure (PMI)** in the console navigation tree. The proxy server collection page displays.
2. Select the proxy server from the collection list.
3. Click **Custom**. The Custom monitoring level panel displays.
4. In the navigation tree, expand **Proxy Module**. Select the statistics you want to view, then click **Enable**.

Monitoring traffic through the proxy server

You can monitor traffic, such as requests and connection statistics, through the proxy server.

Before you begin

You should know the machines and nodes that will belong to the proxy server cluster before completing these steps, because the product must be installed on those machines.

1. Ensure that the proxy server is running and there is some traffic flowing through the proxy server.

2. Obtain the proxy server MBean and invoke the operation to get the route statistics as follows:
 - a. Start **wsadmin**.
 - b. Get all of the traffic statistics through the proxy server using one of the following commands.

Using the JACL command

```
$AdminControl queryNames type=ProxyServer,*  
set proxymbean <cut and paste the MBean identifier from the previous command output>  
$AdminControl getAttribute $proxymbean stats
```

Using the Jython command

```
print AdminControl.getAttribute((AdminControl.queryNames("type=ProxyServer,*")), "stats")
```

The **\$AdminControl queryName** command lists all of the proxy server MBeans. There will be one per active proxy server in the cell. Set the *proxymbean* variable to the appropriate proxy server MBean from the output of the previous command.

Results

The traffic that flows through the proxy server can now be monitored.

Overview of the custom error page policy

The custom error page policy is a feature that enables the proxy server to use an application to generate an HTTP error response. With this capability, the administrator can return a polished error page when the proxy server generates an error, or when a content server returns an unsuccessful response.

The following action describes scenarios for how the error page policy is used when it is configured:

- **Internal error**

1. The client sends the following request to the proxy server: GET /house/rooms/kitchen.jpg HTTP/1.1.
2. The proxy server generates an internal error because no servers map to the request (HTTP 404 – File not found).
3. The error policy is configured to handle HTTP 404 responses, so it sends a request to the error page application to retrieve error content to send to the client. The request URI and HTTP response code are included as query parameters in the request to the error page application. If the configured error page application URI is /ErrorPageApp/ErrorPage, then the request URI to the error page application is: /ErrorPageApp/ErrorPage?responseCode=404&uri=/house/rooms/kitchen.jpg. The query parameters “responseCode” and “uri” are sent to the error page application by default.
4. The proxy server returns an HTTP 404 response with content from the error page application.

- **Remote error**

1. The client sends the following request to the proxy server: GET /house/rooms/kitchen.jpg HTTP/1.1
2. The proxy server forwards the request to the homeserver.companyx.com content server.
3. The homeserver.companyx.com content server is unable to locate the /house/rooms/kitchen.jpg file and sends an HTTP 404 response (File not found) to the proxy server.
4. The error policy is configured to handle HTTP 404 responses, so it sends a request to the error page application to retrieve error content to send to the client. The request URI and HTTP response code are included as query parameters in the request to the error page application. If the configured error page application URI is /ErrorPageApp/ErrorPage, then the request URI to the error page application is: /ErrorPageApp/ErrorPage?responseCode=404&uri=/house/rooms/kitchen.jpg. The query parameters “responseCode” and “uri” are sent to the error page application by default.
5. The proxy server returns an HTTP 404 response with content from the error page application.

A sample error application is available in the <WAS_INSTALL_ROOT>/installableApps/HttpErrorHandler.ear file.

Request mapping

This topic provides an overview of how the proxy server matches an HTTP request that is received to an application that is deployed in the cell or routing rule.

Unlike the Apache Web server or Caching Proxy, which have flat configuration files with routing precedence that is inherent to the ordering of directives, the proxy server uses a best match mechanism to determine the installed application or routing rule that corresponds to a request. The virtual host or URI patterns determine the best match for a Web module or routing rule. For applications that are deployed in clusters, the proxy server maintains affinity (Secure Sockets Layer ID, cookie, and URL rewriting), otherwise, a weighted round-robin approach is used to select the target server. The following examples address various routing scenarios for when routing rules and applications are deployed within the same cell.

Proxy environment. A WebSphere proxy server called proxy1 is active in the same cell as the applications and routing rules. All of the applications and routing rules are enabled in the cell for proxy1, and PROXY_HTTP_ADDRESS for proxy1 is set to 80.

Virtual host	Host name	Port
default_host	host1.company.com	80
	host1.company.com	9080
	*	80
proxy_host	host2.company.com	80
	*	443
	*	80
server_host	host3.company.com	80

URI group name	URI patterns
ALL	/*
ROOMS	/kitchen/*, /bathroom/*, /bedroom/*
CONFLICT	/WM2C/*

Generic server cluster name	Protocol	Host	Port
CLUSTER1	HTTP	webserver1.company.com	9081
		webserver2.company.com	9083
CLUSTER2	HTTP	host47.company.com	8088
		host48.company.com	8088
CLUSTER2-SSL	HTTPS	host47.company.com	8443
		host48.company.com	8443

Routing rule name	Virtual host	URI group	Action
ALLTOCLUSTER1	proxy_host	ALL	Generic server cluster - CLUSTER1
ROOMTOCLUSTER2	proxy_host	ROOMS	Generic server cluster - CLUSTER2
ALLTOCLUSTER2	server_host	ALL	Generic server cluster - CLUSTER2

Routing rule name	Virtual host	URI group	Action
REDIRECTTOCONFLICT	default_host	CONFLICT	Redirect - http://www.conflict.com

Application name	Context root	Web module name	Virtual host	Web module URI patterns
App1	/WM1A/	Web Mod A	default_host	wm1a.jsp
	/WM1B/	Web Mod B	default_host	wm1b.jsp
App2	/WM2C/	Web Mod C	default_host	/*, wm2c.jsp
	/WM2D/	Web Mod D	default_host	/*, wm2d.jsp

Example 1: Basic request. The proxy1 proxy receives the following request:

```
GET /WM1A/wm1a.jsp HTTP/1.1
Host: host1.company.com
```

Result. The `wm1a.jsp` file is sent as the response. The `ALLTOCLUSTER1` routing rule is a possible match, but Web Mod A is chosen as the best match by proxy1 because the combination of its context root and URI pattern `/WM1A/wm1a.jsp` is a better match than `/*`. Web Mod A is also chosen as the best match because its virtual host contains the `host1.company.com:80` alias, which is a more specific match than the `*:80` wild card alias.

Example 2: Routing rules that use the same URI group and different virtual hosts . The proxy1 proxy receives the following request:

```
GET /index.html HTTP/1.1
Host: host3.company.com
```

Result. The proxy1 proxy maps the request to the `ALLTOCLUSTER2` routing rule, and a response is received from a server in `CLUSTER2`. The `ALLTOCLUSTER1` routing rule is a possible match and can handle the request if the `ALLTOCLUSTER2` routing rule did not exist. However, the `ALLTOCLUSTER2` rule is the best match because its virtual host (`server_host`) explicitly lists `host3.company.com`.

Example 3: Routing rules that use same virtual host and different URI groups. The proxy1 proxy receives the following request:

```
GET /kitchen/sink.gif HTTP/1.1
Host: host2.company.com
```

Result. The proxy1 proxy maps the request to the `ROOMSTOCLUSTER2` routing rule and a server from the `CLUSTER2` cluster sends a response. The `ALLTOCLUSTER1` routing rule is a possible match, but the `ROOMSTOCLUSTER2` rule is the best match because its URI group contains a pattern `/kitchen/*` that is a better match for the request URI `/kitchen/sink.gif`.

Example 4: Routing rule URI group conflicts with URI pattern of a Web module that uses the same virtual host. The proxy1 proxy receives the following request:

```
GET /WM2C/index.html HTTP/1.1
Host: host1.company.com
```

Result. Indeterminate. It is unknown whether Web Mod C or the `REDIRECTTOCONFLICT` routing rule handles the request because they use the same virtual host and have the same URI pattern. In such cases, the ID `DWCT0007E` message is displayed in `SystemOut.log` file for the proxy1 proxy. In this example, changing the `REDIRECTTOCONFLICT` routing rule to use a different virtual host resolves the problem.

Example 5: The PROXY_HTTP_ADDRESS address is not in the virtual host. Assume that the proxy1 proxy address, PROXY_HTTP_ADDRESS, is changed to 81, while all of the other configuration information remains the same. The proxy1 proxy receives the following request:

```
GET /index.html HTTP/1.1
Host: host1.company.com:81
```

Result. The proxy1 proxy is unable to handle the request because the PROXY_HTTP_ADDRESS address is not available in a virtual host and will send an HTTP 404 response back to the client.

Session failover in the proxy server

This article describes how the proxy server handles session failover.

You can enable memory-to-memory replication on an application server to maintain session state in multiple servers. In this case, a private header is added to the response which identifies the backup servers for the session. The proxy server reads this header and maintains a list of backup servers for a session. If the proxy server fails to route to the primary server, it tries to route to the backup servers. If none of the backup servers are available, a deterministic algorithm selects one of the available servers; consequently, multiple proxy servers will route to the same server.

If the set of servers that are hosting a session changes, the private response header causes the proxy server to update its list of servers for the session. It is possible that the set of servers is updated, but the proxy server has not yet received an updated response header. In this case, the proxy server routes to a server that does not contain the session data. If this occurs, the backend server obtains the session data from a server that contains the session data. There is no functional difference in this case; however, there is a performance difference due to the cost of obtaining the session data from another server.

Installing a Session Initiation Protocol proxy server

The Session Initiation Protocol (SIP) proxy server initiates communication and data sessions between users. It delivers a high performance SIP proxy capability that you can use at the edge of the network to route, load balance, and improve response times for SIP dialogs to backend SIP resources. The SIP proxy provides a mechanism for other components to extend the base function and support additional deployment scenarios. This topic provides information to install a Session Initiation Protocol (SIP) proxy server.

Before you begin

Ensure that you have a SIP application installed. The SIP container will not listen on a port without a SIP application installed. For a proxy server to be useful, you should have a cluster of SIP application servers.

About this task

The SIP proxy design is based on the HTTP proxy architecture and can be considered a peer to the HTTP proxy. Both the SIP and the HTTP proxy are designed to run within the same proxy server and both rely on a similar filter-based architecture for message processing and routing.

A SIP proxy serves as the initial point of entry, after the firewall, for SIP messages that flow into and out of the enterprise. The SIP proxy acts as a surrogate for SIP application servers within the enterprise. In fact, the nodes that host the SIP proxy servers host the public SIP domain of the enterprise. As a surrogate, you can configure the SIP proxy with rules to route to and load balance the clusters of SIP containers. The SIP proxy is capable of securing the transport, using secure sockets layer (SSL), and the content, using various authentication and authorization schemes.

The SIP proxy is also responsible for establishing outbound connections to remote domains on behalf of the back-end SIP containers and clients that reside within the domain that is hosted by the proxy. Another important feature of the SIP proxy is its capability to protect the identity of the back-end SIP containers from the SIP clients.

Note: If you created a cell, deployment manager, and node when you installed the product, the first five steps do not apply and you can skip to step six.

1. **Optional:** Launch the profile creation wizard and create a profile.
2. **Optional:** Launch the profile creation wizard and create a deployment manager profile.
3. **Optional:** Start the deployment manager.
4. **Optional:** Federate the node that contains your newly-created profile into the deployment manager cell. You can federate the node in one of the following ways:
 - Type `<WAS_home>profiles/<name_of_profile>/bin/addNode <deployment_manager_host_name> 8879`
8879 is the default bootstrap port. The server that is being federated should not be running when completing this task from a command line.
 - From the administrative console, click **System Administration > Nodes > Add nodes** .

The server in the node that is being federated needs to be running when federating through the deployment manager administrative console.

5. **Optional:** Create a cluster in the administrative console by clicking **Servers > Clusters > WebSphere application server clusters > New**. Add the federated server to the cluster.

Note: After you create the cluster, the name of this cluster is listed as an option in the **Default cluster** field on the SIP proxy settings page for the proxy server. Select this new cluster as the default cluster for the proxy server. Do not leave the value **none** as the value for the **Default cluster** field. **none** is the default value for this field.

6. Create a proxy server on the node that you just federated, or on another node in the cell that contains the cluster that the proxy will sit in front of by clicking **Servers > Server Types > WebSphere proxy servers > New**.

Note: You must select the SIP protocol.

You do not have to install the proxy server on a machine that also includes one of the servers in the cluster.

Results

After you choose a default cluster, your SIP proxy server will be functional. However, if the SIP proxy server is fronted by an IP sprayer, you must set up the loopback address and modify the inbound channel chains to make the SIP proxy server functional. See the load balancer documentation for information about setting up the loopback address.

Note: The virtual host for your SIP container ports must be defined, and the SIP container(s) must be restarted after adding the new virtual host.

Trusting SIP messages from external domains

The general approach for providing secure communications between two independent domains or communities (each maintaining distinct directories) relies on *identity assertion*, where a trust relationship is established between two distinct domains using a certificate exchange during the setup of the physical Secure Sockets Layer (SSL) connection between the two domains.

About this task

Authentication of Session Initiation Protocol (SIP) messages that are sent by end users needs to occur only in the local domain for the user. All user messages pass through the SIP container local domain

before being sent on to the external domain. If a message is received from a external domain over a secured connection that is mutually authenticated in the manner described as follows, it is assumed that the message is authenticated by the external domain because of the trust relationship. An administrator can enable support for external domains in the SIP proxy as follows:

1. Enable client authentication within the SSL repertoire that is assigned to all the inbound channel chains (or endpoints) that are to receive inbound connections from external domains.
2. Ensure that all trusted certificate authorities are set up in the trust store that is assigned to the SSL repertoires mentioned in the previous step. Set up the asymmetric key pair (public and private keys) for the local domain, with the proper chain of certificates that is associated with the local domain.
3. Configure the distinguished names (DNs) that are associated with the external domains to support. The DN is part of the X.509 certificate that is sent by the external domain server when the SSL connection is set up. Within the configuration model, each SIP external domain entry includes a field for the external DN.
4. Assuming that the SIP infrastructure is deployed within each domain, provide the DN to the external domain administrator that is included in the local domain's public certificate. With this action, the external domain administrator can configure the proper external DN.

With this approach, the Java Secure Socket Extension (JSSE) is responsible for authorizing the certificate that is received over a new inbound connection from a external domain. This authorization is based on the agreed upon certificate authorities whose certificates are set up in the local trust store. If the external domain certificate is authorized, it is then the responsibility of the SIP proxy to filter the connections, based on the DN that is associated with the external domain certificate. The proxy also validates outbound connections by ensuring that the DN that is received in the remote server certificate matches the DN configured for the external domain.

The SIP proxy must recognize when identity assertion is in use so that it can inform the SIP container that no message authentication is required over this mutually authenticated connection. This communication is done by adding the P-Preferred-Identity SIP header, which is described in RFC 3325, in all SIP messages that are sent from the proxy to the SIP container that arrive over the authenticated connection. The SIP container only recognizes this header when it is received from a device that resides in the trusted domain, specifically the SIP proxy. It is up to the SIP proxy to remove this header from any inbound messages that are received over any connections to remote devices that are not considered part of the trusted domain. You can also use this header to support the addition of proxy authentication.

Tracing a Session Initiation Protocol proxy server

You can trace a Session Initiation Protocol (SIP) proxy server, starting either immediately or after the next server startup.

About this task

To trace a SIP proxy server, complete the following steps:

1. Start the product, and open the administrative console.
2. In the administrative console, click **Troubleshooting** → **Logs and trace**.
3. Select the name of the server for the SIP proxy server.
4. From the General Properties section, click **Diagnostic Trace Service**.
5. Select one of the following options:

Option	Description
Configuration	To start tracing after the next server startup
Runtime	To start tracing immediately

6. Replace the content of the trace specification with the following code:

```
*=info:com.ibm.ws.sip.*=all:com.ibm.ws.proxy.*=all
```

7. Make sure that the **Enable trace with following specification** check box is checked.
8. Click **Apply** → **Save**.

What to do next

When the changes take effect, SIP proxy server tracing messages display in *WASProductDir/logs/serverName/trace.log* on the SIP proxy server node, where *WASProductDir* is the fully-qualified path name of the directory where the product is installed and *serverName* is the name of the specific instance of the application server that is running the SIP proxy server to be traced.

High availability and workload management with Session Initiation Protocol proxy server

The Session Initiation Protocol (SIP) high availability solution assumes that all messages that belong to the same dialog are handled by the same container. If a container fails, all of the sessions that were handled by that container are picked up by the other servers in that container replication domain, and are activated immediately. All subsequent messages that belong to a session from the failed container are sent to the new container in charge of that session.

Note: The SIP proxy server does not administer transaction-level failover, which are calls that are made in the middle of a transaction. The SIP proxy server administers failover of stable calls, which are calls that are not made in the middle of a transaction.

High availability manages the following:

- Scalability – The ability to add more servers to the cluster to handle increased loads.
- Load balancing – The ability to distribute the load across all of the servers in the cluster so no server is overloaded while there are other servers that are not utilized.
- Fail over – The ability to recover from a failure in one or more of the components in the solution.

The SIP high availability solution uses the following components:

- SIP container - Maintains all of the sessions and launches all of the applications.
- SIP proxy - Manages a large number of client connections, routes incoming messages to the appropriate SIP container, and creates outbound connections to clients and other domains.
- Network Dispatcher - Provides a single IP for the cluster and round-robins between proxies.
- Unified Clustering Framework (UCF) - Communicates routing information between the SIP container and the SIP proxy. Using UCF, the SIP proxy routes messages to the least-loaded SIP container or to a container that is taking over sessions for a failed server.

Note: If you add SIP containers to a cluster while traffic is flowing, you should add them one container at a time so that the system can go through the bootstrap process for the container without draining resources from the entire cluster. If you add one container at a time, only the added container goes through the bootstrap process as opposed to all of the containers in the cluster.

Manage failover in the SIP proxy by:

1. Replicating the data in the sessions between SIP containers so that other containers are able to activate the failed sessions in case of a server failure.
2. Activating the failed sessions on the rest of the servers immediately when a failure is detected because the SIP sessions might have timers associated with them.
3. Routing incoming messages that belong to failed sessions to the new server that is handling the session.

Load balancing with the Session Initiation Protocol proxy server

Load Balancer for IBM WebSphere Application Server can help maximize the potential of your Web site by providing a powerful, flexible, and scalable solution to peak-demand problems. Configuration of a load balancer is critical to prevent any single points of failure at Session Initiation Protocol (SIP) proxy. Configuration of a load balancer is required when more than one SIP proxy is deployed.

Before you begin

Before you begin, complete these tasks.

Install Load Balancer for IBM WebSphere Application Server. See the information provided in the Edge Components Information Center. This information center is available on the WebSphere Application Server Library page.

From the Load Balancer for IBM WebSphere Application Server, complete the steps for setting up server machines for load balancing. Ensure that you set up a loopback address for your operating system. See the Load Balancer Administration Guide in the Edge Components Information Center. Start the Session Initiation Protocol (SIP) proxy server.

About this task

Complete these steps to integrate the SIP proxy server with the Load Balancer.

1. Start the Load Balancer.
 - a. From the command prompt, type `dsserver start`.
 - b. Then type `ladmin` to start the administrative console for the Load Balancer.
 - c. From the administrative console, right click **Dispatcher**, and then select **Connect to Host**.
 - d. Right click on the hostname and select **Start Executor**.
2. Start the configuration wizard for the Load Balancer.
 - a. Select default host.
 - b. Type a cluster address. You should not be able to ping the cluster address before the Executor starts. You must specify this same value for host when you create a user-defined port.
 - c. Type a port number, such as 5060. You must specify this same value for port when you create a user-defined port.
 - d. Add servers to the port. Add each server to which the Load Balancer will proxy traffic. In your configuration, the load-balanced server is the proxy server for your configuration.
 - e. Start an advisor by typing the name of the advisor. For example, for HTTP traffic, start the HTTP advisor. For SIP traffic, start the SIP advisor. The advisor tells the manager whether or not a specific port is accepting traffic.
3. Alias the cluster address on the SIP proxy server loopback adapter. See the topic entitled "Setting up server machines for load balancing" in the *Load Balancer for WebSphere Application Server Administration Guide, Version 6.1*.
4. Configure an IP sprayer from the administrative console.
 - a. From the administrative console, click **Servers > Server Types > WebSphere proxy servers > proxy_server_name > SIP proxy server settings > SIP proxy settings**.
 - b. Under General Properties, in the IP sprayer configuration section, select the checkbox for the IP sprayer from which you want the SIP proxy server to receive traffic: **Enable TCP sprayer**, **Enable SSL sprayer**, or **Enable UDP sprayer**.
 - c. Enter a value for the **Host**. This is the host for your load balancer.
 - d. Enter a value for the **Port**. This is the port for your load balancer.
 - e. Click **Apply**, and then click **Save**.
5. Define SIP proxy server custom properties from the administrative console.

- a. From the administrative console, click **Servers > Server Types > WebSphere proxy servers > proxy_server_name > SIP proxy server settings > SIP proxy settings**.
- b. In the Additional Properties section, click **Custom properties**.
- c. Specify a value for these SIP proxy custom properties, and then click **OK**.

Note: These settings are now available from the SIP proxy settings page in the administrative console and do not require that a custom property be set, however the custom properties are supported. If the custom property is set, the value of the custom property is used even if you set the value in the administrative console.

LBIPAddr

SIPAdvisorMethodName

6. Set up a loopback address that represents the IP address for the load balancer. See the load balancer documentation for information about setting up loopback addresses.
7. Create a user-defined port from the WebSphere Application Server administrative console.
 - a. From the administrative console, click **Servers > Server Types > WebSphere proxy servers > proxy_server_namePorts > New**.
 - b. Select User-defined port.
 - c. Enter SIP_LB_Address for the **Port Name**.
 - d. Enter a value for the **Host**. This is the host for your load balancer.
 - e. Enter a value for the **Port**. This is the port for your load balancer.
 - f. Enter a value for **PROXY_SIP_ADDRESS** This is the actual hostname for the proxy server machine.
 - g. Click **Apply**, and then click **Save**.
8. Modify the SIP proxy transports from the administrative console.
 - a. From the administrative console, click **Servers > Server Types > WebSphere proxy servers > proxy_server_name > SIP container transport chains > UDP_SIP_PROXY_CHAIN > UDPInbound Channel**.
 - b. Under SIP proxy transports, modify **UDP_SIP_PROXY_CHAIN**.
 - c. From the **Port** drop-down list, select **SIP_LB_Address**
 - d. Click **Apply**, and then click **Save**.
9. Restart the proxy server to save your changes.

SIP proxy settings

The SIP proxy settings page contains general configuration items that affect outbound transport configuration, toleration of IP Sprayer devices, and access logging configuration.

To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers > proxy_server_name > SIP proxy settings**.

The **SIP proxy settings** panel allows you to define attributes and policies that control the behavior of the SIP proxy server.

Default cluster

Specifies the default cluster name.

The **Default cluster** field must be set to a valid cluster before any SIP messages are routed through the proxy server. The default cluster indicates which cluster of application servers should receive SIP traffic when there are no cluster selection rules defined, or when none of the existing cluster selection rules match.

Data type	Valid cluster name
------------------	--------------------

Default None

Retry-after header value

Specifies the amount of time, in seconds, that the SIP proxy waits before it returns the SIP request because the SIP containers are overloaded, or the SIP proxy cannot locate available servers to which it can route the request.

Range 1 to 1000
Data type Integer
Default 5

Logging

Specifies whether to enable or disable access logging.

When **Enable access logging** is checked, specifies the access log maximum size and proxy access log path name. When this control is unchecked, access log maximum size and proxy access log settings are disabled.

Data type Boolean
Default Unchecked

Access log maximum size

Specifies the maximum size, in megabytes (MB), of the access log before it rolls over.

Range 1 to 65535
Data type Integer
Default 20

Proxy access log

Specifies the location of the SIP proxy access log.

Range Valid path name
Data type String
Default `$(SERVER_LOG_ROOT)/sipproxy.log`

Container facing network interface

This section contains fields that configure the container-facing network interfaces.

Note: The value asterisk (*) means let the OS decide which interface to use.

UDP interface

Specifies the network interface for all User Datagram Protocol (UDP) data that goes to and from the SIP container. The value for this setting is the hostname or IP address to use for all communications between the SIP proxy and the SIP containers when the network is segmented.

Note: If a value is specified for this setting, you must also specify a value for the UDP port setting.

Data type String
Default *

UDP port

Specifies the UDP port for SIP container communications, such as the specific port required to pass through a firewall that separates the SIP proxy and the SIP containers.

Note: If a value is specified for this setting, you must also specify a value for the UDP interface.

Range	1 to 65535
Data type	Integer
Default	*

TCP interface

Specifies the network interface for all Transmission Control Protocol (TCP) data that goes to and from the SIP container or containers.

Data type	String
Default	*

TLS interface

Specifies the network interface for all Transport Layer Security (TLS) data that goes to and from the SIP container or containers.

Data type	String
Default	*

Load balancer health checking

This section contains fields that configure the load balancer used to load balance the SIP proxy server and perform health checks.

Load balancer members (IP address 1 and 2)

Specifies the IP addresses of the load balancers that source the SIP health checks. The IP addresses (1 and 2) ensure that the message is an actual SIP advisor request before sending a response back to it.

Data type	String
Default	None

SIP health check method name

Specifies the health check method name sent to the SIP proxy from the load balancer.

Data type	String
Default	None

Tolerate a specific number of negative health checks

Specifies whether to allow negative health checks.

Data type	Boolean
Default	False

Maximum negative health checks

Specifies the number of SIP health checks that the load balancer can perform that return negative results, indicating that there is a communication problem between the load balancer and the SIP proxy, before this SIP proxy informs the other SIP proxies that it is no longer processing data.

Range	1 through 10
Data type	Integer
Default	3

SIP network outage detection

Specifies whether to enable or disable outage detection.

When **Enable SIP network outage detection** is checked, specifies the KEEPALIVE interval and maximum KEEPALIVE failures. When this control is unchecked, network outage detection is disabled.

Data type	Boolean
Default	Unchecked

Keep alive interval

Specifies the interval, in milliseconds, at which the KEEPALIVE packets are sent to the SIP containers.

Range	1 through 1000
Data type	Integer
Default	3000

Maximum keep alive failures

Specifies the number of KEEPALIVE requests that are sent without getting a response before considering this server down.

Range	1 through 10
Data type	Integer
Default	3

Overload protection

Specifies whether to enable or disable overload protection.

When **Enable proxy managed overload protection** is checked, specifies the maximum throughput allowed and overload error response. When this control is unchecked, maximum throughput and overload error response fields are disabled.

Data type	Boolean
Default	True

Max throughput factor

Specifies the degree of overload protection at the proxy and is a percentage of the maximum messages per averaging period for the SIP container.

Range	1 through 100
Data type	Integer
Default	90

Overload response code

Specifies the overload response code. When the proxy detects an overload condition, it will respond to the request with the overload response code, if it is set.

Data type	Integer
Default	503

Overload response reason

Specifies the overload response reason phrase. When the proxy detects an overload condition, it will respond to the request with the overload response phrase, if it is set.

Data type	String
Default	Service unavailable

SIP external domains collection

The external domain collection panel provides create, remove and update capabilities for the external domain routing configuration.

To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers > proxy_server_name > SIP proxy settings > External Domains**.

New

Create a new external domain entry. Clicking **New** launches the External Domain Detail panel.

Delete

Used to remove an external domain entry.

Domain

The domain string that is mapped to the associated protocol, host, and port configuration.

Distinguished name

The name that is associated with the external domain. It is used when SSL client authentication is enabled to limit connections from an external domain.

Protocol

This field contains the host name that the SIP Proxy will write to outbound requests so that the receiving agent will connect back to the IP Sprayer.

Host

The host used to make the SIP connection associated with the domain.

Port

The port used to make the SIP connection associated with the domain.

SIP external domains

The external domain detail panel configures the properties for external domain routing.

To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers > proxy_server_name > SIP proxy settings > External Domains > New**.

Domain

The SIP domain that is mapped to the protocol, host, and port that is specified in the fields on this panel. The SIP proxy server matches the domain that is found in the TO header of a SIP message to this value and uses the related information to connect to the specified SIP service.

Range	Valid SIP domain name, with the addition of an optional preceding * as a wildcard.
Setting recommendations	None

Protocol

The protocol that the SIP proxy server uses to connect to the SIP service.

Range	TCP, SSL, and UDP
Default	TCP
Setting recommendations	None

Distinguished name

The name that is associated with the external domain. Used when SSL client authentication is enabled to limit connections from an external domain.

Range	Any string
Default	blank
Setting recommendations	None

Host

The host that the SIP proxy server uses to connect to the SIP service.

Range	Valid host name or IP address
Default	blank
Setting recommendations	None

Port

The port that the SIP proxy server uses to connect to the SIP service.

Range	1 to 65535
Default	blank
Setting recommendations	None

SIP routing rules collection

Routing rules enable an administrator to direct Session Initiation Protocol (SIP) traffic to a specific cluster when there is more than one cluster running SIP applications in a WebSphere Application Server Network Deployment cell.

Routing rules are not needed in cases where there is one cluster running SIP. Rules are only applied to the initial message of a SIP conversation and not all SIP traffic. To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers > *proxy_server_name* > SIP proxy settings > Routing rules**.

New

Clicking **New** launches the Routing Rule Detail Panel for creating a new rule.

Delete

Deletes selected rules.

Enable

Sets the enabled attribute of the selected rules to true.

Disable

Sets the enabled attribute of the selected rules to false.

Set order

Launches the Set Order panel.

Select

Allows the user to select multiple rows to be affected by the Delete, Enable and Disable actions.

Order

The order column represents the order in which the rules are evaluated. This is critical since a SIP message may match more than one rule.

Cluster

The name of the cluster to which the rule will route SIP traffic.

Condition

A concatenation of the list of conditions associated with the rule. Condition is not sorted.

Enabled/Disabled

Indicates whether the rule is enabled, and thus considered for evaluation.

SIP routing rules set order

It is possible that a SIP message can match more than one routing rule but the SIP proxy will stop evaluating at the first match. Routing rules are evaluated in the order that they appear in the configuration file. The set order routing rule panel enables the administrator to change this ordering.

Routing rules are not needed in cases where there is one cluster running SIP. Rules are only applied to the initial message of a SIP conversation and not all SIP traffic. To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers > *proxy_server_name* > SIP proxy settings > Routing rules > Set Order**.

Move Up

Clicking **Move Up** moves the selected rule up one row.

Move Down

Clicking **Move Down** moves the selected rule down one row.

Select

Enables the user to select one row that will be acted on by the Move Up and Move Down actions.

Cluster

The name of the cluster that the rule will direct SIP requests to.

Condition

A concatenation of the list of conditions that are associated with the rule.

Enabled

Indicates whether the rule is enabled and thus considered for evaluation.

SIP routing rules detail

The routing rule detail panel provides the ability to create and modify individual rules. Since the structure of conditions is complex, the user will need to utilize a different set of panels to change the conditions associated with a rule.

Routing rules are not needed in cases where there is one cluster running SIP. Rules are only applied to the initial message of a SIP conversation and not all SIP traffic. To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers > *proxy_server_name* > SIP proxy settings > Routing rules > New**.

Enabled

Enables a rule to be removed from consideration without requiring that it be deleted. This field is checked by default.

Data type	Boolean
Default	true
Setting recommendations	none
Setting dependencies	none

Target Cluster

The name of the cluster that the SIP message will be routed to if the message attributes match the conditions.

Data type	String
Default	First cluster in the list.
Range	List of existing clusters or “null cluster” to indicate that a request should be rejected.
Setting recommendations	None
Setting dependencies	None

SIP rule condition collection

Each rule contains a list of conditions that are combined using a logical AND operator. This means that all of the conditions need to be true for the rule to apply. This panel enables the user to manage the set of conditions.

To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers > proxy_server_name > SIP Proxy Server Settings > Routing rules > target_cluster > Conditions**.

New

Create a new condition. Clicking **New** launches the Rule Condition Detail panel.

Delete

Removes a condition from the collection.

Type

Indicates which aspect of a SIP message the condition applies to.

Value

The value that will be compared to the aspect of the SIP message indicated by type.

SIP rule condition detail

Routing rules direct SIP requests that match the condition to the selected cluster. The condition setting specifies which SIP messages match the rule. Rules only apply to the initial message of a SIP conversation and not all SIP traffic. Use this panel to create or modify a rule.

Note: Routing rules are not needed in cases where there is only one cluster running SIP.

To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers > proxy_server_name > SIP Proxy Server Settings > Routing rules > target_cluster > Conditions > New**.

Condition type: Method

Select this option to use **Condition type: Method**. Selecting this condition type activates the Condition value field.

Default Selected

Condition value

Specifies a fixed list of predefined method types when **Condition type: Method** is selected.

Range INVITE, REGISTER, REFER, SUBSCRIBE, UNSUBSCRIBE, PUBLISH, MESSAGE, OPTIONS, INFO

Default INVITE

Condition type: Defaults

Select this option to use **Condition type: Defaults**. Selecting this condition type activates the Type and Condition value fields. You can select between the fixed condition list, **Type**, and free form entry, **Condition value**, for this condition type.

Default Not selected

Type

Specifies a fixed list of predefined attributes when **Condition type: Defaults** is selected.

Range TO, FROM, REQUEST URI, SOURCE ADDRESS, DESTINATION ADDRESS

Default TO

Condition value

Specifies a generic SIP message condition when **Condition type: Defaults** is selected.

Default Empty string

Condition type: Header

Specifies whether to enable the header for a condition rule. You must specify a header name and header value pair if this setting is enabled.

Default Not selected

Header name

Specifies the header name for the condition rule. If the value specified for this setting is *Header1*, then a packet sent to the proxy with this name specified as the header name satisfies the condition for the defined rule.

Default Empty string

Header value

Specifies the header value for the condition rule. If the value specified for this setting is *Value1*, then a packet sent to the proxy with this value specified as the header value satisfies the condition for the defined rule.

Default Empty string

SIP proxy inbound channel detail

This panel displays the configuration details of the SIP proxy inbound channel.

To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers > proxy_server_name > Transport chain > SIP proxy inbound channel** .

Transport channel name

Specifies the name of the SIP proxy inbound channel.

The name field cannot contain the following characters: # \ / , : ; " * ? < > | = + & % ' .

This name must be unique across all channels in a WebSphere Application Server environment. For example, a SIP proxy inbound channel and a TCP transport channel cannot have the same name if they reside within the same system.

Data type String

Discrimination weight

Specifies the priority this channel has in relation to the other channels in this transport chain. This property is only used when port sharing is enabled and the channel chain includes multiple channels to which it might forward data. The channel in the chain with the lowest discrimination weight is the first one given the opportunity to look at incoming data and determine whether or not it owns that data.

Data type Positive integer
Default 0

Troubleshooting the proxy server

This topic helps you to solve problems that you might encounter with your proxy server.

About this task

Proxy server errors are logged in the SystemOut.log, proxy.log, or local.log files. Consult the following list if you are having problems with your proxy server.

- **The proxy server was created successfully, but I am unable to start it.** Check the SYSOUT file for port conflicts. Use the `netstat -a` command to see if any of the endpoints that are associated with the proxy server are already being used. You can find the ports in the administrative console by clicking **Servers > Proxy servers > <server_name> > Ports**.

If the proxy server fails to start when attempting to start it as a non-privileged user on UNIX systems, check for the following message in the logs:

```
ChannelFrame E CHF0029E: Error initializing chain HTTPS_PROXY_CHAIN because of
exception
com.ibm.wsspi.channel.framework.exception.RetryableChannelException: Permission
denied
TCPPEndpoint E TCPC0003E: TCP Channel TCP_7 initialization failed. The socket
bind failed for host * and port 80. The port may already be in use.
```

Change the ports of the PROXY_HTTP_ADDRESS and PROXY_HTTPS_ADDRESS transport chains to values greater than 1024.

- **The proxy server routes requests to the Web container over an administration port.** The proxy server is located in front of several Web containers. The configuration requires that the Web containers listen to the non-default ports such as 9061, 9081, and so on. This scenario is the default case when multiple application servers are on the same machine, which forces new and different ports to be used in the configuration. In this scenario, the proxy server might route an application request to the Web container over the administration port of 9061, instead of using the expected port of 9081.
Add the listening port numbers of the Web container to the virtual host that is associated with the target application. This process will ensure that the proxy server routes the request to the Web container over the correct port number.
- **The proxy server started, but I am unable to access the application resources through the endpoints for the proxy server.** Ensure that the endpoints for the proxy server are among the host aliases in the virtual host that are associated with the application.
- **The proxy server routes to another core group.** Verify that core group bridges exist between the core groups in the cell, and that the processes that are chosen to be bridges are restarted. If there is a firewall between the core group, verify that the correct ports are open for core group bridge traffic.
- **The proxy server is unable to route requests to another cell.** Review the core group bridge settings. If the HMGR0149E error message is logged on any server, usually on a server acting as a core group bridge, then the security settings for the cell need to be modified to allow communication.
- **Receiving a blank page when making a request to the proxy.** Consider the following actions:

- Update the virtual host. Ensure that the target application and routing rule are assigned to a virtual host that includes the proxy server listening ports (default: HTTP 80, HTTPS 443). Add the proxy server listening ports to the application, or routing rule virtual host, or use the proxy_host virtual host.
- Stop the conflicting process. Check your system to ensure that no other process (for example, Apache, IBM HTTP Server, and so on) is running that uses the proxy server ports (default: HTTP 80, HTTPS 443). If this problem occurs, the proxy server seems to start normally, but is unable to receive requests on the affected listening port. Check your system as follows:
 1. Stop the proxy server.
 2. Query your system using **netstat** and **ps** commands to determine if an offending process is using the port on which the proxy server is listening.
 3. If an offending process is found, stop the process and configure your system so that the process is not started during system startup.
- Enable proxy routing. Ensure that proxy routing is enabled for the Web module of the application. Proxy routing is enabled by default, so if no proxy properties are modified, disregard this solution. Otherwise, see “Customizing routing to applications” on page 230 for instructions on modifying the proxy properties.
- Test direct request. Ensure that the target application is installed by making a request directly to the application server. If a response is not received, then the problem is with the application server and not the proxy server. Verify this case by going through the proxy server after you can receive a response directly from the application server.
- **HTTP 404 (File not found) error received from the proxy server.** Consider the following actions:
 - Update the virtual host. Ensure that the target application and routing rule are assigned to a virtual host that includes the proxy server listening ports (default: HTTP 80, HTTPS 443). Add the proxy server listening ports to the application, or routing rule virtual host, or use the proxy_host virtual host.
 - Enable proxy routing. Ensure that proxy routing is enabled for the Web module of the application. Proxy routing is enabled by default, so if no proxy properties are modified, disregard this solution. Otherwise, see “Customizing routing to applications” on page 230 for instructions on modifying the proxy properties.
 - Test direct request. Ensure that the target application is installed by making a request directly to the application server. If a response is not received, then the problem is with the application server and not the proxy server. Verify this case by going through the proxy server when you can receive a response directly from the application server.
- **Unable to make Secure Sockets Layer (SSL) requests to application or routing rule.** Ensure that the virtual host of the application or routing rule includes a host alias for the proxy server SSL port (default: 443).
- **Unable to connect to the proxy server...request times out.** Stop the conflicting process. Check your system to ensure that no other process (for example, Apache, IBM HTTP Server, and so on) is running that uses the proxy server ports (default: HTTP 80, HTTPS 443). If this situation occurs, the proxy server seems to start normally, but is unable to receive requests on the affected listening port. Check your system, as follows:
 1. Stop the proxy server.
 2. Query your system using **netstat** and **ps** commands to determine if an offending process is using a port on which the proxy server is listening.
 3. If an offending process is found, stop the process and configure your system so that the process is not started during system startup.
- **Did not receive a response from the error page application when the HTTP error occurred (for example, 404).** Ensure that the error page URI is entered correctly. Also, make sure that the Handle remote errors option is selected if you are handling HTTP error responses from back-end servers. For more detailed information, refer to “Overview of the custom error page policy” on page 239 and the custom error page policy section of “Proxy server settings” on page 191.
- **What packages do I enable when tracing the proxy server?** All of the following packages are not needed for every trace, but if unsure, use all of them:

- *=info
 - WebSphere Proxy=all
 - GenericBNF=all
 - HAManager=all
 - HTTPChannel=all
 - TCPChannel=all
 - WLM*=all
 - DCS=all
 - ChannelFrameworkService=all
 - com.ibm.ws.dwl.*=all
 - com.ibm.ws.odc.*=all
- **How do I enable SSL on/off load?** SSL on/off load is referred to as the transport protocol in the administrative console, and transport protocol is a Web module property. Refer to “Customizing routing to applications” on page 230 to see how to configure Web module properties. No SSL on/off load or transport protocol properties exist for routing rules because the transport protocol is inherent to the generic server cluster that is specified in the routing rule.
 - **When fronted by IBM HTTP Server or a Web server plug-in, how do I configure the proxy server so I do not have to add a port for it to the virtual host?** For the proxy server to trust the security-related information that is included in a request, such as private headers that the product adds, add the originator of the request to the proxy server trusted security proxies list. For example, add an IBM HTTP Server or a Web server plug-in that sends requests to the proxy server, to the proxy server trusted security proxies list. The Web server plug-in can then send product added private header information that, contains the virtual host information of a request. If the proxy does not trust these private headers from the Web server plug-in, or from any client, the proxy server adds its own private headers, which requires the addition of proxy server HTTP and HTTPS ports to the virtual host. Typically, when a Web server plug-in is used with the proxy server, the intent is to use the proxy server as a back-end server. Therefore, you must add the plug-in as a trusted security proxy to avoid having to expose the proxy server ports. “Routing requests from a plug-in to a proxy server” on page 236 provides more information about configuring a Web server plug-in to use with the proxy server. “Proxy server settings” on page 191 provides more information about setting up trusted security proxies.
 - **The proxy server seems to hang under stress, or Too Many Files Open exceptions display in ffcd or SystemErr.log.** Under high connection loads, the number of file system descriptors might become exhausted and the proxy server may seem to hang and drop Too Many Files Open exceptions in the ffcd directory or in the SystemError.log file because it is unable to open a socket. To alleviate this problem, modify one or more of the following parameters at the operating system level, and at the proxy server level to optimize the use of connections for the proxy server:

– **Windows** **Operating system tuning for Windows 2000, 2003, and XP**

- TcpTimedWaitDelay - Determines the time that must elapse before TCP/IP releases a closed connection and reuse its resources. This interval between closure and release is known as the TIME_WAIT state, or twice the maximum segment lifetime (2MSL) state. During this time, reopening the connection to the client and server costs less than establishing a new connection. By reducing the value of this entry, TCP/IP releases closed connections faster and can provide more resources for new connections. Adjust this parameter if the running application requires rapid release, the creation of new connections, or an adjustment because of a low throughput caused by multiple connections in the TIME_WAIT state.

