IBM Informix Dynamic Server
December 2004

DB2 Information Management Software

IBM Informix Dynamic Server and High Availability on Sun Cluster 3.x

Srinivasrao Madiraju,
Pradeep Kulkarni and
Michael Nemesh
IBM Software Group

Contents

2 Overview
2 IBM Informix Dynamic Server 9.40
3 Sun Cluster 3.x
9 IBM IDS: Installation and configuration in Sun Cluster 3.x environment
14 SC3.1 Update 2 IBMids agent
33 Customizing SC3.x resource properties for IBM IDS
35 Cluster verification testing
40 Cluster topology
43 Optimizing cluster file system performance
50 Summary
51 Appendix A: Example sqlhosts
52 Appendix B: Example onconfig
58 Appendix C: Error messages

Overview

Businesses today demand application service continuity at a low cost. Sun Cluster software is the industry's premier availability platform for improving the predictability and resilience of business-critical applications and other applications such as databases. This white paper covers the information related to setup and testing of IBM Informix® Dynamic Server (IDS) 9.40 on Sun Cluster 3.1 (SC3.x) Update 2 in a Sun Solaris 9 SPARC environment with the IBMids agent.

The reader should be familiar with the motivation for the implementation of a highly available (HA) solution and with basic terminology in this field. However, some crucial overview material will be presented to prepare the reader for the installation of the IBMids agent.

IBM Informix Dynamic Server 9.40

The IBM Informix Dynamic Server 9.40 engine represents the next evolution of database technology by merging the world-class performance and scalability of data systems architecture with cutting-edge object-relational technology. IBM IDS 9.40 sets new standards for scalability, reliability and availability to meet the unpredictable challenges of On Demand Business. IBM IDS 9.40 provides the essential qualities needed for today's always-available, Web application–based online transaction processing (OLTP). IDS 9.40 also is the fastest IBM IDS engine, providing exceptional performance, availability, security, simplified manageability and support of business partner applications.

Sun Cluster 3.x

Sun Cluster 3.x represents the next generation of the Sun Cluster product line, which delivers an easy-to-use, highly available and scalable clustering solution that is tightly integrated with Solaris. It is also a service-level management platform for deploying highly available application services. It delivers higher service levels at lower cost of operation and risk.

By tightly coupling Sun server, storage and networking solutions, Sun Cluster 3.x provides the maximum level of service availability and performance for a cluster system.

The servers (nodes) in a cluster communicate through private interconnects. These interconnects carry important cluster information, such as data and a cluster "heartbeat." This heartbeat lets the servers in the cluster know the status of the other servers within the cluster, ensuring that each server is operational. If a server goes offline and ceases its heartbeat, the rest of the devices in the cluster isolate the server and fail over applications and data from the failing node to another node. This failover process is quick and transparent to the users of the system. By exploiting the redundancy in the cluster, Sun Cluster helps ensure high levels of availability.

A typical Sun Cluster configuration has several components:

Hardware:

- *Servers with local storage (storage devices hosted by one node)*
- *Shared storage (storage devices hosted by more than one node)*
- *Cluster interconnect for private communication among the cluster nodes*
- *Public network interfaces for connectivity to the outside world*
- *Administrative workstation for managing the cluster*

Software:

- *Solaris Operating Environment running on each cluster node*
- *Sun Cluster 3.1 software running on each cluster node*
- *Data Services—applications with agents and fault monitors—running on one or more cluster nodes*

Failover

Failover is the process by which the cluster automatically relocates a service from a failed primary node to a designated secondary node. With failover, Sun Cluster software provides high availability. When a failover occurs, clients might see a brief interruption in service and might need to reconnect after the failover has finished. However, clients are not aware of a change in the physical server that provides the service.

By allowing other nodes in a cluster to automatically host workloads when the primary node becomes nonfunctional, Sun Cluster can significantly reduce downtime and increase productivity by providing high availability service to all users.

Multi-host disks

Sun Cluster requires multi-host disk storage—disks that can be connected to more than one node at a time. In the Sun Cluster environment, multi-host storage allows disk devices to become highly available. Disk devices utilizing multi-host storage can tolerate single node failures since a physical path to the data via the alternate server node still exists.

Multi-host disks can be accessed globally through a primary node; this node is said to master the disk. If client requests are accessing the data through one node and that node fails, the requests are routed to another node that has a direct connection to the same disks.

A volume manager provides for mirrored or RAID-5 configurations for data redundancy of multi-host disks. Currently, Sun Cluster supports Solaris Volume Manager and VERITAS Volume Manager as volume managers, and the RDAC RAID-5 hardware controller on several hardware RAID platforms. Combining multi-host disks with disk mirroring and striping protects against both node failure and individual disk failure.

Global devices

Sun Cluster uses global devices to provide cluster-wide, highly available access to any device in a cluster, from any node, without regard to where the device is physically attached. All disks are included in the global namespace with an assigned device ID and are configured as global devices; therefore, the disks themselves are visible from all cluster nodes.

Cluster file systems, global file systems and failover file systems

A cluster file system, also referred to as a global file system, is a proxy between the kernel on one node and the underlying file system volume manager running on a node that has a physical connection to the disk(s). Cluster file systems are dependent on global devices with a physical connection to one or more nodes. The cluster file system is independent of the underlying file system and volume manager. Currently, cluster file systems can be built on the UNIX® file system (UFS) using either Solaris Volume Manager or VERITAS Volume Manager. The data only becomes available to all nodes if the file systems on the disks are mounted globally as a cluster file system.

The HAStoragePlus resource type is designed to make non-global file system configurations such as UFS and VERITAS file system highly available. HAStoragePlus bypasses the global file service layer completely, which can lead to a significant performance increase for disk I/O–intensive data services.

HAStoragePlus can work with any file system, even those that might not work with the global file service layer. If a file system is supported by the Solaris operating system, it will work with HAStoragePlus. File systems that were used previously with HAStorage as global file systems can be used with HAStoragePlus as failover file systems (FFSs) by simply changing their automount entry in /etc/vfstab to "no" and removing the "global" mount option. For example, GFS entries like:

```
/dev/vx/dsk/scdg2/vol05 /dev/vx/rdsk/scdg2/vol05 /global/informix
    ufs 2 yes global,logging
```

would be changed to an FFS entry such as:

```
/dev/vx/dsk/scdg2/vol05 /dev/vx/rdsk/scdg2/vol05 /global/informix
    ufs 2 no logging
```

Device groups

In a Sun Cluster environment, all multi-host disks must be under control of the Sun Cluster framework. Disk groups, managed by either Solaris Volume Manager or VERITAS Volume Manager, are first created on the multi-host disk. Then, they are registered as Sun Cluster disk device groups. A disk device group is a type of global device. Multi-host device groups are highly available. Disks are accessible through an alternate path if the node currently mastering the device group fails. The failure of the node mastering the device group does not affect access to the device group except for the time required to perform recovery and consistency checks. During this time, all requests are blocked (transparently to the application) until the system makes the device group available.