View or set this value as follows:

1. Use the **regedit** command and access the HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\TCPIP\Parameters registry subkey to create a new REG_DWORD value named TcpTimedWaitDelay.
2. Set the value to decimal 30, which is Hex 0x0000001e. This value sets the wait time to 30 seconds.

3. Stop and restart your system.

Default value	0xF0, which sets the wait time to 240 seconds (4 minutes).
Recommended value	A minimum value of 0x1E, which sets the wait time to 30 seconds.

- MaxUserPort - Determines the highest port number that TCP/IP can assign when an application requests an available user port from the system. View or set this value as follows:
 1. Use the **regedit** command, access the HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\TCPIP\Parameters registry subkey, and create a new REG_DWORD value named MaxUserPort.
 2. Set this value to at least decimal 32768.
 3. Stop and restart your system.

Default value	None
Recommended value	At least decimal 32768.

- **Linux** Operating system tuning for Linux

- timeout_timewait parameter - Determines the time that must elapse before TCP/IP releases a closed connection and can reuse its resources. This interval between closure and release is known as the TIME_WAIT state, or twice the maximum segment lifetime (2MSL) state. During this time, reopening the connection to the client and server costs less than establishing a new connection. By reducing the value of this entry, TCP/IP can release closed connections faster, providing more resources for new connections. Adjust this parameter if the running application requires rapid release, the creation of new connections, and a low throughput due to many connections sitting in the TIME_WAIT state.

View or set this value by issuing the following command to set the timeout_timewait parameter to 30 seconds:

```
echo 30 > /proc/sys/net/ipv4/tcp_fin_timeout
```

- Linux file descriptors (ulimit) - Specifies the number of open files that are supported. The default setting is typically sufficient for most applications. If the value set for this parameter is too low, a file open error, memory allocation failure, or connection establishment error might display. View or set this value by checking the UNIX reference pages on the ulimit command for the syntax of different shells. Set the ulimit command to 65535 for the KornShell shell (ksh), by issuing the ulimit -n 65535 command. Use the ulimit -a command to display the current values for all limitations on system resources.

Default value	1024
Recommended value	65535

- **AIX** Operating system tuning for AIX

- TCP_TIMEWAIT - Determines the time that must elapse before TCP/IP releases a closed connection and can reuse its resources. This interval between closure and release is known as the TIME_WAIT state, or twice the maximum segment lifetime (2MSL) state. During this time, reopening the connection to the client and server costs less than establishing a new connection. By reducing the value of this entry, TCP/IP can release closed connections faster, providing more resources for new connections. Adjust this parameter, if the running application requires rapid release or the creation of new connections, or if a low throughput occurs due to many connections sitting in the TIME_WAIT state.

View or set this value by issuing the following command to set the TCP_TIMEWAIT state to 15 seconds:

```
/usr/sbin/no -o tcp_timewait =1
```

- AIX file descriptors (ulimit) - Specifies the number of open files that are permitted. The default setting is typically sufficient for most applications. If the value set for this parameter is too low, errors can occur when opening files or establishing connections, and a memory allocation error might display. To prevent WebSphere Application Server from running short on resources, remove the upper limits (ulimit) for resources on the user account on which the WebSphere Application Server process runs.

View or set this value by changing the ulimit settings as follows:

1. Open the command window.
2. Type **smitty users** to open the AIX configuration program.
3. Select **Change** or **Show Characteristics** of a user.
4. Type the name of the user account on which the WebSphere Application Server runs.
5. Press **Enter**.
6. Change the following settings to the indicated value:

Soft FILE Size	-1
Soft CPU Time	-1
Soft STACK Size	-1
Soft CORE File Size	-1
Hard FILE Size	-1
Hard CPU Time	-1
Hard STACK Size	-1
Hard CORE File Size	-1

7. Press **Enter** to save changes.
8. Log out and log into your account.
9. Restart the product.

Default value	2000
Recommended value	unlimited

– **Solaris** Operating system tuning for Solaris

- **TCP_TIME_WAIT_INTERVAL** - Notifies TCP/IP on how long to keep the connection control blocks closed. After the applications complete the TCP/IP connection, the control blocks are kept for the specified time. When high connection rates occur, a large backlog of the TCP/IP connections accumulate and can slow server performance. The server can stall during certain peak periods. If the server stalls, the **netstat** command shows that many of the sockets that are opened to the HTTP server are in the **CLOSE_WAIT** or **FIN_WAIT_2** state. Visible delays can occur for up to four minutes, during which time the server does not send any responses, but CPU utilization stays high with all of the activities in system processes.

View or set this value by using the **get** command to determine the current interval and the set command to specify an interval of 30 seconds. For example:

```
ndd -get /dev/tcp tcp_time_wait_interval  
ndd -set /dev/tcp tcp_time_wait_interval 30000
```

Default value	240000 milliseconds, which is equal to 4 minutes.
Recommended value	60000 milliseconds

– **Proxy server tuning**

- Persistent requests - A persistent request is one that is sent over an existing TCP connection. You can maximize performance by increasing the number of requests that are received over a TCP connection from a client. The value should represent the maximum number of embedded objects, for instance GIF and so on, in a Web page +1.

View or set this value in the WebSphere Application Server administrative console by clicking **Servers > Proxy Servers > *server_name* > Proxy server transports > HTTP_PROXY_CHAIN/HTTPS_PROXY_CHAIN**

Default value	100
Recommended value	A value that represents the maximum number of embedded objects in a Web page + 1.

- Outbound connection pool size - The proxy server pools outbound connections to target servers and the number of connections that resides in the pool is configurable. If the connection pool is depleted or empty, the proxy server creates a new connection to the target server. Under high concurrent loads, increase the connection pool size should to a value of the expected concurrent client load to achieve optimal performance.

View or set this value in the WebSphere Application Server administrative console by clicking **Servers > Proxy Servers > *server_name* > HTTP Proxy Server Settings**. In the Content Server Connection section, increase the maximum connections per server field to a value that is equal to or greater than the expected maximum number of connected clients. Save your changes, synchronize the changes to the proxy server node, and restart the proxy server.

Recommended value	Value consistent to the expected concurrent client load.
-------------------	--

- Outbound request time-out - Often times, the back-end application servers that are fronted by the proxy server may be under high load and may not respond in an adequate amount of time, therefore the connections on the proxy server may be tied up from waiting for the back-end application server to respond. Alleviate this by configuring the amount of time the proxy server waits for a response from the target server. This is the Outbound Request Time-out value. By managing the amount of time the proxy server waits for a slow back-end application server, connections are freed up faster and used for other request work.

View or set this value in the administrative console by clicking **Servers > Server Types > WebSphere proxy servers > *proxy_server_name* > HTTP Proxy Server Settings**. In the Content Server Connection section, set Outbound Request Time-out to a value that represents the acceptable response time from the point of view of the client.

Default value	120
Recommended value	A value that represents the acceptable response time from the point of view of the client.

Troubleshooting request routing and workload management through the proxy server

This section provides information for how to troubleshoot request traffic that flows through the proxy server.

Before you begin

You will need to know the machines and nodes that will belong to the proxy server cluster, because the product needs to be installed on those machines. You will also need to know the URL for the applications, application deployment, and cluster definition details. The proxy server should be started.

About this task

You can use the proxy server MBean to determine how requests are routed to applications, and subsequently, to a particular application server. If the request is being routed incorrectly, you can disable routing to specific applications or reconfigure the routing rules.

1. Obtain the Dynamic Route MBean for the proxy server and invoke the operation to generate routing information for the URI. Start **wsadmin** and get all of the Dynamic Route MBeans as follows:

```
$AdminControl queryNames  
type=DynamicRoute,*
```

```
set routembean <cut and paste the MBean Identifier from the previous command output>
```

```
$AdminControl invoke $routembean debugRouting {http://*/urlpattern all}
```

Use an asterisk (*) to match all of the virtual hosts, or explicitly specify a virtual host. For example, `http://proxy_name:80/urlpattern`. The **set routembean** command should correspond to the MBean from the output of the previous command.

The proxy server will start generating routing-related information for all subsequent HTTP requests that match the specified virtual host and URL pattern to the `SystemOut.log` file.

2. Send representative workload traffic through the proxy server.
3. Analyze the routing information in the proxy server `SystemOut.log` file.
4. Make required changes to application routing to enable or disable routing through the proxy server, using the administrative console, by clicking **Applications > Enterprise Applications**.
5. Repeat steps two through four until the routing of all requests are satisfied.
6. Disable gathering routing information using **wsadmin** as follows:

```
$AdminControl invoke $routembean  
stopDebugRouting
```

Results

The proxy server and the applications are correctly configured for external access.

Session Initiation Protocol overload protection

Session Initiation Protocol (SIP) overload protects the system from two overload conditions: memory overload and CPU overload. Container managed overload protection (CMOP) and proxy managed overload protection (PMOP) allow for real-time protection based on the overload settings information.

SIP container managed overload protection

In a stand-alone server deployment, SIP container managed overload protection (CMOP) provides the only defense against both memory and CPU overload conditions. An administrator can set up several static thresholds using SIP container settings. When these thresholds are exceeded, the container begins to drop new requests by responding to any requests that initiate new dialogs with a 503 response until the container is no longer overloaded. This includes settings that affect memory and CPU usage.

In an ND deployment, CMOP allows the container to use the same SIP container settings to notify the proxy server when the container is in an overloaded state. After the proxy server receives this notification, it begins to drop new requests instead of forwarding them to the containers. All memory overload conditions in ND are prevented by CMOP regardless of the configuration.

Overload protection is calculated based on the virtual memory settings and the maximum throughput that a container can handle. You can specify a value for the following SIP container settings from the administrative console for CMOP.

- Maximum application sessions

- Maximum messages per averaging period
- Maximum response time
- Maximum dispatch queue size

You can also specify a value for these SIP container custom properties for CMOP.

- `thread.message.queue.max.size`
- `weight.overload.watermark`

SIP proxy managed overload protection

SIP Proxy Managed overload protection (PMOP) is the best line of defense against container overload. In a typical deployment, CMOP alone does not provide optimal results. The following conditions might occur if CMOP is deployed without deploying PMOP at the proxy server.

- The on or off mechanism might become too granular
- Admission rates might fluctuate
- An absolute cap on load might be difficult to establish
- Unstable loads might be sent to the containers

When PMOP is deployed, the proxy server utilizes admission rate controllers for each container. When a container is overloaded, instead of either accepting or rejecting new load for a period of time, new workload can be sent to the backend containers without completely shutting down the flow of new traffic. This allows the proxy server to offer a consistent load to the containers without exceeding the maximum values for the container settings.

The SIP proxy server calculates a maximum value for message throughput to each backend container based on a percentage of the configured Maximum Messages per Averaging Period (MMAP) setting specified for the container. The maximum value for message throughput is called the Maximum Throughput Factor (MTF).

MTF is disabled by default and can only be enabled by specifying a value for the `maxThroughputFactor` custom property. The value specified for the MTF custom property should be less than 100 percent to prevent CPU overload at the container. For example, you might set this value to 90 percent.

When the value of the MTF custom property is set to less than 100 percent, the total throughput to the container should never exceed the maximum value specified for the MMAP container setting. This process protects the container from handling excessive loads when coming out of an overloaded condition.

The MTF value should always be specified when stable and accurate overload protection is required. Specifying the MTF setting provides the best results for loads that range up to twice the capacity of the system. You should consider your system capacity when configuring for overload protection.

The algorithm utilized by the SIP proxy server relies on several features to provide stable and accurate loads to the backend container.

- Per-server rate control managed at the proxy server
- Auto-adjusting, per second admission rate controller
 - Ratio of in-dialog to non-dialog averages used to control rate
 - Auto-rate reduction when in an overloaded state
- Ability to absorb fast load transitions
- Burst tolerance to allow for short occasional load burst without triggering overload
- Stabilization control to prevent excessive overload when there is a transition in the cluster

You can specify a value for the following SIP proxy server custom properties for PMOP.

- burstResetFactor
- deflatorRatio
- dropOverloadPackets
- inDialogAveragingPeriod
- maxThroughputFactor
- outDialogAveragingPeriod
- perSecondBurstFactor
- proxyTransitionPeriod
- sipProxyStartupDelay

Configuring SIP quorum support using the default core group

You can configure quorum to avoid inconsistency among Session Initiation Protocol (SIP) containers, or repeated errors from a proxy server. If quorum is enabled before a network partition occurs, the correct routing decisions can be made. A network partition can occur when a set of SIP containers are disconnected from the network and then reconnected.

Before you begin

Determine whether you want to configure the quorum feature for an entire core group, or for specific clusters within a core group. The topic *Implications of high availability group policy settings* provides additional information about quorum functionality.

About this task

All SIP containers publish a set of unique identifiers (IDs) to the SIP proxy server. Each ID represents a set of SIP sessions. The SIP proxy server is able to make the correct routing decisions based on the one-to-one mapping of these IDs to the SIP containers.

A network partition occurs whenever a network device within the topology of the cell fails. As a result, some portion, or partition, of the cell disconnects from the other portion, or partition.

Note: If a network partition evenly splits the cluster members, half of the cluster members are arbitrarily selected to be in the quorum. This split might cause a restart of the partition while it is still connected to clients. Therefore, you should configure your system to minimize evenly split partitions. You should create a set of three groupings: three data centers, three blade centers, and three groupings of cluster members.

Whenever network connectivity is interrupted, a backup SIP container takes ownership of all IDs that were managed by the disconnected SIP container. The backup container then publishes ownership of these IDs to the SIP proxy server so that the proxy server can make the correct routing decisions.

When the network connection for the primary container is restored, the primary container begins publishing ID ownership to the SIP proxy server to indicate that it owns the same IDs as the backup container. If the SIP proxy server has two destinations for each ID, it is impossible for the SIP proxy server to consistently make the correct routing decisions. To avoid this problem, you must configure the SIP quorum feature for your proxy servers before a network partition occurs.

The SIP quorum feature can be enabled for an entire core group, such as `DefaultCoreGroup`, or for specific clusters within a core group. When the quorum feature is enabled on the Default SIP Quorum core group policy for a core group, the feature is enabled for all clusters in the core group. If the SIP quorum feature is only enabled for some of the clusters in a core group, then you must modify the default policy, and create a duplicate core group policy setting to enable quorum support for those clusters.

You must enable the Default SIP Quorum core group policy, or configure a duplicate core group policy with quorum enabled for each core group that requires quorum support. The purpose of the high availability group that uses the core group policy is to track quorum between the SIP containers, or SIP proxy servers, in a cluster.

Complete these steps to configure the quorum feature using an additional SIP quorum core group policy.

Note: Only one cluster name can be applied to a policy. Repeat this procedure for each cluster that requires the SIP quorum feature to create a new policy with the appropriate match criteria. Each SIP quorum policy should have at the most three match criteria.

1. In the administrative console, click **Servers > Core Groups > Core group settings > *core_group_name***.
2. Click **Policies > New**.
3. Select **All active policy**, and then click **Next**.
4. Specify a unique name for the policy in the **Name** field, and then enter a description in the **Description** field.
5. Select **Quorum**, and then click **OK**. You will receive a warning message that indicates that you must define at least one match criteria for this policy.
6. In the Additional Properties section, click **Match criteria > New**.
7. Specify *policy* in the **Name** field, and `AllActiveQuorumPolicy` in the **Value** field. Both of these fields are case sensitive, and both values must be entered as shown in this step.
8. Click **Apply**.
9. Click the name of your policy in the navigation breadcrumb on this administrative console page, and then, in the Additional Properties section, click **Match criteria > New** again.
10. This time, specify *type* in the **Name** field, and `SIP_QUORUM` in the **Value** field. As previously mentioned, both of these fields are case sensitive, and both values must be entered as shown in this step.
11. Click **Apply**.
12. Perform the following actions if you only want this new policy to apply to a specific cluster within the core group. Skip this step if you want this new policy to apply for the entire core group.
 - a. Click the name of your policy in the navigation breadcrumb on this administrative console page, and then, in the Additional Properties section, click **Match criteria > New** again.
 - b. This time, specify `IBM_hc` in the **Name** field, and the name of a cluster, to which you want to apply this policy, in the **Value** field. Both of these fields are case sensitive. Therefore, make sure that you enter the cluster name exactly as it is defined in the core group.
 - c. Click **Apply**.
13. Click **Save** to save your configuration changes.
14. Restart the affected servers.

If the new SIP quorum core group policy applies to the entire core group, you must restart every server that is part of that core group.

If the new SIP quorum core group policy only applies to a specific cluster within the core group, you must restart all of the members of that cluster.
15. Repeat these steps if you are only enabling SIP quorum for specific clusters in the core group.

Only one cluster name can be associated with a policy. If you have multiple clusters in this core group, for which you want to enable SIP quorum, but you are not enabling SIP quorum for the entire core group, you must create an entirely new policy for each cluster for which you are enabling SIP quorum. For each new policy, you must specify the three match criterias `type=SIP_QUORUM`, `policy=AllActiveQuorumPolicy`, and `IBM_hc=cluster_name`.

Results

The majority of the members that are included in a SIP container or a proxy server cluster must be started before quorum is achieved. The members of a proxy server cluster do not start to listen for SIP requests until quorum is achieved.

When a SIP proxy server is disconnected from the network, a SIP proxy server in the minority partition of the cell continues to handle SIP requests. The minority partition of a cell is the partition that has connectivity to the fewest cluster members.

When a SIP container is disconnected from the network, the SIP container in the minority partition is automatically restarted to clear all information about the logical partitions that were previously managed. The SIP containers in the majority partition take ownership of the logical partitions from the disconnected SIP container and publish these IDs to the SIP proxy servers.

Configuring the SIP proxy for network outage detection

When SIP traffic and the high availability (HA) manager traffic are not on the same network, failures that affect SIP traffic might not be detected by the HA manager, resulting in loss of SIP traffic. You can configure network outage detection on the same network as the SIP traffic to ensure that failures that affect SIP traffic can be detected by the HA manager.

About this task

Define these settings for network outage detection to ensure that SIP network failures are detected by the HA manager.

1. From the administrative console, select **Servers > Server Types > WebSphere proxy servers > *proxy_server_name* > SIP proxy settings**. Scroll to the SIP network outage detection settings on this page.
2. Select **Enable SIP network outage detection** and enter the **Keep alive** interval. The proxy will send keep alive messages to the server at the configured keep alive interval.
3. Specify the **Maximum keep alive failures** to send before a server failure response is sent.

Results

If the proxy misses the number of responses configured as `keepAliveFailures`, or the server misses that number of keep alive requests, it will determine that SIP traffic should no longer be sent between the proxy and the server. The proxy will stop sending SIP messages to that server, and the server will restart and wait for connectivity to be restored before resuming operation.

Administering proxy actions

You can create new proxy actions or you can administer an existing proxy action. A proxy server action is an event that is performed when an HTTP request or an HTTP response is received by the a proxy virtual host. Some examples of proxy server actions include caching actions, rewriting actions, compression actions, header modification actions, and routing actions.

Before you begin

A proxy action cannot be performed unless it is associated with a proxy rule expression. A proxy rule expression is only evaluated when it is associated with a proxy virtual host. A proxy action can be created or administered without a proxy rule expression or a proxy virtual host, but it cannot be used without them.

About this task

Proxy actions are associated with proxy rule expressions. If a proxy rule expression evaluates to true, then all of the proxy actions specified in the proxy rule expression configuration are performed. Complete these steps to create a new proxy action or to administer an existing proxy action for the proxy server.

1. From the administrative console, select **Servers > Server Types > WebSphere proxy servers > *proxy_server_name* > Proxy actions**.
2. Optional: Manage an existing proxy action configuration.
 - a. Click *proxy_action_name* if you want to view or modify the settings for an existing proxy action. If you have made changes to the proxy action configuration, click **OK** to save the changes.
 - b. Click **Delete** to remove an existing proxy action that has been selected.
3. If you want to create a caching proxy action, then click **New Caching Action**.
 - a. Enter the name of the proxy action in the Action name field. The Action name is used as the unique identifier for this proxy action configuration. The Action name must be unique with the cell and cannot include any of the following characters: # \ / , ; " * ? < > | = + & % '.
 - b. Select the **Enable caching** check box to enable caching.
 - c. Enter the value in seconds for the **Default expiration** field. The Default expiration field specifies the amount of time, in seconds, before a cached response expires.
 - d. Enter the value in seconds for the **Last modified** factor. The Last modified factor field specifies the amount of time before a response is cached if the response does not have explicit HTTP expiration headers.
 - e. Click **OK** to save the proxy action configuration.
4. If you want to create a compression action, then click **New HTTP Request Compression Action** to create an action for requests, or click **New HTTP Response Compression Action** to create an action for responses.
 - a. Enter the name of the proxy action in the **Action name** field. The Action name is used as the unique identifier for this proxy action configuration. The Action name must be unique with the cell and cannot include any of the following characters: # \ / , ; " * ? < > | = + & % '.
 - b. Select the appropriate compression type from the Compression type menu.
 - c. Select the context-types for which compression should occur. You can selected multiple content-types to be compressed. Click **New** to add a new content type to the list. If you want to remove a content type from the list, then select the content type to be removed, then click **Delete**.
 - d. Click **OK** to save the proxy action configuration.
5. If you want to create a header action, click **New HTTP Request Header Action** to create an action for requests, or click **New HTTP Response Header Action** to create an action for responses.
 - a. Enter the name of the proxy action in the **Action name** field. The Action name is used as the unique identifier for this proxy action configuration. The Action name must be unique with the cell and cannot include any of the following characters: # \ / , ; " * ? < > | = + & % '.
 - b. Enter the name of the HTTP header to be modified in the **Header name** field.
 - c. Select the appropriate action from the **Header modify action** menu. The available header modify actions include:
 - Set
 - Append
 - Edit
 - Remove
 - d. Enter the value to be used for the Header modify action in the **Header value** field.
 - e. Enter an expression to be performed on the header value in the **Header value expression** field. The Header value expression is evaluated and the modify action is performed if the evaluation returned a match.

- f. If you are creating a HTTP Request Header action, then select the appropriate methods for the Header modify action being performed. You can select multiple methods. Click **New** to add a new method to the list. If you want to remove a method from the list, then select the method to be removed and click **Delete**.
 - g. If you are creating a HTTP Response Header action, then select the appropriate status codes for the Header modify action being performed. You can selected multiple status codes to be included. Click **New** to add a new status code to the list. If you want to remove a status code from the list, select the status code to be removed and click **Delete**.
 - h. Click **OK** to save the proxy action configuration.
6. If you want to create a rewriting action proxy action, then click **New Rewriting Action**.
 - a. Enter the name of the proxy action in the **Action name** field. The Action name is used as the unique identifier for this proxy action configuration. The Action name must be unique with the cell and cannot include any of the following characters: # \ / , ; " * ? < > | = + & % '.
 - b. Select the type of rewriting action to be performed in the Rewriting action types menu. The follow elements can be rewritten using this type of proxy action:
 - Absolute URL response
 - Redirect location header
 - Redirect status code
 - Relative URL response
 - Set cookie domain
 - Set cookie path
 - c. Enter the subject URL pattern to be rewritten in the **From pattern** field.
 - d. Enter the resulting URL pattern after the rewrite has occurred in the **To pattern** field.
 - e. Optional: Select **Enable passive rewrite** to defer the rewriting of the URI until the subsequent request for that URI is sent by the client.
 - f. Optional: In the **Cookie name** field, enter the name of the cookie for which the domain or the path are to be rewritten. This field is only valid for the Set-Cookie type of rewriting actions.
 - g. Optional: In the **Limit URL pattern** field, specify a constraint on the URL patterns to rewrite in the response message. Limiting the URL pattern prevents the proxy server from rewriting all URL patterns in the response message of a certain page. This field is only valid for the absolute URL response action type or the relative URL response action type.
 - h. Optional: In the **Limit Cookie domain** field, specify a constraint to limit the rewriting of the cookie domain to only a set of specified domains. If no domains are specified, then all domains are rewritten. This field is only valid for the Set-Cookie type of rewriting action.
 - i. Optional: In the **Limit Cookie path** field, specify a constraint to limit the rewriting of the cookie path to only a set of specified paths. If no paths are specified, then all the paths are rewritten. This field is only valid for the Set-Cookie type of rewriting action.
 - j. Click **OK** to save the proxy action configuration.
 7. If you want to create a routing proxy action, then click one of the following: **New Application Server Route**, **New Generic Server Cluster Route**, **New Fail Route**, **New Redirect Route**, or **New Local Route**.
 - a. Enter the name of the proxy action in the **Action name** field. The Action name is used as the unique identifier for this proxy action configuration. The Action name must be unique with the cell and cannot include any of the following characters: # \ / , ; " * ? < > | = + & % '.
 - b. If you are creating a new application server route, then follow these additional steps:
 - 1) Enter the beginning time for this routing rule in the **Start time** field. If the start time is specified using a 12-hour clock, then click **AM** or **PM**. If the start time is specified using a 24-hour clock, then click **24-hour**.

- 2) Enter the finishing time for this routing rule in the **End time** field. If the end time is specified using a 12-hour clock, then click **AM** or **PM**. If the end time is specified using a 24-hour clock, then click **24-hour**.
 - 3) Select **Include** or **Exclude** from the Action menu to specify the type of rule being configured.
 - 4) Select the applications servers that will follow this rule from the Available application servers menu.
 - 5) Click **>**.
 - 6) If you want to remove an application server from the Enabled application servers menu, then select that server from the Enabled application servers menu and click **<**.
- c. If you are creating a new generic server cluster route, then follow these additional steps:
- 1) Select the Generic server cluster that will follow this rule from the **Generic server cluster name** menu.
 - 2) Select either **Active Affinity** or **Passive Affinity** for the affinity type.
 - 3) If you selected Active Affinity, then in the **Default expiration** field, enter the expiration time in seconds.
 - 4) If you selected Passive Affinity, then in the **Cookie name** field, enter the name of the cookie to be used by the proxy server to manage affinity.
 - 5) If you selected Passive Affinity, then select the generic server mappings for which this cookie will be used to manage affinity. If you need to create any additional mappings, click **New**. If you need to remove existing cookie mappings, select the appropriate mapping and click **Delete**.
 - 6) Click **New Time Mapping**.
 - 7) Enter the beginning time for this routing rule in the **Start time** field. If this rule should be applied all the time, then selected **24-hour**.
 - 8) Enter the finishing time for this routing rule in the **End time** field. If this rule should be applied all the time, then selected **24-hour**.
 - 9) Select **Include** or **Exclude** from the Action menu to specify the type of rule being configured.
 - 10) Select the cluster members that will follow this rule from the Available generic server cluster members menu.
 - 11) Click **>**.
 - 12) If you want to remove cluster member from the Enabled generic server cluster members menu, then selected that cluster member from the Enabled generic server cluster members menu and click **<**.
 - 13) Click **OK** to return to set the time-of-day rules and continue creating your generic server cluster route configuration.
- d. If you are creating a failure route, follow this additional step: In the **Fail status code** field, Enter the status code that must be used to indicate a request was not successful.
- e. If you are creating a redirection route, follow this additional step: In the **Redirect URL** field, enter the URL that must be used to redirect the inbound request.
- f. If you are creating a local route, follow this additional step: Confirm that the static file document root is correct. If the value that is listed is not the document root that you want, click **Edit**. See “Administering proxy virtual hosts” on page 287 for more information on changing the static file document root.
8. Click **OK** to finish creating your proxy action rule.

Proxy server actions

Proxy server actions are used in association with proxy rule expressions. If a proxy rule expression evaluates to true, then all the proxy actions associated with the rule expression are performed. Some examples of proxy server actions include caching actions, rewriting actions, compression actions, header modification actions, and routing actions.

Caching actions

Caching actions are set to determine whether or not a response will be cached. A caching action specifies the last modified factor and the default expiration to define how a response is cached.

Rewriting actions

Rewriting actions define how the proxy server rewrites uniform resource locators (URL). A rewriting action is used to rewrite elements of a response message. This is often done to mask the back-end server identity with that of the proxy server. The follow elements can be rewritten using this type of proxy action:

- Absolute URL response
- Redirect location header
- Redirect status code
- Relative URL response
- Set-Cookie

Compression actions

HTTP Compression actions are set to compress the request message body to the server or response message body to the client. The supported compression type standards for these proxy actions are Deflate and Gzip.

Header modification actions

Header modification actions are implemented to perform a header modify action on a specified HTTP header. The available header modify actions include:

- Set
- Append
- Edit
- Remove

The header modify action is performed only if the expression matches when the specified value is applied. For HTTP requests, the header modify action is performed on the HTTP methods you specify. For HTTP responses, the header modify action is performed on the HTTP status codes you specify.

Routing actions

Routing actions are used to route requests when a given rule expression is matched. The following types of routing actions are available:

- Application server routes
Application server routing actions allow you to specify time of day mappings for your application servers. These mappings include or exclude an application server for the routing of requests during a specified time of day. If multiple time of day mappings are configured, the order they are matched is the same as they appear in the application server route configuration.
- Generic server cluster routes
Generic server cluster routing actions work similar to application server routes but apply to generic server clusters instead of application servers.
- Fail routes
Fail routing actions are used to return a failure status code to an inbound request. The value of the failure status code is specified in the fail routing action configuration.
- Redirect routes

Redirect routing actions are used to redirect an inbound request to a different URL. The URL the request is being redirected to is specified in the redirection routing action configuration.

- **Local routes**

Local routing actions are used to pass an inbound request to be served by the local web applications deployed for the cell.

Proxy actions collection

Use this page to administer actions for the proxy server. Proxy actions include creating, modifying, or deleting rules that affect caching, compression, headers, rewriting, and routing for the proxy server. The Proxy actions collection panel allows you to configure proxy actions from one interface.

To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers > proxy_server_name > Proxy actions**.

Caching actions

The Caching actions table contains the following fields. Caching actions are set to determine whether a response is cached.

Action name

Specifies the name of the caching action. A proxy action name must be unique within a cell and cannot contain an invalid character. The name field cannot contain the following characters: # \ / , ; " * ? < > | = + & % '.

Enable caching

Specifies to enable or disable caching.

Default expiration

Specifies the default expiration, in seconds, that is used to determine the validity of cached responses for the URI that is associated with the cache rule.

Last modified factor

Specifies the period of time since the last modification.

Compression actions

Compression actions are set to compress the request message to the server or response message to the client. The Compression actions table contains the following fields.

Action name

Specifies the name of the compression action. A proxy action name must be unique within a cell and cannot contain an invalid character. The name field cannot contain the following characters: # \ / , ; " * ? < > | = + & % '.

Compression action type

Specifies whether the action type is a request or a response compression action.

Compression type

Specifies the compression type standard to apply to the outbound request sent to the target server or the compression type standard to apply to the response sent to the client.

Header actions

Header actions allow you to add, modify, or delete request and response headers. The Header actions table contains the following fields.

Action name

Specifies the name of the header action. A proxy action name must be unique within a cell and cannot contain an invalid character. The name field cannot contain the following characters: # \ / , ; " * ? < > | = + & % '.

Header action type

Specifies whether the action type is a request or a response header action.

Header modify action

Specifies the action to be taken on a request or response header, such as set, append, edit, or remove.

Header name

Specifies the header name to be sent in the request.

Header value

Specifies a user-defined value for a request or response header.

Rewriting actions

A rewrite action can modify inbound requests handled by the proxy server. Rewriting actions define how the proxy server rewrites the URL of a response message; for example, to mask the back-end server identity with that of the proxy server. The Rewriting actions table contains the following fields.

Action name

Specifies the name of the rewriting action. A proxy action name must be unique within a cell and cannot contain an invalid character. The name field cannot contain the following characters: # \ / , ; " * ? < > | = + & % '.

Rewriting action type

Specifies a rewriting action type from a predefined list. See the following table for details.

Rewriting action type	Description
Absolute URL response	Rewrites an absolute URL from matching attributes in a response.
Redirect location header	Rewrites the URL in the relocation header in the HTTP response.
Redirect status code	Specifies the redirect status code in the first line of a response message.
Relative URL response	Rewrites a relative URL in tag attributes a response.
Set cookie domain	Rewrites the domain attribute of the set cookie header.
Set cookie path	Rewrites the path attribute of the set cookie header.

From pattern

Specifies the original URL pattern in the 302 response header from the target server. The pattern can include the asterisk (*) as a wild card symbol. A URL pattern can have one or more asterisks.

To pattern

Specifies the resulting pattern after the rewrite. The pattern can include the asterisk (*) as a wild card symbol. A URL pattern can have one or more asterisks.

Routing action

Routing actions define routes to local file system resources for static file serving. The Routing actions table contains the following fields.

Action name

Specifies the name of the routing action. A proxy action name must be unique within a cell and cannot contain an invalid character. The name field cannot contain the following characters: # \ / , ; " * ? < > | = + & % '.

Routing type

Specifies a routing type from a predefined list: Application server route, Generic server route, Fail route, Redirect route, and Local route.

Note: For more detailed configuration information, see the individual proxy action settings topics.

Caching action settings

You can configure caching action settings or a proxy server. Caching actions are set to determine whether or not a response will be cached.

To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers > proxy_server_name > Proxy actions > action_name**.

Name

Specifies a user-defined symbolic name for a caching action.

A caching action name must be unique and cannot contain an invalid character. The name field cannot contain the following characters: # \ / , ; " * ? < > | = + & % '.

Enable caching

Specifies whether or not to enable caching for the proxy server. If caching is enabled, then the proxy server does not cache the uniform resource identifier (URI) group that is associated with this cache rule.

You must enable caching before you can provide a value for Default expiration or Last modified factor. If a value is specified for Last modified factor, then the value for Default expiration is not used.

Default expiration

Specifies the amount of time, in seconds, before a cached response expires. The default expiration determines the validity of cached responses for the URI that is associated with the cache rule.

Last modified factor

Specifies the amount of time, in seconds, before a response is cached if the response does not have explicit HTTP expiration headers. The last modified time header must be specified in the response. The value is determined based on when an HTML file was last changed to prevent caching frequently modified files for long periods of time.

HTTP compression action settings

You can configure settings for an HTTP request compression action or an HTTP response compression action for a proxy server. Compression actions are set to compress the request message to the server or response message to the client.

To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers > proxy_server_name > Proxy actions > action_name**.

Name

Specifies a user-defined symbolic name for an HTTP compression action.

An HTTP compression action name must be unique and cannot contain an invalid character. The name field cannot contain the following characters: # \ / , ; " * ? < > | = + & % '.

Compression type

For requests, a compression type specifies the compression type standard to apply to the outbound request body sent to the target server. For responses, compression type specifies the compression type standard to apply to the response body sent to the client.

The following standards are supported: Gzip and Deflate. If you specify Any, either standard can be used.

Gzip

Deflate

Any

Content types

Specifies the content types for which compression is applied.

HTTP header action settings

You can configure settings for an HTTP request header action or an HTTP response header action for a proxy server. Use header modification actions to add, modify, or delete request and response headers.

To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers > proxy_server_name > Proxy actions**.

Action name

Specifies a user-defined symbolic name for an HTTP header action.

An HTTP header action name must be unique and cannot contain an invalid character. The name field cannot contain the following characters: # \ / , ; " * ? < > | = + & % '.

Header name

Specifies the name of the HTTP header to set, append, edit, or remove from a request header or a response header.

Header modify action

Specifies the action to be taken on a request or response header, such as set, append, edit, or remove.

Table 8. Header actions

Types of actions	Description
Set	Adds a new value for a request or response header. If a value does not exist, this value is added. If a value does exist, this value replaces the current value.
Append	Adds a value to the current value for a request or response header. If a value does not exist for the header, then this value is added. If a value does exist for the header, then this value is added to the end of the current value.
Edit	Changes the current value to a different value for a request or response header. If a value exists for the header, then this value is changed equivalent to a regular expression search and replace.
Remove	Removes the specified header. If the header exists, then the header is removed. If the header does not exist, then no action is taken.

Header value

Specifies a user-defined value for a request or response header.

Header value expression

Specifies a regular expression applied to the value of the Header name field. If a match exists, then the value specified for the header value field replaces the current value of the header name field.

Method names (HTTP header request action)

Specifies the HTTP method to which the action applies, such as GET or POST. If a value is not specified, then the action is applied for all method names.

Method names apply to HTTP header request actions only.

Status codes (HTTP header response action)

Specifies the return status codes to which the action applies, such as 200 or 503. If a value is not specified, then the action is applied for all status codes.

Status codes apply to HTTP header response actions only.

Rewrite action settings

You can configure settings to implement a rewrite action for inbound requests handled by the proxy server. Rewriting actions define how the proxy server rewrites elements of the uniform resource locators (URL of a response message). Rewrite actions are often done to mask the back-end server identity with that of the proxy server.

To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers > proxy_server_name > Proxy actions > action_name**.

Action name

Specifies a user-defined symbolic name for a rewriting action.

A rewriting action name must be unique and cannot contain an invalid character. The name field cannot contain the following characters: # \ / , ; " * ? < > | = + & % '.

Rewriting action type

Specifies the type of rewriting action to perform. You can specify the following rewriting action types: Absolute URL response, Redirect location header, Redirect status code, Relative URL response, Set-Cookie.

Table 9. Rewriting action types

Action type	Description
Absolute URL response	<p>Rewrites the absolute URI in the tag attribute in the HTTP response. The proxy server scans the response for an attribute matching the From pattern. If a match of the From pattern occurs, then the proxy rewrites the response based on the To pattern. For example:</p> <pre>frPattern = '/(.*)' toPattern = '/prefix/\$1'</pre> <p>The tag <code></code> is changed to <code></code>.</p>
Redirect location header	<p>Rewrites the URI in the relocation header in the HTTP response. For example:</p> <pre>fromPattern = 'http:(.*)' toPattern = 'https:\$1'</pre> <p>The location header: "Location: http://www.ibm.com" is changed to "Location: https://www.ibm.com."</p>
Redirect status code	<p>Specifies the redirect status code in the first line of a response message, such as 301 or 302.</p>

Table 9. Rewriting action types (continued)

Action type	Description
Relative URL response	<p>Rewrites an Relative URL in tag attributes a response. The proxy server scans the response for an attribute matching the From pattern. If a match of the From pattern occurs, then the proxy will rewrite the response based on the To pattern. For example:</p> <pre>fromPattern = '/(.*)' toPattern = '/prefix/\$1'</pre> <p>The tag <code></code> is changed to <code></code>.</p>
Relative URL response:Passive	<p>Instead of rewriting the response directly, the proxy server will inject a cookie in the response header. For example: If a request for <code>"/myimages/1.jpg"</code> is resent from the browser with the cookie, then the proxy server will recreate the request URI as <code>"/prefix/myimages/1.jpg"</code>. This feature requires a browser that supports cookies and for each session, only one passive rule can be defined.</p>
Set-Cookie_Domain	<p>Rewrites the domain attribute of the set cookie header. For example:</p> <pre>fromPattern = '(.*)' toPattern = '\$1.cn'</pre> <p>The set cookie header: <code>"Set-Cookie: JSESSIONID: abcdefg; domain="www.ibm.com""</code> is changed to be <code>"Set-Cookie: JSESSIONID: abcdefg; domain="www.ibm.com.cn""</code></p>
Set-Cookie_Path	<p>Rewrites the path attribute of the set cookie header. For example:</p> <pre>frPattern = '(.*)' toPattern = '/prefix\$1'</pre> <p>The set cookie header: <code>"Set-Cookie: JSESSIONID: abcdefg; domain="www.ibm.com"; path="/"</code> is changed to <code>"Set-Cookie: JSESSIONID: abcdefg; domain="www.ibm.com"; path="/prefix/"</code>.</p>

From pattern

Specifies the original URL pattern in the response from the target server. The pattern can include the following wild card symbol: `*`. A URL pattern can have one or more asterisks (`*`).

To pattern

Specifies the resulting pattern after the rewrite. The pattern can include the following wild card symbol: `*`. A URL pattern can have one or more asterisks (`*`).

Enable passive rewrite

Specifies whether or not to defer the rewriting of the URI until the subsequent request for that URI is sent by the client. Enabling passive rewrite prevents the proxy server from rewriting all the links in the response before sending the response back to the client.

Cookie name

Specifies the cookie for which path or domain attributes are rewritten. This setting is only valid when the action type is Set-Cookie path or Set-Cookie domain.

Limit URL pattern

Specifies a constraint on the URL patterns to rewrite in the response message. Limiting the URL pattern prevents the proxy server from rewriting all URL patterns in the response message of a certain page. This setting is only valid when the action type is absolute URL response or relative URL response.

Limit cookie domain

Specifies a constraint to limit the rewriting of the cookie domain to only a set of specified domains. If no domains are specified, then all domains are rewritten. This field is only valid when the rewriting action type specified is Set cookie domain.

Limit cookie path

Specifies a constraint which limits rewriting the cookie path to the specified paths. If no paths are specified, then all paths are rewritten. This field is only valid when the rewriting action type specified is Set cookie path.

Route action settings

You can configure settings for a route action for a proxy server. Add a route action to define routes to local file system resources for static file serving.

This topic is for local route actions, fail route actions, redirect route actions, and application server route actions. Some settings only apply to a specific type of route action. See the topic about generic server route actions for information about generic server route actions.

To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers > proxy_server_name > Proxy actions > action_name**.