Resource Group Manager

The cluster facility known as the Resource Group Manager (RGM) provides the mechanism for high availability. The RGM runs as a daemon on each cluster node and automatically starts and stops resources on selected nodes according to preconfigured policies. The RGM allows a resource to be highly available in the event of a node failure or reboot by stopping the resource on the affected node and starting it on another. The RGM also automatically starts and stops resource-specific monitors that can detect resource failures and relocate failing resources onto another node. It can also monitor other aspects of resource performance.

Data services

A data service is a third-party application that has been configured to run on a cluster rather than on a single server. A data service consists of an application, specialized Sun Cluster configuration files and Sun Cluster management methods that start, stop and monitor the application.

The Sun Cluster software supplies data service methods that are used to control and monitor the application within the cluster. These methods run under the control of the RGM, which uses them to start, stop and monitor the application on the cluster nodes. These methods, along with the cluster framework software and multi-host disks, help enable applications to become highly available data services. As highly available data services, they can prevent significant application interruptions after any single failure within the cluster. The failure could be a node, an interface component or the application itself. The RGM also manages resources in the cluster, including instances of applications and network resources—logical hostnames and shared addresses.

Resource types, resource groups and resources

A resource type consists of a software application to be run on the cluster, control programs used as callback methods by the RGM to manage the application as a cluster resource and a set of properties that form part of the static configuration of a cluster. The RGM uses resource type properties to manage resources of a particular type.

A resource, which inherits the properties and values of its resource type, is an instance of the underlying application running on the cluster. Each instantiation requires a unique name within the cluster. Each resource must be configured in a resource group.

The RGM brings all resources in a group online and offline together on the same node. When the RGM brings a resource group online or offline, it invokes callback methods on the individual resources in the group. The nodes on which a resource group is currently online are referred to as its primaries or its primary nodes.

A resource group is mastered by each of its primaries. Each resource group has an associated NodeList property—set by the cluster administrator—which identifies all potential primaries or masters of the resource group.

IBM Informix Dynamic Server:
Installation and configuration in a Sun Cluster 3.x environment

The installation of IBM IDS in a Sun Cluster environment closely mirrors that which is documented in the appropriate *Installation Guide for Informix Dynamic Server.*

Cluster file system/failover file system considerations

At this stage, it is presumed that Sun Cluster is configured according to the information given in the *Sun Cluster 3.1 Installation Guide.* It is recommended that at least one failover file system be defined and available, because it will be used to host the Informix user's home directory and Informix binaries directory (INFORMIXDIR) for each IBM IDS instance to be installed.

Briefly, there are three distinct binary location strategies: they can exist on a global file system, on a failover file system or on each physical node on a local file system. It is recommended that the IBM IDS binaries be placed on a failover file system for the following reasons:

- *Access to the binaries will be highly available (special steps will need to be taken to ensure that access to the binaries will be highly available if they are locally installed and mounted)*
- *Installation of binary images needs to be performed only once (per set of cluster nodes that share the global binary installation mount points)*
- *There is no chance of having mismatched binary code levels across the nodes of the cluster*
- *Nodes that are added to the cluster can automatically access the IBM IDS binary*
- *Easy-to-use backup files created by OnBar*

It is strongly recommended that the failover file system be used as the instance home directory mount point, although it is not required; otherwise, potential availability issues may arise with respect to the instance home directory, which must be considered should a failover file system, or at least a cluster file system, not be utilized for this task.

User and group creation considerations

On each physical node that will participate in the cluster, an Informix group and user account must be created for the Informix database administrator (DBA) account.

If Network Information Service (NIS) or NIS+ are in use, groups and users must be created on the NIS server prior to executing the install programs. If using local server authentication, repeat the following steps on all the nodes in the cluster. The users and groups must be defined identically across all nodes in the cluster; ensure that the user id, group id, user home directory, account passwords and user shell are consistent across all nodes in the cluster.

For example, use the following Solaris command to create the Informix group on each server in a local server authentication environment:

```
groupadd -g 1001 informix
```

Use the following Solaris command to create the Informix user account without creating home:

```
useradd -g informix -u 1001 -d /global/informix -m -s /bin/ksh
    informix
```

Note that for these commands, we presume that /global/informix is the mount point of a cluster file system that will be used to host the INFORMIXDIR directory. Please ensure that the account passwords are consistent across all nodes in the cluster (use the Solaris `passwd` command to set them if necessary).

Miscellaneous considerations

At this point, you are advised to consult the appropriate *Installation Guide for Informix Dynamic Server* for specifics on installation and initializations. Another important point, unchanged from a typical IBM IDS installation on Solaris, is to ensure that the correct kernel parameters have been applied at each physical node in the cluster. These parameters can be found in the release notes delivered with the IBM IDS software.

Installation of IBM IDS binary

It is recommended that the software be installed in a failover file system that is mountable on both the primary and backup system, as needed. The database server should be installed with the standard procedures prescribed for that version of the server on the primary machine and initialized like a normal installation. Once that is complete, the following steps should be followed depending on the IBM IDS version being installed.

IBM IDS 7.x—The same procedure should be followed on the backup system. Although a savvy DBA or system administrator may be able to create the necessary links for the install to work correctly on the second machine, rerunning the installer on the second system will not affect the primary system, but will make sure that all necessary steps have been completed on both machines.

IBM IDS 9.x—On the backup system, run the RUNASROOT.xxxxx files in the same order as they were run on the primary machine. If a failover file system is used, the file system must be unmounted on the primary system and mounted on the secondary system before this can be completed.

Installation of user-defined routines (UDRs) and DataBlades—If you are using UDRs and/or DataBlades, make sure you install them on a global or failover file system. Otherwise, you must install them on each cluster node and maintain similar paths for the files on each.

Configuration of a particular IBM IDS instance

First, become superuser on all nodes. Then configure the /etc/nsswitch.conf file as follows so that the Sun Cluster HA for IBM IDS data service starts and stops correctly if a switchover or failover occurs.