Action name

Specifies a user-defined symbolic name for a route action.

A route action name must be unique and cannot contain an invalid character. The name field cannot contain the following characters: # \ / , ; " * ? < > | = + & % '.

Static file document root: /home/action1 (local route action)

Specifies the root directory on the file system where static files are located. Click **Edit** to access the Proxy Settings page to set the static file document root for a proxy server or an individual proxy server virtual host.

Static file document root applies to local route actions only.

Fail status code (fail route action)

Specifies the HTTP status code to send to the client indicating that the request failed. Possible values are integers with a value greater than zero.

Fail status code applies to fail route actions only.

Redirect URL (redirect route action)

Specifies the URL to send to the client in a 302 redirect response.

Redirect URL applies to redirect route actions only.

Time of day rules (application server and generic server cluster routes)

Specifies a time rule, which is typically used to configure a maintenance window. You can specify a start time and an end time to include or exclude routing to a particular cluster member.

Click **New Time Mapping** to access the Time Mapping settings page to specify a time interval for the time mapping. You can create, delete, or edit time mappings.

Table 10. Time mapping settings

Setting	Description
Time	Specifies the start time and end time for the time mapping.
Action	If the action is set to include, then the server routes actions to the cluster members specified in the time mapping. If the action is set to exclude, the server does not route actions to the specified cluster members during this time interval.

Generic server cluster route action settings

You can configure a generic server cluster route action for a proxy server. Add a generic server cluster route action to define routes for inbound requests to specific generic server clusters.

To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers > proxy_server_name > Proxy actions > action_name**.

Action name

Specifies a user-defined symbolic name for a generic server cluster route action.

A generic server cluster route action name must be unique and cannot contain an invalid character. The name field cannot contain the following characters: # \ / , ; " * ? < > | = + & % '.

Generic server cluster name

Specifies the generic server cluster name provided when the generic server cluster was created. This is a user-defined field. The name field cannot contain the following characters: # \ / , ; " * ? < > | = + & % '.

The name specified must be unique among generic server clusters and cannot begin with a period or a space. A space does not generate an error, but leading and trailing spaces are automatically deleted.

Active affinity

Specifies whether the proxy server will manage the application affinity to the target generic server. Active affinity is used when a generic server application cannot manage active affinity.

The Default expiration field specifies the amount of time, in seconds, that the active affinity cookie set by the proxy server is valid. The cookie is required to maintain affinity and ensure that the request is returned to the correct server within the configured time.

Passive affinity

Specifies whether the proxy server will maintain affinity based on a specified cookie that is set by the target generic server. Passive affinity is used when a generic server sets an affinity cookie that the proxy can use to manage affinity to a generic server.

The administrator specifies a cookie name and then maps the cookie values to each generic server.

Cookie name

Specifies the cookie that is sent by the server and used by the proxy to determine affinity.

Cookie mapping

Specifies the cookie settings to use for a particular generic server.

Table 11. Cookie mapping settings

Action type	Description
Cookie value	Specifies the value for the cookie that is set by the server. The cookie value is mapped to the server defined by the host and port.
Host	Specifies the host name for the target server.
Port	Specifies the port for the target server.

Time of day rules

Specifies a time rule, which is typically used to configure a maintenance window, for generic server cluster routes. You can specify a start time and an end time to include or exclude routing to a particular cluster member.

Click **New Time Mapping** to access the settings page to specify a time interval for the time mapping. You can create, delete, or edit time mappings.

Table 12. Time mapping settings

Setting	Description
Time	Specifies the start time and end time for the time mapping.
Action	If the action is set to include, then the server routes actions to the cluster members specified in the time mapping. If the action is set to exclude, the server does not route actions to the specified cluster members during this time interval.

Time mapping settings

You can configure time mapping settings for the proxy server that set routing rules to be in effect during specific time intervals. These settings can be specified for application server members or for generic server cluster members.

To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers > proxy_server_name > Proxy actions > routing_action_name > time_mapping**.

Note: Select **Proxy Virtual Host Configuration** under Proxy Actions and create a new routing action name if you have not already configured one.

With the time mapping panel, you can make specific servers active or inactive during specified time intervals.

Start time

Specifies the start time, in hours and minutes, for when the routing action being created is valid. The time entered for this field uses a 24-hour clock.

End time

Specifies the end time, in hours and minutes, for when the routing action being created is no longer valid. The time entered for this field uses a 24-hour clock.

Action

Specifies to include or exclude the application server or generic server cluster members when a routing request is sent during the specified Start time and End time.

Include

Specifies the servers to include in routing decisions during the specified time period.

Exclude

Specifies the servers to exclude from routing decisions during the specified time period.

Application server members

Specifies the available application servers that can use the specified time mappings. You can add or remove application servers. The servers listed in the Enabled list are included or excluded from routing requests as selected.

Note: This field only displays if you configured an application server member.

Generic server cluster members

Specifies the available generic server clusters that can use the specified time mappings. You can add or remove generic cluster members. The servers listed in the Enabled list are included or excluded from routing requests as selected.

Note: This field only displays if you configured a generic server cluster member.

Administering custom advisors for the proxy server

You can administer custom advisors for the proxy server. Custom advisors allow for more specific determination of target application server availability by sending protocol level requests to back-end servers. Custom advisors are unique based on the combination of the business level application ID and the composition unit ID.

About this task

Complete these steps to administer actions for the proxy server.

1. From the administrative console, select **Servers > Server Types > WebSphere proxy servers > proxy_server_name > Custom advisor policies**.
2. Click the name of a custom advisor to change settings.
3. Specify the business level application ID.
4. Specify the composition unit ID.
5. Specify the poll interval using seconds in the Poll interval field..
6. Specify the connection timeout using seconds in the Connect timeout field.
7. Specify the input/output timeout using seconds in the I/O timeout field.
8. Optional: Select **Enable logging** if you would like to use logging for this custom advisor. If you enable logging, then follow these steps:
 - a. Select **Enable log file wrapping** if you would like wrapping turned on.
 - b. Specify the maximum size of the log file using megabytes in the Log file size field.
9. Create mappings to be used by your custom advisor using one of the following sub-steps:
 - a. "Configuring stand-alone application server mappings" on page 282
 - b. "Configuring application server cluster mappings" on page 283
 - c. "Configuring generic server cluster mappings" on page 285
10. Optional: Click **Delete** to remove a selected mapping from the collection.

Custom advisor policies

Custom advisor policies help to determine target application server availability. Custom advisors are Java code modules that work within the proxy server to provide information about the application's availability to the proxy server selection code.

Custom advisor policies provide a mechanism to interpret an application protocol response message to determine if an application server, a cluster, or a generic server cluster should be used by the proxy server when requests are made. A custom advisor is able to verify that the application is available, functioning properly, and has access to all required resources.

The proxy server periodically performs an advisor cycle. During an advisor cycle it will call the `isUsable()` method defined on the custom advisor for each available application server, cluster, and generic server cluster targeted by that custom advisor. The `isUsable()` method passes an `AdvisableServer` object, which is used to determine the address, port, and protocol for the targeted application servers, clusters, and generic server clusters.

The custom advisor sends a request to the targets and if communication is successful, then receives their responses. Using the responses, the custom advisor determines if the target is usable or is not usable. If the target is not usable, then the target will be marked as unavailable and will not be used for selection. A target that is marked as unavailable will not be used for selection for this application again, until it is determined to be available by a future advisor cycle. If the target is determined to be usable, a value of `true` is returned, and the target will continue to be used for selection.

Custom advisors collection

Use this page to administer custom adviser policies. With custom advisors, you can determine the availability of a specific target application server by sending protocol level requests to back-end servers. Custom advisor policies are unique based on the combination of the business-level application ID and the composition unit ID.

To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers > *proxy_server_name* > Custom advisor policies**.

Business-level application ID

Specifies a unique identifier for a business level application. A business level application is a configuration that is stored in the product configuration repository.

Composition unit ID

Specifies a unique identifier for a composition unit. A composition unit is a registered custom adviser asset that has additional configuration information, which you specified when adding the asset to the application.

Custom advisor policy settings

You can configure settings for a custom advisor policy. Custom advisor policies allow for more specific determination of target application server availability by sending protocol level requests to back-end servers. Custom advisor policies are unique based on the combination of the business-level application ID and the composition unit ID.

To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers > *proxy_server_name* > Custom advisor policies > *bla_id***.

Business-level application ID

Specifies a unique identifier for a business level application. A business-level application is a configuration that is stored in the product configuration repository.

Composition unit ID

Specifies a unique identifier for a composition unit. A composition unit is a registered custom adviser asset that has additional configuration information, which you specified when adding the asset to the application.

Poll interval

Specifies the amount of time, in seconds, between requests for custom advisors.

Connect timeout

Specifies the amount of time, in seconds, to wait before a connection attempt fails.

I/O timeout

Specifies the amount of time, in seconds, for a custom advisor to wait before read and write operations succeed. If the timeout is reached, then the server stops.

Enable logging

Specifies whether or not logging is enabled. By default, the check box is cleared.

If logging is enabled, then the proxy server logs error messages to a log file. If logging is not enabled, log messages are not added to a log file.

Enable log file wrapping

Specifies whether information at the beginning of the log file is overwritten when the maximum log file size is reached. The default value is True.

Log file size

Specifies the size, in megabytes, of the log file.

Custom advisor mappings

Specifies the application servers, application server clusters, or generic server clusters the custom advisor monitors.

Click **New custom advisor mapping** to access the Custom advisor settings page to create a new custom advisor mapping for a custom advisor.

Configuring stand-alone application server mappings

You can configure a stand-alone application server mapping. Stand-alone application server mappings specify that a stand-alone application server is mapped to a custom advisor.

Before you begin

Stand-alone application server mappings cannot be configured until a proxy server has been configured and a custom advisor has been deployed.

About this task

Complete these steps to configure a stand-alone application server mapping.

1. From the administrative console, select **Servers > Server Types > WebSphere proxy servers > proxy_server_name > Custom advisors policies > custom_advisor_namestandalone_application_server_mapping_name**.
2. Click **New Standalone Application Server Mapping** to begin configuring a new stand-alone application server mapping.
3. In the Cell name menu, select the cell where the server is located that the custom advisor will monitor.
4. In the Node name menu, select the node where the server is located that the custom advisor will monitor. Only the nodes contained within the cell you selected will be available in the Node name menu.
5. In the Server name menu, select the server the custom advisor will monitor. Only the server located on the node you selected will be available in the Server name menu.
6. Specify the application to be monitored. Only the applications deployed on the server you selected will be available in the Application name menu.
7. From the menu, choose a transport chain.
8. Click **OK**.

Stand-alone application server cluster mapping settings

You can configure settings for a stand-alone application server cluster mapping. Stand-alone application server cluster mappings are specified only for a stand-alone application server cluster custom advisor.

To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers > proxy_server_name > Custom advisors > custom_advisor_name > standalone_application_server_cluster_mapping**.

Cell name:

Specifies a logical name for the cell.

A cell name must be unique and cannot contain an invalid character. The name field cannot contain the following characters: # \ / , ; " * ? < > | = + & % '.

Node name:

Specifies the name of the node. A node is a logical grouping of managed servers.

A node name must be unique and cannot contain an invalid character. The name field cannot contain the following characters: # \ / , ; " * ? < > | = + & % '.

Server name:

Specifies the display name for the server.

A server name must be unique and cannot contain an invalid character. The name field cannot contain the following characters: # \ / , ; " * ? < > | = + & % '.

Application name:

Specifies the name of the application.

An application name must be unique and cannot contain an invalid character. The name field cannot contain the following characters: # \ / , ; " * ? < > | = + & % '.

Transport chain:

A transport chain consists of one or more types of channels, each of which supports a different type of I/O protocol, such as TCP, HTTP, or DCS.

Transport chains enable communication through transport channels, or protocol stacks, which are usually socket based.

Configuring application server cluster mappings

You can configure an application server cluster mapping to specify which application server clusters a custom advisor will monitor.

Before you begin

Application server cluster mappings cannot be configured until a proxy server has been configured and a custom advisor has been deployed.

About this task

Complete these steps to configure an application server cluster mapping.

1. From the administrative console, select **Servers > Server Types > WebSphere proxy servers > proxy_server_name > Custom advisors policies > custom_advisor_nameapplication_server_cluster_mapping_name**.
2. Optional: Click **New Application Server Cluster Mapping** to begin configuring a new application server cluster mapping.
3. In the Cell name menu, select the cell of the server that the custom advisor will monitor.
4. In the Cluster name menu, select the cluster of the server that the custom advisor will monitor. Only the available clusters within the cell you selected are available in the Cluster name menu.
5. Specify the application to be monitored. Only the applications that are deployed on the cluster you selected are available in the Application name menu.
6. From the menu, choose a transport chain.
7. Select the cluster members that the custom advisor will monitor from the Available application server cluster members menu.
8. Click > to add the selected cluster members to be monitored by the custom advisor.
9. Optional: If you want to remove a cluster member from the Selected application server cluster members menu, select the cluster members to remove and click <.
10. Click **OK**.

Application server cluster mapping settings

You can configure settings for an application server cluster mapping. Application server cluster mappings are specified only for an application server cluster custom advisor.

To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers > proxy_server_name > Custom advisorscustom_advisor_name > application_server_cluster_mapping**.

Cell name:

Specifies a logical name for the cell.

A cell name must be unique and cannot contain an invalid character. The name field cannot contain the following characters: # \ / , ; " * ? < > | = + & % '.

Cluster name:

Specifies a logical name for the proxy cluster.

A cluster name must be unique among proxy clusters within the containing cell and cannot contain an invalid character. The name field cannot contain the following characters: # \ / , ; " * ? < > | = + & % '.

Application name:

Specifies the name of the application.

An application name must be unique and cannot contain an invalid character. The name field cannot contain the following characters: # \ / , ; " * ? < > | = + & % '.

Transport chain:

A transport chain consists of one or more types of channels, each of which supports a different type of I/O protocol, such as TCP, HTTP, or DCS.

Transport chains enable communication through transport channels, or protocol stacks, which are usually socket based.

Application server cluster members:

Specifies the application server clusters that use the specified cluster mapping. You can add or remove application server cluster members.

An application server cluster is a cluster of application servers. Application servers host a common set of resources and can receive routing actions as a unit.

Node name

Specifies the name of the node. A node is a logical grouping of managed servers. A node name must be unique and cannot contain an invalid character. The name field cannot contain the following characters: # \ / , ; ; " * ? < > | = + & % '.

Server name

Specifies the display name for the server. A server name must be unique and cannot contain an invalid character. The name field cannot contain the following characters: # \ / , ; ; " * ? < > | = + & % '.

Configuring generic server cluster mappings

You can configure a generic server cluster mapping. A generic server cluster allows you to configure external servers into a logical cluster that can be used by the proxy server to route requests. Generic servers host a common set of resources and can receive routing actions as a unit. Generic server cluster mappings are configured to specify which generic server clusters a custom advisor will monitor.

Before you begin

Generic server cluster mappings cannot be configured until a proxy server has been configured and a custom advisor has been deployed.

About this task

Complete these steps to configure a generic server cluster mapping.

1. From the administrative console, select **Servers > Server Types > WebSphere proxy servers > proxy_server_name > Custom advisors policies > custom_advisor_namegeneric_server_cluster_mapping_name**.
2. Optional: Click **New Generic Server Cluster Mapping** to begin configuring a new generic server cluster mapping.
3. In the Cluster name menu, select the generic server cluster the custom advisor will monitor is located.
4. Select the generic cluster members the custom advisor will monitor from the Available generic server cluster members menu.
5. Click > to add the generic server cluster members to be monitored by the custom advisor.
6. Optional: If you want to remove a generic cluster member from the Selected generic server cluster members menu, then select the generic cluster member to remove and click <.
7. Click **OK**.

Generic server cluster mapping settings

You can configure settings for a generic server cluster mapping. Generic server cluster mappings are specified only for a generic server cluster custom advisor.

To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers > proxy_server_name > Custom advisors > custom_advisor_name > generic_server_cluster_mapping**.

Cluster name:

Specifies a logical name for the proxy cluster.

A cluster name must be unique among proxy clusters within the containing cell and cannot contain an invalid character. The name field cannot contain the following characters: # \ / , ; " * ? < > | = + & % '.

Generic server cluster members:

Specifies the generic server clusters that use the specified time mappings. You can add or remove generic server cluster members.

A generic server cluster is a cluster of generic servers. Generic servers host a common set of resources and can receive routing actions as a unit. Generic servers, such as Web servers, are not managed by the application server.

Host name

Specifies the IP address, domain name system (DNS) host name with domain name suffix, or the DNS host name for the generic server cluster.

Port Specifies the port where the generic server cluster accepts client requests. Specify a port value in conjunction with the host name.

Creating custom advisors for the proxy server

Use the following steps to create and deploy a custom advisor to a proxy server. Custom advisor policies allow for more specific determination of target application server availability. Custom advisors are Java code modules written that work within the proxy server to provide information about application server availability to the proxy server selection code.

About this task

Complete these steps to create and deploy a custom advisor for the proxy server.

1. Create the custom advisor code.

Custom advisors are written in the Java language. A custom advisor extends the `com.ibm.wsspi.advisor.AbstractCustomAdvisor` class in the `proxy.jar` file that is included with WebSphere Application Server. Custom advisors use the defined methods of the `AbstractCustomAdvisor` class to obtain the information about the advisor. A custom advisor also must implement the following elements:

- A constructor method that takes a `CustomAdvisorConfigObject` object and calls the `super(caConfigObject)` method, for example

```
public AbstractHttpProxyCustomAdvisor(CustomAdvisorConfigObject caConfigObject) {  
    super(caConfigObject);  
}
```

- An `isUsable()` method that takes an `AdvisableServer` object; for example

```
public boolean isUsable(AdvisableServer aServer) throws CustomAdvisorException;
```

The `initialize` method is called after the `AbstractCustomAdvisor` construction, but before the `isUsable` method is called. This process allows the custom advisor to perform any additional steps after the base class completes initialization, but before the `isUsable` method is called, which ensures that the `initialize` method is only called once. If overridden, then the `initialize()` method must call the `super.initialize` method, for example:

```
protected void initialize() {  
    super.initialize();  
}
```

For more information about the required routines and the other methods available to a custom advisor, see the application programming interface (API) reference section of the information center. From the information center navigation, scroll to the Reference section and click APIs - Application Programming Interfaces. A list of the product API specifications displays in alphabetic order.

There are two exception classes that need to be considered when creating the custom advisor:

- The CustomAdvisorException can be created by the isUsable method of the custom advisor to tell the AbstractCustomAdvisor that the custom advisor must not call the isUsable method again until the next advisor cycle.
- The NoLogConfiguredException is created by the AbstractCustomAdvisor if the no logging file has been configured for the custom advisor, but logging is enabled.

The httpcustomadvisor.jar file can be used as a sample of a custom advisor. This file contains an AbstractHttpProxyCustomAdvisor.java class that extends the com.ibm.wsspi.advisor.AbstractCustomAdvisor and implements the isUsable() and initialize() methods.

2. Compile your custom advisor code. After you have created the Java source code for your custom advisor, you must compile it using the AbstractCustomAdvisor code that is included with WebSphere Application Server. To access the abstract custom advisor classes in the com.ibm.wsspi.advisor package, add the proxy.jar file to your Java class path. The proxy.jar file is located in the \${WAS_INSTALL_ROOT}/plugins directory.

3. Create the advisor-context.xml file.

After compiling the custom advisor code, you will need create the advisor-context.xml file. This file is used to identify the code as a custom advisor Java archive (JAR) file when it is imported as an asset and then added as a compilation unit to a business-level application (BLA). When the custom advisor JAR asset is added to a BLA and then targeted to a proxy server, the Content Distribution Framework (CDF) support will distribute and copy all the BLA artifacts of the custom advisor to the appropriate configuration information on the targets specified.

The advisor-context.xml contains the class name for the custom advisor to be run and the custom advisor name. The format of the advisor-context.xml file must follow the advisor-context.xsd schema in the proxy.jar file. You can use an XML schema tool to assist in creating and setting the appropriate information. The required configuration information is defined as follows:

```
<advisor-name> SomeCustomAdvisor </advisor-name>
<advisor-class> com.ibm.wlm.test.customadvisor.SomeCustomAdvisor </advisor-class>
<description> Some Custom Advisor Description </description>
<display-name> Some display name </display-name>
```

4. Create the custom advisor BLA. Package your compiled custom advisor class files and the advisor-context.xml into a JAR file. This JAR file is then used when creating the custom advisor BLA to be installed and deployed to the proxy server. The following example shows the commands to use to install a custom advisor as a BLA.

```
$AdminTask importAsset {-source C:/proxy/testadvisor.jar -storageType FULL}
$AdminTask createEmptyBLA {-name myBLA}
$AdminTask addCompUnit {-blaid myBLA -cuSourceID assetname=testadvisor.jar,assetversion=1.0 -MapTargets {{.* ProxyServer}} -CustomAdvisorCUOptions
{"type=Cluster,cellName=yourCellName,clusterName=yourClusterName
,applicationName=myBLA" default default default 1000}}
$AdminConfig save
```

See [Setting up business-level applications using scripting](#) for more information on setting up a BLA.

5. Configure your deployed custom advisor. See “Administering custom advisors for the proxy server” on page 280 for additional details.

Administering proxy virtual hosts

Virtual hosting allows a single proxy server to host multiple domains and ports on a single IP address and port. A proxy virtual host can be created for each Web domain the proxy server is hosting, or wild card characters can be used to host multiple Web domains with a single proxy virtual host.

About this task

Note: A proxy virtual host allows a single proxy server to host multiple domains and ports on a single IP address and port. A proxy virtual host consists of the name and the port representing the Web domain and a set of proxy rule expressions to perform specified proxy actions when a defined criteria exists. Additionally, each proxy virtual host can override the server scope configuration of the proxy server to have configuration elements defined specifically for that virtual host. Proxy

virtual hosts use a set of proxy server actions and proxy rule expressions. Proxy rules expressions are evaluated when inbound requests are received by the proxy virtual host. If the expression is evaluated to be true, any proxy server actions specified by the proxy rule expression are performed

Complete these steps to administer or create a new proxy virtual host.

1. From the administrative console, select **Servers > Server Types > WebSphere proxy servers > proxy_server_name > Virtual hosts > virtual_host_name**.
2. Optional: Click **New** to access the virtual hosts settings page to configure settings for a new virtual host.
3. Specify a name for the virtual host. The name of the virtual host must match the Web domain it represents. If the web domain is www.proxy1.com, the name of the proxy server host must be www.proxy1.com. The asterisk symbol, *, can be used as a wild card character to represent all Web domains. If the proxy virtual host is *:80, then all inbound requests on port 80 are handled by that proxy virtual host regardless of what Web domain is being requested.
4. Specify the port for the virtual host. The port of the virtual host must match the port used by the web domain it represents. If the Web domain uses port 80, then the port of the proxy virtual host must also be 80. The asterisk symbol, *, can be used as a wild card character to represent all ports. If the proxy virtual host is www.proxy1.com:*, then all inbound requests to www.proxy1.com are handled by that proxy virtual host regardless of what port is being used.
5. Choose one or more proxy rule expressions for the virtual host. Proxy rule expressions allow for proxy actions to be performed when the expression evaluates to true. See “Proxy rule expressions” on page 292 and “Proxy server actions” on page 269 for more information on proxy rule expressions and proxy server actions.
6. Optional: Move a proxy virtual host up or down in the list to ensure the correct proxy virtual host is used. The usage of wild card characters for virtual proxy hosts creates a situation where an inbound request might match multiple proxy virtual hosts. In this scenario, the request will be handled by the first proxy virtual host that matches the request.
7. Optional: Click **Edit** to change the selected proxy rule expression.
8. Optional: Click **Proxy Virtual Host Settings** to override the server scoped settings for static file serving, logging, or error page policy.
 - a. If you want to override the static file serving settings for this proxy virtual host, click **Static File Serving**, select **Customize for this virtual host**, and then specify a new value for the **Static file document root** field.
 - b. If you want to override the logging settings for this proxy virtual host, click **Logging**, select **Customize for this virtual host**, and then specify new values for one or more of the following fields:
 - **Enable access logging**
 - **Access log maximum size**
 - **Proxy access log**
 - **Cache access log**
 - **Local access log**
 - c. If you want to override the error page policy settings for this proxy virtual host, click **Error Page Policy**, and then select **Customize for this virtual host**, and then specify new values for one or more of the following fields:
 - **Error page generation application URI**
 - **Handle errors generated by the proxy server**
 - **Handle errors generated by the application server**
 - **Headers to forward to error page application**
 - **HTTP status codes that are to be recognized as errors**

- d. Click **OK** to save your proxy virtual host settings, and return to the previous administrative console page.
9. Click **OK** to save all of your other changes.

Proxy virtual hosts

Virtual hosting allows a single proxy server to host multiple domains and ports on a single IP address and port.

A proxy virtual host consists of the name and port of a Web domain and a set of proxy rule expressions that perform proxy actions for specific criteria. Additionally, each proxy virtual host can override the proxy server configuration to have configuration elements defined specifically for that virtual host. The following settings can specify virtual host settings in place of the server scope settings:

- Logging
- Custom error pages
- Static file serving

Proxy virtual hosts use proxy server actions and proxy rule expressions. Proxy rule expressions and proxy server actions are only used for proxy virtual hosts. When the proxy virtual host receives inbound requests, the proxy rule expressions are evaluated. If the expression is evaluated to be true, any proxy server actions that are specified by the proxy rule expression are performed. The following proxy server actions can be specified when an expression evaluates to true:

- Routing rules
- Caching rules
- URL rewriting rules
- Header modification rules
- Compression rules

A different proxy virtual host can be created for the proxy server to represent each Web domain that the proxy server is hosting. For example, a request for `www.proxy1.com` on port 80 uses the configuration specified for `www.proxy1.com:80`. A request for `www.proxy2.com` on port 80 uses the configuration specified for `www.proxy2.com:80`. You can use a wild card character to specify that a proxy virtual host can be used for all Web domains or all ports. For example, `www.proxy1.com:*` specifies that a proxy virtual host can be used for all requests for the Web domain `www.proxy1.com` regardless of the port. A proxy virtual host for `*:80` specifies that it can be used for all requests on port 80 regardless of the Web domain.

After creating a proxy server with the needed proxy virtual hosts, the HTTP protocol allows multiple Web domains to be hosted by a single server process. When an inbound request is received by the proxy server, it matches the proxy virtual host located in the inbound request message to the appropriate configuration for that proxy virtual host. If a request matches multiple proxy virtual hosts because wild card characters were used, the proxy virtual host that is first in the list of proxy virtual hosts is used.

Proxy virtual hosts collection

Use this page to administer proxy virtual hosts. A proxy virtual host allows a single proxy server to host multiple domains or ports on a single IP or port. Each proxy virtual host consists of a domain name and a port.

To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers > *proxy_server_name* > Proxy virtual hosts**.

Virtual host name

Specifies a user-defined symbolic name for a proxy virtual host.

A virtual host name must be unique and cannot contain an invalid character. The virtual host name is a reflection of the Web domain it represents. For example, use a virtual host name of `www.proxy1.com` to represent the `www.proxy1.com` Web domain. The asterisk symbol, `*`, can be used in the name field to indicate that this proxy virtual host matches all Web domains. The name field cannot contain the following characters: `# \ / , ; " * ? < > | = + & % ' .`

Proxy rule expressions

Specifies a rule expression for a proxy virtual host. A proxy rule expression consists of operands and operators that is defined by an administrator. If a proxy rule expression evaluates to true, then the proxy actions that are configured for the rule are run.

Enabled

Specifies whether or not a proxy virtual host is enabled.

If a proxy virtual host is enabled, then the configuration for the proxy virtual host is used when a user makes a request for the proxy virtual host. Otherwise, another matching proxy virtual host configuration is used. If no other proxy virtual host configuration matches the configuration for the requested proxy virtual host, then the server-scoped configuration is used.

Proxy virtual host settings

You can configure settings for a proxy virtual host. A proxy virtual host allows a single proxy server to host multiple domains or ports on a single IP or port. Each proxy virtual host consists of a domain name and a port.

To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers > proxy_server_name > Virtual hosts > virtual_host_name**.

Virtual host name

Specifies a user-defined symbolic name for a proxy virtual host.

A virtual host name must be unique and cannot contain an invalid character. The virtual host name is a reflection of the Web domain it represents. For example, use a virtual host name of `www.proxy1.com` to represent the `www.proxy1.com` Web domain. The asterisk symbol, `*`, can be used in the name field to indicate that this proxy virtual host matches all Web domains. The name field cannot contain the following characters: `# \ / , ; " * ? < > | = + & % ' .`

Virtual host port

Specifies the port number associated with the Web domain that the proxy virtual host represents. The asterisk symbol, `*`, can be used in the virtual host port field to indicate that this proxy virtual host matches requests for all ports of the Web domain that are specified by the Virtual host name field..

Proxy rule expressions

Specifies a rule expression for a proxy virtual host. If a proxy rule expression evaluates to true, then the proxy actions that are configured for the rule are run. Click **Edit** to change a proxy rule expression. Proxy rule expressions are only available for proxy virtual hosts.

You can enable proxy rule expressions to complete certain proxy actions for a proxy virtual host. Click the arrows to include or exclude proxy rule expressions. Proxy rule expressions are evaluated in the order that they are listed. You can move a proxy rule expression up or down in the list to determine when the rule expression is evaluated.

Available proxy rule expressions

Specifies the rule expressions that can be added to a virtual host.

Proxy rule expressions

Specifies the rule expressions that were added to the virtual host.

Proxy virtual host settings details

Use this panel to override some proxy server scoped settings with values to be used for the proxy virtual host.

To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers > proxy_server_name > Virtual hosts > virtual_host_name > Proxy virtual host settings**.

General Properties

The three types of configurations available on this page are static file serving, logging, and error page policy. For each of these types, you can choose to use the server scoped value for this setting or you can override the settings with new values that are used only for this proxy virtual host.

- If you choose **Use global server settings**, then the server scoped values for that type are displayed and those values will be used for the proxy virtual host.
- If you choose **Customize for the virtual host**, then the values that you enter in the fields for that type are used for the proxy virtual host instead of the server scoped values.

Static file serving

This type of configuration specifies the information needed for the proxy server to perform static file serving.

- **Static file document route:** The location on the file system of the static files to be served

Logging

This type of configuration specifies the logging options for proxied requests and stored cache requests.

- **Enable access logging:** Specifies whether or not access logging will be used.
- **Access log maximum size:** Specifies the maximum size of each access in megabytes.
- **Proxy access log:** Specifies the log to be used to log responses that are received from remote servers.
- **Cache access log:** Logs responses that are served from the local cache.
- **Local access log:** Logs all non-cache local responses, for example, redirects and internal errors.

Error page policy

This type of configuration specifies settings to support the use of customized error pages to be displayed when errors occur during the processing of requests.

- **Error page generation application URI:** If a valid URI to an installed application is not provided, the custom error page policy does not handle requests.
- **Handle errors generated by the proxy server:** Specifies if errors generated by the proxy server should be handled with the custom static error pages stored on the local file system. If this is not selected then the default error messages will be used instead of any customer error pages.
- **Handle errors generated by the application server:** Specifies if errors generated by the back-end server should be handled with the custom static error pages stored on the local file system. If this is not selected, then the default error messages are used instead of any customer error pages.
- **Headers to forward to error page application:** Specifies a list of the headers from the original request to forward to the error page generation application.
- **HTTP status codes that are to be recognized as errors:** Specifies a list of the status codes in a response that should be directed to the error page generation application.

Administering proxy rule expressions

You can administer proxy rule expressions to make proxy actions more granular in scope. Proxy rule expressions are configurations assigned to a proxy virtual host.

Before you begin

About this task

A proxy rule expression consists of operands and operators that are defined by an administrator. When an expression evaluates to true, the proxy action rules associated with that proxy rule expression are performed.

Complete these steps to create a new proxy rule expression or to administer an existing proxy rule expression.

1. From the administrative console, select **Servers > Server Types > WebSphere proxy servers > proxy_server_name > Proxy Virtual Host Configuration > Proxy rule expressions**.
2. Click **New** to create a new proxy rule expression. To administer an existing proxy rule expression, Click **rule_expression_name**.
3. Specify the name for the rule expression.
4. Click **Subexpression builder** to add an expression. You can also create the expression by entering the expression manually. The Subexpression builder is only designed to generate a subset of the available options when creating an expression. For example the Subexpression builder does not allow the usage of parenthesis in an expression. If a complex expression is needed, then type the expression into the field instead of using the Subexpression builder. For some examples of valid expressions, see “Proxy rule expressions.” To use the Subexpression builder to create your expression, repeat these steps for each subexpression that you want to create.
 - a. From the **Logical operator** menu, choose a logical operator. If this the first subexpression in this field, then the logical operator will be removed when it is generated.
 - b. From the **Select operand** menu, choose an operand.
 - c. From the **Operator** menu, choose an operator.
 - d. From the **Select <operand_name>** menu, choose the name of the operand to be used. The name and available selections for this menu is dependant on the operand specified in step b. For example, if you selected **cell** in step b, then the name of this menu will **Select cell** and the menu will list the available cells as menu options.
 - e. Click **Generate subexpression** to add the new expression to the Subexpressions list.
 - f. Click **OK**.
5. Enable a proxy action for the proxy rule expression. You can enable multiple proxy actions. All the enabled proxy actions will be performed when the proxy rule expression evaluates to true.
6. Optional: If the a proxy action needs to be modified, select that proxy action, and click **Edit**.
7. Optional: Move a proxy action up or down in the list to change the order in which the proxy actions will be performed. If the sequence the proxy actions are performed is important for the rules you specified, ensure that the order of the proxy actions in the list corresponds to the order they must be performed.
8. Click **OK**.

Proxy rule expressions

Use proxy rule expressions to make configuration rules more granular in scope by specifying information within the rule. Proxy rule expressions are assigned to a proxy virtual host. When a request is handled by the proxy virtual host, the proxy rule expressions associated with that proxy virtual host are evaluated. If any of the proxy rule expressions evaluates to true, then all the proxy actions specified in the proxy rule expression configuration are performed.

Proxy rule expressions have operands and operators that are created and managed by an administrator. With operands, you can configure proxy rule expressions based on the following criteria:

- cell
- application

- module
- uri
- urigroup

Operands are combined with operators to define the proxy rule expressions. Operators can be applied with words or symbols.

Word	Symbol
AND	&&
OR	

Example 1

In the following example, the expression evaluates to true if the target cell name is *mycell*. Otherwise, the expression evaluates to false.

```
cell=mycell
```

Example 2

In the following example, the expression evaluates to true if the target cell name is *mycell*, and the application name is *myapp*. Otherwise, the expression evaluates to false.

```
cell=mycell AND application="myapp"
```

Example 3

In the following example, the expression evaluates to true if the target cell name is *mycell*, and the target application name is *myapp1* or *myapp2*. Otherwise, the expression evaluates to false.

```
cell=mycell && (application="myapp1" || application="myapp2")
```

Example 4

In the following example, the expression evaluates to true if the target cell name is *mycell*, and the target application is not named *myapp*. Otherwise, the expression evaluates to false.

```
cell=mycell AND application!myapp
```

Example 5

In the following example, the expression evaluates to true if the request URI matches the pattern */proxy1/**. Otherwise, the expression evaluates to false.

```
uri="/proxy1/*"
```

Proxy rule expressions collection

Use this page to administer proxy rule expressions for the proxy server. Proxy rule expressions allow you to make configuration rules more granular in scope by specifying information within the rule. A proxy rule expression is associated with a virtual proxy host and consists of operands and operators that are defined by an administrator. When an expression evaluates to true, the proxy action rules associated with that proxy rule expression are performed.

To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers > server_name > Proxy Virtual Host Configuration > Proxy rule expressions**.

Expression name

Specifies a user-defined symbolic name for a proxy rule expression.

An expression name must be unique and cannot contain an invalid character. The name field cannot contain the following characters: # \ / , ; " * ? < > | = + & % '.

Expression

Specifies the expression that is evaluated when an inbound request is made to the proxy virtual host.

An expression evaluates to true or false and consists of operands and operators. Expressions can be entered manually on the Proxy rule expressions settings page. The **Subexpression builder** can also be used on that page to generate the expressions.

Table 13. Rule expression operands

Operands	Description
Cell	Specifies the cell that contains the back end server to which the request is mapped.
Application	Specifies the Java Platform, Enterprise Edition (Java EE) application that a given request maps to on the back end server.
Module	Specifies the J2EE application module to which a given request is mapped.
URI	Specifies the URI for an inbound request.
URIgroup	Specifies a group of URIs to match against the URI. If the inbound request is in the URI group, then the expression evaluates to true.

Proxy actions

Specifies the proxy actions configured for the rule expression. Proxy actions are performed when the proxy rule expression evaluates to true.

You can configure the following proxy actions for a proxy rule expression:

- header actions
- caching action
- compression actions
- rewriting actions
- routing actions

Proxy rule expression settings

You can configure settings for a proxy rule expression. A proxy rule expression consists of operands and operators that are defined by an administrator. If a proxy rule expression evaluates to true, then all proxy actions configured for that rule expression are run.

To view this administrative console page, click **Servers > Server Types > WebSphere proxy servers > proxy_server_name Proxy Virtual Host Configuration > Proxy rule expressions > expression_name**.

Expression name

Specifies a user-defined symbolic name for a proxy rule expression.

An expression name must be unique and cannot contain an invalid character. The name field cannot contain the following characters: # \ / , ; " * ? < > | = + & % '.

Expression

Specifies the expression that is evaluated when an inbound request is made to the proxy virtual host. You can create expressions manually by entering the text into this field, or you can use the **Subexpression builder** to generate a new expression.

Specify the following settings for the **Subexpression builder**. You can use the following operands in your expression:

Table 14. Rule expression operands

Operands	Description
Cell	Specifies the cell that contains the server to which the request is mapped.
Application	Specifies the J2EE application that a given request maps to on the back end server.
Module	Specifies the J2EE application module to which a given request is mapped.
URI	Specifies the URI for an inbound request.
URIgroup	Specifies a group of URIs to match against the URI. If the inbound request is in the URI group, then the expression evaluates to true.

Proxy actions

Specifies the proxy actions configured for the rule expression.

You can configure the following proxy actions for a proxy rule expression:

- header actions
- caching action
- compression actions
- rewriting actions
- routing actions

Click **Edit** to modify a proxy action from this panel. Click the arrows to include or exclude proxy actions. You can move a proxy action up or down in the list to determine when the proxy action is performed.

Creating a custom filter and deploying it to a proxy server

You can create a custom filter for your proxy server if you need to have the proxy server perform a function, such as customized logging, that not provided with through normal product settings.

Before you begin

- Determine where you are going to store the artifact that contains the filter you create. You can import this artifact from either a local or remote file system.
- You must know the name of the proxy server on which you want to install the custom filter.
- You must start the administrative console that is used to control this proxy server, it is not already started.

About this task

A filter provides an optional, secondary control over a function, that is beyond the control that is normally provided through typical product settings. For example, an applications might use a logging filter mechanism to suppress all of the events that have a particular message key.

1. Create a custom filter.
 - a. Create a class that extends `com.ibm.wsspi.HttpDefaultFilter`.
 - b. Override the abstract `doFilter` method, and, optionally, the `init` and `destroy` methods. If you override the `init` and `destroy` methods, you must call the super versions of these methods to preserve filter lifecycle functionality.

- c. Write a new doFilter method, that uses the HttpProxyServiceContext interface.

This new doFilter method can be written such that it:

- Changes any artifacts that are associated with the request or the response, such as the response itself, the response code, or the headers.
- Changes and artifacts that are associated with both the request and the response.
- Obtains information about either the request or the response.
- Obtains information about both the request and the response.

- d. Use the proxy server filter managed bean (MBean), proxyFilterMbean, to determine the correct ordinal for the filter.

```
proxyFilterMbean = AdminControl.queryNames('type=ProxyServerFilterBean,*')
AdminControl.invoke(proxyFilterMbean, 'viewAllFilters')
```

The order, in which all of the filters on the proxy server are processed, displays.

- e. Create the filter descriptor file, filter-context.xml

This file is used to define all of the filters that have this descriptor. The following example illustrates a basic version of a filter-context.xml file.

In this example, the descriptor determines where and when the given filter is executed on both the request and the response. The filter point determines where the filter is executed. In this example, the filter executes after the request is received. The ordinal determines when the filter gets processed relative to other filters at the same filter point. The higher the ordinal, the higher the filter is in the processing queue.

```
<?xml version="1.0" encoding="UTF-8">
<filter-context xmlns="http://www.ibm.com/2003/FilterContextSchema">
  <description>Proxy Filter Sample descriptor</description>
  <display-name>Proxy Sample Filter</display-name>

  <filter>
    <filter-name>HttpRequestFilter</filter-name>
    <filter-class>com.ibm.ws.proxy.sample.HttpRequestFilter</filter-class>
    <description>HTTP sample filter to execute at REQUEST filter point</description>
    <display-name>HTTP Request Sample</display-name>
    <protocol-name>HTTP</protocol-name>
    <filter-point>RequestReceived</filter-point>
    <ordinal>1000</ordinal>
  </filter>
</filter-context>
```

- f. Bundle the compiled .class files for the filter, and the filter descriptor into a JAR file.

2. Import the artifact (JAR file) that contains the custom filter.

- a. In the administrative console, click **Applications > Application Types > Assets > Import**.
- b. Select either **Local file system** or **Remote file system** to indicate where the JAR file is located.
- c. Specify the fully qualified name of the JAR file.

The fully qualified name of the JAR file includes the directory path to where the file is located, and the file name. If you do not know the fully qualified name of the JAR file, you can use the browse function to locate the file.

3. Create a business level application (BLA) that includes this artifact.

- a. In the administrative console, click **Applications > New Application > New Business-level application**.
- b. In the **Name** field, specify a name for the new application that you are creating. You can also specify a description of the application in the **Description** field.
- c. Click **Apply**.