On each node that can master the logical host that runs the Sun Cluster HA for IBM IDS data service, include one of the following entries for group in the /etc/nsswitch.conf files:

```
group:
group: files
group: files [NOTFOUND=return] nis
group: file [NOTFOUND=return] nisplus
```

Refer to the *Sun Cluster 3.1 Installation Guide* for further nsswitch.conf entries required by SC3.1.

The Sun Cluster HA for IBM IDS data service uses the `su` user command to start and stop the database node. The user is typically informix. The network information name service might become unavailable when a cluster node's public network fails.

Adding one of the preceding entries for group ensures that the `su(1M)` command does not refer to the NIS/NIS+ name services if the network information name service is unavailable.

Configure the failover file system for the Sun Cluster HA for IBM IDS data service. If raw devices contain the databases, configure the global devices for raw-device access. See the *Sun Cluster 3.1 Installation Guide* for information on how to configure global devices.

When you use the Solaris Volume Manager, configure the IBM IDS software to use UFS logging on mirrored meta devices or raw-mirrored meta devices. See the Solaris Volume Manager documentation for more information on how to configure raw-mirrored meta devices.

On the primary node, create the directory to be used for the software install on a failover file system:

```
cd /global/informix/
mkdir ids940
chown informix:informix ids940
chmod 755 ids940
```

The IBM IDS software should be extracted into the previously created directory and the standard procedures for software installation should be followed for a normal installation with some minor exceptions that can be done in parallel. Add a hostname alias, like dbserver, on each system in the /etc/hosts file as shown below.

Example: /etc/hosts for a primary system named suninfo

```
#
# Internet host table
#
127.0.0.1 localhost
192.168.0.201 pelican
192.168.0.202 suninfo loghost dbserver
```

and a backup system named pelican

```
#
# Internet host table
#
127.0.0.1 localhost
192.168.0.201 pelican loghost dbserver
192.168.0.202 suninfo
```

Next make the appropriate entry in /etc/services for the service name to be used in the sqlhosts file on each machine:

Example: /etc/services entry

```
online940   1528/tcp
online940mc  1529/tcp
```

Then configure the onconfig and sqlhosts files with the appropriate information for the instance based upon those entries. This configuration only needs to be done on one machine since the software is installed on a global file system. Examples of these files are found in Appendix A. Once it has been verified that the instance can be started and stopped with normal procedures, one more step needs to be performed on the backup system. Execute the RUNASROOT.xxxxx in the same order as on the primary machine to create the necessary links for the server. You can now attempt to create the stores database (or an empty test database if you prefer). Create the database on the path of the global device that you plan to use for storage of the actual production database. Ensure that the create database command completes successfully. Proceed to remove the test database via the drop database command. The instance is now ready to be made highly available.

SC3.1 Update 2 IBMids agent

The IBMids agent is a software product consisting of the methods required to start, stop and monitor IBM IDS in a Sun Cluster environment. These methods are implemented as shell scripts, which allow for maximum flexibility in the hands of a skilled user. This agent must be installed separately on each node of the cluster. General instructions for installing and setting up IBMids agent for IDS are discussed in detail in the next sections.

Cluster topology

The IBMids agent fully supports all three cluster topology classes supported by SC3.x:

- *Clustered pairs/idle standby*
- *N+1 (or Star)*
- *Pair+N*

We will discuss examples of using each cluster topology class later in this paper. Note however, that the nomenclature is somewhat fluid (that is, there are a variety of names used interchangeably in the HA industry—we will try to conform to those stated).

Logical hostname/IP failover

Both a logical hostname and the IP address it maps to must be associated with a particular IBMids instance. Client programs will access the IBM IDS database instance using this logical hostname instead of the physical hostname of a server in the cluster. This logical hostname is the entry point to the cluster, preventing the client program from addressing the physical servers directly. This logical hostname/IP address combination is configured on the clients (via the `setnet32` command or the open database connectivity driver manager) as a logical hostname resource, which must then be added to the same resource group as the instance resource.

If a failure occurs, the entire resource group including the instance and the logical hostname will fail over to the backup server. This floating IP setup provides high-availability IBM IDS service to client programs. The hostname must map to an IP address and the mapped hostname and IP address must be configured on all nodes in the cluster. More information on configuration for public IP addresses can be found in the *Sun Cluster 3.1 Installation Guide*. The following is a /etc/hosts file with the logical hostname entry for the example systems.

Example: /etc/hosts with logical hostname entry for suninfo

```
#
# Internet host table
#
127.0.0.1 localhost
192.168.0.201 pelican
192.168.0.202 suninfo loghost
192.168.0.203 dbserver
```

and backup system pelican

```
#
# Internet host table
#
127.0.0.1 localhost
192.168.0.201 pelican loghost
192.168.0.202 suninfo
192.168.0.203 dbserver
```

Logical hostname considerations

In the case of IBM IDS, the highly available logical hostname/IP address must be collocated (located within the same SC3.x resource group) as the instance itself. This will guarantee that the logical hostname/IP address will always be local to the IBM IDS instance. Otherwise, if they do not exist within the same SC3.x resource group, the possibility exists that a particular physical node may host the logical hostname/IP address while a different physical node may host the IBM IDS instance itself. It is suggested that the dbservername entry in the onconfig file coincide with the sqlhosts entry that contains the logical/failover hostname and the /etc/services entry that matches the portlist for the IBM IDS resource. The correlation of these files is shown in the examples provided in this document.

Agent utilities

Three utilities (startids, stopids and removeids) supplied with IBM IDS and located in /opt/IBMids/util are used to control the installation, start, stop and removal of a basic IBMids agent in a Sun Cluster 3.x environment. These scripts are designed to help with the initial setup of an IBMids agent, but their capabilities are limited. Note that although a number of other components are included in the package, only these three items can be executed directly by the individual user. A man-style document is provided in the man directory for each of the three supplied methods in /opt/IBMids.

Note: The /opt/IBMids/man path should be added to the manpath environment variable.