4. Create a BLA composition unit (CU) from the artifact.

- a. In the **Deployed assets** table, click **Add > Add asset**.
- b. Select the name of the artifact that you imported in the first step, and then click **Continue**.

- c. Change the composition unit settings as needed, and then click **Modify target**.
- d. Select the proxy server on which you want to deploy this CU from the list of available deployment targets, and then click **OK**.

When you click **OK** the product maps the composition unit to the selected proxy server.

- e. Specify the relationship options for this composition unit.
- f. Click **Finish**.

To verify that the product successfully added the created this CU, click **Applications > Business-level applications > *application_name***. If the product successfully adds the CU, the name of the CU is shown in the list of deployed assets for this BLA.

5. Start the proxy server.
6. Start the BLA that contains the filter.

Results

The filter is running on the proxy server.

What to do next

Use the proxy server filter MBean, `proxyFilterMbean`, to verify that the filter is installed on the proxy server and that it is being processed in the correct order relative to the other filters that are deployed on the proxy server. If you need to change the order in which this filter is processed, run the `modifyOrdinal` command against the `proxyFilterMbean` MBean.

Configuring denial of service protection for the proxy server

You can configure a pair of properties on your proxy server or DMZ Secure Proxy Server for IBM WebSphere Application Server to limit your risk against denial of service attacks involving the buffering of large HTTP payloads.

Before you begin

Denial of service protection for the proxy server or the DMZ Secure Proxy Server for IBM WebSphere Application Server is not done during the creation of these servers. The proxy server or the DMZ Secure Proxy Server for IBM WebSphere Application Server that will include denial of service protection must already exist before following these steps.

About this task

Note:

Protection is now included to guard against a type of security breach known as a denial of service attack. This type of attack can typically send more traffic to a network address than the data buffers were designed to accommodate. The proxy server and DMZ Secure Proxy Server for IBM WebSphere Application Server have several properties that can be configured to limit your risk against denial of service attacks involving the buffering of large HTTP payloads.

A denial of service attack is a malicious type of security breach to a computer system that does not usually result in the theft of information or other security loss. This type of attack can typically send more traffic to a network address than the data buffers were designed to accommodate resulting in memory exhaustion. HTTP allows for the body of a message to be sent to an HTTP server as an HTTP request or an HTTP response. The body can be sent to the HTTP server in a series of sequential network writes instead of being sent in one large network write. This process is known as Transfer-Encoding chunking. The maximum size of a Transfer-Encoding: chunked response body and a Transfer-Encoding: chunked request body can be set to determine how much data is buffered before a network write is performed.

1. Click **Servers>Proxy Servers>***proxy_server_name*.
2. Under Proxy Settings, expand **HTTP Proxy Server Settings** and click **Denial of service protection**
3. Set the appropriate buffer size in kilobytes for **Maximum request body buffer size**. Buffering can lead to better performance because it will decrease the number of network writes for large payloads. The size of the buffer must not be configured too high or memory exhaustion can occur. To determine the optimal values for your environment, gradually increase the size of the buffer until the proper balance has been achieved.
4. Set the appropriate buffer size in kilobytes for **Maximum response body buffer size**. The same precaution must be taken when setting Maximum response body buffer size as were indicated for Maximum request body buffer size.

Results

After these steps have been properly complete, denial of service protection will be in place for your proxy server or DMZ Secure Proxy Server for IBM WebSphere Application Server.

Denial of Service Protection

Use these settings to define denial of service protection for the proxy server. The buffer chunk sizes are used to provide protection against denial of service attacks. It is very important to tune these settings to balance the level of protection with the performance impacts that can be experienced.

To view this administrative console page, click **Servers** → **Proxy Servers** → **<server_name>** → **HTTP Proxy Server Settings** → **Denial of Service Protection**.

Maximum request body buffer chunk size

This field sets the maximum amount of the request body data that is buffered in memory before it is sent. The proxy server and secured proxy server can buffer multiple network reads of a Transfer-Encoding: chunked request body before performing a network write. This can improve performance because buffering the data will decrease the number of network writes required for large payloads. This field should be adjusted with caution. If the buffer size is set too high, memory exhaustion, which is a common intent of denial of service attacks, might be experienced. You will need to test different buffer sizes to find the optimal level of balance between protection and performance for your system. The default value of this field is 32 kilobytes.

Maximum response body buffer chunk size

This field sets the maximum amount of the response body data that is buffered in memory before it is sent. The proxy server and secured proxy server can buffer multiple network reads of a Transfer-Encoding: chunked response body before performing a network write. This can improve performance because buffering the data will decrease the number of network writes required for large payloads. This field should be adjusted with caution. If the buffer size is set too high, memory exhaustion, which is a common intent of denial of service attacks, might be experienced. You will need to test different buffer sizes to find the optimal level of balance between protection and performance for your system. The default value of this field is 32 kilobytes.

Tuning the security properties for the DMZ Secure Proxy Server for IBM WebSphere Application Server

When creating a DMZ Secure Proxy Server for IBM WebSphere Application Server, default security levels of high, medium and low are available. In addition to the predefined configuration levels, you can modify the security settings for your DMZ Secure Proxy Server for IBM WebSphere Application Server. When you choose to customize the settings, a qualitative value of high, medium or low is still assigned to inform you of the overall security level of your DMZ Secure Proxy Server for IBM WebSphere Application Server.

Before you begin

A DMZ Secure Proxy Server for IBM WebSphere Application Server must be installed before these steps can be completed. The DMZ Secure Proxy Server for IBM WebSphere Application Server profile must be registered with the AdminAgent for this panel to be available.

About this task

Installing the DMZ Secure Proxy Server for IBM WebSphere Application Server in the DMZ rather than the secured zone presents new security challenges. The DMZ Secure Proxy Server for IBM WebSphere Application Server has been equipped with various capabilities to provide protection for meeting these challenges. In addition to the predefined security configurations for your DMZ Secure Proxy Server for IBM WebSphere Application Server you can also tune the settings to customize the protection.

1. Click **Servers** → **Proxy Servers** → **<secured_proxy_server_name>** → **Custom security settings** to open up the Proxy security settings panel.
2. Choose your administration option for your DMZ Secure Proxy Server for IBM WebSphere Application Server.
 - Local Administration - Security level: medium and high.
This option allows two different types of administration. Managing the DMZ Secure Proxy Server for IBM WebSphere Application Server entirely using the wsadmin tool and loading updated profiles imported from inside the cell using the wsadmin tool are both considered Local Administration.
 - Remote Administration- Security level: low
3. Choose your routing option for your DMZ Secure Proxy Server for IBM WebSphere Application Server.
 - Static routing - Security level: high
 - Dynamic routing - Security level: low and medium
4. Choose your startup permission option for your DMZ Secure Proxy Server for IBM WebSphere Application Server.
 - Run as an unprivileged user - Security level: medium and high
 - Run as privileged user - Security level: low
5. Optional: If Run as an unprivileged user is selected, enter the user name or the user group whose identify the server should assume after startup has completed.
6. Choose your custom error page policy option for your DMZ Secure Proxy Server for IBM WebSphere Application Server.
 - Local error page handling - Security level: high
If you choose to use local error page handling, you need to select which error responses should use custom error messages. Select Handle local errors for responses generated by the proxy server and select Handle remote errors for responses generated by the backend server. Both options may be selected to use custom error messages for local and remote errors. Manage your error code mappings to determine the custom error pages to be used for specific responses.
 - Remote error page handling - Security level: low and medium
If you choose to use remote error page handling to include custom errors, you need to select which error responses should be customized. Select Handle local errors for responses generated by the proxy server and select Handle remote errors for responses generated by the backend server. Both options may be selected to use custom error messages for both local and remote errors. Manage the headers that should be sent to the custom error application and what status codes are to be recognized as errors.

Results

You have finished customizing the security settings for your DMZ Secure Proxy Server for IBM WebSphere Application Server. A qualitative value of high, medium or low has been calculated based on the settings you have chosen to demonstrate the Current DMZ Security level.

DMZ Secure Proxy Server for IBM WebSphere Application Server start up user permissions

The overall security level of the DMZ Secure Proxy Server for IBM WebSphere Application Server can be hardened by reverting the server process to run as an unprivileged user after startup. Although the DMZ Secure Proxy Server for IBM WebSphere Application Server must be started as a privileged user, changing the server process to run as an unprivileged user provides additional protection for local operating resources.

Like the proxy server, the DMZ Secure Proxy Server for IBM WebSphere Application Server must start under a privileged user because it requires authorization to initialize privileged ports. Ports lower than 1024 are considered privileged ports. After these ports are initialized and access to the protected ports is no longer required, it is possible to change the user association of the DMZ Secure Proxy Server for IBM WebSphere Application Server process. Altering the server process to run using the privileges of a user or a group that does not have authority to access the local operation system resources adds a layer of protection to those resources. The firewall helps protect local operating system resources for the proxy server, but as the DMZ Secure Proxy Server for IBM WebSphere Application Server is installed in the DMZ, this type of protection becomes a higher priority. Although changing the user association of the server process for the DMZ Secure Proxy Server for IBM WebSphere Application Server is not required, continuing to run as a privileged user does not use the extra layer of protection for local operation resources that is provided when the server process is changed to run as an unprivileged user.

Run as unprivileged user	This is considered a high and medium security level setting.
Run as privileged user	This is considered a low security level setting.

DMZ Secure Proxy Server for IBM WebSphere Application Server routing considerations

This topic summarizes some of the security implications that must be considered when choosing how your DMZ Secure Proxy Server for IBM WebSphere Application Server will match incoming HTTP requests to an application or routing rule.

The DMZ Secure Proxy Server for IBM WebSphere Application Server can be configured to route requests either statically or dynamically. Using static routing specifies routing is performed using a flat configuration file using routing precedence that is inherent to the ordering of the directives. Requests can also be routed dynamically using a best match mechanism that determines the installed application or routing rule that corresponds to a specific request. The DMZ Secure Proxy Server for IBM WebSphere Application Server will dynamically discover the best route to a destination and distribute to servers with like protocols. It is considered more secure to use static routing than dynamic routing. This is because dynamic routing requires the secured proxy server to communicate with the application server to map requests where as static routing is self contained by the secure proxy server requiring no additional communication.

Static Routing	This is considered a high security level setting.
Dynamic Routing	This is considered a low and medium security level setting.

DMZ Secure Proxy Server for IBM WebSphere Application Server administration options

The DMZ Secure Proxy Server for IBM WebSphere Application Server is administered differently than the WebSphere proxy server. The DMZ Secure Proxy Server for IBM WebSphere Application Server is a separate binary installed in the DMZ. Installing the DMZ Secure Proxy Server for IBM WebSphere Application Server in the DMZ requires that administration be managed differently for security reasons.

Several administrative options are available for administering the DMZ Secure Proxy Server for IBM WebSphere Application Server to provide different levels of balance between security and usability.

The most secure way to administer the DMZ Secure Proxy Server for IBM WebSphere Application Server is locally using the wsadmin tool. The DMZ Secure Proxy Server for IBM WebSphere Application Server does not have web container therefore local administration can only be done via the command line. Using the wsadmin commands locally to manage the DMZ Secure Proxy Server for IBM WebSphere Application Server is the most secure option available because it does not require any external listening ports to be opened.

The DMZ Secure Proxy Server for IBM WebSphere Application Server configurations can also be managed within the network deployment application server cell and then imported locally using the wsadmin commands. The configurations are maintained inside the cell as configuration only profiles. The profiles are registered with the Admin Agent and are then managed using the administrative console. After you implement any changes to the profile, you export the configuration to an configuration archive (CAR) file using the exportProxyProfile or exportProxyServer wsadmin commands. After you transmit the CAR file to the local DMZ Secure Proxy Server for IBM WebSphere Application Server installation using ftp, the CAR file is imported using the importProxyProfile or importProxyServer wsadmin commands. This option is also considered to be local administration.

The DMZ Secure Proxy Server for IBM WebSphere Application Server can be managed remotely from the Job Manager console. You can create, delete, start, stop and modify the DMZ Secure Proxy Server for IBM WebSphere Application Server configuration using the Job Manager console. However, typically before managing the DMZ Secure Proxy Server for IBM WebSphere Application Server using the Job Manager console, the DMZ Secure Proxy Server for IBM WebSphere Application Servers would be created and configured using one of the other administrative options explained above. Using the Job Manager console is useful for centrally managing multiple DMZ Secure Proxy Server for IBM WebSphere Application Servers but is less secure than the other options because it requires the SOAP connector listener port to be opened for communication with the Job Manager console.

Error handling security considerations for the DMZ Secure Proxy Server for IBM WebSphere Application Server

The overall security level of the DMZ Secure Proxy Server for IBM WebSphere Application Server is partially determined by the choices made regarding the handling of custom errors.

You can define a custom error page for each error code or a group of error codes on errors generated by the proxy server or the application server. This is done using HTTP status codes in responses to generate uniform customized error pages for the application. For security reasons, you can ensure that the error pages are read from the local file system instead of being forwarded to a custom remote application. Choosing this option limits the code path and eliminates the need for a potentially unauthorized application to be run as the error message is generated based on a flat file. For more information about the error handling for the secured proxy server, see “Overview of the custom error page policy” on page 239.

The following security level settings are used when evaluating a custom security level. Local error page handling is used for all of the predefined security levels.

Local error page handling	This is considered a high security level setting.
Remote error page handling	This is considered a medium and low security level setting.

Proxy security level properties

These settings describe the attributes and policies that define the security level of a secured proxy server. The overall security level of the secured proxy server is set to the weakest level of security assigned to any of the individual settings.

To view this administrative console page, click **Servers > Proxy Servers > server_name > Custom security settings**. This panel will only be available for a secure proxy server profile that has been registered with the AdminAgent.

Current security level

A qualitative security level based on an evaluation of the current security related configuration values.

The possible values for Current[®] DMZ Security are high, medium, low. During creation of the secured proxy server, default configurations of high, medium and low are available. You are also able to customize these security settings resulting in the Current DMZ Security level being calculated by the system. Each custom setting has an assigned value of high, medium or low. The overall security level is equal to value of the setting that is considered the least secure. For example, to have an overall security level of high, all settings must be configured to the values associated with a high level of security. If any of the settings are configured with a less secure value, the overall security level is the value of that setting.

Administration

Table 15. Administration options

Option	Used as the default value in the predefined security levels	Description
Local administration	The default value for the Medium and the High security levels	Specifies that administration of the secure proxy server can only be performed using wsadmin commands performed locally on the system.
Remote administration	The default value for the Low security level	Specifies that remote administration of the secure proxy server is permitted.

Routing

Table 16. Routing options

Option	Used as the default value in the predefined security levels	Description
Static routing	The default value for the High security level	Specifies that the proxy server will make routing determinations from routing information based on flat files on the file system. This is for Hypertext Transfer Protocol (HTTP) only
Dynamic routing	The default value for the Low and the Medium security levels	Specifies that the proxy server will dynamically discover the best route to a destination and distribute to servers with like protocols.

Start-up permissions

Table 17. Start-up permission options

Option	Used as the default value in the predefined security levels	Description
Run as an unprivileged user	The default value for the Medium and the High security levels	Specifies that the server process will revert to a predefined unprivileged user after start-up has completed.
Run as a privileged user	The default value for the Low security level	Specifies that the server process does not revert to an unprivileged user after startup. It is a requirement that the proxy server start under a privileged user as it initializes privileged ports. Ports lower than 1024 are considered privileged ports. Under this setting, the effective user of the server process continues to be the privileged user. This setting does not provide additional hardening to the access of the server process to the local operation system resources. This is considered a low security level setting.

Custom Error Page Policy

Table 18. Error page options

Option	Used as the default value in the predefined security levels	Description
Local error page handling	The default value for the Low, the Medium and the High security levels	Specifies that error responses will be generated from flat custom error page files stored locally on the local file system.
Remote error page handling	None	Specifies to route error responses to a remote custom application deployed on a back-end server. This application will generate a custom response for the error

Local error page handling

- **Handle errors generated by the proxy server**

Specifies if errors generated by the proxy server should be handled with the custom static error pages stored on the local file system. If this is not selected then the default error messages will be used instead of any custom error pages.

- **Handle errors generated by application servers**

Specifies if errors generated by the backend server should be handled with the custom static error pages stored on the local file system. If this is not selected then the default error messages will be used instead of any custom error pages.

- **Error mappings**

Specifies the error codes to match with specific static error pages stored on the file system. You can use a relative file path under the configured static file document root to assign a custom error file to be used for a specific error code or group of error codes. The wildcard character, *, is used to assign error files to groups of error codes.

Remote error page handling

- **Error page generation application URI**

Specifies the URI for the custom error page generation application.

- **Handle errors generated by the proxy server**

Specifies if errors generated by the proxy server should be handled with the custom error application deployed on the application server. If this is not selected then the default error messages will be used instead of any custom error pages.

- **Handle errors generated by application servers**

Specifies if errors generated by the backend server should be handled with the custom error application deployed on the application server. If this is not selected then the default error messages will be used instead of any custom error pages.

- **Headers to forward to Error page Application**

Specifies a list of the headers from the original request to forward to the error page generation application.

- **HTTP status codes that are to be recognized as errors**

Specifies a list of the status codes in a response that should be directed to the error page generation application.

Configuring a DMZ Secure Proxy Server for IBM WebSphere Application Server using the administrative console

You can create a DMZ Secure Proxy Server for IBM WebSphere Application Server inside of a cell using the administrative console. The server can then be exported to a node in the demilitarized zone (DMZ) where the configuration is then imported. After the server is created on a node in the DMZ, administration can be done locally or it can be done using the Job Manager console.

Before you begin

Before you begin, complete these tasks.

- Install the DMZ Secure Proxy Server for IBM WebSphere Application Server code, see the Installing the DMZ Secure Proxy Server for IBM WebSphere Application Server image topic for more details.
- Create a secure proxy server profile using either the Profile management tool (PMT) or using the `manageprofile` command. See Creating a secure proxy profile for more information on creating the profile using the PMT.
- If you're going to manage the secure proxy server using the Job Manager, you also need to create an administrative agent, see Creating a management profile with an administrative agent server for additional details.
- Register and start the secure proxy server profile.

About this task

The DMZ node does not have a Web container; therefore, you must create a secure proxy server from the administrative console or using the command line. The secure proxy server configurations can be managed within the network deployment application server cell and then imported locally using the `wsadmin` commands. The configurations are maintained inside the cell as configuration-only profiles. The profiles are registered with the Admin Agent and are then managed using the administrative console. After you implement any changes to the profile, you export the configuration to a configuration archive (CAR) file using the `exportProxyProfile` or `exportProxyServer` `wsadmin` commands. After you transmit the CAR file to the local secure proxy server installation using FTP, the CAR file is imported using the `importProxyProfile` or `importProxyServer` `wsadmin` commands. Complete these steps to create a secure proxy server on a network deployment node in the DMZ.

1. From the administrative console, select **Servers > Server Types > WebSphere proxy servers**.
2. Click **New** to access the Proxy Server Creation wizard.
3. Select the DMZ Secure Proxy Server for IBM WebSphere Application Server node.
4. Complete the steps in the wizard to create a new secure proxy server. This new secure proxy server will only be used as a configuration. It cannot be started inside of the cell.
5. Save the configuration.
6. Using the `wsadmin` tool, connect to the secure proxy server profile. See Starting the `wsadmin` scripting client for more information.
7. Export your configuration to be used inside of the DMZ; you can export the entire profile or export the server. See the following examples:
 - a. Optional: To export the profile, run the following command:

```
AdminTask.exportProxyProfile(['-archive', 'c:\myCell.car'])
```
 - b. Optional: To export the server, run the following command:

```
AdminTask.exportProxyServer(['-archive c:\myServer.ear -nodeName node1 -serverName proxy1'])
```
8. Using FTP, transfer the configuration archive to the secure proxy server node.
9. Start the `wsadmin` tool on the secure proxy server profile.
10. Import the entire profile or import the server.
 - a. Optional: Use the `importProxyProfile` command to import the profile, as the following example demonstrates:

```
AdminTask.importProxyProfile(['-archive', 'c:\myCell.car'])
AdminTask.importProxyProfile(['-archive', '/myCell.car'])
```
 - b. Optional: Use the `importProxyServer` command to import the server, as the following example demonstrates:

```
AdminTask.importProxyServer(['-archive c:\myServer.ear -nodeInArchive node1 -serverInArchive proxy1'])
AdminTask.importProxyServer(['-archive /myServer.ear -nodeInArchive node1 -serverInArchive proxy1'])
```
11. Save the configuration changes.

Results

Successful completion of this procedure results in deployment of the DMZ Secure Proxy Server for IBM WebSphere Application Server on a node in the DMZ.

What to do next

You can now start and begin to use your DMZ Secure Proxy Server for IBM WebSphere Application Server.

Configure secure routing for a DMZ Secure Proxy Server for IBM WebSphere Application Server

The DMZ Secure Proxy Server for IBM WebSphere Application Server can be configured to route requests statically or dynamically.

Before you begin

Configure your profiles and security properties before you configure routing. See the topic Tuning the security properties for the DMZ Secure Proxy Server for IBM WebSphere Application Server. Decide whether you want to configure static or dynamic routing.

About this task

Static routing is performed using a flat configuration file. Static routing is considered more secure than dynamic routing. With dynamic routing, requests are routed through a best match mechanism that determines the installed application or routing rule that corresponds to a specific request. The secure proxy server will dynamically discover the best route to a destination and distribute to servers with like protocols.

The secure routing options are summarized as follows:

- Use static routing with the `exportTargetTree` command.
- Use dynamic routing by setting up a core group bridge (CGB) tunnel. See the topic Configuring communication with a core group that resides on a DMZ Secure Proxy Server for IBM WebSphere Application Server.

Use the following procedure to configure static or dynamic secure routing.

- To configure static routing, follow these steps:
 1. Set the secure proxy server to use static routing, which is the default level after install. Do this either by setting the overall security level to `high` or by setting the custom security level for the routing property to `static`.
 2. Using the `wsadmin` tool, query for the `TargetTreeMbean` mbean.

```
mbean=AdminControl.queryNames('*:*',type=TargetTreeMbean,process=dmgr')
```
 3. Invoke the `exportTargetTree` method on the `TargetTree` mbean to a specified xml file.

```
AdminControl.invoke(mbean, 'exportTargetTree', '/opt/IBM/WebSphere/AppServer/targetTree.xml')
```
 4. Using the Deployment Manager (`dmgr`) command line, transfer the `targettree.xml` file from the `dmgr` to the proxy server's profile root `/staticRoutes` directory. The file is transferred from the `dmgr` to the proxy server by FTP or some other protocol.
 5. Start the proxy server from the system command line: `ProfileRoot/startServer proxy_server_name`.
- To configure dynamic routing, follow these steps:
 1. Configure the CGB in the WAS cell. See the topic Configuring communication with a core group that resides on a DMZ Secure Proxy Server for IBM WebSphere Application Server.

2. Export the tunnel template settings to a file. From the wsadmin tool, use the exportTunnelTemplate command to export the settings, as in the following example:

```
AdminTask.exportTunnelTemplate('[-tunnelTemplateName exportedTunnelTemplate  
-outputFileName tunnelTemplate1.props]')
```

3. Import the tunnel template settings into the DMZ proxy config, as in the following example:

```
AdminTask.importTunnelTemplate('[-inputFileName tunnelTemplate1.props  
-bridgeInterfaceNodeName DMZNode01 -bridge InterfaceServerName DMZProxyServer01]')
```

4. Start the proxy server from the system command line: ProfileRoot/startServer proxy_server_name.

Results

Completing this procedure results in configuring secure routing for a DMZ Secure Proxy Server for IBM WebSphere Application Server.

What to do next

You can now start and begin to use the DMZ Secure Proxy Server for IBM WebSphere Application Server.

Related tasks

Configuring communication with a core group that resides on a DMZ Secure Proxy Server for IBM WebSphere Application Server

This task describes the steps that you must perform to establish communication between a cell inside of a firewall, and a DMZ Secure Proxy Server for IBM WebSphere Application Server outside of the firewall.

“Tuning the security properties for the DMZ Secure Proxy Server for IBM WebSphere Application Server” on page 298

When creating a DMZ Secure Proxy Server for IBM WebSphere Application Server, default security levels of high, medium and low are available. In addition to the predefined configuration levels, you can modify the security settings for your DMZ Secure Proxy Server for IBM WebSphere Application Server. When you choose to customize the settings, a qualitative value of high, medium or low is still assigned to inform you of the overall security level of your DMZ Secure Proxy Server for IBM WebSphere Application Server.

Related reference

“WebSphere DMZ Secure Proxy Server for IBM WebSphere Application Server”

The DMZ Secure Proxy Server for IBM WebSphere Application Server can be used to provide a secure platform for your proxy server.

“DMZ Secure Proxy Server for IBM WebSphere Application Server routing considerations” on page 300

This topic summarizes some of the security implications that must be considered when choosing how your DMZ Secure Proxy Server for IBM WebSphere Application Server will match incoming HTTP requests to an application or routing rule.

WebSphere DMZ Secure Proxy Server for IBM WebSphere Application Server

The DMZ Secure Proxy Server for IBM WebSphere Application Server can be used to provide a secure platform for your proxy server.

Note: The DMZ Secure Proxy Server for IBM WebSphere Application Server installation allows you to install your proxy server in the demilitarized zone (DMZ), while reducing the security risk that might occur if you choose to install an application server in the DMZ to host a proxy server. The risk is reduced by removing any functionality from the application server that is not required to host the proxy servers, but that could pose a security risk. Installing the secure proxy server in the DMZ rather than the secured zone presents new security challenges. However, the secure proxy server is equipped with capabilities to provide protection from these challenges.

The following capabilities are available to harden the security of the DMZ Secure Proxy Server for IBM WebSphere Application Server and to determine the level of security to assign.

- Startup user permissions

The secure proxy server process can be changed to run as an unprivileged user after startup. Although the secure proxy server must be started as a privileged user, changing the server process to run as an unprivileged user provides additional protection for local operating resources.

- Routing considerations

The secure proxy server can be configured to route requests to target servers based on static routing information or dynamic information. Static routing means the server obtains the routing information from local flat files. Dynamic routing means the server obtains the routing information from a Hypertext Transfer Protocol (HTTP) tunnel connection from the proxy server to a server in the secure zone. It is more secure to use static routing as the use of dynamic routing requires an additional connection through the inner firewall. Static routing is only applicable to HTTP requests.

- Administration options

The secure proxy server does not contain a Web container, and therefore is unable to host the administrative console. It is better to not have a Web container on a DMZ Secure Proxy Server for IBM WebSphere Application Server since hosting application artifacts is considered a security risk and adds an unnecessary footprint to the proxy server. The secure proxy server is installed separately and has several different administrative options that have security implications.

- Error handling

Custom error pages can be used by the secure proxy server for specific error codes or groups of error codes. A custom error page application can be used to generate error messages or flat custom error page files can be stored locally on the file system and used during run time. Choosing to use flat custom error pages instead of a custom error application provides a higher level of security. Choosing this option limits the code path and eliminates the need for a potentially unauthorized application to be run when an error page is needed.

- Denial of service protection

Denial of service protection is provided with the inclusion of two properties: Maximum request body buffer chunk size and Maximum response body buffer chunk size. These properties must be tuned to balance the level of protection with the performance overhead that might be experienced if these properties are set incorrectly.

When creating the DMZ Secure Proxy Server for IBM WebSphere Application Server, you can choose any of the default security levels: High, Medium or Low.

- Low DMZ security level

Table 19. Low DMZ security level default values

Setting	Default value
Startup permissions	Run as a privileged user
Routing	Dynamic routing
Administration	Remote Administration
Error handling	Local error page handling

- Medium DMZ security level

Table 20. Medium DMZ security level default values

Setting	Default value
Startup permissions	Run as an unprivileged user
Routing	Dynamic routing
Administration	Local Administration
Error handling	Local error page handling

- High DMZ security level

Table 21. High DMZ security level default values

Setting	Default value
Startup permissions	Run as an unprivileged user
Routing	Static routing
Administration	Local Administration
Error handling	Local error page handling

Note: The High DMZ security level cannot be used for SIP proxy servers, because static routing cannot be used for the SIP proxy server.

In addition to these predefined settings, you can customize the settings to better serve your requirements. If you choose to customize the settings, your DMZ Secure Proxy Server for IBM WebSphere Application Server will still be assigned a qualitative categorization of your security level called the *current* security level. Each custom setting has been assigned a value of High, Medium or Low. The current security level is equal to the value of the least secure setting being used. To achieve a current security level of High, only settings assigned the high value can be configured. To achieve a current security level of Medium, only settings with values of High or Medium can be used. A current security level of Low will be used if any settings that are assigned the value of Low are set.

An additional change to enhance the protection for the DMZ Secure Proxy Server for IBM WebSphere Application Server is the switch from a Java Development Kit (JDK) to a Java Runtime Environment (JRE). Switching from a JDK to a JRE removes the inclusion of a compiler on the installation. This change is beneficial because the compiler could possibly be used for malicious purposes in the event of a security breach.

Chapter 5. Creating a proxy server cluster

A proxy server cluster consists of a group of proxy servers. You can create a cluster of proxy servers that can route requests to applications in a cell.

Before you begin

Version 7 of the product must be installed on the nodes that will belong to the proxy server cluster. You must also know whether DNS, Load Balancer, or proxy will be used to route requests to the proxy server cluster.

About this task

To create a proxy server cluster in the administrative console, click **Servers > Clusters > Proxy server clusters > New** to start the Create a new proxy server cluster wizard.

Results

You have successfully created a proxy server cluster.

What to do next

Click **Servers > Clusters > Proxy server clusters**, select the newly created cluster, and then click **Start** to start the cluster, or click on the name of the new cluster to change its configuration settings, or add additional members to the cluster.

Proxy server cluster collection

Use this page to view information about and change configuration settings for a proxy server cluster. A proxy cluster consists of a group of proxy servers.

To view this administrative console page, click **Servers > Clusters > Proxy server clusters** .

To define a new proxy server cluster, click **New** to start the Create a new proxy server cluster wizard.

Name

Specifies a logical name for the proxy cluster. The name must be unique among proxy clusters within the containing cell.

Supported Protocols






Indicates the protocol or protocols that the proxy server is configured to handle.

This information is based on the types of transport channels that are included in the transport chains that are configured for the proxy server. For example, if a transport chain includes an HTTP channel, HTTP displays in this field. If a transport chain includes both a SIP and an HTTP channel, SIP,HTTP displays in this field.

Status

This field indicates whether the proxy cluster is partially started, started, partially stopped, stopped, or unavailable. If the status is unavailable, the node agent is not running in that node and you must restart the node agent before you can start the proxy server.

After you click **Start** or **Ripplestart** to start a proxy cluster, each server that is a member of that proxy cluster launches if it is not already running. When the first member launches, the status changes to partially started. The status remains partially started until all proxy cluster members are running. When all proxy cluster members are running, the state changes to running and the status changes to started. Similarly, when you click **Stop** or **ImmediateStop** to stop a proxy cluster, the status changes to partially stopped when the first member stops, then changes to stopped when all proxy cluster members are not running.

	Started	All of the proxy cluster members are running.
	Partially started	At least one of the proxy cluster members is running.
	Stopped	All of the proxy cluster members have stopped running.
	Partially stopped	At least one of the proxy cluster members has stopped running.
	Unavailable	Status cannot be determined. All of the members of a proxy cluster with an unavailable status might, in fact, be running but the proxy cluster has an unavailable status because the proxy server that is running the administrative console cannot communicate with all of the proxy cluster members. The node agent should be started on all proxy nodes to ensure that the correct status is shown.

Proxy server cluster settings

Use this page to view a list of the proxy server clusters that are defined for your system, and the status of each of these clusters. You can also use this page to view or change the configuration settings for a specific proxy server cluster or to view the local topology of a specific proxy server cluster.

To view a list of the proxy server clusters that are defined for your system, and the status of each of these clusters, in the administrative console click **Servers > Clusters > Proxy server clusters**.

To display the topology of your proxy server clusters, click **Servers > Clusters > Proxy server clusters > proxy_server_cluster_name**, and then click the **Local Topology** tab.

To view or change the configuration settings of a proxy server cluster, in the administrative console click **Servers > Clusters > Proxy server clusters > proxy_server_cluster_name**.

Cluster name

Specifies a logical name for the proxy cluster. The name must be unique among proxy clusters within the containing cell.

Bounding node group name

Specifies the node group that forms the boundaries for this proxy cluster.

A node group is a collection of proxy server nodes. A node is a logical grouping of managed proxy servers, usually on a system that has a distinct IP host address. All proxy servers that are members of a proxy cluster must be on nodes that are members of the same node group. Nodes that are organized into a node group need enough capabilities in common to ensure that proxy clusters formed across the nodes in the node group can host the same application in each proxy cluster member. A node must be a member of at least one node group and can be a member of more than one node group.

Create and manage node groups by clicking **System administration > Node groups** in the administrative console.

Local Topology tab

Use this page to display, in a tree format, a list of all of the application server proxy clusters defined for your environment. The list shows all of the nodes and proxy cluster members that are included in each proxy cluster contained in a cell.

Proxy server cluster member collection

Use this page to view and manage proxy servers that belong to a proxy cluster.

To view this administrative console page, click **Servers > Clusters > Proxy server clusters > proxy_cluster_name > Cluster members**.

Proxy servers that are part of a proxy cluster are referred to as proxy cluster members. A copy of the first proxy cluster member that you create is stored as part of the proxy cluster data and becomes the template for all additional proxy cluster members that you create.

You can use this page to perform the following actions:

- Create a New proxy cluster member. To create a new member, click **New**.
A copy of the first proxy cluster member that you create is stored as part of the proxy cluster data and becomes the template for all additional proxy cluster members that you create. Any individual configuration change that you make to a proxy cluster member does not affect the configuration settings of the proxy cluster member template.
You can use wsadmin commands to modify the proxy cluster member template, or you can click **Servers > Clusters > Proxy server clusters > proxy_cluster_name > Cluster members**, and then click **Templates**. Any change that you make to the template does not affect existing proxy cluster members.
- Delete a proxy cluster member. To delete a proxy cluster member, select the proxy cluster member you want to delete, then click **Delete**.
- Create or edit proxy cluster templates. To work with proxy cluster templates, click **Templates**.
- Start a proxy cluster member. To start a proxy cluster member, select the server that you want to start, then click **Start**.
- Stop a proxy cluster member. To stop a proxy cluster member, select the server that you want to stop, then click **Stop**.

Member name

Specifies the name of the server in the proxy cluster. On most platforms, the name of the server is the process name. The name must match the (object) name of the proxy server.

Node


Specifies the name of the node for the proxy cluster member.




Version

Specifies the version of the product on which the proxy cluster member runs.

Status

This field indicates whether the proxy cluster member is started, stopped, partially stopped, or unavailable. If the status is unavailable, the node agent is not running in that node and you must restart the node agent before you can start the server.

	Started	The proxy cluster member is running.
---	----------------	--------------------------------------

	Partially stopped	The proxy cluster member is in the process of changing from a started state to a stopped state.
	Stopped	The proxy cluster member is not running.
	Unavailable	Status cannot be determined. A proxy cluster member with an unavailable status might, in fact, be running but has an unavailable status because the proxy cluster member that is running the administrative console cannot communicate with this proxy cluster member.

Proxy cluster member settings

Use this page to manage the members of a proxy cluster. A proxy cluster of proxy servers is managed together and participate in workload management.

To view this administrative console page, click **Servers > Clusters > Proxy server clusters > proxy_cluster_name > Cluster members > cluster_member_name**.

Member name

Specifies the name of the proxy server in the proxy cluster. On most platforms, the name of the server is the process name. The member name must match the name of one of the servers that are listed on the proxy servers page.

Run in development mode

Enabling this option may reduce the startup time of an proxy server. This might include Java virtual machine (JVM) settings, such as disabling bytecode verification and reducing Just in Time (JIT) compiler compilation costs. Do not enable this setting on production servers.

Specifies that you want to use the JVM settings, **-Xverify** and **-Xquickstart**, on startup. After selecting this option, save the configuration and restart the server to activate development mode.

The default setting for this option is `false`, which indicates that the server is not started in development mode. Setting this option to `true` specifies that the server is started in development mode, using settings that decrease server startup time.

Data type Boolean
Default false

Parallel start

Specifies whether to start the server on multiple threads. When you start the server on multiple threads, the server components, services, and applications start in parallel rather than sequentially, which might shorten the startup time.

The default setting for this option is `true`, which indicates that the server uses multiple threads when it starts. Setting this option to `false` specifies that the server uses a single thread when it starts, which might lengthen startup time.

Data type Boolean
Default true

Start components as needed

Select this property if you want the proxy server components started as they are needed by an application that is running on this proxy server.

When this property is selected, proxy server components are dynamically started as they are needed. When this property is not selected, all of the proxy server components are started during the startup process. Therefore, selecting this property usually results in improved startup time because fewer components are started during the startup process.

Note: If you are running other WebSphere products on top of the Application Server, make sure that those other products support this functionality before you select this property.

Proxy cluster member templates collection

Use this page to view the list of proxy cluster member templates that exist for this proxy cluster.

To view the proxy cluster member templates that are available for creating a new member of a proxy cluster, in the administrative console, click **Servers > Clusters > Proxy server clusters > *proxy_cluster_name* > Cluster members > Templates**.

To edit the attributes of an existing proxy cluster member template, click the name of the template.

Usually, only one template exists that you can use to create additional members for a proxy cluster. However, if a proxy cluster includes nodes that are at different versions of the product, a different template exists for each of these versions. For example, if a proxy cluster has proxy cluster members residing on both a Version 6.1 node and a Version 7.0 node, the proxy cluster has two templates. The Version 6.1 template is used when you create an additional proxy cluster member on the Version 6.1 node, and the Version 7.0 template is used when you create an additional proxy cluster member on the Version 7.0 node.

If you modify a template, all new proxy cluster members are created with the server property settings of the modified template. However, the property settings of existing proxy cluster members do not change. If you make any change to a proxy cluster member template, you should make the same change to all of the existing proxy cluster members.

Name

Specifies the name of the proxy cluster member template.

Platform

Specifies the operating system platform to which this template applies.

Version

Specifies the version of the product to which the template applies.

Description

Specifies a description of this proxy cluster member template. This field is optional and might be blank.

Proxy cluster member template settings

Use this page to modify proxy cluster member template attributes.

To view this administrative console page, click **Servers > Clusters > Proxy server clusters > proxy_cluster_name > Cluster members > Templates > cluster_member_template_name**. The following attributes are displayed:

Name

Displays the name of the proxy cluster member template.

Run in development mode

Enabling this option may reduce the startup time of a proxy server. This might include Java virtual machine (JVM) settings, such as disabling bytecode verification and reducing Just in Time (JIT) compiler compilation costs. Do not enable this setting on production servers.

Specifies that you want to use the JVM settings, **-Xverify** and **-Xquickstart**, on startup. After selecting this option, save the configuration and restart the server to activate development mode.

The default setting for this option is `false`, which indicates that the server is not started in development mode. Setting this option to `true` specifies that the server is started in development mode, using settings that decrease server startup time.

Data type	Boolean
Default	false

Parallel start

Specifies whether to start the proxy server on multiple threads. When you start the proxy server on multiple threads, the proxy server components, services, and applications start in parallel rather than sequentially, which might shorten the startup time.

The default setting for this option is `true`, which indicates that the proxy server uses multiple threads when it starts. Setting this option to `false` specifies that the proxy server uses a single thread when it starts, which might lengthen startup time.

Data type	Boolean
Default	true

Start components as needed

Select this property if you want the proxy server components started as they are needed by an application that is running on this proxy server.

Note: When this property is selected, proxy server components are dynamically started as they are needed. When this property is not selected, all of the proxy server components are started during the startup process. Therefore, selecting this property usually results in improved startup time because fewer components are started during the startup process.

Avoid trouble: If you are running other WebSphere products on top of the Application Server, make sure that those other products support this functionality before you select this property.

Creating a proxy cluster: Basic proxy cluster settings

Use this page to enter the basic settings for a proxy cluster.

To view this administrative console page, click **Servers > Clusters > Proxy server clusters > New**.

Proxy cluster name

Specifies the name of the proxy cluster. The proxy cluster name must be unique within the cell.

Supported Protocols

Indicates the protocol or protocols that the proxy server is configured to handle.

This information is based on the types of transport channels that are included in the transport chains that are configured for the proxy server. For example, if a transport chain includes an HTTP channel, HTTP displays in this field. If a transport chain includes both a SIP and an HTTP channel, SIP, HTTP displays in this field.

Creating a proxy cluster: Create first proxy cluster member

Use this page to specify settings for the first proxy cluster member.

There are two ways to create the first member of a proxy cluster:

- You can create the first member when you create a new proxy cluster.
To create a new proxy cluster, in the administrative console, click **Servers > Clusters > Proxy server clusters > New**.
- You can create an empty proxy cluster and then add a first member after you finish creating the proxy cluster.
To create a proxy cluster member for an existing proxy cluster, in the administrative console, click **Servers > Clusters > Proxy server clusters > *proxy_cluster_name* > Cluster members > New**.

When you create the first proxy cluster member, a copy of that member is stored as part of the proxy cluster data and becomes the template for all additional proxy cluster members that you create.

When adding servers to a proxy cluster, note that you can either remove a proxy server from a proxy cluster by deleting the proxy server from the list of proxy cluster members, or by deleting the server from the Proxy Servers collection.

Member name

Specifies the name of the proxy server that is created for the proxy cluster.

The member name must be unique on the selected node.

Select node

Specifies the node on which the proxy server resides.

Generate unique HTTP ports

Specifies that a unique HTTP port is generated for the proxy server. By generating unique HTTP ports for the proxy server, you avoid potential port collisions and configurations that are not valid.

Select basis for first proxy cluster member:

Specifies the basis you want to use for the first proxy cluster member.

- If you select **Create the member using a proxy server template**, the settings for the new proxy server are identical to the settings of the proxy server template you select from the list of available templates.
- If you select **Create the member using an existing proxy server as a template**, the settings for the new proxy server are identical to the settings of the proxy server you select from the list of existing proxy servers.
- If you select **Create the member by converting an existing proxy server**, the proxy server you select from the list of available proxy servers becomes a member of this proxy cluster.

- If you select **None. Create an empty proxy cluster**, a new proxy cluster is created but it does not contain any proxy cluster members.

Note: The basis options are available only for the first proxy cluster member. All other members of a proxy cluster are based on the proxy cluster member template, which is created from the first proxy cluster member.

Creating a proxy cluster: Create additional proxy cluster members

Use this page to create additional members for a proxy cluster. You can add a member to a proxy cluster when you create the proxy cluster or after you create the proxy cluster. A copy of the first proxy cluster member that you create is stored as part of the proxy cluster data and becomes the template for all additional proxy cluster members that you create.

To add members to a proxy cluster, in the administrative console, click **Servers > Clusters > Proxy server clusters > New > *proxy_cluster_name* > Cluster members > New**. After you enter the required information about the new proxy cluster member, click **Add Member** to add this member to the proxy cluster member list.

After adding a proxy cluster member, you might need to change one or more of the property settings for this proxy cluster member, or another proxy cluster member that you just added. To change one or more of the editable property settings (Member name, Node, and Generate unique ports) for any proxy cluster member that you just added, other than the first proxy cluster member, select that proxy cluster member, and then click **Edit**. When you finish changing the property settings, click **Update Member** to save your changes.

If you decide not to create a particular proxy cluster member, select the member and then click **Delete**.

You cannot edit or delete the first proxy cluster member or an already existing proxy cluster member.

If you create additional proxy cluster members immediately after you create the first proxy cluster member, the list of proxy cluster members includes a checklist in front of the names of these additional proxy cluster members. However, a check box does not appear in front of the name of the first proxy cluster member because you cannot delete this member or edit its settings. To modify the first proxy cluster member, click **Previous**.

Similarly, if you are adding proxy cluster members to a proxy cluster that already has existing members, the existing members appear in the list of proxy cluster members but a check box does not appear in front of the names of these proxy cluster members. To delete one of these existing members or to change the settings of one of these proxy cluster members, in the administrative console click **Servers > Clusters > Proxy server clusters > *proxy_cluster_name* > Cluster members**, and then select the member that you want to delete or whose configuration settings you want to change.

Member name

Specifies the name of the proxy server that is created for the proxy cluster.

The member name must be unique on the selected node.

Select node

Specifies the node on which the proxy server resides.

In a mixed cell environment, you can use any server from within the node group to create a new proxy cluster member. For example, if the node group to which the proxy cluster belongs consists of a V7.0 node, and a V6.1 node, you can use a server from either the V6.1 or the V7.0 node to create a new proxy cluster member.

Generate unique HTTP ports

Specifies that a unique HTTP port is generated for the proxy server. By generating unique HTTP ports for the proxy server, you avoid potential port collisions and configurations that are not valid.

Creating a proxy cluster: Summary settings

Use this administrative console page to view and save settings when you create a proxy cluster or proxy cluster member.

You can view this administrative console page whenever you create a new proxy cluster or a new proxy cluster member. This summary page displays your configuration changes before you commit the changes and the new proxy cluster or proxy cluster member is created.

To create a cluster, in the administrative console, click **Servers > Clusters > Proxy server clusters > New**.

To create a proxy cluster member for an existing proxy cluster, in the administrative console, click **Servers > Clusters > Proxy server clusters > New > *proxy_cluster_name* > Cluster members > New**. After you enter the required information about the new proxy cluster member, click **Add Member** to add this member to the proxy cluster member list.

The bounding node group of the proxy cluster is based on the first application server that is added as a member of the proxy cluster. The settings for this first member become the settings for the proxy cluster member template that is then used to create additional proxy cluster members.

To select a different bounding node group, in the administrative console, click **Servers > Clusters > Proxy server clusters > New > *proxy_cluster_name* > Cluster members > New**. Then select the appropriate node group for the new proxy cluster member. When you select a different node group, another proxy cluster member template is automatically created for that node group, if one does not already exist.

Review the changes to your configuration. Click Finish to complete and save your work.

Chapter 6. Managing a proxy server cluster

You can manage a proxy server cluster using either the administrative console or the wsadmin tool.

About this task

You can use either the administrative console or wsadmin to manage a proxy server cluster. To use the administrative console to manage your proxy server clusters, in the administrative console, click **Servers > Clusters > Proxy server clusters**. You have the following options: New, Delete, Start, Stop, Ripplestart, and ImmediateStop.

- To start one or more proxy server clusters, select the proxy server clusters and click either **Start** or **RippleStart**.
 - Start launches the server process of each member of the cluster by calling the node agent for each server to start the servers. After all servers are running, the state of the cluster changes to running. If the call to a node agent for a server fails, the server does not start.
 - RippleStart combines stopping and starting operations. It first stops and then restarts each member of the cluster. For example, your cluster contains 3 cluster members named server_1, server_2 and server_3. When you click RippleStart, server_1 stops and restarts, then server_2 stops and restarts, and finally server_3 stops and restarts. Use the RippleStart option instead of manually stopping and then starting all of the application servers in the cluster.
- To stop one or more proxy server clusters, select the proxy server clusters and click either **Stop** or **ImmediateStop**.
 - Stop halts each server in a manner that allows the server to finish existing requests and allows failover to another member of the cluster. When the stop operation begins, the cluster state changes to partially stopped. After all servers stop, the cluster state becomes stopped.
 - ImmediateStop stops the server quickly without regard to existing requests. The server ignores any current or pending tasks. When the stop operation begins, the cluster state changes to partially stopped. After all servers stop, the cluster state becomes stopped.
- To change the configuration settings, click the name of the proxy server cluster to access the Configuration tab.
- To create a new proxy server cluster, click **New**. Type the name for the cluster in the **Cluster name** field.
- To delete an existing proxy server cluster, click **Delete** and then click **OK**.
- To add new members to a proxy server cluster, click the name of an existing proxy server cluster, and click **Cluster members > New**. Type the name for the proxy server in the **Member name** field.

Results

You have successfully modified a proxy server cluster.

Chapter 7. Setting up caching in the proxy server

This topic provides information on caching static and dynamic content in the proxy server.

About this task

Complete the following steps to configure a proxy server such that it can cache static and dynamic content.

1. Configure the object cache instance for size, disk offload location, and other such capabilities, in the administrative console. Click **Servers > Server Types > WebSphere proxy servers > proxy_server_name > HTTP proxy server settings > Proxy cache instance config**. Repeat these steps on any nodes that have a proxy server.
2. Select the proxy cache store instance and enable configuration attributes such as cache size, disk offload, and cache replication. For disk offload, it is recommended that the location be set to a dedicated disk partition.
3. Enable caching at the proxy server, in the administrative console. Click **Servers > Server Types > WebSphere proxy servers > proxy_server_name > HTTP proxy server settings > Proxy settings** page in the administrative console.
4. Select **Enable caching** and choose a cache instance from the drop-down box.
 - a. To enable dynamic content to be cacheable with the proxy server, in the administrative console, click **Servers > Server Types > WebSphere proxy servers > proxy_server_name > HTTP proxy server settings > Proxy settings**, and then select **Cache dynamic content**. You enable cacheability and invalidation of dynamic content when you enable servlet caching on the application server, and specifying the cache criteria in a cachespec.xml file that is associated with that application. Invalidation is received by connecting to the cache update URI that is associated with the invalidation servlet hosted on the application server cluster.

Dynamic content is content that an application, that is hosted on an application server, generates. A proxy server caches dynamic content only if the content is identified as edge cacheable in the cachespec.xml file for the application. All of the information that describes the cache, such as the ID to use for the cache, dependency identifiers for invalidation, and expiration times, is also defined in the cachespec.xml file. Proxy Server uses the ESI protocol to obtain this information from the file.

See the *Administering applications and their environment* PDF for more information on how to set up a cachespec.xml file for an application.

Cached dynamic content can be invalidated by events in the application server. The ESI Invalidation Servlet, that is contained in the DynacacheEsi.ear application, propagates these invalidation events from the application server to the proxy server. The DynacacheEsi.ear is shipped with the product, and must be deployed in the cluster with the application that is generating the dynamic content for dynamic caching at the proxy server to function properly.

- b. Static caching is enabled by default when caching is enabled for the proxy server. *Static content* is Web content that is public and accompanied by HTTP response headers, such as EXPIRES and LAST_MODIFIED_TIME, that describe how long the response can be cached. The proxy server uses the HTTP 1.1 RFC (2616), which specifies how content should be treated and includes capabilities such as VARY header support for caching variants of the same resource Uniform Resource Identifier (URI).

Static cache rules collection

This topic lists the static cache rules for a proxy server. From this topic you can create, delete, or modify a static cache rule.

To view this administrative console topic, click **Servers > Server Types > WebSphere proxy servers > proxy_server_name > HTTP Proxy Server Settings > Static cache rules**.

The security proxy, or the intermediary that is the entry point into the enterprise, is responsible for terminating SSL connections with the client, authenticating the request, propagating the connection characteristics for the client, and any other credentials to the application server in the enterprise.

The proxy server enables security proxies to be identified so that private headers that are set in the request are propagated as they are to the application servers. Identify the list of security proxies that are trusted by the proxy server using the trusted security proxies field. To access this administrative console field, click **Servers > Server Types > WebSphere proxy servers > *proxy_server_name* > HTTP proxy server settings > Proxy settings**. You can find trusted security proxies in the **Security** section of this administrative console page.

URI Groups

The URI group name is a user-specified name.

The name field cannot contain the following characters: # \ / , : ; " * ? < > | = + & % ' .

The name that is defined must be unique among URI groups and cannot begin with a period or a space. A space does not generate an error, but leading and trailing spaces are automatically deleted.

Disable caching for this URI group

Specifies whether or not caching is disabled.

The default is `false`, which indicates that caching is enabled for the URI group.

Default expiration

The default expiration that you set for the cached response for the URI that is associated with this cache rule.

The default expiration value is in seconds.

Last modified factor

Use this field to derive the cache expiration value for a response if it does not have HTTP expiration headers and when it has a `LastModifiedTime` header in the response.

Name of the virtual host

A virtual host that is configured using the virtual host service. This virtual host is associated with the proxy server. This attribute is one of the elements in a request that is matched by the proxy server to determine if this rule is activated.

Static cache rule settings

Use this topic to configure a cache rule that is associated to a URI group for the proxy server. HTTP 1.1 defines a set of rules for proxy servers to cache content. Static cache rules enable these default rules to be overridden for a given address space. Before the rules have any meaning, you must enable caching on the **Servers > Server Types > WebSphere proxy servers > *proxy_server_name* > HTTP Proxy Server Settings > Proxy settings** administrative console page.

To view this administrative console page, click **Servers > Proxy Servers > *server_name* > HTTP Proxy Server Settings > Static cache rules > *URI_group***.

You can edit proxy server setting fields on the Configuration tab.

URI groups

URI groups, along with the virtual host, define the scope of the address space to have cache customizations performed.

The name field cannot contain the following characters: # \ / , : ; " * ? < > | = + & % ' .

The name that is defined must be unique among URI groups and cannot begin with a period or a space. A space does not generate an error, but leading and trailing spaces are automatically deleted.

Disable caching for this URI group

Disables caching for this address space. A user may wish to disable caching for a set of content servers that are known to contain sensitive or highly-personalized information.

The default is `false`, which indicates that caching is enabled for the URI group.

Default expiration

The default expiration value, in seconds, that is used to determine the validity of a cached response when all other HTTP 1.1 caching-related response headers do not give guidance. The default value is sufficient in most environments.

The default expiration value is in seconds.

Last modified factor

The percentage of a last-modified header for a response that determines the validity of a cached response when the response does not have explicit HTTP expiration headers. The default value is sufficient in most environments.

The default for this field is `0.0`.

Name of the virtual host

A virtual host that is configured using the virtual host service. This virtual host is associated with the proxy server. This attribute is one of the elements in a request that is matched by the proxy server to determine if this rule is activated.

The default for this field is `none`.

Chapter 8. Using Session Initiation Protocol to provide multimedia and interactive services

Follow these procedures for creating SIP applications and configuring the SIP container.

About this task

Session Initiation Protocol (SIP) is used to establish, modify, and terminate multimedia IP sessions including IP telephony, presence, and instant messaging. A *SIP application* in WebSphere is a Java program that uses at least one Session Initiation Protocol (SIP) servlet. A SIP servlet is a Java-based application component that is managed by a SIP servlet container.

The servlet container is a part of an application server that provides the network services over which requests and responses are received and sent. The servlet container decides which applications to invoke and in what order. A servlet container also contains and manages a servlet through its lifecycle.

SIP servlet containers can employ SIP proxy servers as surrogates to handle load balancing and security issues. For more information about the SIP proxy server, see: [Session Initiation Protocol proxy server](#).

This topic is divided into the following subsections:

- [Configure the SIP container](#): Information and instructions for configuring SIP container properties and timers.
- [Developing SIP applications](#): Reference information for developers.
- [Deploying SIP applications](#): Information for installing, starting, and stopping, your applications.
- [Securing SIP applications](#): Instructions for enabling security providers and setting up a trust association interceptor (TAI).
- [Tracing a SIP container](#): Troubleshoot SIP applications through traces on the SIP container.

SIP in WebSphere Application Server

WebSphere Application Server delivers rich SIP functionality throughout its infrastructure.

Session Initiation Protocol (SIP) has grown considerably since it first became an IETF standard in 1999. SIP was originally intended purely for video and audio but has now grown to become the control protocol for many interactive services, particularly in the peer-to-peer realm. SIP, and the standards surrounding SIP, provide the mechanisms to look up, negotiate, and manage connections to peers on any network over any other protocol.

The SIP Servlet 1.0 specification allows enterprise applications to use SIP and to support SIP-predominant applications in the Java EE environment.

WebSphere Application Server also provides tooling for the development environment and high performing Edge Components to handle distributed application environments.

In the application server, the Web container and SIP container are converged and are able to share session management, security and other attributes. In this model, an application that includes SIP servlets, HTTP servlets, and portlets can seamlessly interact, regardless of the protocol.

High availability, offered by the Network Deployment (ND) version of the product, of the converged applications is made possible because of the tight integration of HTTP and SIP in the base application server.

In front of a clustered application sits the proxy server, managing the traffic and workload of the SIP and HTTP traffic to the container. This proxy server is a stateless SIP proxy and a HTTP reverse proxy together, which uses the unified clustering framework and high availability (HA) manager services of the ND package to seamlessly monitor the health of the servers. The proxy server also can act as a stand-alone stateless SIP proxy in front of the SIP container in the application server when no HTTP traffic is present.

The proxy server uses the unified clustering framework and HA manager services of the ND package to perform failover work, when necessary. With the converged proxy and converged container, session failover is done with affinity to the application, allowing the HTTP and SIP sessions to be tied together automatically. Having the SIP and HTTP sessions automatically tied together from the container to the Proxy is another way the application server solution excels in converged environments.

It's important to note that the SIP function in the proxy server is stateless. The SIP RFC defines two types of proxy servers, one stateful and one stateless. Normally, a SIP proxy is a stateful instance and stateless proxies are specified as such. A stateful proxy participates in the call flows and is implemented using SIP servlets.

The stateless SIP proxy functionality in the proxy server allows the proxy to handle the workload, routing, and session affinity needs of the SIP container with less complexity. Being stateless, the proxy server can be fronted by a simple IP sprayer such as the load balancer component included in the ND package. If a proxy server fails, the affinity is to the container and not to the proxy itself so there is one less potential failure along the message flow.

SIP Infrastructure

The SIP infrastructure is a multi-tiered architecture made up of SIP containers, SIP proxies and an IP sprayer. The SIP container is a general purpose SIP application server. The SIP infrastructure consists of:

- SIP container – Web container extension that implements JSR 116 plus a SIP protocol stack that implements all pertinent RFCs.
- SIP proxy – Stateless edge device that handles I/O concentration, load balancing, and other functions, in a similar manner to the reverse HTTP proxy. This is not the same as the SIP proxy defined by RFC 3261.
- Load balancer – SIP enabled to interoperate with SIP proxies and SIP containers. The extendable SIP proxy handles session affinity, load balancing, and failover. The load balancer functions as a highly available IP sprayer to dispatch messages to the proxies.

SIP is a key element for many new applications, especially when converged with HTTP, including:

- Click-To-Call
- Voice over IP
- Third Party Call Control and Call Monitoring
- Presence and Instance Messaging

SIP applications

A *SIP application* is a Java program that uses at least one Session Initiation Protocol (SIP) servlet.

A SIP servlet is a Java-based application component that is managed by a SIP servlet container and that performs SIP signaling. Like other Java-based components, servlets are platform-independent Java classes that are compiled to platform-neutral bytecode that can be loaded dynamically into and run by a Java-enabled SIP application server. Containers, sometimes called servlet engines, are server extensions that handle servlet interactions. SIP servlets interact with clients by exchanging request and response messages through the servlet container.

SIP is used to establish, modify, and terminate multimedia IP sessions including IP telephony, presence, and instant messaging. "Presence" in this context refers to user status such as "Active," "Away," or "Do not disturb." The standard that defines a programming model for writing SIP-based servlet applications is JSR 116.

SIP container

This product complies with the following SIP standards:

IETF
JCP

For a complete list of the supported Internet Engineering Task Force (IETF) and Java Community Process (JCP) industry standards, see the "Compliance with industry SIP standards" topic linked below.

SIP industry standards compliance

The product implementation of Session Initiation Protocol (SIP) complies with industry standards for both a SIP container and SIP applications.

SIP container

This product complies with the following SIP standards:

IETF
JCP

This product also complies with the Internet Engineering Task Force (IETF) and Java Community Process (JCP) industry standards for SIP. The following table contains a list of the IETF and JCP standards.

Table 22. WebSphere Application Server complies with these SIP standards

Standard	Description
RFC 2543	SIP: Session Initiation Protocol
RFC 3261	SIP: Session Initiation Protocol
RFC 3262	Reliability of provisional responses in SIP
RFC 3265	SIP-specific event notification
RFC 3326	The Reason Header field for the SIP
RFC 3515	The SIP Refer method
RFC 3824	Using E.164 numbers with the SIP
RFC 3903	SIP Extension for event state publication
RFC 3263	Locating SIP servers Note: SIP does not support use of DNS procedures for a server to send a response to a back-up client if the primary client fails.

SIP applications

This product complies with standards for SIP applications.

Table 23. Compliance with standards for SIP applications

Standard	Description
RFC 2848	The PINT Service Protocol: Extensions to SIP and Session Description Protocol (SDP) for internet protocol (IP) access to telephone call services
RFC 2976	The SIP INFO method
RFC 3050	Common gateway interface for SIP

Table 23. Compliance with standards for SIP applications (continued)

Standard	Description
RFC 3087	Control of service context using SIP request-URI
RFC 3264	An offer and answer model with SDP
RFC 3266	Support for IPv6 in SDP
RFC 3312	Integration of resource management and SIP
RFC 3313	Private SIP extensions for media authorization
RFC 3319	Dynamic Host Configuration Protocol (DHCPv6) options for SIP servers
RFC 3327	SIP Extension Header field for registering non-adjacent contacts
RFC 3372	SIP for telephones (SIP-T): context and architectures
RFC 3398	Integrated Services Digital Network (ISDN) User Part (ISUP) to SIP mapping
RFC 3428	SIP extension for instant messaging
RFC 3455	Private Header (P-Header) extensions to the SIP for the 3rd-Generation Partnership Project (3GPP)
RFC 3578	Mapping of Integrated Services Digital Network (ISDN) User Part (ISUP) overlap signaling to the SIP
RFC 3603	Private SIP proxy-to-proxy extensions for supporting the PacketCable distributed call signaling architecture
RFC 3608	SIP Extension Header field for service route discovery during registration
RFC 3665	SIP basic call flow examples
RFC 3666	SIP Public Switched Telephone Network (PSTN) call flows
RFC 3680	A SIP event package for registrations
RFC 3725	Best current practices for third-party call control (3pcc) in the SIP
RFC 3840	Indicating user agent capabilities in the SIP
RFC 3842	A message summary and message waiting indication event package for the SIP
RFC 3856	A presence event package for the SIP
RFC 3857	A watcher information event template package for the SIP
RFC 3959	The early session disposition type for the SIP
RFC 3960	Early media and ringing tone generation in the SIP
RFC 3976	Interworking SIP and intelligent network (IN) applications
RFC 4032	Update to the SIP preconditions framework
RFC 4092	Usage of the SDP Alternative Network Address Types (ANAT) semantics in the SIP
RFC 4117	Transcoding services invocation in the SIP using third-party call control (3pcc)
RFC 4235	An invite-initiated dialog event package for the SIP
RFC 4240	Basic network media services with SIP
RFC 4353	A framework for conferencing with the SIP
RFC 4354	A SIP event package and data format for various settings in support for the push-to-talk over cellular (PoC) service
RFC 4411	Extending the SIP Reason Header for preemption events
RFC 4457	The SIP P-user-database Private-Header (P-Header)
RFC 4458	SIP URIs for applications such as voicemail and interactive voice response (IVR)
RFC 4483	A mechanism for content indirection in SIP messages
RFC 4497	Interworking between the SIP and QSIG

Table 23. Compliance with standards for SIP applications (continued)

Standard	Description
RFC 4508	Conveying feature tags with the SIP REFER method

Runtime considerations for SIP application developers

You should consider certain product runtime behaviors when you are writing Session Initiation Protocol (SIP) applications.

Container may accept non-SIP URI schemes

The SIP container will not reject a message if it doesn't recognize the scheme in the request URI because the container cannot know which URI schemes are supported by the applications. SIP elements may support a request URI with a scheme other than sip or sips, for example, the pres: scheme has a particular meaning for presence servers, but the container does not recognize it. It is up to the application to determine whether to accept or to reject a specific scheme. SIP elements may translate non-SIP URIs using any mechanism available, resulting in SIP URIs, SIPS URIs, or other schemes, like the tel URI scheme of RFC 2806 [9].

Directing requests in a multiple-container environment

In a multiple-container environment (SIP proxy plus SIP containers), when your application sends a request intended initially to be sent externally but later received, it should use the host and ports of the front-most load balancing element (either an IP sprayer for multiple SIP proxies, or the SIP proxy if only one exists). If the application uses the host name of a container instead of the front-most element, the request may be lost in the event of a failure.

For example, an application sends an INVITE request to itself, but the request must pass through an external accounting system through a pushed Route header. The application should set the INVITE request's URI to the host and port of the foremost element to ensure that failover occurs. The request will be routed to the accounting system via the pushed Route, and then sent back to the front load balancing element for processing.

Invoking session listener events

SipSessionListener and SipApplicationSessionListener events are invoked only if an application requests the corresponding session object. You do this by using in your application the method shown in Table 24.

Table 24. Methods that invoke session listener events

Event	Method
SipSessionListener	getSession()
SipApplicationSessionListener	getApplicationSession()

Session activation and passivation

During normal operation, this product never migrates a session from one server to another. Session migration occurs only as a result of a server failure. Therefore the SipSessionActivationListener method's passivation callback is never invoked. However, the activation callback is invoked when a failure forces session failover to a different server.

External resources

If a SIP application performs intensive I/O or accesses an external database, it may be blocked for several milliseconds. If possible, use asynchronous APIs for these resources. Under stress, a blocked SIP application may trigger a Request Timeout or re-transmission.

SIP application attributes

Avoid hanging large objects or BLOBs as SIP Session attributes (via `SIPSession.setAttribute` API). This may damage the overall performance when combined with high availability (HA). The same recommendation applies for `SIPApplicationSession.setAttribute`. In most cases, the large object can be replaced by several simple or composed strings.

SIP IBM Rational Application Developer for WebSphere framework

This page provides information about the SIP IBM Rational Application Developer for WebSphere framework.

WebSphere Application Server includes IBM Rational Application Developer for WebSphere to meet all the basic development needs for Java EE applications. Included in IBM Rational Application Developer for WebSphere is support for developing SIP servlet applications. IBM Rational Application Developer for WebSphere provides graphical deployment descriptor editors and basic wizards to get you started writing SIP servlets.

IBM Rational Application Developer for WebSphere also includes many other pieces that integrate well in WebSphere Application Server deployments, such as the Unit Test Environment, which provides the WebSphere Application Server servlet container to run SIP servlets in the development phase of the product, as well as tools for server automation and application packaging.

IBM Rational Application Developer for WebSphere supports:

- SIP servlet development (JSR 116)
- Converged SIP/HTTP applications
- Import/Export SAR packages
- SIP samples (call forward, call block, third party call)

SIP container

A *SIP container* is a Web application server component that invokes the Session Initiation Protocol (SIP) action servlet and that interacts with the action servlet to process SIP requests.

The servlet container provides the network services over which requests and responses are received and sent. It decides which applications to invoke and in what order. The container also contains and manages servlets through their life cycle.

A SIP servlet container manages the network listener points on which it listens for incoming SIP traffic. A listener point is a combination of transport protocol, IP address, and port number. The SIP servlet container supports the transport protocols UDP, TCP, and TLS over TCP.

The SIP servlet container can employ a SIP proxy server to route, load balance, and improve response times between SIP requests and back-end SIP container resources. For more information about the SIP proxy server, see: [Session Initiation Protocol proxy server](#).

SIP converged proxy

SIP in WebSphere offers a converged proxy.

The SIP converged proxy:

- Handles SIP and HTTP
- Provides application level session failover, regardless of protocol
- Fronts clusters of containers, SIP or HTTP
- Provides a highly scalable I/O concentration
- Handles session affinity
- Provides a framework for extending the base functions of proxy using an API consistent with proxy flows (Proxy Filter Layer)
- Contains support for failover and load balancing
- Provides first pass protocol validation
- Provides a framework for secure proxy functions – SSL termination, Outbound SSL, Client Side Certificates, etc.
- Allows for augmentation by our other products such as WebSphere XD
- Provides DMZ support.

SIP proxy setup considerations

- A single SIP proxy can front multiple SIP clusters.
- An IP sprayer is required for load balancing when deploying multiple SIP proxies into a single cell.
- Each SIP proxy must be configured with a default cluster. This is used to route inbound messages that do not match a cluster routing rule.
- When deploying converged applications, both HTTP and SIP should be enabled on the proxy.
- SIP proxies can be clustered.

SIP port relationships

When multiple servers, either containers or proxies are on the same host, each container or proxy must be configured with its own port.

SIP cluster routing and the default cluster

- A single SIP proxy can front multiple SIP clusters.
- Each SIP proxy must be configured with a default cluster which is used to route all messages that do not have an associated cluster routing rule.
- You can define cluster routing rules at each proxy. These dictate how messages are routed to the various backend clusters being fronted.
- By changing the default cluster, a SIP proxy can reroute messages to a new cluster containing an upgraded version of the deployed applications.

SIP high availability

SIP uses the high availability features in that are included in the product to offer a comprehensive high availability solution.

The following topics describe how high availability is implemented for this product.

- High availability manager
- “Replicating SIP sessions” on page 411
- “SIP session affinity and failover” on page 332

- “SIP cluster routing” on page 336
- “Upgrading SIP applications” on page 410
- “SIP IP sprayer” on page 337
- Setting up a high availability environment

SIP high availability architectural considerations

- Each container is handling state replication and SIP traffic.
- UDP/TCP/Multicast is used for state replication and SIP protocol.
- Complete session state is replicated between all servers in the replication domain.
- Each replication domain is optimal at two servers. Three servers per replication domain does not perform as well because each server must maintain a copy of all session data in the replication domain, but on a failure only half the servers replicated data is activated on any of the remaining servers.
- The number of SIP proxy servers per blade is dependent on the number of cluster addresses configured at the IP Sprayer.

SIP proxy configuration considerations

- No call state information is stored at the SIP proxy.
- If the IP sprayer is configured for MAC forwarding:
 - The SIP proxy must listen on the loopback address that corresponds to a cluster address configured at the IP sprayer.
 - The IP sprayer configuration dictates the number of SIP proxies per node (or blade).
 - Each configured cluster address can have at most one corresponding SIP proxy instance per node.

Core groups and Distribution and Consistency Services (DCS)

A core group is a set of processes that handles elections and heartbeats. The default order can be changed. The core group coordinator:

- Maintains all group information – group name, group members, the policy of the group, and the state of each group member
- Assigns singleton service to group members and handles failover of singleton services.

For more information on core groups, see Core groups (high availability domains)

DCS failure detection

The Core group failure detection mechanisms and algorithms are documented in Core group discovery and failure detection protocols

DCS failure detection includes:

- Active failure detection – a tunable heartbeat mechanism monitoring:
 - Time between heartbeats
 - Heartbeats lost before declaring failure
- TCP KEEP_ALIVE / Sockets closing
- Looking at hardware assist features

SIP session affinity and failover

SIP in WebSphere provides session affinity and failover.

SIP session affinity

A single SIP container in a cluster will handle all the messages associated with a single dialog. If a container fails in the middle of a dialog, a single server in the cluster will take over responsibility for the dialog. It is the SIP proxy's responsibility to maintain session affinity based on the session identifier (which includes the logical server name). Logical server information is published by the SIP container and consumed by the SIP proxy via Workload Management System (WLM).

Routing SIP messages based on the session ID

The `ibmsid` is embedded in various SIP messages and is used to route to specific sessions running on the SIP application server. Generally speaking, `VIA` headers are always used to route responses. The container will always embed an `ibmsid` in the `VIA` associated with the SIP app server. Here is an example of such a `VIA` header:

```
Via: SIP/2.0/UDP 9.51.252.69:5063;ibmsid=sipcontainer1.1153242645968.4_2_2;branch=z9hG4bK920196437955379
```

For proxy applications, the `ibmsid` (or session ID) is inserted in the initial request received from the UAC in the `Record-Route`. The UAS returns the same `Record-Route` with the session ID in the response. The UAC returns the `Route` header with the session ID in subsequent request within the dialog. For example:

```
Record-Route: <sip:protocol2.databeam.com:5060;transport=udp;ibmsid=sipcontainer1a.1138119214953.4_2_2;lr>
```

In this example the UAS returns the same `Record-Route` with the session ID in the response, and the UAC returns the `Route` header with the session ID in subsequent request within the dialog.

For UAC and UAS applications, the container acting as UAC or UAS will insert the Session ID into `To` tag (UAS) or `From` tag (UAC) (i.e. `To` tag `local.1132518053302_2_2`). The same `To` or `From` tag is included in subsequent request.

Encoded URIs will also contain an `ibmappid` (very similar to an `ibmsid`), which can then be sent in subsequent HTTP request. An important point here is that the IBM SIP infrastructure does not support transaction level failover. It only supports dialog failover for stable calls.

Here are the general rules for how the IBM SIP infrastructure decides which address to use for contact headers it embeds in outbound SIP messages:

- A standalone SIP application server will use its own address in contact headers that it needs to insert into SIP messages.
- A SIP application server fronted by a stateless SIP proxy will use the address of the SIP proxy in contact headers it needs to insert into SIP messages. The SIP application server discovers this address via WLM.
- A SIP application server fronted by a stateless SIP proxy which is then fronted by an IP Sprayer will use the address of the IP Sprayer in contact headers it needs to insert into SIP messages. The address of the IP sprayer must be configured at the SIP proxy through the administrative console and this address is published to the SIP application server through UCF.

Failover in the middle of a call

When a server fails, the sessions associated with the failed server are activated on the remaining containers in the replication domain. Once active, the containers handling the failed sessions publish the session location to the proxy servers fronting the cluster via WLM. When a message associated with one of the failed dialogs arrives at the proxy, the proxy pulls the session ID from the SIP message and uses that to look up the new container. When the failed server is restarted, it is added back to the cluster. It will then handle only newly created dialogs.

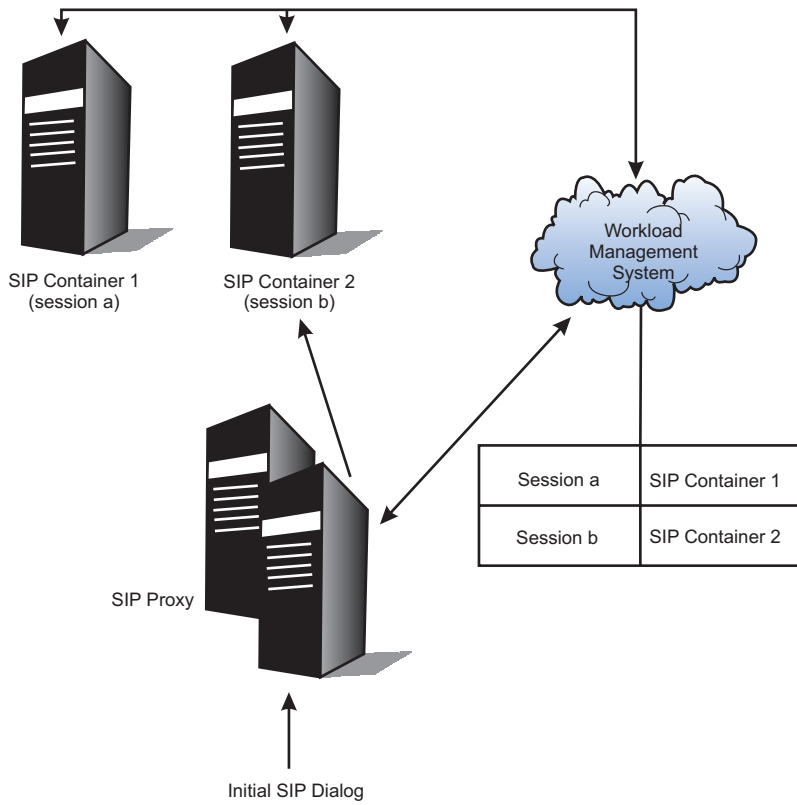


Figure 1. SIP Container Failover within a Cluster – Before

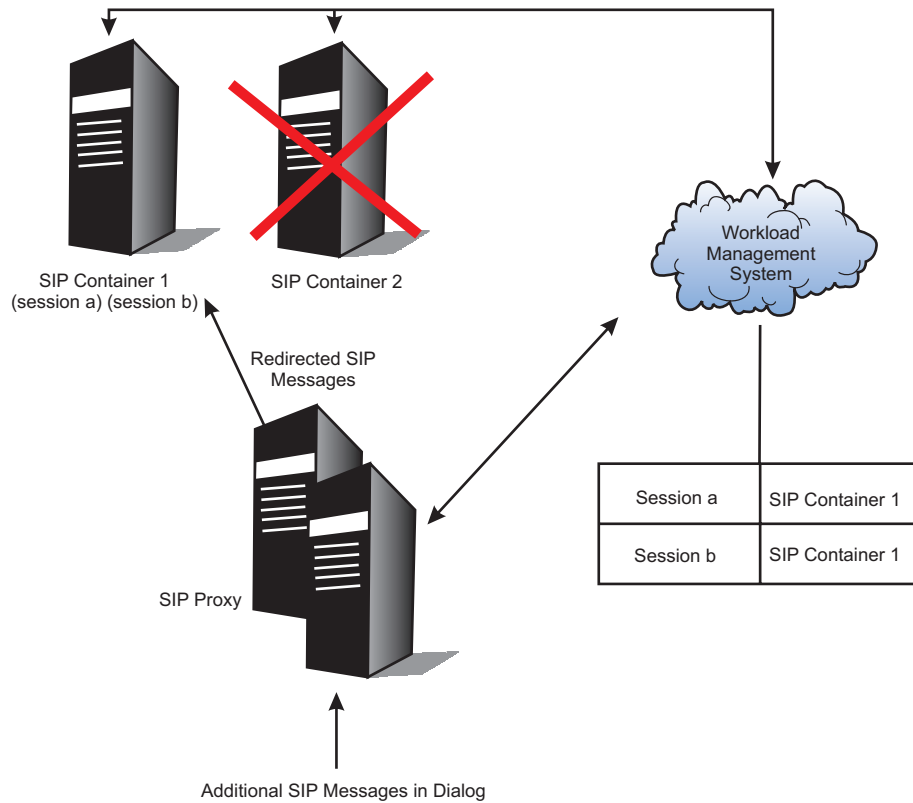


Figure 2. SIP Container Failover within a Cluster – After

Converged applications and failover

All messages associated with a converged HTTP/SIP Session are routed by the proxy to the same backend container using encoded URIs and SIP session affinity. HTTP and SIP utilize the same replication topology (they share the same replication settings). If a failure occurs, both HTTP and SIP requests associated with a failed dialog are routed to the same new backend server. Jsession cookies take precedence over Encoded URIs within the proxy when affinity targets are being determined.

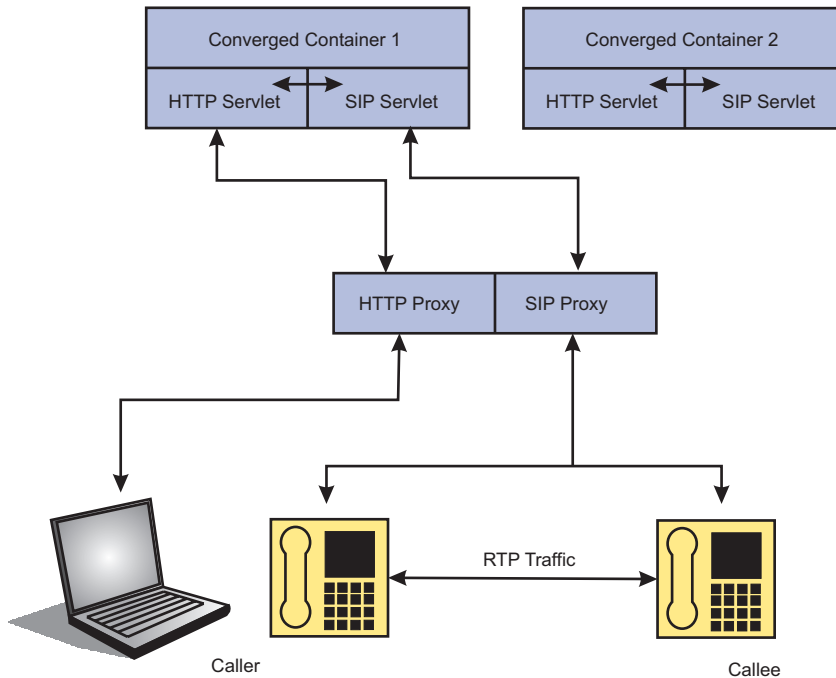


Figure 3. Converged Application Example

SIP cluster routing

Cluster routing rules is the method used to define how the SIP proxy routes messages to specific clusters in the cell.

A single SIP proxy can front multiple SIP clusters. If you are running more than one cluster of applications on the backend and want to use a single set of proxy servers, the proxy tier will need certain information in order to distinguish which cluster an *initial* request (one not in a pre-existing dialog with no affinity or route built-in) should be routed to.

In order to do this, SIP uses a set of cluster routing rules which allows the proxy to properly distinguish each request. Unlike HTTP, which can usually do this using the URI, there may be numerous other factors applicable in SIP.

Each SIP proxy is configured with a default cluster which is used to route all messages that do not have an associated cluster routing rule. Cluster routing rules, which dictate how messages get routed to the various backend clusters being fronted, can be defined at each proxy.

Cluster routing rules

Routing rules can be based on:

- Message type
- To: field
- From: field
- Destination address
- Source address
- Arbitrary header name with a unique value to distinguish the cluster

Cluster routing rules are used to route only messages associated with new dialogs. Once a dialog is established, a method to route to the specific cluster is embedded in SIP messages.

SIP IP sprayer

SIP in WebSphere provides an IP sprayer.

Scaling a deployment to include multiple SIP proxies requires a front end load balancer. The WebSphere Application Server Edge Components Load Balancer provides this capability. It provides a feature called the SIP advisor that detects outages so that messages only flow to healthy SIP proxies. The health detection is done by sending a SIP OPTIONS message over TCP to the SIP proxies and looking for a response. The SIP proxy is configured with the host information from the load balancer, so it can detect that an OPTIONS message was sent for health checking. If an error response occurs, or if there is no response, or if the TCP connection fails, the SIP proxy is removed from the load balancer's active list. The SIP advisor continues to look for the SIP proxy after a failure.

Configuring the IP sprayer

- MAC forwarding is used to load balance the SIP proxies.
- The SIP advisor must be started.
- The interval between health check requests is configurable (default = 7 seconds).

Configuring the SIP proxy

- Use the IP sprayer configuration panel to set the load balancer's host name and port. This will be used in outgoing SIP messages.
- The loopback device on the SIP proxy should be configured with the address the load balancer is listening on (cluster address, virtual IP address).
- The SIP proxy transport chains must be configured to listen on the loopback address.

SIP proxies publish client facing host interface information to the containers they are fronting via WLM. When fronted by an IP sprayer, the proxy will publish the IP sprayer information instead of its own host information. This information is inserted into Record-Route headers by the container for proxy applications running on the container. The SIP proxy also inserts this information into its via headers for outbound requests.

Configuring the SIP container

Configure the Session Initiation Protocol (SIP) container to adjust message response times or set a custom property.

About this task

Use the administrative console to configure SIP container settings. Complete the following steps to find and configure the SIP container settings.

1. Start WebSphere Application Server.
2. From the administrative console, click **Servers** → **Application servers** → *serverName*.
3. From **Container Settings**, expand **SIP Container Settings**, and click **SIP Container**. Select the container settings that you want to change.
 - From **General Properties**, you can configure session, message, and response time maximums. See *Session Initiation Protocol container settings* and *SIP container custom properties*.
 - From **Additional Properties**, you can define custom properties, manage transport chains or inbound channel settings, or configure the session manager.
4. After configuring the SIP container, click **Apply** to save the changes.
5. Restart WebSphere Application Server.

Results

Changes to the SIP container settings take effect after you restart WebSphere Application Server.

SIP container custom properties

You can add any of the following custom properties to the configuration settings for a Session Initiation Protocol (SIP) container.

To specify custom properties for a specific SIP container, navigate to the custom properties page, and then specify a value for the custom property.

Note: The custom properties are supported as the primary method of configuration. Therefore, if a custom property is set and then you set the corresponding setting in the administrative console, the custom property value is used.