Startids utility. This method will register and selectively start the appropriate resources and resource groups for a specified instance. It will normally attempt to bring the resource group online unless the `-x` option is used, which is recommended for an initial installation. Commonly, this will be the first script called, because it will perform all necessary steps to prepare IBM IDS for SC3.x control including prompting for IDS-specific information and validating the user's responses before configuring the agent. To create an IBMids resource type for the example systems, the following `startids` command can be used:

```
/opt/IBMids/util/startids
```

Alternatively, the following SC3.x commands could be run from the command line:

```
# Register resource type <IBM.ids>
  scrgadm -a -t IBM.ids

# Create failover resource group <ids-harg>
  scrgadm -a -g ids-harg -h suninfo,pelican

# Create logical host resource <dbserver>
  scrgadm -a -L -g ids-harg -l dbserver -n nafo0@suninfo,nafo0@pelican

# Create HAStoragePlus resource <ids-hasp>
  scrgadm -a -t SUNW.HAStoragePlus -g ids-harg -j ids-hasp \
  -x GlobalDevicePaths=ids-ds1 \
  -x FilesystemMountPoints=/global/informix

# Bring resource group <ids-harg> online due to HAStoragePlus
  dependency
  scswitch -Z -g ids-harg

# Create resource <ids-hars> for the resource type <IBM.ids>
  scrgadm -a -j ids-hars -g ids-harg -t IBM.ids \
  -y scalable=false \
  -y Port_list=1528/tcp,1529/tcp \
  -y Network_resources_used=dbserver \
```

```
  -y Resource_dependencies=ids-hasp \
  -x Confdir_list="" \
  -x START_TIMEOUT=300 \
  -x STOP_TIMEOUT=300 \
  -x PROBE_TIMEOUT=30 \
  -x INFORMIXDIR=/global/informix/ids940 \
  -x INFORMIXSERVER=dbserver_shm \
  -x ONCONFIG=onconfig.940 \
  -x INFORMIXSQLHOSTS=/global/informix/ids940/etc/sqlhosts

# Bring the resource group <ids-harg> online
  scswitch -Z -g ids-harg
```

After the initial setup with the `startids` command, the resource group can be started with the following command:

```
./startids -h dbserver
```

Stopids utility. This method will execute the SC3.x commands required to bring a highly available IBM IDS instance offline. It will not remove any resources or resource groups from the cluster. To stop the instance for the example systems, the following `stopids` command can be used:

```
/opt/IBMids/util/stopids
```

Alternatively, the following SC3.x commands could be run from the command line:

```
# Offline resource group <ids-harg> on all nodes
  scswitch -z -g ids-harg -h ""
# Disable the resource <ids-hars>
  scswitch -n -j ids-hars
# Disable the resource <ids-hasp>
  scswitch -n -j ids-hasp
# Disable the network resource <dbserver>
  scswitch -n -j dbserver
```

Removeids utility. This method will execute the commands required to remove the IBMids agent from the SC3.1 cluster. It will remove the resources and groups created by the `startids` command. Essentially, this method is the inverse of `startids` and will generally be called if the database instance is no longer required to be highly available. To remove the IBMids resource type, resource groups and resources for the example systems, the following `removeids` command can be used:

```
/opt/IBMids/util/removeids
```

Alternatively, the following SC3.x commands can be run from the command line:

```
# Offline the resource group <ids-harg>
  scswitch -z -g ids-harg -h ""
# Disable the resource <ids-hars>
  scswitch -n -j ids-hars
# Remove the resource <ids-hars>
  scrgadm -r -j ids-hars
# Disable the resource <ids-hasp>
  scswitch -n -j ids-hasp
# Remove the resource <ids-hasp>
  scrgadm -r -j ids-hasp
# Disable the resource <dbserver>
  scswitch -n -j dbserver
# Remove the resource <dbserver>
  scrgadm -r -j dbserver
# Unmanage the resource group <ids-harg> ...
  scswitch -u -g ids-harg
# Remove the resource group <ids-harg> ...
  scrgadm -r -g ids-harg
# Remove the resource type <IBM.ids>
  scrgadm -r -t IBM.ids
```

In this agent setup example, we will create a highly available IBM IDS instance. Figure 1 shows the example cluster in diagrammatic form.

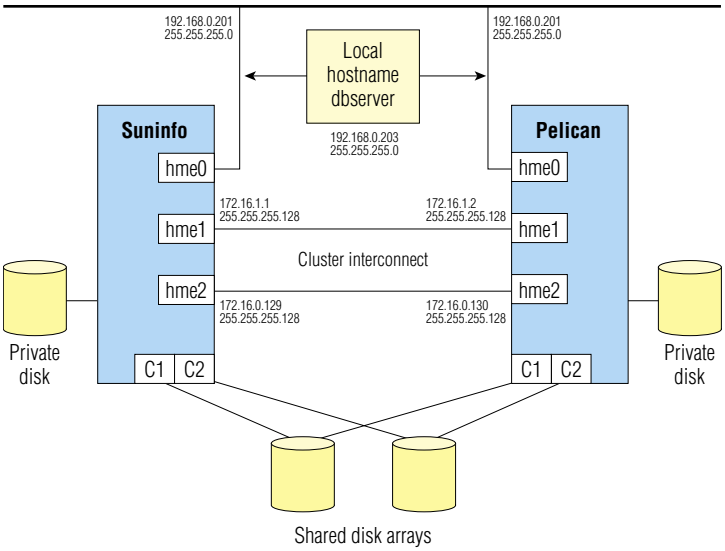

Figure 1: Example of a highly available IBM IDS instance

In terms of the existing SC3.1 infrastructure, we will use the `scstat`, `scrgadm` and `scdidadm` commands to display the information for the cluster (blank lines removed to save space):

```
# scstat -p
------------------------------------------------------------------
-- Cluster Nodes --
Node name Status
--------- ------
Cluster node: suninfo Online
Cluster node: pelican Online
------------------------------------------------------------------
-- Cluster Transport Paths --
Endpoint Endpoint Status
-------- -------- ------
Transport path: suninfo:hme2 pelican:hme2 Path online
Transport path: suninfo:hme1 pelican:hme1 Path online
```

```
------------------------------------------------------------------
-- Quorum Summary --
Quorum votes possible: 3
Quorum votes needed: 2
Quorum votes present: 3
-- Quorum Votes by Node --
Node Name Present Possible Status
--------- ------- -------- ------
Node votes: suninfo 1 1 Online
Node votes: pelican 1 1 Online
-- Quorum Votes by Device --
Device Name Present Possible Status
----------- ------- -------- ------
Device votes: /dev/did/rdsk/d4s2 1 1 Online
------------------------------------------------------------------
-- Device Group Servers --
Device Group Primary Secondary
------------ ------- ---------
Device group servers: ids-ds1 suninfo pelican
Device group servers: dbspace-ds1 suninfo pelican
Device group servers: sbspace-ds1 suninfo pelican
-- Device Group Status --
Device Group Status
------------ ------
Device group status: ids-ds1 Online
Device group status: dbspace-ds1 Online
Device group status: sbspace-ds1 Online
------------------------------------------------------------------
------------------------------------------------------------------
# scrgadm -p
Res Type name: SUNW.LogicalHostname
Res Type description: Logical Hostname Resource Type
Res Type name: SUNW.SharedAddress
Res Type description: HA Shared Address Resource Type
# scdidadm -L
1 suninfo:/dev/rdsk/c0t0d0 /dev/did/rdsk/d1
```

```
2 suninfo:/dev/rdsk/c0t1d0 /dev/did/rdsk/d2
3 suninfo:/dev/rdsk/c0t6d0 /dev/did/rdsk/d3
4 pelican:/dev/rdsk/c1t3d0 /dev/did/rdsk/d4
4 suninfo:/dev/rdsk/c1t3d0 /dev/did/rdsk/d4
5 pelican:/dev/rdsk/c1t4d0 /dev/did/rdsk/d5
5 suninfo:/dev/rdsk/c1t4d0 /dev/did/rdsk/d5
6 pelican:/dev/rdsk/c1t5d0 /dev/did/rdsk/d6
6 suninfo:/dev/rdsk/c1t5d0 /dev/did/rdsk/d6
7 pelican:/dev/rdsk/c2t3d0 /dev/did/rdsk/d7
7 suninfo:/dev/rdsk/c2t3d0 /dev/did/rdsk/d7
8 pelican:/dev/rdsk/c2t4d0 /dev/did/rdsk/d8
8 suninfo:/dev/rdsk/c2t4d0 /dev/did/rdsk/d8
9 pelican:/dev/rdsk/c2t5d0 /dev/did/rdsk/d9
9 suninfo:/dev/rdsk/c2t5d0 /dev/did/rdsk/d9
10 pelican:/dev/rdsk/c0t0d0 /dev/did/rdsk/d10
11 pelican:/dev/rdsk/c0t1d0 /dev/did/rdsk/d11
12 pelican:/dev/rdsk/c0t6d0 /dev/did/rdsk/d12
```

You will notice that logical disks 1–3 and 10–12 look like the same devices but have different logical device names. These are local devices and not part of the shared arrays.

In this example, we will create a simple highly available IBM IDS instance along with an associated logical hostname/IP address. Our IBM IDS instance name (dbservername) is dbserver_tcp with dbserver_shm as a dbserveralias and the logical hostname in /etc/hosts is dbserver. In the following examples, the `-v` option shows the actual `scrgadm` commands that are run by the utilities and the occasional use of `|grep .` is just to remove blank lines to shorten output.

First, use the startids utility to register the instance with the Sun Cluster 3.1 infrastructure (user input is bold):

```
suninfo # cd /opt/IBMids/util
./startids -x -v
Registering resource type <IBM.ids>...
scrgadm -a -t IBM.ids
done.
Creating failover resource group <ids-harg>...
scrgadm -a -g ids-harg
done.
Enter the logical/failover hostname: dbserver
Enter the nafo list (i.e. nafo0@primary,nafo0@backup):
nafo0@suninfo,nafo0@pelican
Creating logical host resource <dbserver>...
scrgadm -a -L -g ids-harg -l dbserver -n nafo0@suninfo,nafo0@pelican
done.
Do you want to use HAStoragePlus(y/n)? y
Which do you want to enter for HAStoragePlus devices?
1 Global Devices/Device Groups
2 Local File Systems
3 Both
4 Don't Know
Enter the number for your choice: 3
One moment...
Enter the Global Device or Device Group one at a time and press return.
To finish just press return.
Enter device: ids-ds1
Enter device:
Is the following list of devices correct?
ids-ds1
Is the list correct(y/n)? y
Enter the file system mount points one at a time and press return.
To finish just press return.
```

```
Enter file system: /global/informix
Enter file system:
Is the following list of file systems correct?
/global/informix
Is the list correct(y/n)? y
Creating HAStoragePlus resource <ids-hasp>...
scrgadm -a -t SUNW.HAStoragePlus -g ids-harg -j ids-hasp -x
GlobalDevicePaths=ids-ds1 -x FilesystemMountPoints=/global/informix
done.
Enter the full directory path for the IBM IDS software:
/global/informix/ids940
Select your onconfig file from the following list.
1 onconfig.940
2 onconfig.demo
3 onconfig.std
4 Other
Enter the number for your onconfig file: 1
Found file "sqlhosts" in "/global/informix/ids940/etc".
Do you want to use this file for communications info (y/n)? y
Found "1528/tcp" entry in /etc/services based upon "dbserver"
     hostname.
Do you want to use this value for the portlist(y/n)? y
Choose the value for INFORMIXSERVER from the following list.
1 dbserver_shm
2 dbserver_tcp
3 dbserver_imc
Enter the number for the INFORMIXSERVER value: 1
The default for START_TIMEOUT is 300 seconds.
Do you want to use this value(y/n)? y
The default for STOP_TIMEOUT is 300 seconds.
Do you want to use this value(y/n)? y
The default for PROBE_TIMEOUT is 30 seconds.
Do you want to use this value(y/n)? y
```

```
The following values will be used to install the Informix agent.
INFORMIXDIR=/global/informix/ids940
INFORMIXSERVER=dbserver_shm
ONCONFIG=onconfig.940
INFORMIXSQLHOSTS=/global/informix/ids940/etc/sqlhosts
PORT_LIST=1528/tcp,1529/tcp
START_TIMEOUT=300
STOP_TIMEOUT=300
PROBE_TIMEOUT=30
Are these values what you want to use(y/n)? y
Bring resource group <ids-harg> online, for HAStoragePlus
     dependency...
scswitch -Z -g ids-harg
done.
Creating resource <ids-hars> for the resource type <IBM.ids>...
scrgadm -a -j ids-hars -g ids-harg -t IBM.ids -y
PORT_LIST=1528/tcp,1529/tcp -y NETWORK_RESOURCES_USED=dbserver -y
RESOURCE_DEPENDENCIES=ids-hasp -x CONFDIR_LIST="" -x
     START_TIMEOUT=300
-x STOP_TIMEOUT=300 -x PROBE_TIMEOUT=30 -x TRACE_AGENT=false -x
     INFORMIXDIR=/global/informix/ids940
-x INFORMIXSERVER=dbserver_shm
-x ONCONFIG=onconfig.940
-x INFORMIXSQLHOSTS=/global/informix/ids940/etc/sqlhosts
done.
Not starting service. To start, enter the following command.
scswitch -Z -g ids-harg
Use the scstat command to determine the status of the cluster.
suninfo # scstat|grep .
------------------------------------------------------------------
-- Cluster Nodes --
Node name Status
--------- ------
Cluster node: suninfo Online
Cluster node: pelican Online
------------------------------------------------------------------
-- Cluster Transport Paths --
```

```
Endpoint Endpoint Status
-------- -------- ------
Transport path: suninfo:hme2 pelican:hme2 Path online
Transport path: suninfo:hme1 pelican:hme1 Path online
---------------------------------------------------------------------
-- Quorum Summary --
Quorum votes possible: 3
Quorum votes needed: 2
Quorum votes present: 3
-- Quorum Votes by Node --
Node Name Present Possible Status
--------- ------- -------- ------
Node votes: suninfo 1 1 Online
Node votes: pelican 1 1 Online
-- Quorum Votes by Device --
Device Name Present Possible Status
----------- ------- -------- ------
Device votes: /dev/did/rdsk/d4s2 1 1 Online
---------------------------------------------------------------------
-- Device Group Servers --
Device Group Primary Secondary
------------ ------- ---------
Device group servers: ids-ds1 suninfo pelican
Device group servers: dbspace-ds1 suninfo pelican
Device group servers: sbspace-ds1 suninfo pelican
-- Device Group Status --
Device Group Status
------------ ------
Device group status: ids-ds1 Online
Device group status: dbspace-ds1 Online
Device group status: sbspace-ds1 Online
---------------------------------------------------------------------
-- Resource Groups and Resources --
Group Name Resources
---------- ---------
Resources: ids-harg dbserver ids-hasp ids-hars
-- Resource Groups --
Group Name Node Name State

---------- --------- -----
```

```
Group: ids-harg suninfo Online
Group: ids-harg pelican Offline
-- Resources --
Resource Name Node Name State Status Message
------------- --------- ----- --------------
Resource: dbserver suninfo Online Online -
LogicalHostname online.
Resource: dbserver pelican Offline Offline
Resource: ids-hasp suninfo Online Online
Resource: ids-hasp pelican Offline Offline
Resource: ids-hars suninfo Offline Offline
Resource: ids-hars pelican Offline Offline
```