1. In the administrative console, expand **Servers > Server Types > WebSphere application servers > *server_name*** to open the configuration tab for the server.
2. From **Container settings**, expand **SIP Container settings**, and click **SIP container**.
3. From **Additional properties**, select **Custom Properties** → **New**.
4. On the settings page, type the custom property to configure in the **Name** field, and then type the value of the custom property in the **Value** field.
5. Click **Apply** or **OK**.
6. Click **Save** on the console task bar to save your configuration changes.
7. Restart the server.

The following list of SIP container custom properties is provided with the product. These properties are not shown on the settings page for the container.

enable.system.headers.modify

Specifies whether the application has access to headers that are otherwise restricted.

Data type	String
Default	False

replicate.with.confirmed.dialog.only

Specifies whether to replicate the application session, even when no dialogs are confirmed. If the value is set to `false`, then the application session is replicated immediately after the session is created. Otherwise, the application session is only replicated when an associated dialog is confirmed.

Data type	String
Default	False

com.ibm.sip.sm.lnm.size

Specifies the number of logical names in the application server. Each SIP object that can be replicated, such as a SIP session, is associated with a logical name. All objects with the same logical name are replicated to the same back-up container. The proxy can route messages to the correct container using the logical name found in the message. The value must be greater than 1.

Data type	String
Default	10

auth.int.enable

Specifies the **auth-int** quality of protection (QOP) for digest authentication. Digest authentication defines two types of QOP: **auth** and **auth-int**. By default, **auth** is used. When this custom property is set to True, the highest level of protection is used, which is the **auth-int** QOP.

Data type	String
Default	False

DigestPasswordServerClass

Specifies the Java class name that implements the PasswordServer interface.

Data type	String
Default	LdapPasswordServer

com.ibm.websphere.sip.security.tai.usercachecleanperiod

Specifies the clean security subject cache period in minutes.

Data type	String
Default	15

com.ibm.ws.sip.tai.DisableSIPBasicAuth

Specifies whether to allow basic authentication for SIP.

Data type	String
Default	False

com.ibm.websphere.sip.security.digest.ldap.cachecleanperiod

Specifies the clean Lightweight Directory Access Protocol (LDAP) cache period in minutes.

Data type	String
Default	120

pws_atr_name

Specifies the LDAP attribute name that stores the user password.

Data type	String
Default	userpassword

javax.sip.transaction.invite.auto100

Specifies whether to automatically reply to invite requests with a 100 Trying response. Disabling this property might increase the number of invite retransmissions.

Data type	String
Default	True

Note: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core group that contains a mixture of Version 7.0 and Version 6.x processes.

Note: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use this custom property.

javax.sip.force.connection.reuse

Specifies whether to force reuse of inbound connections for outbound requests. This custom property is only relevant for stream transports, such as Transmission Control Protocol (TCP) and Transport Layer Security (TLS). Disabling this property causes the container to create a separate connection for outbound requests, even if an existing connection is already established to the same peer address. The connection is automatically reused if the top Via header in the inbound request contains an alias parameter. (<http://www.ietf.org/internet-drafts/draft-ietf-sip-connect-reuse-07.txt>)

Data type	String
Default	TrueFalse

Note: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core group that contains a mixture of Version 7.0 and Version 6.x processes.

Note: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use this custom property.

javax.sip.hide.message.body

Specifies to hide message content in logs. Set the value of this property to true to remove the message body text from SIP messages printed in the log files. This property only affects the representation of the messages in log files.

Data type	String
Default	False

Note: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core group that contains a mixture of Version 7.0 and Version 6.x processes.

Note: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use this custom property.

javax.sip.PATH_MTU

Specifies the maximum transmission unit, in bytes, for outbound User Datagram Protocol (UDP) requests. The SIP stack measures the size of a request before sending it out on the UDP channel. If the request is larger than the value specified for **PATH_MTU-200** (1300 bytes by default), then the transport is switched from UDP to TCP before transmission. Increase this value to send larger requests over the UDP channel; however, messages might be truncated or dropped. See the RFC 3261-18.1.1 specification for details.

Data type	String
Default	1500

Note: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core group that contains a mixture of Version 7.0 and Version 6.x processes.

Note: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use this custom property.

javax.sip.OUTBOUND_PROXY

Specifies the fixed address for routing all outbound SIP messages. The format is *address:port/transport*, such as *1.2.3.4:5065/tcp*.

Note: Do not use this property if the container is fronted by an application server SIP proxy.

Data type	String
Default	null

Note: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core group that contains a mixture of Version 7.0 and Version 6.x processes.

Note: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use this custom property.

javax.sip.trace.msg.in

Specifies whether to print incoming messages to a system.out file.

Data type	String
Default	False

javax.sip.trace.msg.out

Specifies whether to print outbound messages to a system.out file.

Data type	String
Default	False

javax.sip.stat.report.interval

Specifies the amount of time, in milliseconds, for reporting dispatch and timer statistics to a system.out file. A value of zero indicates no report.

Data type	String
Default	0

javax.sip.detect.pre.escaped.params

Specifies whether to prevent the container from re-escaping Uniform Resource Identifier (URI) parameters that were pre-escaped by the application.

Enabling this property provides the application with more control over escaping URI parameters, when calling the **javax.servlet.sip.SipFactory.createURI()** and the **javax.servlet.sip.SipURI.setParameter()** parameters.

By default, the container only escapes characters that it must encode according to the RFC 3261 25.1 specification. In some cases, however, escaping additional characters might be required. Due to a limitation in the JSR 116 (5.2.1) specification, the application cannot perform its own escaping. Because of this limitation, attempts by the application to encode URI parameters causes the container to re-encode the percent sign. If the value of this property is set to true, the container cannot re-encode the percent sign.

Setting the value to true is not in compliance with the JSR 116 (5.2.1) specification, but provides the application with greater control over URI parameter escaping. APAR PK37192 describes the problem and the workaround.

Data type	String
Default	False

DigestPasswordServerClass

Specifies types of user registries that are supported, except LDAP. To configure DigestTAI without the LDAP user registry, complete the following steps.

1. Create a class that implements this interface: `com.ibm.ws.sip.security.digest.DigestPasswordServer` 1
2. Add these properties to the `SlpDigestTAI` custom property.

name =

DigestPasswordServerClass value =

3. Ensure that all users declared by the **impl** class are declared in the user registry configured for the product security.

Data type	String
Default	impl

javax.sip.bind.retries

Specifies the amount of time, in milliseconds, between attempts to start the SIP channel if the SIP port is busy with another process during server startup.

Data type	String
Default	60

javax.sip.bind.retry.delay

Specifies the delay, in milliseconds, between attempts to start the SIP channel if the SIP port is busy with another process during server startup.

Data type	String
Default	5000

javax.sip.transaction.timer.t1

Specifies the amount of time, in milliseconds, for a network round trip delay for timer T1 for the RFC 3261 specification. The value is used as a base for calculating some timers and is relevant for all types of transactions, such as client, server, invite, and non-invite transactions.

Data type	String
Default	500

Note: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core group that contains a mixture of Version 7.0 and Version 6.x processes.

Note: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use this custom property.

javax.sip.transaction.timer.t2

Specifies the maximum time in milliseconds before retransmitting non-invite requests and invite responses for timer T2 for the RFC 3261 specification.

Data type	String
Default	4000

Note: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core group that contains a mixture of Version 7.0 and Version 6.x processes.

Note: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use this custom property.

javax.sip.transaction.timer.t4

Specifies the maximum amount of time, in milliseconds, for a message to remain in the network. This value is used as a base for calculating other timers for timer T4 for the RFC 3261 specification.

Data type	String
Default	5000

Note: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core group that contains a mixture of Version 7.0 and Version 6.x processes.

Note: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use this custom property.

javax.sip.transaction.timer.a

Specifies, for UDP only, the amount of time, in milliseconds, before retransmitting invite requests for timer A for the RFC 3261 specification. This property is relevant for the invite client transaction.

Data type	String
Default	javax.sip.transaction.timer.t1

Note: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core group that contains a mixture of Version 7.0 and Version 6.x processes.

Note: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use this custom property.

javax.sip.transaction.timer.b

Specifies the amount of time, in milliseconds, for the invite client transaction timeout timer (timer B) for the RFC 3261 specification.

Data type	String
Default	64*javax.sip.transaction.timer.t1

Note: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core group that contains a mixture of Version 7.0 and Version 6.x processes.

Note: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use this custom property.

javax.sip.transaction.timer.d

Specifies the wait time, in milliseconds, before retransmission of the invite response for timer D for the RFC 3261 specification. This property is relevant for the invite client transaction.

Data type	String
Default	32000

Note: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core group that contains a mixture of Version 7.0 and Version 6.x processes.

Note: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use this custom property.

javax.sip.transaction.timer.e

Specifies, for UDP only, the amount of time, in milliseconds, before the retransmission of the initial non-invite request for timer E for the RFC 3261 specification. This property is relevant for the non-invite client transaction.

Data type	String
Default	javax.sip.transaction.timer.t1

Note: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core group that contains a mixture of Version 7.0 and Version 6.x processes.

Note: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use this custom property.

javax.sip.transaction.timer.f

Specifies the amount of time, in milliseconds, for the non-invite transaction timeout timer (timer F) for the RFC 3261 specification. This property is relevant for the non-invite client transaction.

Data type	String
Default	64*javax.sip.transaction.timer.t1

Note: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core group that contains a mixture of Version 7.0 and Version 6.x processes.

Note: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use this custom property.

javax.sip.transaction.timer.g

Specifies the amount of time, in milliseconds, before retransmission of an initial invite response for timer G for the RFC 3261 specification. This property is relevant for the invite server transaction.

Data type	String
Default	javax.sip.transaction.timer.t1

Note: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core group that contains a mixture of Version 7.0 and Version 6.x processes.

Note: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use this custom property.

javax.sip.transaction.timer.h

Specifies the amount of time, in milliseconds, to wait for an acknowledgement (ACK) receipt for timer H for the RFC 3261 specification. This property is relevant for the invite server transaction.

Data type String
Default 64*javax.sip.transaction.timer.t1

Note: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core group that contains a mixture of Version 7.0 and Version 6.x processes.

Note: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use this custom property.

javax.sip.transaction.timer.i

Specifies the amount of time in milliseconds to wait for an ACK retransmission for timer I for the RFC 3261 specification. This property is relevant for the invite server transaction.

Data type String
Default javax.sip.transaction.timer.t4

Note: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core group that contains a mixture of Version 7.0 and Version 6.x processes.

Note: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use this custom property.

javax.sip.transaction.timer.j

Specifies the amount of time in milliseconds to wait for non-invite request retransmission for timer J for the RFC 3261 specification. This property is relevant for the non-invite server transaction.

Data type String
Default 64*javax.sip.transaction.timer.t1

Note: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core group that contains a mixture of Version 7.0 and Version 6.x processes.

Note: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use this custom property.

javax.sip.transaction.timer.k

Specifies the amount of time, in milliseconds, to wait for non-INVITE response retransmissions for timer K for the RFC 3261 specification. This property is relevant for the non-invite client transaction.

Data type String
Default javax.sip.transaction.timer.t4

Note: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core group that contains a mixture of Version 7.0 and Version 6.x processes.

Note: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use this custom property.

javax.sip.transaction.timer.non.invite.server

Specifies the amount of time, in milliseconds, for an Application Programming Interface (API) timer for the application to respond to a non-invite request. This property is relevant for non-invite server transactions.

This timer is not defined in the RFC specification. This property is needed to stop the transaction if the application does not generate a final response to the request. The timer starts when the request arrives in the stack and stops when a response is generated by the application. If no response is generated before the timer stops, then the transaction completes.

Data type	String
Default	34000

javax.sip.transaction.timer.invite.server

Specifies the amount of time, in milliseconds, for the timer to keep the invite server transaction in the complete state. This timer is not defined in the RFC specification.

To avoid creating a new server transaction when a client retransmits an invite request, keep the completed server transaction for a period of time before removing invite retransmissions. This timer is started when the transaction changes to the terminated state. When the timer completes, the transaction is removed.

Data type	String
Default	32000

javax.sip.transaction.timer.cancel

Specifies the amount of timer, in milliseconds, for the timer to keep the cancelled client transaction in the proceeding state before completing the cancelled transaction for the RFC 3261 9.1 specification. This property is relevant for the invite client transaction.

Data type	String
Default	64*javax.sip.transaction.timer.t1

SIP_RFC3263_nameserver

Specifies whether to allow a SIP URI to be resolved through Domain Name System (DNS) into the IP address, port, and transport protocol of the next hop.

The value of the property is a string containing one or two address and port tuples, where two tuples are separated by a space. The following examples specify a one address and port tuple or a two address and port tuple.

```
dottedDecimalAddress@.port  
hostname.domain@port  
IPV6address@port
```

The following example values represent a single tuple.

- 1.2.3.4@53
- example.com@53
- a:b:c::d@53

The following example values represent two tuples separated by a space.

- 1.2.3.4@53 example.com@53
- a:b:c::d@53 9.32.211.14@53

Data type	String
------------------	--------

Default null

Note: This custom property is deprecated. Do not use this custom property unless you are running in a mixed cell environment that includes at least one core group that contains a mixture of Version 7.0 and Version 6.x processes.

Note: If you are running in a mixed cell environment, and you have core groups that contain a mixture of Version 7.0 and Version 6.x processes, you must continue to use this custom property.

thread.message.queue.max.size

Specifies the maximum number of events allowed in the container threads queue. When this number is exceeded, the proxy server is notified that the container is overloaded and requests for new sessions are not accepted. Instead, the container returns an error message that indicates that the container is temporarily unavailable.

This value represents the total number of messages for all queues and reflects the state of the CPU. When the CPU approaches 100%, the maximum value for this custom property is reached quickly. Configure your system to limit the queue size and prevent the queue from reaching this threshold.

Data type String
Default 1000

weight.overload.watermark

Specifies the threshold value for the internal weight calculated by the container. When the container calculates the internal weight to be higher than the value specified, an overloaded container becomes available for service again.

This custom property represents a percentage of the maximum internal weight, such as 30 percent when the default value is set. When the high-water mark, or maximum threshold, is exceeded, the container waits until the weight drops below the maximum weight. This value cannot exceed 10.

Data type String
Default 3

sip.container.heartbeat.enabled

Specifies whether or not SIP network outage detection is enabled for the SIP container. SIP network outage detection allows the SIP proxy to send keepalive messages to the SIP container if the value of this property is set to true.

If the value is set to `false` for the SIP container, then this property has no effect on the SIP proxy. However, if the value is set to `true` for the SIP container, the value should also be set to `true` for the SIP proxy to ensure that keepalive messages are answered at the SIP container and not presented to the application.

Data type String
Default true

Using DNS procedures to locate SIP servers

The Session Initiation Protocol (SIP) can use Domain Name Server (DNS) procedures for a client to resolve a SIP Uniform Resource Identifier (URI).

About this task

WebSphere Application Server provides support for the RFC 3263 standard. This allows a SIP URI to be resolved through DNS into the IP address, port, and transport protocol of the next hop to contact.

Note: SIP does not support use of DNS procedures for a server to send a response to a back-up client if the primary client fails.

Complete these steps to configure WebSphere Application Server to support the RFC 3263 standard.

1. Start WebSphere Application Server.
2. From the administrative console, expand **Servers**, and click **Application servers** → **serverName**.
3. Under **General Properties**, check the **Enable locating SIP servers using DNS NAPTR records** checkbox, then fill in the **Primary DNS server name** and **Secondary DNS server name** fields.
4. Click **Apply** to save your changes.
5. Restart WebSphere Application Server.

What to do next

You must configure your DNS server in order for RFC 3263 support to work for the SIP container. The following example is a BIND db file for configuring RFC 3263 support on a DNS server.

```
; Copyright (C) 2004 Internet Systems Consortium, Inc. ("ISC")
; Copyright (C) 2001 Internet Software Consortium.
;
; Permission to use, copy, modify, and distribute this software for any
; purpose with or without fee is hereby granted, provided that the above
; copyright notice and this permission notice appear in all copies.
;
; THE SOFTWARE IS PROVIDED "AS IS" AND ISC DISCLAIMS ALL WARRANTIES WITH
; REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY
; AND FITNESS. IN NO EVENT SHALL ISC BE LIABLE FOR ANY SPECIAL, DIRECT,
; INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM
; LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE
; OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR
; PERFORMANCE OF THIS SOFTWARE.

; $Id: include.db,v 1.2.206.1 2004/03/06 10:22:13 marka Exp $

; Test $INCLUDE current domain name and origin semantics

example.com. 43200 IN SOA ns.example.com. email.example.com. ( 2003032001 10800 3600 604800 86400 )
;
example.com.      43200 IN NS      ns.example.com.
;
ns.example.com.  43200 IN A      10.0.0.20
sipserver1.example.com. 43200 IN A      10.0.0.21
sipserver2.example.com. 43200 IN A      10.0.0.22
sipserver3.example.com. 43200 IN A      10.0.0.23
;
router.example.com. 43200 IN CNAME sipserver3
;
sipserver1.example.com. 43200 IN AAAA   fec0:0:0:0:0:0:abcd
sipserver2.example.com. 43200 IN AAAA   fec0:0:0:0:0:0:abba
;
_sip_udp.example.com. 43200 IN SRV 2 0 5060 sipserver1.example.com.
_sip_udp.example.com. 43200 IN SRV 2 0 5060 sipserver2.example.com.
_sip_tcp.example.com. 43200 IN SRV 1 4 5060 sipserver1.example.com.
_sip_tcp.example.com. 43200 IN SRV 1 2 5060 sipserver2.example.com.
_sips_tcp.example.com. 43200 IN SRV 0 1 5061 sipserver1.example.com.
_sips_tcp.example.com. 43200 IN SRV 0 0 5061 sipserver2.example.com.
;
```

```
example.com. 43200 IN NAPTR 0 0 "s" "SIPS+D2T" "" _sips._tcp.example.com.
example.com. 43200 IN NAPTR 1 0 "s" "SIP+D2T" "" _sip._tcp.example.com.
example.com. 43200 IN NAPTR 2 0 "s" "SIP+D2U" "" _sip._udp.example.com.
```

SIP container settings

Use this page to configure the SIP container settings for Session Initiation Protocol (SIP).

To view this administrative console page, click **Application servers** → *serverName* → **SIP container settings** → **SIP container**.

Related tasks

../ae/ttrb_confighangdet.dita

Related information

Server collection

Use this topic to learn how to navigate within the administrative console to the pages where you can view information about the application servers, generic servers, Java message service (JMS) servers, and Web servers that are defined for your system.

Use SIP proxy for external domains

Specifies that, when true, the SIP container assumes the existence of the SIP Proxy running on another server process and therefore, delegates routing outbound traffic to that component.

Data type	Boolean
Default	True

Maximum application sessions

Specifies the maximum number of SIP application sessions that the container manages. When the maximum is reached, no new SIP conversations are started. When the maximum is exceeded in a clustered environment, the server does not forward new dialogs until the number of application sessions no longer exceeds the maximum.

Application sessions are typically created by new incoming calls, but can also be created by other events. The application session count does not impact failover, but applies only to new sessions that are created as a result of incoming calls.

When application sessions are transferred from one application server to another due to failover, the active application server inherits the sessions created on the failed server. In addition, the servlet might create a new application session in the SIP container by calling `SipFactory.createApplicationSession()`.

New application sessions created for events other than starting SIP conversations are not controlled by this setting. But all new application sessions are included when computing the maximum number of application sessions allowed. Thus, all active application sessions, including those not related to starting SIP conversations, can cause the maximum to be exceeded.

Data type	Integer
Default	120000 (recommended)
Range	1 <= n <= java.lang.Integer.MAX_VALUE

Maximum messages per averaging period

Specifies the maximum amount of SIP messages processed per averaging period. The averaging period is the period of time during which the average number of messages received by the container is calculated.

This average is used to determine the load for the container and to determine if the number of messages is approaching the maximum. When the maximum is exceeded, the stand-alone server or the proxy server continues to handle all in-dialog messages. Other non-dialog requests are rejected. When a container is in

an overloaded state, the proxy server returns a 503 error.

Data type	Integer
Default	5000 (recommended)
Range	1 <= n <= java.lang.Integer.MAX_VALUE

Maximum dispatch queue size

Specifies the size of the internal dispatch queue. When the maximum queue size threshold is reached, the container queue becomes overloaded and begins to drop requests for new sessions. In this case, the container does not report its overloaded state to the proxy server.

Configure your system to limit the queue size to prevent the queue from reaching this threshold. If the internal queue reaches the overloaded state, incoming UDP packets are dropped until the queue is no longer in an overloaded state. Limiting the queue size enables better recovery if the CPU is used by other processes or threads and prevents the container from reaching out-of-memory conditions. When the value is set to 0, the queue size is unlimited.

Data type	Integer
Default	5000 (recommended)
Range	0 <= n <= java.lang.Integer.MAX_VALUE

Maximum response time

Specifies the maximum response time, in milliseconds, for an application. When the amount of time is exceeded, the container notifies the clustering framework that it is unavailable. You can disable this feature in the administrative console by deselecting the check box and specifying a value of 0.

Use the maximum SIP response time setting cautiously because the calculated response time does not match the behavior of all applications. For requests, such as INVITE requests, where the responses are generated as a result of a user interaction, the calculated response time is extensive. However, the extensive response time is not caused by a delay in the SIP container. Therefore, you should not calculate the response time as a load factor. The recommended applications for effective calculation of response time are applications that respond immediately without a user interaction. The subscribe and register applications are relevant examples.

Data type	Integer
Default	0
Range	1 <= n <= java.lang.Integer.MAX_VALUE

Averaging period in milliseconds

Specifies the amount of time, in milliseconds, to use for calculating the maximum messages per averaging period. This setting is the sliding window during which the SIP container counts the number of messages sent to the container.

Data type	Integer
Default	1000 (recommended)
Range	1000 <= n <= java.lang.Integer.MAX_VALUE

Statistic update rate

Specifies control over the interval for which the container calculates averages and publishes statistics to Performance Monitoring Infrastructure.

Data type	Integer
Default	1000 (recommended)
Range	1000 <= n <= java.lang.Integer.MAX_VALUE

Locating SIP servers using DNS

Specifies whether to enable locating SIP servers using DNS (Directory Name Service).

A SIP Uniform Resource Identifier (URI) can be resolved through DNS into the Internet Protocol (IP) address, port, and transport protocol of the next hop.

The value for the **Primary DNS server name** or **Secondary DNS server name** fields is a string containing one address and port tuple. The following examples specify an address and port tuple.

Note: The container contacts the primary DNS server first; however, if the primary DNS server is unavailable, then the container contacts the secondary DNS server. If the secondary DNS server remains responsive, the container does not attempt to contact the primary DNS server.

dottedDecimalAddress@.port

hostname.domain@port

IPV6address@port

Data type	Boolean
Default	False

Primary DNS server name

Specifies an IP address and port tuple for the primary DNS server. If this server is not available, then the container sends a response to the secondary DNS server.

Data type	String
Default	empty string.

Secondary DNS server name

Specifies an IP address and port tuple for the secondary DNS server.

Data type	String
Default	empty string.

Thread pool

Specifies the available thread pools that you can select from the drop-down list for the SIP container to use when dispatching work. If you do not select a thread pool from the drop-down list, a default thread pool, which is automatically created by the container, is used.

It is recommended that you create a dedicated WebSphere thread pool for SIP applications. For general usage, this is a minimum of 15 threads and a maximum of 30 threads (with one thread per queue). This becomes convenient when combined with WebSphere hung thread detection. A hung thread can block many SIP messages, so it is important it be detected as soon as possible. However, the default hung thread detection threshold is too long for most SIP scenarios, so it is recommended you change it to 30 seconds. See the "Configuring the hang detection policy" topic (linked below) for the exact property names.

Data type	menu list
Default	None

SIP stack settings

Use this page to configure values for the Session Initiation Protocol (SIP) stack settings that are different from those specified in RFC 3261. The default values are the same as those specified for RFC 3261.

To view this administrative console page, click **Servers** → **Application servers** → *serverName* → **SIP container settings** → **SIP container** → **SIP stack**.

Automatically reply to INVITE with "100 Trying" response

Specifies whether to automatically reply to INVITE requests with a 100 Trying response. Disabling this setting might increase the number of INVITE retransmissions.

Data type	Boolean
Default	True

Hide message body

Specifies whether to hide message content in log files. Set the value to True to remove the message body text from SIP messages printed in the log files. This setting only affects the representation of the messages in log files.

Data type	Boolean
Default	False

Outbound connection timeout

Specifies the amount of time, in milliseconds, for creating outbound connections. This setting is relevant only for stream transports, such as Transmission Control Protocol (TCP) and Transport Layer Security (TLS).. The default value of 0 provides an infinite amount of time.

Data type	Integer
Default	0

Maximum transmission unit

Specifies the maximum transmission unit, in bytes, for outbound User Datagram Protocol (UDP) requests. The SIP stack measures the size of a request before sending it out on the UDP channel. If the request is higher than the value specified for PATH_MTU-200, 1300 bytes by default, then the transport is switched from UDP to TCP before transmission.

Increase this value to send larger requests over the UDP channel; however, messages might be truncated or discarded. See the RFC 3261-18.1.1 specification for details.

Data type	Integer
Default	1500

Outbound proxy

Specifies the fixed address for routing all outbound SIP messages. The format is address:port/transport, such as 1.2.3.4:5065/tcp.

Note: Do not use this setting if the container is fronted by an application server SIP proxy.

Data type	String
Default	empty string

SIP timers settings

Use this page to set values for the Session Initiation Protocol (SIP) timers that are different from those specified in RFC 3261. SIP timers provide a mechanism for session expiration. The default values are the same as those specified for RFC 3261.

To view this administrative console page, click **Servers** → **Application servers** → *serverName* → **SIP container settings** → **SIP container** → **SIP stack** → **SIP timers**.

The following SIP timers can be configured from the administrative console.

T1

Specifies the amount of time, in milliseconds, for a network round trip delay for timer T1 for the RFC 3261 specification. The value is used as a base for calculating some timers and is relevant for all types of transactions, such as client, server, invite, and non-invite transactions.

Data type	Integer
Default	500

A Specifies, for UDP only, the amount of time, in milliseconds, before retransmitting invite requests for timer A for the RFC 3261 specification. This setting is relevant for the invite client transaction.

Data type	Integer
Default	T1

B Specifies the amount of time, in milliseconds, for the invite client transaction timeout timer (timer B) for the RFC 3261 specification.

Data type	Integer
Default	T1*64

E Specifies, for UDP only, the amount of time, in milliseconds, before the retransmission of the initial non-invite request for timer E for the RFC 3261 specification. This setting is relevant for the non-invite client transaction.

Data type	Integer
Default	800

F Specifies the amount of time, in milliseconds, for the non-invite transaction timeout timer (timer F) for the RFC 3261 specification. This setting is relevant for the non-invite client transaction.

Data type	Integer
Default	24000

G Specifies the amount of time, in milliseconds, before retransmission of an initial invite response for timer G for the RFC 3261 specification. This setting is relevant for the invite server transaction.

Data type	Integer
Default	T1

H Specifies the amount of time, in milliseconds, to wait for an acknowledgement (ACK) receipt for timer H for the RFC 3261 specification. This setting is relevant for the invite server transaction.

Data type	Integer
Default	T1*64

J Specifies the amount of time, in milliseconds, to wait for non-invite request retransmission for timer J for the RFC 3261 specification. This setting is relevant for the non-invite server transaction.

Data type	Integer
Default	T1*64

T2

Specifies the maximum amount of time, in milliseconds, before retransmitting non-invite requests and invite responses for timer T2 for the RFC 3261 specification.

Data type	Integer
Default	4000

T4

Specifies the maximum amount of time, in milliseconds, for a message to remain in the network. This value is used as a base for calculating other timers for timer T4 for the RFC 3261 specification.

Data type	Integer
Default	5000

I Specifies the amount of time, in milliseconds, to wait for an ACK retransmission for timer I for the RFC 3261 specification. This setting is relevant for the invite server transaction.

Data type	Integer
Default	T4

K Specifies the amount of time, in milliseconds, to wait for non-INVITE response retransmissions for timer K for the RFC 3261 specification. This setting is relevant for the non-invite client transaction.

Data type	Integer
Default	T4

D

Specifies the amount of time to wait, in milliseconds, before retransmission of the invite response for timer D for the RFC 3261 specification. This setting is relevant for the invite client transaction.

Data type	Integer
Default	32000

SIP digest authentication settings

Use this page to configure Session Initiation Protocol (SIP) digest authentication settings that are different from those specified in RFC 3261. The default values are the same as those specified for RFC 3261.

To view this administrative console page, click **Security** → **Global Security** → **Authentication** → **Web and SIP Security** → **SIP digest authentication**.

Enable digest authentication integrity

Specifies the authentication integrity (auth-int) quality of protection (QOP) for digest authentication. Digest authentication defines two types of QOP: auth and auth-int. By default, basic authentication (auth) is used. If the value is set to True, the auth-int QOP is used, which is the highest level of protection.

Data type	Boolean
Default	False

Enable SIP basic authentication

Specifies the authentication (auth) quality of protection (QOP) for digest authentication. Digest authentication defines two types of QOP: auth and auth-int. By default, basic authentication (auth) is used. If the value is set to True, basic authentication will be performed. It will not be processed by the Trust Association Interceptor.

Data type	Boolean
Default	True

Enable multiple use of nonce

Specifies whether to enable multiple uses of the same nonce. If you use the same nonce more than once, then less system resources are required, however, your system is not as secure.

Data type	Boolean
Default	False

Enable nonce maximum age

Specifies the amount of time, in milliseconds, for which a nonce is valid. If the value is set to 1, then the amount of time is considered to be infinite.

Data type	Integer
Default	1

LDAP cache clean intervals

Specifies the amount of time that must expire, in minutes, before the LDAP cache is cleaned.

Data type	Integer
Default	120

LDAP password attribute name

Specifies the LDAP attribute name that stores the user password .

Data type	String
Default	userpassword

User cache clean intervals

Specifies the amount of time that must expire, in minutes, before the security subject cache is cleaned.

Data type	Integer
Default	15

Digest password server class

Specifies the Java class name that implements the PasswordServer interface.

Data type	String
Default	LdapPasswordServer

Hashedcredentials

Specifies the name of the LDAP field that contains the hashed credentials. If a value is specified for this setting, then this setting overrides the pws_atr_name setting.

Data type	String
Default	empty string

Hashedrealm

Specifies the realm for hashed credentials, if the hashed credentials setting is enabled.

Data type String
Default empty string

Configuring SIP timers

You can configure SIP timers to set values for the Session Initiation Protocol (SIP) timers that are different from default values specified in RFC 3261. SIP timers provide a mechanism for session expiration. The default values are the same as those specified for RFC 3261.

Before you begin

Before you begin

Ensure that clusters and standalone servers are created and federated.

About this task

You can set values for the SIP timers from the administrative console.

Note: If you have already used the custom properties to specify a value for a SIP timer, then the custom property value is the primary value. Therefore, the SIP timer value specified from the administrative console is not used.

1. From the administrative console, click **Servers** → *server_name* → **SIP container** → **SIP stack** → **SIP timers**.
2. Specify a value for a timer by clicking the Use Custom Value check box beside the timer.
3. Specify a value for the timer in the Value column.
4. Click **OK**.
5. Restart the application server.

Results

The container uses the times specified in the administrative console and not the RFC defaults.

SIP timer summary

Request for Comments (RFC) 3261, “SIP: Session Initiation Protocol,” specifies various timers that SIP uses.

Table 25 summarizes for each SIP timer the default value, the section of RFC 3261 that describes the timer, and the meaning of the timer.

Table 25. Summary of SIP timers

Timer	Default value	Section	Meaning
T1	500 ms	17.1.1.1	Round-trip time (RTT) estimate
T2	4 sec.	17.1.2.2	Maximum retransmission interval for non-INVITE requests and INVITE responses
T4	5 sec.	17.1.2.2	Maximum duration that a message can remain in the network
Timer A	initially T1	17.1.1.2	INVITE request retransmission interval, for UDP only
Timer B	64*T1	17.1.1.2	INVITE transaction timeout timer
Timer C	> 3 min.	16.6 bullet 11	Proxy INVITE transaction timeout
Timer D	> 32 sec. for UDP 0 sec. for TCP and SCTP	17.1.1.2	Wait time for response retransmissions

Table 25. Summary of SIP timers (continued)

Timer	Default value	Section	Meaning
Timer E	initially T1	17.1.2.2	Non-INVITE request retransmission interval, UDP only
Timer F	64*T1	17.1.2.2	Non-INVITE transaction timeout timer
Timer G	initially T1	17.2.1	INVITE response retransmission interval
Timer H	64*T1	17.2.1	Wait time for ACK receipt
Timer I	T4 for UDP	17.2.1	Wait time for ACK retransmissions
	0 sec. for TCP and SCTP		
Timer J	64*T1 for UDP	17.2.2	Wait time for retransmissions of non-INVITE requests
	0 sec. for TCP and SCTP		
Timer K	T4 for UDP	17.1.2.2	Wait time for response retransmissions
	0 sec. for TCP and SCTP		

Configuring digest authentication for SIP

You can configure Session Initiation Protocol (SIP) digest authentication settings that are different from those specified in RFC 3261. The default values are the same as those specified for RFC 3261.

Before you begin

Before you begin

Ensure that clusters and standalone servers are created and federated.

About this task

You can set values for the SIP timers from the administrative console.

Note: If you have already used the custom properties to specify a value for a SIP timer, then the custom property value is the primary value. Therefore, the SIP timer value specified from the administrative console is not used.

1. From the administrative console, click **Security** → **Global security** → **Web and SIP security** → **Digest authentication**.
2. Specify a value for one or more settings.
3. Click **OK**.
4. Restart the application server.

Results

The container uses the digest authentication values specified in the administrative console and not the RFC defaults.

Performing controlled failover of SIP applications

You can take an active application out of the loop via a controlled failover.

Before you begin

SipContainerMBean is used to initiate a server quiesce through wsadmin (command line interface). This MBean is used to set the container's weight to 0, which prevents new messages from being routed to it.

WebSphere Application Server PMI is used to monitor the server's active sessions. The remaining active sessions can be watched by enabling a counter on the server being quiesced. The server can be shut down once the number of active sessions reaches an acceptable level. A script can be written to monitor active sessions and shut down the server when an acceptable threshold is achieved.

About this task

Quiescing a single server

1. On an ND machine, start the wsadmin utility:
 - a. Go to `<nd_installation_path>/bin`
 - b. Run the command: `./setupCmdLine.sh`
 - c. Run the command: `./wsadmin.sh`
 - d. Verify that received: `wsadmin>`
2. Run the command: `set scBean [$AdminControl queryNames type=SipContainerMBean,process=<server name>,*]`
3. Run the command: `$AdminControl invoke $scBean quiesce true`

From the admin console, command line or scripts

Use the following commands to stop application servers from the admin console, command line or scripts:

- **Stop:** Quiesces the application server. The sessions will failover to another server. It is important to manually quiesce the server on shutdown.
- **Immediate Stop:** Stops the server, but bypasses the normal server quiesce process that supports in-flight requests to complete before shutting down the entire server process. This shutdown mode is faster than the normal server stop processing, but some application clients can receive exceptions.
- **Terminate:** Deletes the application server process. Use this if immediate stop fails to stop the server.

SIP PMI counters

The Session Initiation Protocol (SIP) provides the following counters in the WebSphere Performance Monitoring Infrastructure (PMI) to monitor the performance of SIP.

Counter definitions

Sip container module

Name	Key	Description	Granularity	Type	Level	Overhead	ID
Incoming traffic	incoming.traffic	Average number of messages handled by the container calculated over configurable period	Per server	CountStatistic	All	Low	3
New SIP App sessions	new.sip.app.session	Average number of new SIP application sessions created in the container and calculated over configurable period	Per server	CountStatistic	All	Low	4
Response Time	response.time	The average amount of time that it takes between when a message gets into the container and when a response is sent from the container.	Per server	CountStatistic	All	Low	5
Queue Size	queue.size	Size of the invoke queue in WebSphere Application Server	Per server	CountStatistic	All	Low	6

Session module

Name	Key	Description	Granularity	Type	Level	Overhead	ID
Number of active SIP sessions	active.sip.sessions	The number of SIP sessions belongs to each application	Per application	CountStatistic	All	Low	11
Number of active SIP application sessions	active.sip.app.sessions	The number of SIP application sessions belongs to each application	Per application	CountStatistic	All	Low	12

Inbound request module

Name	Key	Description	Granularity	Type	Level	Overhead	ID
Number of Inbound NOT SIP STANDARD requests	inbound.request.other	The number of inbound NOT SIP STANDARD requests that belong to each application	Per application	CountStatistic	All	Low	60
Number of Inbound REGISTER requests	inbound.request.register	The number of inbound REGISTER requests that belong to each application	Per application	CountStatistic	All	Low	61
Number of Inbound INVITE requests	inbound.request.invite	The number of inbound INVITE requests that belong to each application	Per application	CountStatistic	All	Low	62
Number of Inbound ACK requests	inbound.request.ack	The number of Inbound ACK requests that belong to each application	Per application	CountStatistic	All	Low	63
Number of Inbound OPTIONS requests	inbound.request.options	The number of Inbound OPTIONS requests that belong to each application	Per application	CountStatistic	All	Low	64
Number of Inbound BYE requests	inbound.request.bye	The number of Inbound BYE requests that belong to each application	Per application	CountStatistic	All	Low	65
Number of Inbound CANCEL requests	inbound.request.cancel	The number of Inbound CANCEL requests that belong to each application	Per application	CountStatistic	All	Low	66
Number of Inbound PRACK requests	inbound.request.prack	The number of Inbound PRACK requests that belong to each application	Per application	CountStatistic	All	Low	67
Number of Inbound INFO requests	inbound.request.info	The number of Inbound INFO requests that belong to each application	Per application	CountStatistic	All	Low	68
Number of Inbound SUBSCRIBE requests	inbound.request.subscribe	The number of Inbound SUBSCRIBE requests that belong to each application	Per application	CountStatistic	All	Low	69
Number of Inbound NOTIFY requests	inbound.request.notify	The number of Inbound NOTIFY requests that belong to each application	Per application	CountStatistic	All	Low	70
Number of Inbound MESSAGE requests	inbound.request.message	The number of Inbound MESSAGE requests that belong to each application	Per application	CountStatistic	All	Low	71
Number of Inbound PUBLISH requests	inbound.request.publish	The number of Inbound PUBLISH requests that belong to each application	Per application	CountStatistic	All	Low	72
Number of Inbound REFER requests	inbound.request.refer	The number of Inbound REFER requests that belong to each application	Per application	CountStatistic	All	Low	73
Number of Inbound UPDATE requests	inbound.request.update	The number of Inbound UPDATE requests that belong to each application	Per application	CountStatistic	All	Low	74

Inbound info response module

Name	Key	Description	Granularity	Type	Level	Overhead	ID
Number of Inbound 100 responses	inbound.response.info.100	The number of Inbound 100 (Trying) responses that belong to each application	Per application	CountStatistic	All	Low	1100
Number of Inbound 180 responses	inbound.response.info.180	The number of Inbound 180 (Ringing) responses that belong to each application	Per application	CountStatistic	All	Low	1180
Number of Inbound 181 responses	inbound.response.info.181	The number of Inbound 181 (Call being forwarded) responses that belong to each application	Per application	CountStatistic	All	Low	1181
Number of Inbound 182 responses	inbound.response.info.182	The number of Inbound 182 (Call Queued) responses that belong to each application	Per application	CountStatistic	All	Low	1182
Number of Inbound 183 responses	inbound.response.info.183	The number of Inbound 183 (Session Progress) responses that belong to each application	Per application	CountStatistic	All	Low	1183

Inbound success response module

Name	Key	Description	Granularity	Type	Level	Overhead	ID
Number of successful Inbound 200 responses	inbound.response.success.200	The number of Inbound 200 (OK) responses that belong to each application	Per application	CountStatistic	All	Low	1200
Number of successful Inbound 202 responses	inbound.response.success.202	The number of Inbound 202 (Accepted) responses that belong to each application	Per application	CountStatistic	All	Low	1202

Inbound redirect responses module

Name	Key	Description	Granularity	Type	Level	Overhead	ID
Number of Inbound 300 responses	inbound.response.redirect.300	The number of Inbound 300 (Multiple choices) responses that belong to each application	Per application	CountStatistic	All	Low	1300
Number of Inbound 301 responses	inbound.response.redirect.301	The number of Inbound 301 (Moved Permanently) responses that belong to each application	Per application	CountStatistic	All	Low	1301
Number of Inbound 302 responses	inbound.response.redirect.302	The number of Inbound 302 (Moved Temporarily) responses that belong to each application	Per application	CountStatistic	All	Low	1302
Number of Inbound 305 responses	inbound.response.redirect.305	The number of Inbound 305 (Use Proxy) responses that belong to each application	Per application	CountStatistic	All	Low	1305
Number of Inbound 380 responses	inbound.response.redirect.380	The number of Inbound 380 (Alternative Service) responses that belong to each application	Per application	CountStatistic	All	Low	1380