The result of the startids processing is that the appropriate IBM IDS resources and resource groups are created and registered with SC3.1.

Note the naming convention of the resources and resource groups and their structure. The start, stop and remove utilities have a rigid structure and are provided for reference rather than agent maintenance. While they can be used to create and maintain a simple agent installation, they will probably not be sufficient for most agent setups.

After verifying that the agent registered properly, you can start the instance with the `startids` command:

```
suninfo # /opt/IBMids/util/startids -v
Bringing the resource group <ids-harg> online...
scswitch -Z -g ids-harg
done.
```

The result of the online processing is that the ids-harg resource group and its associated resources are online and controlled by Sun Cluster 3.1. You should see the resources brought online for the appropriate node (For example, on the physical hostname suninfo, you should see that it hosts the logical hostname resource dbserver as well as the ids-hars and ids-hasp resources). There are two additional supplied utilities to discuss: stopids and removeids.

At some point, you may wish to take an IBM IDS instance offline to perform maintenance on the system. This process may require you to bring the database engine down for an extended period of time. Directly issuing the `onmode -uky` command will be ineffective, because SC3.1 will interpret the absence of resources caused by the successful completion of the `onmode` command as a failure and attempt to restart the appropriate database instance resources. Instead, to accomplish this task, you must offline the resources as follows:

```
suninfo # /opt/IBMids/util/stopids -v
Offlining resource group <ids-harg> on all nodes ...
scswitch -z -g ids-harg -h ""
done.
Disabling the resource <ids-hars> ...
scswitch -n -j ids-hars
done.
Disabling the resource <ids-hasp> ...
scswitch -n -j ids-hasp
done.
Disabling the resource <dbserver> ...
scswitch -n -j dbserver
done.
```

We can verify the command's completion with `scstat`:

```
# scstat|grep .
------------------------------------------------------------------
-- Cluster Nodes --
Node name Status
--------- ------
Cluster node: suninfo Online
Cluster node: pelican Online
------------------------------------------------------------------
-- Cluster Transport Paths --
Endpoint Endpoint Status
-------- -------- ------
Transport path: suninfo:hme2 pelican:hme2 Path online
Transport path: suninfo:hme1 pelican:hme1 Path online
------------------------------------------------------------------
```

```
-- Quorum Summary --
Quorum votes possible: 3
Quorum votes needed: 2
Quorum votes present: 3
-- Quorum Votes by Node --
Node Name Present Possible Status
--------- ------- -------- ------
Node votes: suninfo 1 1 Online
Node votes: pelican 1 1 Online
-- Quorum Votes by Device --
Device Name Present Possible Status
----------- ------- -------- ------
Device votes: /dev/did/rdsk/d4s2 1 1 Online
------------------------------------------------------------------
-- Device Group Servers --
Device Group Primary Secondary
------------ ------- ---------
Device group servers: ids-ds1 suninfo pelican
Device group servers: dbspace-ds1 suninfo pelican
Device group servers: sbspace-ds1 suninfo pelican
-- Device Group Status --
Device Group Status
------------ ------
Device group status: ids-ds1 Online
Device group status: dbspace-ds1 Online
Device group status: sbspace-ds1 Online
------------------------------------------------------------------
-- Resource Groups and Resources --
Group Name Resources
---------- ---------
Resources: ids-harg dbserver ids-hasp ids-hars
-- Resource Groups --
Group Name Node Name State
---------- --------- -----
Group: ids-harg suninfo Offline
Group: ids-harg pelican Offline
```

```
-- Resources --
Resource Name Node Name State Status Message
------------ --------- ----- --------------
Resource: dbserver suninfo Offline Offline -
LogicalHostname offline.
Resource: dbserver pelican Offline Offline
Resource: ids-hasp suninfo Offline Offline
Resource: ids-hasp pelican Offline Offline
Resource: ids-hars suninfo Offline Offline
Resource: ids-hars pelican Offline Offline
```

The result of this action, as shown from the `scstat` output, is that all resources are now offline. No resources associated with the instance ids-harg should be active on either node, and process monitoring has stopped. At this point, SC3.1 will not take any action to protect this instance if it is brought online manually (that is, via `oninit`) and if a failure occurs. Note that at this point, trying to start the instance will fail since the logical hostname is offline. Also, if you are using a failover file system, it is now unmounted, so the `oninit` command will not be found. Alternately, the following command can be used to take the database resource into an unmanaged state:

```
scswitch -n -M -j ids-hars
```

Note: Care must be taken for any changes made in this state. For example, changes to /etc/hosts may affect the logical host resource while it is still enabled, and any changes that are made to the database instance are reflected in the appropriate extension properties.

Let us assume that it is decided to remove this instance permanently from SC3.1 monitoring and control. For this task, you may use the removeids method. Note that this method merely interfaces with SC3.1 to perform these tasks; the instance itself is not dropped or removed.

```
suninfo # /opt/IBMids/util/removeids -v
Offlining the resource group <ids-harg> ...
scswitch -z -g ids-harg -h ""
done.
Disabling the resource <ids-hars> ...
scswitch -n -j ids-hars
done.
Removing the resource <ids-hars> ...
scrgadm -r -j ids-hars
done.
Disabling the resource <ids-hasp> ...
scswitch -n -j ids-hasp
done.
Removing the resource <ids-hasp> ...
scrgadm -r -j ids-hasp
done.
Disabling the network resource <dbserver> ...
scswitch -n -j dbserver
done.
Removing the resource <dbserver> ...
scrgadm -r -j dbserver

done.
Unmanaging the resource group <ids-harg>
scswitch -u -g ids-harg
done.
Removing the resource group <ids-harg>
scrgadm -r -g ids-harg
done.
Removing the resource type <IBM.ids>
scrgadm -r -t IBM.ids
done.
```

If there is any difficulty with registration, un-registration or moving resources online or offline, you may need to consult the system error log (typically in /var/adm/messages).

Another advanced option for debugging is to call the startdb, stopdb and probedb methods directly and observe their behavior. For this type of testing, the logical hostname must be online and the rest of the resource group must be offline.

As stated before, these three utilities are provided for reference rather than cluster manipulation. The use of the `scrgadm` and `scswitch` commands provide for greater flexibility in configuring the IBMids agent. For instance, multiple IBM IDS instances can be started in one resource group or multiple resource groups with multiple logical hostnames being created for each instance. The layout depends on the requirements of the application driving the server.

Customizing SC3.x resource properties for IBM IDS

The IBMids agent is registered with SC3.x as a defined resource type. Each IBM IDS instance is registered with SC3.x as a resource. For resource properties provided by SC3.x, you can change specific attributes via the Sun Cluster administrative commands.

The resource type properties and the resource properties are declared in the Resource Type Registration (RTR) file. The resource property declarations follow the resource type declarations in the RTR file. Entries begin with an open curly bracket and end with a closed curly bracket.

You can change resource properties by registering the resource with a modified RTR file—/opt/IBMids/etc/IBM.ids—but the preferred method is through the SC3.x administrative command `scrgadm`, which can be used to add a new configuration or modify an existing one.

Start_timeout, Stop_timeout and Probe_timeout extension properties

The <method>_timeout properties set a value in seconds after which the SC3.x resource group manager (RGM) concludes that invocation of the method has failed. The MIN value for the <method>_timeout properties is set to 30 seconds. This prevents administrators from setting shorter timeouts, which do not improve switchover/failover performance and can lead to undesired RGM actions (false failovers, node reboot or setting the resource to a STOP_FAILED state, requiring operator intervention).

Setting the <method>_timeout value too short leads to a decrease in overall availability of the IBM IDS data service.

Monitor_retry_count and Monitor_retry_interval extension properties

The number of retries to be performed within a retry_interval before the SC3.x RGM concludes that the resource cannot be successfully started on the current node is called the retry_count. The retry_interval is measured in minutes as opposed to seconds for other time variables.

IBM IDS environment variable extension properties

The IBM IDS instance-specific environment variables INFORMIXDIR, INFORMIXSERVER, INFORMIXSQLHOSTS, ONCONFIG, DB_LANG, SERVER_LOCALE, CLIENT_LOCALE along with LD_LIBRARY_PATH are all stored as extension properties for each resource associated with an IBM IDS instance. The INFORMIXDIRINFORMIXSERVER, INFORMIXSERVER, INFORMIXSQLHOSTS and ONCONFIG extension properties are mandatory for all instances of the IBM IDS agent, while the others are provided for instance-specific needs.

Each of these properties can be changed when the agent instance is disabled, and care must be taken when changing an IBM IDS instance that these properties are also changed accordingly. The property values can be added (`-a`) or changed (`-c`) using the SC3.x `scrgadm` command. For a complete description of customizing the resource properties, please refer to the *Sun Cluster 3.1 Data Services Installation and Configuration Guide*.

Cluster verification testing

An important aspect of a highly available cluster is testing. The purpose of this testing is to gain some confidence that the highly available cluster will function as envisioned for various failure scenarios. The following set of scenarios is recommended as a minimum for cluster testing and verification. Note that to ensure that the cluster continues to function as expected, it is recommended that time be taken to ensure these tests are run whenever a configuration change is made to the cluster instance.

Test 1. This test will perform Sun Cluster 3.1 management commands to ensure that the ids-harg instance can be controlled correctly. First, verify that the instance is accessible from the clients (or locally) and that various database commands complete successfully (for example, `create database`). Take the ids-hars resource and the dbserver resource offline via the following:

```
scswitch -n -j ids-hars
scswitch -n -j dbserver
```

Observe that the ids-hars resource and the logical hostname dbserver appear offline on all nodes in the cluster. From the perspective of a client of this instance, the existing connections are closed.

Test 2. To return the resources to their previous states, bring them online with the following SC3.1 commands in the reverse order that they were taken offline:

```
scswitch -e -j dbserver
scswitch -e -j ids-hars
```

IBM IDS clients can now reconnect to the server.

Test 3. Test the failover of the IBM IDS instance and associated resources from suninfo to pelican. At this point, the cluster is again at its initial state. With the following command, you can move the resources contained within the resource group ids-harg over to the backup node:

```
scswitch -z -g ids-harg -h pelican
```

When complete, the relevant resources will be online on the backup node. You should now see that the ids-hars and dbserver resources along with the ids-harg resource group are now online on the backup machine, that is, pelican. Verify that this is the case by executing the `scstat` command.