Inbound fail response module

Name	Key	Description	Granularity	Type	Level	Overhead	ID
Number of Inbound 400 responses	inbound.response.fail.400	The number of Inbound 400 (Bad Request) responses that belong to each application	Per application	CountStatistic	All	Low	1400
Number of Inbound 401 responses	inbound.response.fail.401	The number of Inbound 401 (Unauthorized) responses that belong to each application	Per application	CountStatistic	All	Low	1401
Number of Inbound 402 responses	inbound.response.fail.402	The number of Inbound 402 (Payment Required) responses that belong to each application	Per application	CountStatistic	All	Low	1402
Number of Inbound 403 responses	inbound.response.fail.403	The number of Inbound 403 (Forbidden) responses that belong to each application	Per application	CountStatistic	All	Low	1403
Number of Inbound 404 responses	inbound.response.fail.404	The number of Inbound 404 (Not Found) responses that belong to each application	Per application	CountStatistic	All	Low	1404
Number of Inbound 405 responses	inbound.response.fail.405	The number of Inbound 405 (Method Not Allowed) responses that belong to each application	Per application	CountStatistic	All	Low	1405
Number of Inbound 406 responses	inbound.response.fail.406	The number of Inbound 406 (Not Acceptable) responses that belong to each application	Per application	CountStatistic	All	Low	1406
Number of Inbound 407 responses	inbound.response.fail.407	The number of Inbound 407 (Proxy Authentication Required) responses that belong to each application	Per application	CountStatistic	All	Low	1407
Number of Inbound 408 responses	inbound.response.fail.408	The number of Inbound 408 (Request Timeout) responses that belong to each application	Per application	CountStatistic	All	Low	1408
Number of Inbound 410 responses	inbound.response.fail.410	The number of Inbound 410 (Gone) responses that belong to each application	Per application	CountStatistic	All	Low	1410
Number of Inbound 413 responses	inbound.response.fail.413	The number of Inbound 413 (Request Entity Too Large) responses that belong to each application	Per application	CountStatistic	All	Low	1413
Number of Inbound 414 responses	inbound.response.fail.414	The number of Inbound 414 (Request URI Too Long) responses that belong to each application	Per application	CountStatistic	All	Low	1414

Number of inbound 415 responses	inbound.response.fail.415	The number of Inbound 415 (Unsupported Media Type) responses that belong to each application	Per application	CountStatistic	All	Low	1415
Number of inbound 416 responses	inbound.response.fail.416	The number of Inbound 416 (Unsupported URI Scheme) responses that belong to each application	Per application	CountStatistic	All	Low	1416
Number of inbound 420 responses	inbound.response.fail.420	The number of Inbound 420 (Bad Extension) responses that belong to each application	Per application	CountStatistic	All	Low	1420
Number of inbound 421 responses	inbound.response.fail.421	The number of Inbound 421 (Extension Required) responses that belong to each application	Per application	CountStatistic	All	Low	1421
Number of inbound 423 responses	inbound.response.fail.423	The number of Inbound 423 (Interval Too Brief) responses that belong to each application	Per application	CountStatistic	All	Low	1423
Number of inbound 480 responses	inbound.response.fail.480	The number of Inbound 480 (Temporarily Unavailable) responses that belong to each application	Per application	CountStatistic	All	Low	1480
Number of inbound 481 responses	inbound.response.fail.481	The number of Inbound 481 (Call Leg Done) responses that belong to each application	Per application	CountStatistic	All	Low	1481
Number of inbound 482 responses	inbound.response.fail.482	The number of Inbound 482 (Loop Detected) responses that belong to each application	Per application	CountStatistic	All	Low	1482
Number of inbound 483 responses	inbound.response.fail.483	The number of Inbound 483 (Too Many Hops) responses that belong to each application	Per application	CountStatistic	All	Low	1483
Number of inbound 484 responses	inbound.response.fail.484	The number of Inbound 484 (Address Incomplete) responses that belong to each application	Per application	CountStatistic	All	Low	1484
Number of inbound 485 responses	inbound.response.fail.485	The number of Inbound 485 (Ambiguous) responses that belong to each application	Per application	CountStatistic	All	Low	1485
Number of inbound 486 responses	inbound.response.fail.486	The number of Inbound 486 (Busy Here) responses that belong to each application	Per application	CountStatistic	All	Low	1486

Number of inbound 487 responses	inbound.response.fail.487	The number of Inbound 487 (Request Terminated) responses that belong to each application	Per application	CountStatistic	All	Low	1487
Number of Inbound 488 responses	inbound.response.fail.488	The number of Inbound 488 (Not Acceptable Here) responses that belong to each application	Per application	CountStatistic	All	Low	1488
Number of inbound 491 responses	inbound.response.fail.491	The number of Inbound 491 (Request Pending) responses that belong to each application	Per application	CountStatistic	All	Low	1491
Number of Inbound 493 responses	inbound.response.fail.493	The number of Inbound 493 (Undecipherable) responses that belong to each application	Per application	CountStatistic	All	Low	1493

Inbound server fail response module

Name	Key	Description	Granularity	Type	Level	Overhead	ID
Number of Inbound 500 responses	inbound.response.serverFail.500	The number of Inbound 500 (Server Internal Error) responses that belong to each application	Per application	CountStatistic	All	Low	1500
Number of Inbound 501 responses	inbound.response.serverFail.501	The number of Inbound 501 (Not Implemented) responses that belong to each application	Per application	CountStatistic	All	Low	1501
Number of Inbound 502 responses	inbound.response.serverFail.502	The number of Inbound 502 (Bad Gateway) responses that belong to each application	Per application	CountStatistic	All	Low	1502
Number of Inbound 503 responses	inbound.response.serverFail.503	The number of Inbound 503 (Service Unavailable) responses that belong to each application	Per application	CountStatistic	All	Low	1503
Number of Inbound 504 responses	inbound.response.serverFail.504	The number of Inbound 504 (Server Timeout) responses that belong to each application	Per application	CountStatistic	All	Low	1504
Number of Inbound 505 responses	inbound.response.serverFail.505	The number of Inbound 505 (Version Not Supported) responses that belong to each application	Per application	CountStatistic	All	Low	1505
Number of Inbound 513 responses	inbound.response.serverFail.513	The number of Inbound 513 (Message Too Large) responses that belong to each application	Per application	CountStatistic	All	Low	1513

Inbound global fail response module

Name	Key	Description	Granularity	Type	Level	Overhead	ID
Number of Inbound 600 responses	inbound.response.globalFail.600	The number of Inbound 600 (Busy Everywhere) responses that belong to each application	Per application	CountStatistic	All	Low	1600
Number of Inbound 603 responses	inbound.response.globalFail.603	The number of Inbound 603 (Decline) responses that belong to each application	Per application	CountStatistic	All	Low	1603
Number of Inbound 604 responses	inbound.response.globalFail.604	The number of Inbound 604 (Does Not Exit Anywhere) responses that belong to each application	Per application	CountStatistic	All	Low	1604
Number of Inbound 606 responses	inbound.response.globalFail.606	The number of Inbound 606 (Not Acceptable Anywhere) responses that belong to each application	Per application	CountStatistic	All	Low	1606

Outbound request module

Name	Key	Description	Granularity	Type	Level	Overhead	ID
Number of Outbound NOT SIP STANDARD requests	outbound.request.other	The number of Outbound NOT SIP STANDARD requests that belong to each application	Per application	CountStatistic	All	Low	80
Number of Outbound REGISTER requests	outbound.request.register	The number of Outbound REGISTER requests that belong to each application	Per application	CountStatistic	All	Low	81
Number of Outbound INVITE requests	outbound.request.invite	The number of Outbound INVITE requests that belong to each application	Per application	CountStatistic	All	Low	82
Number of Outbound ACK requests	outbound.request.ack	The number of Outbound ACK requests that belong to each application	Per application	CountStatistic	All	Low	83
Number of Outbound OPTIONS requests	outbound.request.options	The number of Outbound OPTIONS requests that belong to each application	Per application	CountStatistic	All	Low	84
Number of Outbound BYE requests	outbound.request.bye	The number of Outbound BYE requests that belong to each application	Per application	CountStatistic	All	Low	85
Number of Outbound CANCEL requests	outbound.request.cancel	The number of Outbound CANCEL requests that belong to each application	Per application	CountStatistic	All	Low	86
Number of Outbound PRACK requests	outbound.request.prack	The number of Outbound PRACK requests that belong to each application	Per application	CountStatistic	All	Low	87
Number of Outbound INFO requests	outbound.request.info	The number of Outbound INFO requests that belong to each application	Per application	CountStatistic	All	Low	88
Number of Outbound SUBSCRIBE requests	outbound.request.subscribe	The number of Outbound SUBSCRIBE requests that belong to each application	Per application	CountStatistic	All	Low	89
Number of Outbound NOTIFY requests	outbound.request.notify	The number of Outbound NOTIFY requests that belong to each application	Per application	CountStatistic	All	Low	90
Number of Outbound MESSAGE requests	outbound.request.message	The number of Outbound MESSAGE requests that belong to each application	Per application	CountStatistic	All	Low	91
Number of Outbound PUBLISH requests	outbound.request.publish	The number of Outbound PUBLISH requests that belong to each application	Per application	CountStatistic	All	Low	92
Number of Outbound REFER requests	outbound.request.refer	The number of Outbound REFER requests that belong to each application	Per application	CountStatistic	All	Low	93
Number of Outbound UPDATE requests	outbound.request.update	The number of Outbound UPDATE requests that belong to each application	Per application	CountStatistic	All	Low	94

Outbound info response module

Name	Key	Description	Granularity	Type	Level	Overhead	ID
Number of Outbound 100 responses	outbound.response.info.100	The number of Outbound 100 (Trying) responses that belong to each application	Per application	CountStatistic	All	Low	2100
Number of Outbound 180 responses	outbound.response.info.180	The number of Outbound 180 (Ringing) responses that belong to each application	Per application	CountStatistic	All	Low	2180
Number of Outbound 181 responses	outbound.response.info.181	The number of Outbound 181 (Call being forwarded) responses that belong to each application	Per application	CountStatistic	All	Low	2181
Number of Outbound 182 responses	outbound.response.info.182	The number of Outbound 182 (Call Queued) responses that belong to each application	Per application	CountStatistic	All	Low	2182
Number of Outbound 183 responses	outbound.response.info.183	The number of Outbound 183 (Session Progress) responses that belong to each application	Per application	CountStatistic	All	Low	2183

Outbound success response module

Name	Key	Description	Granularity	Type	Level	Overhead	ID
Number of Outbound 200 responses	outbound.response.success.200	The number of Outbound 200 (OK) responses that belong to each application	Per application	CountStatistic	All	Low	2200
Number of Outbound 202 responses	outbound.response.success.202	The number of Outbound 202 (Accepted) responses that belong to each application	Per application	CountStatistic	All	Low	2202

Outbound redirect response module

Name	Key	Description	Granularity	Type	Level	Overhead	ID
Number of Outbound 300 responses	outbound.response.redirect.300	The number of Outbound 300 (Multiple choices) responses that belong to each application	Per application	CountStatistic	All	Low	2300
Number of Outbound 301 responses	outbound.response.redirect.301	The number of Outbound 301 (Moved Permanently) responses that belong to each application	Per application	CountStatistic	All	Low	2301
Number of Outbound 302 responses	outbound.response.redirect.302	The number of Outbound 302 (Moved Temporarily) responses that belong to each application	Per application	CountStatistic	All	Low	2302
Number of Outbound 305 responses	outbound.response.redirect.305	The number of Outbound 305 (Use Proxy) responses that belong to each application	Per application	CountStatistic	All	Low	2305
Number of Outbound 380 responses	outbound.response.redirect.380	The number of Outbound 380 (Alternative Service) responses that belong to each application	Per application	CountStatistic	All	Low	2380

Outbound fail response module

Name	Key	Description	Granularity	Type	Level	Overhead	ID
Number of Outbound 400 responses	outbound.response.fail.400	The number of Outbound 400 (Bad Request) responses that belong to each application	Per application	CountStatistic	All	Low	2400
Number of Outbound 401 responses	outbound.response.fail.401	The number of Outbound 401 (Unauthorized) responses that belong to each application	Per application	CountStatistic	All	Low	2401
Number of Outbound 402 responses	outbound.response.fail.402	The number of Outbound 402 (Payment Required) responses that belong to each application	Per application	CountStatistic	All	Low	2402
Number of Outbound 403 responses	outbound.response.fail.403	The number of Outbound 403 (Forbidden) responses that belong to each application	Per application	CountStatistic	All	Low	2403
Number of Outbound 404 responses	outbound.response.fail.404	The number of Outbound 404 (Not Found) responses that belong to each application	Per application	CountStatistic	All	Low	2404
Number of Outbound 405 responses	outbound.response.fail.405	The number of Outbound 405 (Method Not Allowed) responses that belong to each application	Per application	CountStatistic	All	Low	2405
Number of Outbound 406 responses	outbound.response.fail.406	The number of Outbound 406 (Not Acceptable) responses that belong to each application	Per application	CountStatistic	All	Low	2406
Number of Outbound 407 responses	outbound.response.fail.407	The number of Outbound 407 (Proxy Authentication Required) responses that belong to each application	Per application	CountStatistic	All	Low	2407
Number of Outbound 408 responses	outbound.response.fail.408	The number of Outbound 408 (Request Timeout) responses that belong to each application	Per application	CountStatistic	All	Low	2408
Number of Outbound 410 responses	outbound.response.fail.410	The number of Outbound 410 (Gone) responses that belong to each application	Per application	CountStatistic	All	Low	2410
Number of Outbound 413 responses	outbound.response.fail.413	The number of Outbound 413 (Request Entity Too Large) responses that belong to each application	Per application	CountStatistic	All	Low	2413
Number of Outbound 414 responses	outbound.response.fail.414	The number of Outbound 414 (Request URI Too Long) responses that belong to each application	Per application	CountStatistic	All	Low	2414

Number of Outbound 415 responses	outbound.response.fail.415	The number of Outbound 415 (Unsupported Media Type) responses that belong to each application	Per application	CountStatistic	All	Low	2415
Number of Outbound 416 responses	outbound.response.fail.416	The number of Outbound 416 (Unsupported URI Scheme) responses that belong to each application	Per application	CountStatistic	All	Low	2416
Number of Outbound 420 responses	outbound.response.fail.420	The number of Outbound 420 (Bad Extension) responses that belong to each application	Per application	CountStatistic	All	Low	2420
Number of Outbound 421 responses	outbound.response.fail.421	The number of Outbound 421 (Extension Required) responses that belong to each application	Per application	CountStatistic	All	Low	2421
Number of Outbound 423 responses	outbound.response.fail.423	The number of Outbound 423 (Interval Too Brief) responses that belong to each application	Per application	CountStatistic	All	Low	2423
Number of Outbound 480 responses	outbound.response.fail.480	The number of Outbound 480 (Temporarily Unavailable) responses that belong to each application	Per application	CountStatistic	All	Low	2480
Number of Outbound 481 responses	outbound.response.fail.481	The number of Outbound 481 (Call Leg Done) responses that belong to each application	Per application	CountStatistic	All	Low	2481
Number of Outbound 482 responses	outbound.response.fail.482	The number of Outbound 482 (Loop Detected) responses that belong to each application	Per application	CountStatistic	All	Low	2482
Number of Outbound 483 responses	outbound.response.fail.483	The number of Outbound 483 (Too Many Hops) responses that belong to each application	Per application	CountStatistic	All	Low	2483
Number of Outbound 484 responses	outbound.response.fail.484	The number of Outbound 484 (Address Incomplete) responses that belong to each application	Per application	CountStatistic	All	Low	2484
Number of Outbound 485 responses	outbound.response.fail.485	The number of Outbound 485 (Ambiguous) responses that belong to each application	Per application	CountStatistic	All	Low	2485
Number of Outbound 486 responses	outbound.response.fail.486	The number of Outbound 486 (Busy Here) responses that belong to each application	Per application	CountStatistic	All	Low	2486

Number of Outbound 487 responses	outbound.response.fail.487	The number of Outbound 487 (Request Terminated) responses that belong to each application	Per application	CountStatistic	All	Low	2487
Number of Outbound 488 responses	outbound.response.fail.488	The number of Outbound 488 (Not Acceptable Here) responses that belong to each application	Per application	CountStatistic	All	Low	2488
Number of Outbound 491 responses	outbound.response.fail.491	The number of Outbound 491 (Request Pending) responses that belong to each application	Per application	CountStatistic	All	Low	2491
Number of Outbound 493 responses	outbound.response.fail.493	The number of Outbound 493 (Undecipherable) responses that belong to each application	Per application	CountStatistic	All	Low	2493

Outbound server fail response module

Name	Key	Description	Granularity	Type	Level	Overhead	ID
Number of Outbound 500 responses	outbound.response.serverFail.500	The number of Outbound 500 (Server Internal Error) responses that belong to each application	Per application	CountStatistic	All	Low	2500
Number of Outbound 501 responses	outbound.response.serverFail.501	The number of Outbound 501 (Not Implemented) responses that belong to each application	Per application	CountStatistic	All	Low	2501
Number of Outbound 502 responses	outbound.response.serverFail.502	The number of Outbound 502 (Bad Gateway) responses that belong to each application	Per application	CountStatistic	All	Low	2502
Number of Outbound 503 responses	outbound.response.serverFail.503	The number of Outbound 503 (Service Unavailable) responses that belong to each application	Per application	CountStatistic	All	Low	2503
Number of Outbound 504 responses	outbound.response.serverFail.504	The number of Outbound 504 (Server Timeout) responses that belong to each application	Per application	CountStatistic	All	Low	2504
Number of Outbound 505 responses	outbound.response.serverFail.505	The number of Outbound 505 (Version Not Supported) responses that belong to each application	Per application	CountStatistic	All	Low	2505
Number of Outbound 513 responses	outbound.response.serverFail.513	The number of Outbound 513 (Message Too Large) responses that belong to each application	Per application	CountStatistic	All	Low	2513

Outbound global fail response module

Name	Key	Description	Granularity	Type	Level	Overhead	ID
Number of Outbound 600 responses	outbound.response.globalFail.600	The number of Outbound 600 (Busy Everywhere) responses that belong to each application	Per application	CountStatistic	All	Low	2600
Number of Outbound 603 responses	outbound.response.globalFail.603	The number of Outbound 603 (Decline) responses that belong to each application	Per application	CountStatistic	All	Low	2603
Number of Outbound 604 responses	outbound.response.globalFail.604	The number of Outbound 604 (Does Not Exit Anywhere) responses that belong to each application	Per application	CountStatistic	All	Low	2604
Number of Outbound 606 responses	outbound.response.globalFail.606	The number of Outbound 606 (Not Acceptable Anywhere) responses that belong to each application	Per application	CountStatistic	All	Low	2606

Developing SIP applications

A SIP application is a set of SIP servlets packaged in a SIP application archive file (SAR).

About this task

A SIP servlet is an application component managed by the SIP container that performs SIP signaling. The programming and deployment models are analogous to Web servlets and therefore will be mapped to the WebSphere administrative model accordingly. It is possible to include Web servlets in a SAR file (along with the required web.xml deployment descriptor) to create what is known as a converged application. See JSR 116 for details on SIP applications, servlets, converged applications, and status codes.

Developing a custom trust association interceptor

When you develop Session Initiation Protocol (SIP) applications, you can create a custom trust association interceptor (TAI).

Before you begin

You may want to familiarize yourself with the general TAI information contained in the Trust Associations documentation. Developing a SIP TAI is similar to developing any other custom interceptors used in trust associations. In fact, a custom TAI for a SIP application is actually an extension of the trust association interceptor model. Refer to the Developing a custom interceptor for trust associations section for more details.

About this task

TAI can be invoked by a SIP servlet request or a SIP servlet response. To implement a custom SIP TAI, you need to write your own Java class.

1. Write a Java class that extends the `com.ibm.wsspi.security.tai.BaseTrustAssociationInterceptor` class and implements the `com.ibm.websphere.security.tai.SIPTrustAssociationInterceptor` interface. Those classes are defined in the `WASProductDir/plugins/com.ibm.ws.sip.container_1.0.0.jar` file, where `WASProductDir` is the fully qualified path name of the directory in which WebSphere Application Server is installed.
2. Declare the following Java methods:

```
public int initialize(Properties properties) throws WebTrustAssociationFailedException;
```

This is invoked before the first message is processed so that the implementation can allocate any resources it needs. For example, it could establish a connection to a database.

`WebTrustAssociationFailedException` is defined in the `WASProductDir/plugins/com.ibm.ws.runtime_1.0.0.jar` file. The value of the `properties` argument comes from the **Custom Properties** set in this step.

```
public void cleanup();
```

This is invoked when the TAI should free any resources it holds. For example, it could close a connection to a database.

```
public boolean isTargetProtocolInterceptor(SipServletMessage sipMsg) throws  
WebTrustAssociationFailedException;
```

Your custom TAI should use this method to handle the `sipMsg` message. If the method returns `false`, WebSphere ignores your TAI for `sipMsg`.

```
public TAIResult negotiateValidateandEstablishProtocolTrust (SipServletRequest req,  
SipServletResponse resp) throws WebTrustAssociationFailedException;
```

This method returns a `TAIResult` that indicates the status of the message being processed and a user ID or the unique ID for the user who is trying to authenticate. If authentication succeeds, the `TAIResult` should contain the status `HttpServletResponse.SC_OK` and a principal. If authentication fails, the `TAIResult` should contain a return code of

HttpServletResponse.SC_UNAUTHORIZED (401), SC_FORBIDDEN (403), or SC_PROXY_AUTHENTICATION_REQUIRED (407). This only indicates whether or not the container should accept a message for further processing. To challenge an incoming request, the TAI implementation must generate and send its own SipServletResponse containing a challenge. The exception should be thrown for internal TAI errors. Table 26 describes the argument values and resultant actions for the negotiateValidateandEstablishProtocolTrust method.

Table 26. Description of negotiateValidateandEstablishProtocolTrust arguments and actions

Argument or action	For a SIP request	For a SIP response
Value of req argument	The incoming request	Null
Value of resp argument	Null	The incoming response
Action for valid response credentials	Return TAIResult.status containing SC_OK and a user ID or unique ID	Return TAIResult.status containing SC_OK and a user ID or unique ID
Action for incorrect response credentials	Return the TAIResult with the 4xx status	Return the TAIResult with the 4xx status

The sequence of events is as follows:

- a. The SIP container maps initial requests to applications by using the rules in each applications deployment descriptor; subsequent messages are mapped based on JSR 116 mechanisms.
- b. If any of the applications require security, the SIP container invokes any defined TAI implementations for the message.
- c. If the message passes security, the container invokes the corresponding applications.

Your TAI implementation can modify a SIP message, but the modified message will not be usable within the request mapping process, because it finishes before the container invokes the TAI.

The com.ibm.wsspi.security.tai.TAIResult class, defined in the *WASProductDir/plugins/com.ibm.ws.runtime_1.0.0.jar* file, has three static methods for creating a TAIResult. The TAIResult create methods take an int type as the first parameter. WebSphere Application Server expects the result to be a valid HTTP request return code and is interpreted as follows:

If the value is HttpServletResponse.SC_OK, this response tells WebSphere Application Server that the TAI has completed its negotiation. The response also tells WebSphere Application Server use the information in the TAIResult to create a user identity.

The created TAIResults have the meanings shown in Table 27.

Table 27. Meanings of TAIResults

TAIResult	Explanation
public static TAIResult create(int status);	Indicates a status to WebSphere Application Server. The status should not be SC_OK because the identity information is provided.
public static TAIResult create(int status, String principal);	Indicates a status to WebSphere Application Server and provides the user ID or the unique ID for this user. WebSphere Application Server creates credentials by querying the user registry.
public static TAIResult create(int status, String principal, Subject subject);	Indicates a status to WebSphere Application Server, the user ID or the unique ID for the user, and a custom Subject. If the Subject contains a Hashtable, the principal is ignored. The contents of the Subject becomes part of the eventual user Subject.

```
public String getVersion();
```

This method returns the version number of the current TAI implementation.

```
public String getType();
```

This method's return value is implementation-dependent.

3. Compile the implementation after you have implemented it. For example: `/opt/WebSphere/AppServer/java/bin/javac -classpath /opt/WebSphere/AppServer/plugins/com.ibm.ws.runtime_1.0.0.jar;/opt/WebSphere/AppServer/lib/j2ee.jar;/opt/WebSphere/AppServer/plugins/com.ibm.ws.sip.container_1.0.0.jar myTAIImpl.java`
 - a. For each server within a cluster, copy the class file to a location in the WebSphere class path (preferably the `WASProductDir/plugin/` directory).
 - b. Restart all the servers.
4. Delete the default WebSEAL interceptor in the administrative console and click **New** to add your custom interceptor. Verify that the class name is dot-separated and appears in the class path.
5. Click the **Custom Properties** link to add additional properties that are required to initialize the custom interceptor. These properties are passed to the `initialize(Properties properties)` method of your implementation when it extends the `com.ibm.websphere.security.WebSphereBaseTrustAssociationInterceptor` as described in the previous step.
6. Save and synchronize (if applicable) the configuration.
7. Restart the servers for the custom interceptor to take effect.

Developing SIP applications that support PRACK

A SIP response to an INVITE request can be final or provisional. Final responses are always sent reliably, but provisional responses typically are not. For cases where you need to send a provisional response reliably, you can use the PRACK (Provisional response acknowledgement) method.

Before you begin

For you to be able to develop applications that support PRACK, the following criteria must be met:

- The client that sends the INVITE request must put a 100rel tag in the Supported or the Require header to indicate that the client supports PRACK.
- The SIP servlet must respond by invoking the `sendReliably()` method instead of the `send()` method to send the response.

About this task

PRACK is described in the following standards:

- RFC 3262 (“Reliability of Provisional Responses in the Session Initiation Protocol (SIP)”), which extends RFC 3261 (“SIP: Session Initiation Protocol”), adding PRACK and the option tag 100rel.
- Section 6.7.1 (“Reliable Provisional Responses”) of JSR 116 (“SIP Servlet API Version 1.0”).
- For an application acting as a proxy, do this:
 - Make your application generate and send a reliable provisional response for any INVITE request that has no tag in the To field.
- For an application acting as a user agent client (UAC), do this:
 - Make your application add the 100rel tag to outgoing INVITE requests. The option tag must appear in either the Supported header or the Require header.
 - Within your application’s `doProvisionalResponse(...)` method, prepare the application to create and send PRACK requests for incoming reliable provisional responses. The application must create the PRACK request on the response’s dialog through a `SipSession.createRequest(...)` method, and it must set the RACK header according to RFC 3262 Section 7.2 (“RACK”).

- The application that acts as an UAC will not receive doPrack() methods. The UAC sends INVITE and receives Reliable responses. When the UAC receives the Reliable response, it sends PRACK a request to the UAS and receives a 200 OK on the PRACK so it should next implement doResponse() in order to receive it.
- For an application acting as a user agent server (UAS), do this:
 - If an incoming INVITE request requires the 100rel tag, trying to send a 101-199 response unreliably by using the send() method causes an Exception.
 - Make the application declare a SipErrorListener to receive noPrackReceived() events when a reliable provisional response is not acknowledged within 64*T1 seconds, where T1 is a SIP timer. Within the noPrackReceived() event processing, the application should generate and send a 5xx error response for the associated INVITE request per JSR 116 Section 6.7.1.
 - Make the application have at most one outstanding, unacknowledged reliable provisional response. Trying to send another one before the first's acknowledgement results in an Exception.
 - Make sure that the application enforces the RFC 3262 offer/answer semantics surrounding PRACK requests containing session descriptions. Specifically, a servlet must not send a 2xx final response if any unacknowledged provisional responses contained a session description.

Setting up SIP application composition

The JSR 116 standard for SIP applications states in section 2.4 that multiple applications may be invoked for the same SIP request. The process of setting up applications to comply with this standard is called application composition.

Before you begin

About this task

Application composition requires that implementations use a cascaded services model. The cascaded services model requires that service applications triggered on the same host are triggered in sequence, as if the triggering occurred on different hosts. Therefore responses flow upstream and hit applications in the reverse order of the corresponding requests.

The JSR 116 standard does not specify how to implement application composition, thus there are many ways to comply with this standard. For WebSphere Application Server, composition of the application depends on the deployed application order, and on the order of mapping rules within the deployment descriptor of each application.

- For an initial incoming request, the SIP container tries each potential rule in order. When the container finds the n^{th} match, the container invokes the corresponding servlet.
- If the servlet must proxy the request, the container scans the rules again to search for additional matches. When the container finds the $(n+1)^{th}$ match, the container invokes the corresponding servlet.
- Any servlet in the same application as the previously invoked servlet is excluded from the matching process. No servlet can be invoked twice for the same SIP request.

You can specify load on start-up priority. The `<load-on-startup>` in the sip.xml defines the order in which servlets are initialized on startup. If this value is lower than zero, the servlets are initialized when the first request is matched to them according to matching rule and composition order. Zero is a legitimate weight for startup initialization order. If this tag does not exist or if it contains a negative value, the servlet does not initialize at startup.

You should also add `<load-on-startup>` to the same tag in the web.xml if you are changing it manually. It is the WebContainer that loads servlets (and siplets), and it looks only at the web.xml. When deploying a SAR, only the sip.xml needs to be changed. The web.xml is automatically constructed correctly after deployment.

The load-on-startup tag embedded in the SIP deployment descriptor tag for a servlet dictates the order that the application is loaded on start up of the server. It does not dictate the order that an application gets called when the application is a member of an application composition chain that matches rules to process a new message coming in.

The starting weight for applications and their modules is specified in the deployment.xml file. The order in which modules pickup requests on composition is evaluated by applications weight first and then modules weight. The following steps can be completed in any order to specify applications weight or modules weight from the administrative console.

1. To specify the applications (EARs) weight, expand **Enterprise Applications** → *applicationName* → **Startup Behavior** and set the startup order.
2. To specify the modules (WARs) weight, expand **Enterprise Applications** → *applicationName* → **Manage Modules** and set the starting weight.
3. Restart the changed applications.

Example

Note:

sip.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE sip-app
PUBLIC "-//Java Community Process//DTD SIP Application 1.0//EN"
"http://www.jcp.org/dtd/sip-app_1_0.dtd">
<sip-app>
  <display-name>SIPSampleProxy</display-name>

  <servlet>
    <servlet-name>SIPSampleProxy</servlet-name>
    <servlet-class>sipes.test.container.proxy.SIPSampleProxy</servlet-class>
    <load-on-startup>1</load-on-startup>
  </servlet>

  <servlet-mapping>
    <servlet-name>SIPSampleProxy</servlet-name>
    <pattern>
      <equal>
        <var>request.uri.user</var>
        <value>SIPSampleProxy</value>
      </equal>
    </pattern>
  </servlet-mapping>

  <proxy-config>
    <sequential-search-timeout>1000</sequential-search-timeout>
  </proxy-config>
  <session-config>
    <session-timeout>12</session-timeout>
  </session-config>
</sip-app>
```

web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE web-app PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN" "http://java.sun.com/dtd/web-app_2_3.dtd">
<web-app id="WebApp">
  <display-name>SIPSampleProxy</display-name>
  <servlet>
    <servlet-name>SIPSampleProxy</servlet-name>
    <display-name>SIPSampleProxy</display-name>
    <servlet-class>sipes.test.container.proxy.SIPSampleProxy</servlet-class>
  </servlet>
  <servlet-mapping>
    <servlet-name>SIPSampleProxy</servlet-name>
    <url-pattern>/SIPSampleProxy</url-pattern>
  </servlet-mapping>
  <welcome-file-list>
    <welcome-file>index.html</welcome-file>
    <welcome-file>index.htm</welcome-file>
    <welcome-file>index.jsp</welcome-file>
    <welcome-file>default.html</welcome-file>
    <welcome-file>default.htm</welcome-file>
    <welcome-file>default.jsp</welcome-file>
  </welcome-file-list>
</web-app>
```


Note:

The following example is for a standalone server.

```
deployment.xml

<?xml version="1.0" encoding="UTF-8" ?>
- <appdeployment:Deployment xmi:version="2.0" xmlns:xmi="http://www.omg.org/XMI"
xmlns:appdeployment="http://www.ibm.com/websphere/appserver/schemas/5.0/appdeployment.xmi"
xmi:id="Deployment_1137951186883">
- <deployedObject xmi:type="appdeployment:ApplicationDeployment" xmi:id="ApplicationDeployment_1137951186883"
deploymentId="0" startingWeight="1" binariesURL="$(APP_INSTALL_ROOT)/OrangeNode08Cell/SipContainerTestSuite.ear"
useMetadataFromBinaries="false" enableDistribution="true" createMBeansForResources="true" reloadEnabled="false"
appContextIDForSecurity="href:OrangeNode08Cell/SipContainerTestSuite"
filePermission=".*\\.dll=755#.*\\.so=755#.*\\.a=755#.*\\.sl=755" allowDispatchRemoteInclude="false"
allowServiceRemoteInclude="false">
<targetMappings xmi:id="DeploymentTargetMapping_1137951186883" enable="true" target="ServerTarget_1137951186883" />
<classloader xmi:id="Classloader_1137951186883" mode="PARENT_FIRST" />
- <modules xmi:type="appdeployment:WebModuleDeployment" xmi:id="WebModuleDeployment_1137951186883"
deploymentId="1" startingWeight="10000" uri="sipunit.war">
<targetMappings xmi:id="DeploymentTargetMapping_1137951186884" target="ServerTarget_1137951186883" />
<classloader xmi:id="Classloader_1137951186884" /> </modules>
<properties xmi:id="Property_1137951186883" name="validateinstall" value="warn" /> </deployedObject>
<deploymentTargets xmi:type="appdeployment:ServerTarget" xmi:id="ServerTarget_1137951186883"
name="server1" nodeName="OrangeNode10" /> </appdeployment:Deployment>
```

SIP servlets

This topic describes SIP servlets.

The SIP Servlet 1.0 specification (JSR 116) is standardized through Java Specification Request (JSR) 116. The idea behind the specification is to provide a Java application programming interface (API) similar to HTTP servlets, which provides an easy-to-use SIP programming model. Like the popular HTTP servlet programming model, some flexibility is limited to optimize ease-of-use and time-to-value.

However, the SIP Servlet API is different in many ways from HTTP servlets because the protocol is so different. While SIP is a request-response protocol, there is not necessarily only one response to every one request. This complexity and a need for a high performing solution meant that it was easier to make the SIP servlets natively asynchronous. Also, unlike HTTP servlets, the programming model for SIP servlets sought to make client requests easy to create alongside the other logic being written because many applications act as a client or proxy to other servers or proxies.

SipServlet requests

Like HTTP servlets, each SIP servlet extends a base `javax.servlet.sip.SipServlet` class. All messages come in through the service method, which you can extend. However, because there is not a one-to-one mapping of requests to responses in SIP, the suggested practice is to extend the `doRequest` or `doResponse` methods instead. When extending the `doRequest` or `doResponse` methods, it is important to call the extended method for the processing to complete.

Each request method, which the specification must support, has a `doxxx` method just like HTTP. In HTTP, methods such as `doGet` and `doPost` exist for GET and POST requests. In SIP, `doInvite`, `doAck`, `doOptions`, `doBye`, `doCancel`, `doRegister`, `doSubscribe`, `doNotify`, `doMessage`, `doInfo`, and `doPrack` methods exist for each SIP request method.

Unlike an HTTP servlet, SIP servlets have methods for each of the response types that are supported. So, SIP servlets include the `doProvisionalResponse`, `doSuccessResponse`, `doRedirectResponse`, and `doErrorResponse` responses. Specifically, the provisional responses (1xx responses) are used to indicate status, the success responses (2xx responses) are used to indicate a successful completion of the transaction, the redirect responses (3xx responses) are used to redirect the client to a moved resource or entity, and the error responses (4xx, 5xx, and 6xx responses) are used to indicate a failure or a specific error condition. These types of response messages are similar to HTTP, but because the SIP Servlet programming model includes a client programming model, it is necessary to have responses handled programmatically as well.

Clarifications of JSR 116

JSR 289 has made some clarifications to JSR 116, as follows:

- JSR 289 Section 4.1.3: Contact Header Field
- JSR 289 Section 5.2: Implicit Transaction State
- JSR 289 Section 5.8: Accessibility of SIP Servlet Messages

SIP SipServletRequest and SipServletResponse classes

The SipServletRequest and SipServletResponse classes are similar to the HttpServletRequest and HttpServletResponse classes.

SipServletRequest and SipServletResponse classes

Each class gives you the capability to access the headers in the SIP message and manipulate them. Because of the asynchronous nature of the requests and responses, this class is also the place to create new responses for the requests. When you extend the doInvite method, only the SipServletRequest class is passed to the method. To send a response to the client, you must call the createResponse method on the Request object to create a response. For example:

```
protected void doInvite(SipServletRequest req) throws
    javax.servlet.ServletException, java.io.IOException {

    //send back a provisional Trying response
    SipServletResponse resp = req.createResponse(100);
    resp.send();
}
```

Because of their asynchronous nature, SIP servlets can seem complicated. However, something as simple as the previous code sample sends a response to a client.

Here is a more complex example of a SIP servlet. With the following method included in a SIP servlet, the servlet blocks all of the calls that do not come from the example.com domain.

```
protected void doInvite(SipServletRequest req) throws
    javax.servlet.ServletException, java.io.IOException {

    //check to make sure that the URI is a SIP URI
    if (req.getFrom().getURI().isSipURI()){
        SipURI uri = (SipURI)req.getFrom().getURI();
        if (!uri.getHost().equals("example.com")) {
            //send forbidden response for calls outside domain
            req.createResponse(SipServletResponse.SC_FORBIDDEN).send();
            return;
        }
    }
    //proxy all other requests on to their original destination
    req.getProxy().proxyTo(req.getRequestURI());
}
```

SIP SipSession and SipApplicationSession classes

Possibly the most complex portions of the SIP Servlet 1.0 specification are the SipSession and SipApplicationSession classes.

SIP SipSession and SipApplicationSession classes

Both of these classes have some useful purposes and can act as the primary place to store data in applications that are designed for distributed or highly available environments.

The SipSession class is the best representative of a specific point-to-point communication between two entities and is the closest to the HttpSession object. Because historically no proxying or forking existed for the HTTP request in HTTP servlets, the need for something higher than a single point-to-point session did

not exist. However, even HTTP users can see the growing need for this type of function since portlets began essentially forking HTTP requests. The SIP users expect the proxying and forking activities that require multiple layers of SIP session management. The SipSession class is the lowest point-to-point layer.

The SipApplicationSession class represents the higher layer of SIP session management. One SipApplicationSession class can own one or more SipSession objects. However, each SipSession class can be related to one SipSession object only. The SipApplicationSession class also supports the attachment of any number of other protocol sessions. Currently, only HTTP sessions are supported by any implementations. The SipApplicationSession class has a getSessions method, which takes the requested protocol type as an argument.

You might find it useful for many applications to combine HTTP and SIP. For example, you might use this approach to tie together HTTP and SIP sessions to monitor a phone call or to start a phone call through a rich HTTP graphical user interface.

Example: SIP servlet simple proxy

This is a servlet example of a simple proxy.

Simple proxy

```
import java.io.IOException;

import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.sip.SipFactory;
import javax.sip.SipRequest;
import javax.sip.SipResponse;
import javax.sip.SipSession;
import javax.sip.SipURI;
import javax.sip.URI;

public class SimpleProxy extends HttpServlet implements Servlet {

    final static private String SHUTDOWN_KEY = new String("shutdown");
    final static private String STATE_KEY = new String("state");
    final static private int INVITE_RECEIVED = 1;

    /* (non-Java-doc)
     * @see javax.sip.SipServlet#SipServlet()
     */
    public SimpleProxy() {
        super();
    }

    /* (non-Javadoc)
     * @see javax.sip.SipServlet#doInvite(javax.sip.SipServletRequest)
     */
    protected void doInvite(SipServletRequest request) throws ServletException,
        IOException {

        //log("SimpleProxy: doInvite: TOP");

        try {
            if (request.isInitial() == true)
            {
                // This should cause the sip session to be created. This sample only uses the session on receiving
                // a BYE but the Tivoli performance viewer can be used to track the creation of calls by viewing the
                // active session count.
                Integer state = new Integer(INVITE_RECEIVED);
                SipSession session = request.getSession();
            }
        }
    }
}
```

```

session.setAttribute(STATE_KEY, state);
    log("SimpleProxy: doInvite: setting attribute");

Proxy proxy = request.getProxy();

SipFactory sipFactory = (SipFactory) getServletContext().getAttribute(SIP_FACTORY);
    if (sipFactory == null) {
        throw new ServletException("No SipFactory in context");
    }

String callingNumber = request.getTo().toString();
if (callingNumber != null)
{
    String destStr = format_lookup(callingNumber);
    URI dest = sipFactory.createURI(destStr);

    //log("SimpleProxy: doInvite: Proxying to dest URI = " + dest.toString());

    if (((SipURI)request.getRequestURI()).getTransportParam() != null)
        ((SipURI)dest).setTransportParam(((SipURI)request.getRequestURI()).getTransportParam());

    proxy.setRecordRoute(true);
    proxy.proxyTo(dest);
}
else
{
    //log("SimpleProxy: doInvite: Request is invalid. Did not contain a To: field.");
    SipServletResponse sipresponse = request.createResponse(400);
    sipresponse.send();
}
else
{
    //log("SimpleProxy: doInvite: target refresh, let container handle invite");
    super.doInvite(request);
}
}
catch (Exception e){
    e.printStackTrace();
}
}

/* (non-Javadoc)
 * @see javax.servlet.sip.SipServlet#doResponse(javax.servlet.sip.SipServletResponse)
 */
protected void doResponse(SipServletResponse response) throws ServletException,
    IOException {
    super.doResponse(response);

    // Example of using the session object to store session state.
    SipSession session = response.getSession();
    if (session.getAttribute(SHUTDOWN_KEY) != null)
    {
        //log("SimpleProxy: doResponse: invalidating session");
        session.invalidate();
    }
}

/* (non-Javadoc)
 * @see javax.servlet.sip.SipServlet#doBye(javax.servlet.sip.SipServletRequest)
 */
protected void doBye(SipServletRequest request) throws ServletException,
    IOException {

    SipSession session = request.getSession();
    session.setAttribute(SHUTDOWN_KEY, new Boolean(true));

```

```

        //log("SimpleProxy: doBye: invalidate session when responses is received.");
        super.doBye(request);
    }

    protected String format_lookup(String toFormat){
        int start_index = toFormat.indexOf('<') + 1;
        int end_index = toFormat.indexOf('>');

        if(start_index == 0){
            //don't worry about it
        }
        if(end_index == -1){
            end_index = toFormat.length();
        }

        return toFormat.substring(start_index, end_index);
    }
}

```

Example: SIP servlet SendOnServlet class

The SendOnServlet class is a simple SIP servlet that would perform the basic function of being called on each INVITE and sending the request on from there.

SendOnServlet class

Function could easily be inserted to log this invite request or reject the INVITE based on some specific criteria.

```

package com.example;
import java.io.IOException;
import javax.servlet.sip.*;
import java.servlet.ServletException;
public class SendOnServlet extends SipServlet {
    public void doInvite(SipServletRequest req)
        throws ServletException, java.io.IOException {
        //send on the request
        req.getProxy().proxyTo(req.getRequestURI);
    }
}

```

The doInvite method could be altered to do something such as reject the invite for some specific criteria simply. In the example doInvite method below, all requests from domains outside of example.com will be rejected with a Forbidden response.

```

    public void doInvite(SipServletRequest req)
    throws ServletException, java.io.IOException {
    if (req.getFrom().getURI().isSipURI()){
        SipURI uri = (SipURI)req.getFrom.getURI();
        if (!uri.getHost().equals("example.com")) {
            //send forbidden response for calls outside domain
            req.createResponse(SipServletResponse.SC_FORBIDDEN, "Calls outside example.com not accepted").send();
            return;
        }
    }
    //proxy all other requests on to their original destination
    req.getProxy().proxyTo(req.getRequestURI());
}

```

SendOnServlet deployment descriptor:

```

<sip-app>
  <display-name>Send-on Servlet</display-name>
  <servlet>
    <servlet-name>SendOnServlet</servlet-name>
    <servlet-class>com.example.SendOnServlet</servlet-class>
  </servlet>

```

```

    <servlet-mapping>
      <servlet-name>SendOnServlet</servlet-name>
      <pattern>
        <equal>
          <var>request.method</var>
          <value>INVITE</value>
        </equal>
      </pattern>
    </servlet-mapping>
  </sip-app>

```

Example: SIP servlet Proxy servlet class

Proxy servlet class

After the initial INVITE, this application will be called on every subsequent SIP message. For each Request and Response, this class will simply print out the action and who it is to or from.

```

package com.example;
import java.io.IOException;
import javax.servlet.sip.*;
import java.servlet.ServletException;
public class ProxyServlet extends SipServlet {
    public void doInvite(SipServletRequest req)
        throws ServletException, java.io.IOException {
        //get the Proxy
        Proxy p=req.getProxy();
        //turn on supervised mode so that all events come through us
        //The default on this is true but it is set to emphasize the function.
        p.setSupervised(true);
        //set record route so we see the ACK, BYE, and OK
        p.setRecordRoute(true);
        //proxy on the request
        p.proxyTo(req.getRequestURI());
    }
    public void doRequest(SipServletRequest req)
        throws ServletException, java.io.IOException {
        System.out.println(req.getMethod()+" Request from "+req.getFrom().getDisplayName());
        super.doRequest(req);
    }
    public void doResponse(SipServletResponse resp)
        throws ServletException, java.io.IOException {
        System.out.println(resp.getReasonPhrase()+" Response from "+resp.getTo().getDisplayName());
        super.doResponse(resp);
    }
}

```

Proxy deployment descriptor

```

<sip-app>
  <display-name>ProxyServlet</display-name>
  <servlet>
    <servlet-name>ProxyServlet</servlet-name>
    <servlet-class>com.example.ProxyServlet</servlet-class>
  </servlet>

  <servlet-mapping>
    <servlet-name>ProxyServlet</servlet-name>
    <pattern>
      <equal>
        <var>request.method</var>
        <value>INVITE</value>
      </equal>
    </pattern>
  </servlet-mapping>
</sip-app>

```

Deploying SIP applications

Use the administrative console to customize your Session Initiation Protocol (SIP) application installation

About this task

When you deploy a Session Initiation Protocol (SIP) application, you can perform various tasks such as installing, starting, stopping, upgrading, and uninstalling the application.

SIP applications are installed as Java Platform, Enterprise Edition (Java EE) applications. You can deploy a SIP application from a graphical interface or from a command line.

Deploying SIP applications through the console

You can deploy a Session Initiation Protocol (SIP) application through the administrative console.

Before you begin

SIP applications are deployed as Java 2 Platform Enterprise Edition (J2EE) applications. In order to process requests, a virtual host must be defined when deploying the SIP application. If there is no virtual host defined for the configured SIP container listen port, the installed application will be inaccessible.

1. Open the administrative console.
In a browser, go to URL `http://hostname:9090/admin`, where *hostname* is the name of the host computer. Enter the appropriate login information, and click **OK**.
2. In the left frame click **Applications** → **Install New Application**.
3. Browse and select a SAR file. Specify the context root, beginning with a slash (/), in the **Context Root** field. For example, if your application is named `ThisApplication`, type `/ThisApplication`.
4. Click **Next** (under the **Context Root** field not beside the WebSphere Status title). If the SAR file has been assembled correctly, the screen will still have the title “Preparing for the application installation”, but the content will change. If an error message appears, check the contents of the SAR file; in particular, verify the `web.xml` file contents, and try to reload the SAR file.
5. Click **Next**. If you see a screen indicating “Application Security Warnings”, click **Continue**.
6. The **Install New Application** screen should appear with “Step 1: Select application options” highlighted. Select the options you need and click **Next**.
7. “Step 2: Map modules to servers” should appear highlighted now. You can choose the cluster or server where you want to install the application’s modules.
 - If you are installing the application in a stand-alone system, click **Next**.
 - If you are installing the application in a clustered system, select **WebSphere:cell=cellname,cluster=cluster_name** in the **Clusters and Servers** field, select the check box beside the Web module that you want to install, and click **Apply** and **Next**.
8. Now “Step 3: Map virtual hosts for Web modules” should appear highlighted. To the right of the application name there should be a drop-down labeled **Virtual Host**.
 - If you are installing the application in a standalone system, set the value of the drop-down to **default_host**, and click **Next**.
 - If you are installing the application in a clustered system, set the value of the drop-down to the name of the virtual host that was chosen during setup, and click **Next**.

Note: You must define a virtual host for your configured SIP container listen port or else you will not be able to access the application.

9. You should now see “Step 4: Summary” highlighted. In the right panel you will see a **Summary of installation options** table that details your selected options and their values. If you need to change an option, click **Previous** to return to the section where you can make your change. Click **Finish** to

install the application with your settings. The screen should display, Application *appname_sar* installed successfully, where *appname* is the name of the application.

10. Click the **Save to Master Configuration** link. A Save to Master Configuration window appears.
11. In the Save to Master Configuration window, click **Save**. The application has now been saved in the current configuration.
12. To confirm that the installation succeeded, in the left frame click **Applications** → **Enterprise Applications**. The newly installed application should appear in the list of installed applications as *appname_sar*.
13. To start the application so that it can service SIP requests, check the box beside *appname_sar*, and click **Start**. You might also want to look at the logs for a successful startup message.

Results

The application can service SIP requests now.

Deploying SIP applications through scripting

You can deploy a Session Initiation Protocol (SIP) application not only from the GUI but also from the command line.

- Launch a scripting client. For more information, see AdminApp object for scripted administration.
- List applications.
- Install standalone archive files. For more information about installation, see Installation options for the AdminApp object.
- Edit application configurations.
- Uninstall applications.

Securing SIP applications

You can apply digest authentication and Trust Association Interceptor (TAI) for a SIP application by applying Lightweight Directory Access Protocol (LDAP) security to the application.

Before you begin

Before you can apply security, you must first deploy an application that has been developed to support security (with the web.xml file configured for security) and roles. The following software must also be installed:

1. Install a supported LDAP server. For a list of supported LDAP servers, see the IBM Web site for WebSphere Application Server supported hardware, software, and APIs.
2. Set up and activate Lightweight Third Party Authentication. For more information, see the Configuring the Lightweight Third Party Authentication mechanism topic.

About this task

To apply LDAP security to a SIP application, click **Applications** → **Enterprise Applications** → ***applicationName*** and complete the following steps:

1. Click **Detail Properties** → **Security role to user/group mapping**.
2. Check **All Authenticated**.
3. Save all changes.
4. Restart the server.

Configuring security for the SIP container

This section provides instructions specific to security for the SIP container.

Before you begin

Before you can configure security for your SIP container, you will need to:

1. Set up and activate Lightweight Third Party Authentication. For more information, see the Lightweight Third Party Authentication section.
2. Install a supported LDAP server.

You may also need to:

- Adjust key group settings. Refer to Lightweight Third Party Authentication key sets and key set groups for LTPA key information.
- Establish and configure Trust Association Interceptor (TAI) settings. Refer to Trust association interceptor settings.

About this task

You must know the name of the key set group and the management scope where the key set group is defined in order to activate and secure LTPA with keys. Refer to Activating Lightweight Third Party Authentication key versions for the setup and activation procedures.

To configure security based on the Lightweight Directory Access Protocol (LDAP), you can configure digest authentication for your supported LDAP server.

- To configure digest authentication and TAI on WebSphere Application Server for Tivoli, select “Configuring digest authentication and TAI for SIP.”
- To configure digest authentication on WebSphere Application Server for Oracle Internet Directory, select “Configuring digest authentication for Oracle Internet Directory” on page 409.

To define an LDAP connection between WebSphere Application Server and LDAP, use the security wizard. It can also be defined by selecting it from available realms and defining the proper connection properties to connect LDAP.

To set up a TAI, you must specify the trust information for any reverse security proxy servers. See Trust association interceptor settings to configure TAI settings.

Configuring digest authentication and TAI for SIP

You can configure digest authentication and Trust Association Interceptor (TAI) for the Session Initiation Protocol (SIP).

Before you begin

Before you can configure digest authentication and TAI, you must either install a supported LDAP server, or configure digest TAI to work without LDAP.

To configure digest TAI to work without LDAP, complete these steps:

1. Create a class that implements the interface: `com.ibm.ws.sip.security.digest.DigestPasswordServer`.
2. In the administrative console, click **Global security > Digest authentication > Custom Properties > New**, and enter `DigestPasswordServerClass` in the **Name** field, and the name of the class that you created in the **Value** field.
3. Ensure that all users that implement the impl class are declared in the user registry configured for WebSphere Application Server security.

LDAP servers automatically provide password support. Unless you enable the LDAP server to use hashed values, the LDAP server stores user passwords and then the request processing component uses these passwords to validate a request. Because this method of authentication exposes user passwords to potential internet theft, you should enable the use of hashed credentials to authenticate a request.

When you enable the use of hashed credentials, the LDAP server stores a hash value for the user, password and realm information. The SIP container then requests this hash value from the LDAP server instead of asking for a user password. This methodology protects the passwords even if the hash data is compromised through internet theft. However, this methodology has the following limitations:

- The LDAP attribute must store a byte value or a string value. Other attribute types are not supported.
- All of your applications must share the same realm, or you must define a different attribute for each realm.
- The hash function might be different than MD5. In this situation, the SIP container sends a algorithm that is different from the calculated value for the attribute. When this situation occurs, user authentication might fail even if the user provided the proper credentials.

To enable the LDAP server to use hashed credentials, you must define the following two custom properties:

- `hashedCredentials=value`, where *value* is the name of LDAP attribute which stores the hash value for user, password, realm
- `hashedCredentialsRealms=value`, where *value* is the realm, on which the hashed value is calculated.

About this task

The SIP container supports digest authentication. When this type of authentication is used, the client does not send a clear text password to the server. Instead, SIP authenticates each request using user data from LDAP. Typically, a component that uses LDAP for authentication, verifies that the response that the client provides equals the response that the component calculates using LDAP data, the component authorizes the request. However,

Howto define: One should d

Complete the following procedure to configure digest authentication and TAI for the SIP container.

1. In the administrative console, click **Security** → **Global security** → **Authentication mechanisms** to verify that **Lightweight Third Party Authentication (LTPA)** is configured for use on your server.
In the **Configuration** tab on the **Authentication mechanisms and expiration** page you should see the **Password** field already filled in.
2. Click **Security** → **Global security**.
 - a. Under **Authentication**, expand **Web security** and click **Trust association**.
 - b. On the **Configuration** tab, in the General properties section, verify that the **Enable trust association** box is selected, and then click **Apply**.
3. On the **Interceptors** page of the administration console look for `com.ibm.ws.sip.security.digest.DigestTAI` in the **Interceptor class name** list.
 - a. If this class name is not present, click **New** to open the Configuration tab and enter `com.ibm.ws.sip.security.digest.DigestTAI` in the **Interceptor class name** field, and then click **Apply**.
 - b. If this interceptor class is present, click `com.ibm.ws.sip.security.digest.DigestTAI` → **Custom Properties** to set up a realm in digest authentication.
 - c. Click **OK**.
4. Click **Security** → **Global security** → **Authentication mechanisms and expiration**, and then click the **Configuration** tab.
 - a. In the **Key generation** section, click **Generate keys**. You do not have to import or export the key.
 - b. In the Cross-cell Single Sign-on section, specify values in the **Password** fields and the **Internal server ID** field.
 - c. Click **OK**.
5. Click **Security** → **Global security**.

- a. If the box **Use Java 2 security to restrict application access to local resources** is selected, click to deselect it.
 - b. In the **User account repository** section of the page, select your LDAP registry from the **Available realm definitions** list.
 - c. Click **Set as current**, and then click **Apply**.
6. Save all changes.
 7. Restart the server.
 8. Verify that the following message appears in the SystemOut.log file after the server restarts:
SECJ0121I: Trust Association Init class com.ibm.ws.sip.security.digest.DigestTAI loaded successfully

If this message does not appear in the log file, digest authentication is not active

Configuring digest authentication for Oracle Internet Directory

You can configure digest authentication for Oracle Internet Directory, an implementation of the Lightweight Directory Access Protocol (LDAP) that uses the Oracle database as a repository for directory entries.

Before you begin

To configure digest authentication for Oracle Internet Directory, you will need to:

- Install Oracle Internet Directory version 9.0.2.
- Set up and activate Lightweight Third Party Authentication. For more information, see the Lightweight Third Party Authentication section.

About this task

Complete the following procedure to configure digest authentication for Oracle Internet Directory on WebSphere Application Server:

1. To set up digest authentication, verify that **Lightweight Third Party Authentication (LTPA)** is configured for use on your server by selecting **Security** → **Global security** → **Authentication mechanisms**. In the **Configuration** tab on the **Authentication mechanisms and expiration** page you should see the **Password** field already filled in.
2. In the administrative console, click **Security** → **Global security**.
 - a. Under **Authentication**, expand **Web security** and click on **Trust association**.
 - b. On the **Configuration** tab, under **General properties**, make sure the **Enable trust association** box is checked. Then click **Apply**.
3. On the **Interceptors** page of the administration console look for `com.ibm.ws.sip.security.digest.DigestTAI` in the **Interceptor class name** list:
 - a. If this class name is not present, click **New** to open the Configuration tab and enter `com.ibm.ws.sip.security.digest.DigestTAI` in the **Interceptor class name** field and click **Apply**. Then proceed to the following steps.
 - b. If this interceptor class is present, you may set up custom properties for it. To do this, click **com.ibm.ws.sip.security.digest.DigestTAI** → **Custom Properties**:
 - c. Click **OK**.
4. Navigate through **Security** → **Global security** → **Authentication mechanisms and expiration** to the **Configuration** tab.
 - a. In the **Key generation** section, click **Generate Keys**. (No import or export of the key is necessary.)
 - b. Under the Cross-cell single sign-on section fill in the **Password** fields.
 - c. Fill in the **Internal server ID** field.
 - d. Click **OK**.
5. Click to **Security** → **Global security**.

- a. If the box **Use Java 2 security to restrict application access to local resources** is checked, then Java 2 security is enabled. Click the box if you want to disable Java 2 security.
 - b. In the **User account repository** section of the page, select your LDAP registry from the **Available realm definitions** drop-down box.
 - c. Click **Set as current** and then click **Apply**.
6. Save all changes.
 7. Restart the server.
 8. Be sure you see the following message appear in the SystemOut.log after the server has restarted:


```
SECJ0121I: Trust Association Init class
com.ibm.ws.sip.security.digest.DigestTAI loaded successfully
```

If this message does not appear in the log, digest authentication has not been activated.

Tracing a SIP container

You can trace a Session Initiation Protocol (SIP) container, starting either immediately or after the next server startup. This tracing writes a record of SIP events to a log file.

About this task

Follow these steps to start tracing a SIP container:

1. Open the administrative console. For more information about the console, read the Using the administrative console chapter of the *Administering applications and their environment* PDF book.
2. In the administrative console, click **Troubleshooting** → **Logs and trace**.
3. Select the name of the server for the SIP container.
4. From the General Properties section, click **Diagnostic Trace Service**.
5. Under the Additional Properties section, click **Change Log Detail Levels**
6. Select one of the following options:

Option	Description
Configuration	To start tracing after the next server startup
Runtime	To start tracing immediately

7. Replace the content of the trace specification with the following code: `com.ibm.ws.sip.*=all=enabled`.

Note: If you want monitor only specific pieces of SIP containers, expand the **com.ibm.ws.sip** section and select the individual items you wish to trace.

8. Make sure that the **Enable trace with following specification** check box is checked.
9. Click **Apply** → **Save**.

What to do next

When the changes take effect (refer to step 6 above), SIP-level tracing messages appear in `WASProductDir/logs/serverName/trace.log`, where `WASProductDir` is the fully qualified path name of the directory in which the product is installed and `serverName` is the name of the specific instance of the application server that is running the SIP container to be traced. These messages include application load events as well as SIP request and response parsing and SIP servlet invocation.

Upgrading SIP applications

Follow these steps to upgrade SIP applications.

About this task

Use the following application upgrade procedure to install a new version of a previously installed application without any outage.

1. Install version 1 of the application in cluster 1.
2. Create cluster 2.
3. Set cluster 1 as the default cluster.
4. Install version 2 of the application in cluster 2.
5. Change the default cluster at proxy to cluster 2.
6. To shut down Version 1, monitor PMI and watch for the application session for this application to come to zero. When it is at zero or a level you consider okay to lose, stop the application.

Results

New calls will be routed to cluster 2. Old calls will continue to be routed to cluster 1.

Replicating SIP sessions

This topic provides an overview of when replication of session and dialog state can take place during typical Session Initiation Protocol (SIP) processing. It also describes the steps that you have to complete to set up a replication domain for SIP sessions.

About this task

The SIP container typically uses the Data Replication Service (DRS) to replicate all state information. Because DRS does not provide a way to confirm when data replication has completed, the only thing that can be quantified is when a piece of state information is queued for replication. Within this topic, references to replication of data only means that the data has been queued for replication.

The SIP container replicates several different types of information. This data falls into two general categories:

- Internal SIP container state information associated with the dialog.
- Application state information associated with the various session objects.

Each of these categories includes a number of different data types, that are described below. Each data object is treated independently. Therefore, a change to an application session object, that causes a replication, does not result in the replication of any internal state information.

Replication of internal state information

Internal state information can be defined as anything related to the state of a dialog being handled by the container. It includes things like cseq, dialog state (initial, early, confirmed, terminated), session expiration, local, remote party, etc. Internal state information is only replicated due to the existence of a dialog. Therefore, no internal SIP-related data will be replicated until the dialog with which it is associated is established. The types of SIP requests that can cause the creation of a dialog include:

- INVITE
- SUBSCRIBE
- REFER

Replication of internal state happens at well-defined, predictable points in the call flow. For example, a dialog is only established at the container when a 2xx response or a 1xx response with a “To” tag is received or sent due to one of the method types listed above. Events that can trigger an internal state replication include:

- Creation of a new SIP dialog
- Expiration of a session due to a session timeout
- The sending of a final response to a UAC
- Creation of an encoded URI
- Handling of any message that results in a change of the internal dialog state

It is important to note that transaction state is NOT replicated in the WebSphere Application Server 6.1 version of the SIP container, only dialog state. Not replicating transaction state reduces the load on all the servers in the replication domain, but can cause problems in failures that happen in the middle of a transaction (for example, loss of some dialog-related SIP messages).

An important difference between a B2BUA and a proxy application is the number of session objects created and replicated. In both cases only a single application session is created, but for the B2BUA, two session objects are created—one for the inbound leg and one for the outbound leg. For a proxy application, only a single session object is created.

Replication of application state information

Application state information is treated differently from internal dialog state information because it does not rely on the existence of a dialog to be replicated. Application state refers to any data that is being maintained by the application through the use of JSR 116 data constructs. This includes:

- `javax.servlet.sip.SipApplicationSession`
- `javax.servlet.sip.SipSession`
- `javax.servlet.sip.ServletTimer`
- Any attribute set on the `SipSession` or the `SipApplicationSession`

Replication of internal state happens at well-defined, predictable points in the call flow, while replication of application state is less predictable because it is generally dependent on when an application creates, invalidates or modifies the session data and timers through JSR 116 APIs. This could be due to the processing of an inbound message, or to the expiration of a SIP timer. All of the following can cause the replication of application-related session data:

- Creation of an application session object
- Creation of a SIP session object
- Creation of a SIP session timer
- Modification of a session object through `setAttribute` or `removeAttribute`
- Invalidation of a SIP session object
- Expiration of a session timer
- Application code sending out a request *

*Causes replication of the `SipSession` and `SipApplicationSession` in order to synchronize the "last access timestamp" with the peer container(s) in the cluster. This is for integrity of future calls to `SipSession.getLastAccessedTime()` and `SipApplicationSession.getLastAccessedTime()`

Replication can occur for requests that do not establish a dialog if an application calls `request.getApplicationSession(true)` and if `addTimer()` and/or `addAttribute()` are called on the resulting application session object. This is needed so that a listener can be called when the timer expires.

SIP failover and replication setup considerations

A SIP cluster that requires replication and failover can consist of many replication domains, each of which contain a set of SIP containers. There is no limit set on the number of containers in a cluster. For performance reasons, each replication domain should contain 2 containers only.

The replication domain should be set to the Entire Domain, which means state is replicated to all containers in the replication domain. The replication mode should be Both client and server.

The distributed session for a container needs to be set to Memory-to-memory replication. Any applications that require session replication must include the `<distributed />` tag in the web.xml and sip.xml files. Both SIP and HTTP will utilize the same replication topology

SIP session replication topology

- Each member replicates all state data to every peer in its replication domain.
- Each replication domain should ideally contain two servers.
- When a failure occurs, the core group coordinator tells the remaining core group members which replicated sessions to activate. These replicated sessions then become part of their active sessions.

Complete the following steps to set up a replication domain for SIP sessions

1. In the administrative console, click **Environment > Replication domains > New**
2. Click **Number of replicas**, and then select **Entire domain**.
3. In the Container settings section, click **Session management**.
4. In the Additional Properties section, click **Distributed environment settings**, and then click **Memory-to-memory replication**.
5. Set **Replication mode** to **Both client and server**.
6. Save your changes.

Results

Memory-to-memory replication is enabled for SIP sessions.

Troubleshooting SIP applications

Use this page to troubleshoot SIP applications.

About this task

SIP container troubleshooting basics

- The Average CPU usage of the system should go no higher than 60%-70%.
- The container should use no more than 70% of the allocated VM heap size. Be sure that the system has enough physical memory to accommodate the VM heap size. Call loads and session timeouts will have a big affect on heap usage.
- The maximum garbage collection (GC) time of the VM on which the container is running should not exceed 500 ms and the average should be less than 400 ms. Verbose GC can be used to measure this and the PMI viewer can be used to view GC times, heap usage, active sessions, etc., in graphical form.

Initial troubleshooting checklist:

- Check the listening ports in the configuration.
- Use `netstat -an` to see listening ports.
- Check to see if virtual hosts are defined
- Check to see if host aliases are defined.
- Is an application installed? Is it started?
- For a proxy configuration: Is a default cluster configured? If proxy and server are on the machine, is there a port conflict?

Results

SIP container symptoms and solutions

If the problem is not resolved, check for specific symptoms.

- **Symptom:** Lots of retransmissions, CPU periodically drops to zero.
Solution: This is typically a DNS issue caused by Reverse DNS lookups and can be confirmed using a tool like Ethereal. If you do a network capture and send lots of DNS queries that contain an IP address and get back a host name in the response, this could be the problem. Make sure that nsd is running if you are on HP or other platforms that require name service caching. (Windows does not require this.) Another solution is to add host names to the /etc/hosts file.
- **Symptom:** Lots of retransmissions, CPU periodically spikes to 100%.
Solution: This is typically due to garbage collection and can be verified by turning on verbose GC (accessible on the admin console) and looking at the length of the GC cycles. The solution here is to enable Generational Garbage Collection by setting the JVM optional args to -Xgcpolicy:gencon.
- **Symptom:** Lots of retransmissions, CPU spikes to 100% for long periods of time and Generational Garbage Collection is enabled.
Solution: This is typically due to SIP session objects either not being invalidated or not timing out over a long period of time. One solution is to set the session timeout value in the sip.xml of the application to a smaller value. The most efficient way to handle this is for the application to call invalidate on the session when the dialog completes (i.e. after receiving a BYE). The following entry in the SystemOut.log file will indicate the session timeout value is for each application installed on the container:

```
SipXMLParser 3 SipXMLParser getAppSessionTTL Setting Expiration time: 7 Minutes, For App: TCK back-to-back user agent"
```
- **Symptom:** Lots of "480 Service Not Available" messages received from the container when sending new INVITE messages to the SIP container. You will also likely see the following message show up in the SystemOut.log when the server is in this state: "LoadManager E LoadManager warn.server.oveloaded".
Solution: This is typically due to one of the SIP container configurable metrics being exceeded. This includes the "Maximum Application Sessions" value and the "Maximum messages per averaging period" value. The solution is to adjust these values higher.
- **Symptom:** Lots of resends and calls are not completing accompanied by OutOfMemory exceptions in the SystemErr.log.
Solution: This usually means that the VM heap size associated with your container is not large enough and should be adjusted upwards. You can adjust this value from the admin console.
- **Symptom:** You receive a "503 Service Unavailable" when sending a SIP request to a SIP proxy.
Solution: This usually means there is no default cluster (or cluster routing rule that matches the message) set up at the proxy. This can also happen when the SIP proxy is configured well but the backend SIP containers are stopped or have crashed.
- **Symptom:** You receive a "404 Not Found" when sending a SIP request to a SIP proxy.
Solution: This usually means there is no virtual host set up for the containers that reside in the default cluster. It could also mean that the servers in the proxy's default cluster do not contain a SIP application or that the message does not match one of the applications installed in the default cluster.
- **Symptom:** An "out of memory" type behavior is occurring.
Solution: This may be due to the maximum heap size being set too low. SIP applications can consume a significant amount of memory because the sessions exist for a long call hold time. The maximum heap size of 512 MB does not provide sufficient memory for the SIP traffic workload. Set the maximum heap size for SIP applications to the minimum recommended value of 768 MB or higher.
- **Symptom:** You receive a "403 Forbidden" when sending a SIP request to a SIP container.
Solution: This usually means there is no appropriate SIP application found to handle the received SIP request (no match rule that matched the message).

Tuning SIP servlets for Linux

This page describes preliminary SIP servlet tuning for Linux 2.6 kernel.

Before you begin

A Session Initiation Protocol (SIP) servlet under load might retransmit messages or drop calls. The UDP socket queues might fill. A review of the verbose garbage collection output might show that there are fairly long garbage collection times, for example, 0.5 to 1.5 seconds. The cause of this problem is that the Ethernet driver, Linux[®] operating system, WebSphere[®] Application Server, or any combination of the items are not tuned for SIP applications. You can apply the following levels of tuning.

Note: The following recommendations have been tested on Red Hat Enterprise Linux 4 only and are provided as is without any implied warranty.

About this task

Linux Ethernet driver

Linux Ethernet driver tuning begins by selecting the best Ethernet driver. For example, the HS20 blades recommended driver is the tg3-3.43b driver (or later), which can be found at the Web site for Broadcom Ethernet NIC Driver Downloads. The following shell commands have been used to tune the Linux kernel Ethernet driver:

```
/sbin/ifconfig eth0 txqueuelen 2000
/sbin/ifconfig eth1 txqueuelen 2000
ethtool -s eth0 autoneg off speed 1000 duplex full
ethtool -A eth0 autoneg off rx on tx on
ethtool -C eth0 adaptive-rx off adaptive-tx off rx-
usecs 20 rx-frames 5 tx-usecs 60 tx-frames 11
ethtool -G eth0 rx 511 rx-jumbo 255 tx 511
```

Depending upon the Ethernet driver that is installed, some of these options might need to change.

Linux kernel

Linux kernel tuning uses the following commands:

```
echo 2097152 > /proc/sys/net/core/rmem_max
echo 2097152 > /proc/sys/net/core/rmem_default
echo 2097152 > /proc/sys/net/core/wmem_max
echo 2097152 > /proc/sys/net/core/wmem_default
echo 10000000 > /proc/sys/net/core/optmem_max
echo 262143 262143 262143 > /proc/sys/net/ipv4/tcp_rmem
echo 262143 262143 262143 > /proc/sys/net/ipv4/tcp_wmem
echo 8388608 8388608 8388608 > /proc/sys/net/ipv4/tcp_mem
echo 400 > /proc/sys/net/unix/max_dgram_qlen
echo 400 > /proc/sys/net/core/message_burst
echo 2800 > /proc/sys/net/core/mod_cong
echo 1000 > /proc/sys/net/core/lo_cong
echo 200 > /proc/sys/net/core/no_cong
echo 2900 > /proc/sys/net/core/nō_cong_thresh
echo 3000 > /proc/sys/net/core/netdev_max_backlog
```

This configuration might not be optimum for a given application and you might need to adjust the configuration to achieve the best performance. However, you might use these values as a starting point.

SIP for WebSphere Application Server

SIP tuning for WebSphere Application Server is completed using the following steps:

1. Create a separate thread pool for the SIP servlet container. Follow this path in the administrative console:
 - a. Click **Server > Application servers > *server_name***.
 - b. Under Additional properties, click **Thread Pools > New**.
 - c. In the Name field, enter *SipContainer*.
 - d. In the Minimum Size and Maximum Size fields, enter *15*. These values should be adequate for most applications.
 - e. Click **OK**.
2. Create custom properties for the SIP Servlet container. Follow this path in the administrative console:
 - a. Click **Server > Application servers > *server_name***.
 - b. Click **SIP container**.
 - c. Under **Additional properties**, click **Custom Properties > New**.
 - d. In the Name field, enter *javax.sip.max.object.pool.size*.
 - e. In the Value field, enter *1000*.
 - f. Click **OK**.
 - g. In the Name field, enter *max.tu.pool.size*.
 - h. In the Value field, enter *1000*.
 - i. Click **OK**.
3. Create custom properties for the SIPUDP channel if User Datagram Protocol (UDP) is the primary transport for SIP traffic. Follow this path in the administrative console:
 - a. Click **Server > Application servers > *server_name***.
 - b. Click **SIP container > Transport Chain > SIPInboundDefaultUDP > UDP Inbound channel (UDP1)**.
 - c. Under **Additional Properties**, click **Custom Properties > New**.
 - d. In the Name field, enter *receiveBufferSocketSize*.
 - e. In the Value field, enter *3000000*.
 - f. Click **OK**.
 - g. In the Name field, enter *sendBufferSocketSize*.
 - h. In the Value field, enter *3000000*.
4. Specify the SIP servlet container general properties. Follow this path in the administrative console:
 - a. Click **Servers > Application Servers > *server_name* > SIP container**.
 - b. Enter the Maximum application sessions value. The Maximum application sessions value can be calculated as: *Maximum call hold time or session timeout x Call rate x Safety factor*.
 - c. Enter the Maximum messages per averaging period value. The Maximum messages per averaging period value can be calculated as: *Maximum call hold time or session timeout x Maximum rate of SIP messages x Safety factor*.
 - d. Enter the Maximum dispatch queue size value. The Maximum dispatch queue size value can be calculated as: *Maximum rate of SIP messages x Maximum latency in SIP processing x Safety factor*.
 - e. Set the thread pool to the newly created SIP container thread pool (to the drop down name "SipContainer").
5. Tune the Java virtual machine (JVM) garbage collection policy. Follow this path in the administrative console:
 - a. Click **Server > Application servers > *server_name***.
 - b. Under Server Infrastructure, click **Java and Process Management > Process Definition**.
 - c. Under **Additional Properties**, click **Java Virtual Machine**.

- d. In the Generic JVM arguments field, enter the following value as one continuous line:
`1"-Xgcpolicy:gencon -Xgc:scvNoAdaptiveTenure,scvTenureAge=8,
stdGlobalCompactToSatisfyAllocate".`

Optional: You also might add a value of 1500 MB to the Initial heap size and Maximum heap size fields. It is also a good practice to enable the **Verbose garbage collection** option during performance testing or tuning operations.

SIP timer summary

Request for Comments (RFC) 3261, "SIP: Session Initiation Protocol," specifies various timers that SIP uses.

Table 25 on page 356 summarizes for each SIP timer the default value, the section of RFC 3261 that describes the timer, and the meaning of the timer.

Table 28. Summary of SIP timers

Timer	Default value	Section	Meaning
T1	500 ms	17.1.1.1	Round-trip time (RTT) estimate
T2	4 sec.	17.1.2.2	Maximum retransmission interval for non-INVITE requests and INVITE responses
T4	5 sec.	17.1.2.2	Maximum duration that a message can remain in the network
Timer A	initially T1	17.1.1.2	INVITE request retransmission interval, for UDP only
Timer B	64*T1	17.1.1.2	INVITE transaction timeout timer
Timer C	> 3 min.	16.6 bullet 11	Proxy INVITE transaction timeout
Timer D	> 32 sec. for UDP	17.1.1.2	Wait time for response retransmissions
	0 sec. for TCP and SCTP		
Timer E	initially T1	17.1.2.2	Non-INVITE request retransmission interval, UDP only
Timer F	64*T1	17.1.2.2	Non-INVITE transaction timeout timer
Timer G	initially T1	17.2.1	INVITE response retransmission interval
Timer H	64*T1	17.2.1	Wait time for ACK receipt
Timer I	T4 for UDP	17.2.1	Wait time for ACK retransmissions
	0 sec. for TCP and SCTP		
Timer J	64*T1 for UDP	17.2.2	Wait time for retransmissions of non-INVITE requests
	0 sec. for TCP and SCTP		
Timer K	T4 for UDP	17.1.2.2	Wait time for response retransmissions
	0 sec. for TCP and SCTP		

Appendix. Directory conventions

References in product information to *app_server_root*, *profile_root*, and other directories infer specific default directory locations. This topic describes the conventions in use for WebSphere Application Server.

Default product locations (distributed)

The following file paths are default locations. You can install the product and other components or create profiles in any directory where you have write access. Multiple installations of WebSphere Application Server Network Deployment products or components require multiple locations. Default values for installation actions by root and non-root users are given. If no non-root values are specified, then the default directory values are applicable to both root and non-root users.

app_client_root

The following list shows default installation root directories for the WebSphere Application Client.

User	Directory
Root	<p>AIX /usr/IBM/WebSphere/AppClient (Java EE Application client only)</p> <p>HP-UX Linux Solaris /opt/IBM/WebSphere/AppClient (Java EE Application client only)</p> <p>Windows C:\Program Files\IBM\WebSphere\AppClient</p>
Non-root	<p>AIX HP-UX Linux Solaris <i>user_home</i>/IBM/WebSphere/AppServer/AppClient (Java EE Application client only)</p> <p>Windows C:\IBM\WebSphere\AppClient</p>

app_server_root

The following list shows the default installation directories for WebSphere Application Server Network Deployment.

User	Directory
Root	<p>AIX /usr/IBM/WebSphere/AppServer</p> <p>HP-UX Linux Solaris /opt/IBM/WebSphere/AppServer</p> <p>Windows C:\Program Files\IBM\WebSphere\AppServer</p>
Non-root	<p>AIX HP-UX Linux Solaris <i>user_home</i>/IBM/WebSphere/AppServer</p> <p>Windows C:\IBM\WebSphere\AppServer</p>

cip_app_server_root

A *customized installation package* (CIP) is an installation package created with IBM WebSphere Installation Factory that contains a WebSphere Application Server Network Deployment product bundled with one or more maintenance packages, an optional configuration archive, one or more optional enterprise archive files, and other optional files and scripts.

The following list shows the default installation root directories for a CIP where *cip_uid* is the CIP unique ID generated during creation of the build definition file.

User	Directory
Root	<p>AIX /usr/IBM/WebSphere/AppServer/cip/cip_uid</p> <p>HP-UX Linux Solaris /opt/IBM/WebSphere/AppServer/cip/cip_uid</p> <p>Windows C:\Program Files\IBM\WebSphere\AppServer\cip\cip_uid</p>
Non-root	<p>AIX HP-UX Linux Solaris user_home/IBM/WebSphere/AppServer/cip/cip_uid</p> <p>Windows C:\IBM\WebSphere\AppServer\cip\cip_uid</p>

component_root

The component installation root directory is any installation root directory described in this topic. Some programs are for use across multiple components. In particular, the Update Installer for WebSphere Software is for use with WebSphere Application Server Network Deployment, Web server plug-ins, the Application Client, and the IBM HTTP Server. All of these components are part of the product package.

gskit_root

IBM Global Security Kit (GSKit) can now be installed by any user. GSKit is installed locally inside the installing product's directory structure and is no longer installed in a global location on the target system. The following list shows the default installation root directory for Version 7 of the GSKit, where *product_root* is the root directory of the product that is installing GSKit, for example IBM HTTP Server or the Web server plug-in.

Directory
<p>AIX HP-UX Linux Solaris product_root/gsk7</p> <p>Windows product_root\gsk7</p>

if_root This directory represents the root directory of the IBM WebSphere Installation Factory. Because you can download and unpack the Installation Factory to any directory on the file system to which you have write access, this directory's location varies by user. IBM WebSphere Installation Factory is an Eclipse-based tool which creates installation packages for installing WebSphere Application Server in a reliable and repeatable way, tailored to your specific needs.

iip_root

This directory represents the root directory of an *integrated installation package* (IIP) produced by the IBM WebSphere Installation Factory. Because you can create and save an IIP to any directory on the file system to which you have write access, this directory's location varies by user. An IIP is an aggregated installation package that can include one or more generally available installation packages, one or more customized installation packages (CIPs), and other user-specified files and directories.

profile_root

The following list shows the default directory for a profile named *profile_name* on each distributed operating system.

User	Directory
Root	<p>AIX /usr/IBM/WebSphere/AppServer/profiles/profile_name</p> <p>HP-UX Linux Solaris /opt/IBM/WebSphere/AppServer/profiles/profile_name</p> <p>Windows C:\Program Files\IBM\WebSphere\AppServer\profiles\profile_name</p>

User	Directory
Non-root	<div style="display: flex; justify-content: space-around; margin-bottom: 5px;"> AIX HP-UX Linux Solaris </div> <code>user_home/IBM/WebSphere/AppServer/profiles/</code> <div style="display: flex; justify-content: space-between; margin-top: 10px;"> Windows C:\IBM\WebSphere\AppServer\profiles\ </div>

plugins_root

The following default installation root is for the Web server plug-ins for WebSphere Application Server.

User	Directory
Root	<div style="display: flex; justify-content: space-around; margin-bottom: 5px;"> AIX </div> <code>/usr/IBM/WebSphere/Plugins</code> <div style="display: flex; justify-content: space-around; margin-bottom: 5px;"> HP-UX Linux Solaris </div> <code>/opt/IBM/WebSphere/Plugins</code> <div style="display: flex; justify-content: space-between; margin-top: 10px;"> Windows C:\Program Files\IBM\WebSphere\Plugins </div>
Non-root	<div style="display: flex; justify-content: space-around; margin-bottom: 5px;"> AIX HP-UX Linux Solaris </div> <code>user_home/IBM/WebSphere/Plugins</code> <div style="display: flex; justify-content: space-between; margin-top: 10px;"> Windows C:\IBM\WebSphere\Plugins </div>

updi_root

The following list shows the default installation root directories for the Update Installer for WebSphere Software.

User	Directory
Root	<div style="display: flex; justify-content: space-around; margin-bottom: 5px;"> AIX </div> <code>/usr/IBM/WebSphere/UpdateInstaller</code> <div style="display: flex; justify-content: space-around; margin-bottom: 5px;"> HP-UX Linux Solaris </div> <code>/opt/IBM/WebSphere/UpdateInstaller</code> <div style="display: flex; justify-content: space-between; margin-top: 10px;"> Windows C:\Program Files\IBM\WebSphere\UpdateInstaller </div>
Non-root	<div style="display: flex; justify-content: space-around; margin-bottom: 5px;"> AIX HP-UX Linux Solaris </div> <code>user_home/IBM/WebSphere/UpdateInstaller</code> <div style="display: flex; justify-content: space-between; margin-top: 10px;"> Windows C:\IBM\WebSphere\UpdateInstaller </div>

web_server_root

The following default installation root directories are for the IBM HTTP Server.

User	Directory
Root	<div style="display: flex; justify-content: space-around; margin-bottom: 5px;"> AIX </div> <code>/usr/IBM/HTTPServer</code> <div style="display: flex; justify-content: space-around; margin-bottom: 5px;"> HP-UX Linux Solaris </div> <code>/opt/IBM/HTTPServer</code> <div style="display: flex; justify-content: space-between; margin-top: 10px;"> Windows C:\Program Files\IBM\HTTPServer </div>
Non-root	<div style="display: flex; justify-content: space-around; margin-bottom: 5px;"> AIX HP-UX Linux Solaris </div> <code>user_home/IBM/HTTPServer</code> <div style="display: flex; justify-content: space-between; margin-top: 10px;"> Windows C:\IBM\HTTPServer </div>

Notices

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only IBM's product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any of IBM's intellectual property rights may be used instead of the IBM product, program, or service. Evaluation and verification of operation in conjunction with other products, except those expressly designated by IBM, is the user's responsibility.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Intellectual Property & Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
USA

Trademarks and service marks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. If these and other IBM trademarked terms are marked on their first occurrence in this information with a trademark symbol (® or ™), these symbols indicate U.S. registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. For a current list of IBM trademarks, visit the IBM Copyright and trademark information Web site (www.ibm.com/legal/copytrade.shtml).

Microsoft and Windows are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Java is a trademark of Sun Microsystems, Inc. in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.