Test 4. This test will verify the failover capabilities of the Sun Cluster 3.1 software itself. Bring the resources back into their initial state (have the ids-harg resource group hosted on suninfo) using the following:

```
scswitch -z -g ids-harg -h suninfo
```

Once the instance and its associated resources are hosted on the suninfo machine, perform a power-off operation on that physical node. This action will cause the internal heartbeat mechanism to detect a physical node failure, and the ids-harg resource group will be restarted on the surviving node, pelican.

Verify that the results are identical to those seen in Test 3 (the ids-harg resource group should be hosted on pelican and the clients should behave similarly in both cases).

Test 5. The purpose of this test is to verify the application level failover in a non-controlled scenario by randomly disabling the process identity numbers (PIDs) of the IDS server and making sure that the instance is restarted on the same node.

Bring the cluster back into its initial state. In this test, verify that the software monitoring is working as expected. To perform this test, issue commands as follows:

```
ps -ef | grep oninit
```

Get the PID of the process group leader, which will be the oninit process whose parent PID is 1.

```
kill -11 <PID>
```

This may take a few moments because it will cause the server to assert fail. The Sun Cluster 3.1 monitor should detect that a required process is not running and attempt to restart the instance on the same node. Verify that this does occur. The client connections should experience a brief delay in service while the restart process continues.

Note that a large number of distinct testing scenarios can be executed, limited only by your resources and imagination. The above sets are meant as a minimum that should be run to test that the cluster is functioning correctly.

Test 6. The purpose of this test is to demonstrate that no failover of the Informix service occurs when the server is down because of a controlled shutdown. Shut down the server using `onmode -ky` and verify that the server comes back online on the same node.

Test 7. When the database server is blocked due to backlogs, make sure that the failover does not happen to the backup node. Fill up the logical logs without doing a backup while transactions are happening. After some time, the server will be blocked. Verify that the server waits until the logical log backup is done and no failover occurs from the current node to the backup node.

Test 8. Verify that no failover happens when the server is blocked because of a long transaction. When a query causes a long transaction, the server will be blocked and the cluster will not have to perform a failover.

Test 9. Perform transactions on the server resulting in more dirty pages by keeping the checkpoint interval high. Crash the node before the checkpoint and verify that the transactions are rolled back and rolled forward accordingly at the time of recovery of the instance during switchover on the backup node.

Test 10. Run client sessions against the database running on the cluster and verify the continuity of the client sessions during the multiple failover operations from node to node on the cluster. The client sessions during the failover time will lose the connection, but the subsequently started sessions should not see any issues because of the change in the node where the database instance is running.

Cluster topology

The IBMids agent fully supports all three cluster topology classes supported by Sun Cluster 3.x: Clustered pairs/idle standby, N+1 (or Star) and Pair+M.

Idle standby

Idle standby is the simplest highly available cluster topology. In this scenario, the primary machine is hosting the production database instance and associated resources. A second machine is idle and available to host the production database instance and associated resources should a failure occur on the primary machine. Note that the second machine can be idle or it can be running a workload (perhaps another IBM IDS instance) to maximize use of resources.

Clustered pairs

In the mutual takeover case, envision a cluster of N nodes as N/2 pairs of nodes. Node number n is responsible for failover support of node number n+1 (and vice versa), node number n+2 is responsible for failover support of node number n+3 (and vice versa) and so on until you reach the Nth node. Note that this scenario requires that N be an even number.

The advantages of this configuration are that in the normal (non-failure) case, all machines are hosting database resources and performing productive work. The primary disadvantage is that in the failure case (that is, during the time period after one hardware resource has failed but before this hardware resource has been repaired), one node is required to support on average twice the workload of any other physical node. Should failures be relatively rare and their duration short, this is a reasonable scenario.

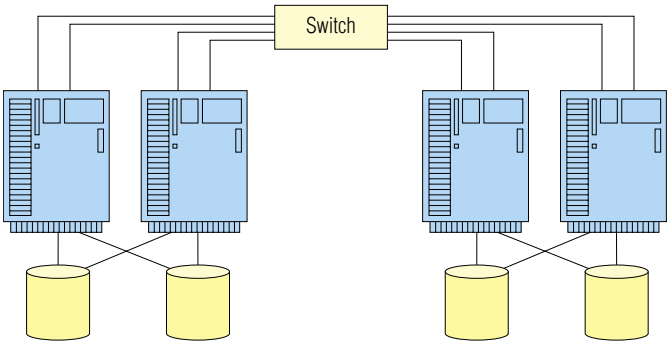

Figure 2: Diagram of clustered pair topology

N+1 takeover

This case relies on an N node cluster, with one defined node as the standby for all N nodes. The advantage of this scenario is that no performance degradation occurs during the failure case. The primary disadvantage is that approximately 1/(N+1) of the aggregate physical computing resource lies unused during the normal operation (presumably overwhelmingly common) case.

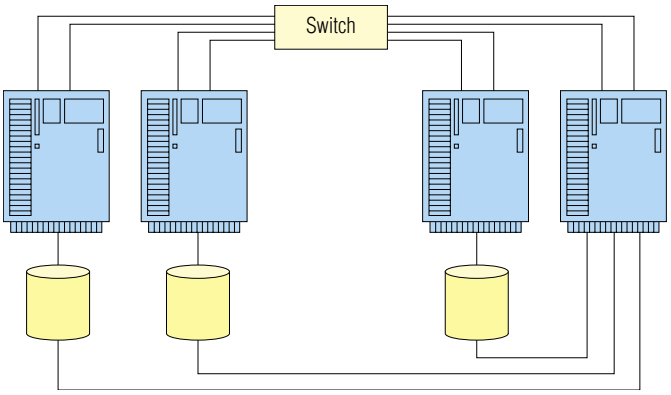

Figure 3: Diagram of N+1 takeover cluster topology

Pair+M (N+M)

This case relies on an N node cluster, with M defined nodes as hot standbys for each of the N nodes. The prime advantage of this configuration is that the environment is fully redundant, in the sense that up to N-1 node failures can be tolerated while still maintaining full database access (subject of course to increased query response times because of capacity constraints when there are less than N nodes in the cluster). In this way, IBM IDS used in conjunction with SC3.x helps ensure full database software redundancy and is most appropriate for environments requiring the highest degree of availability. The primary disadvantage of this configuration is potentially non-local accesses to data on the global device, which will be discussed later in the paper.

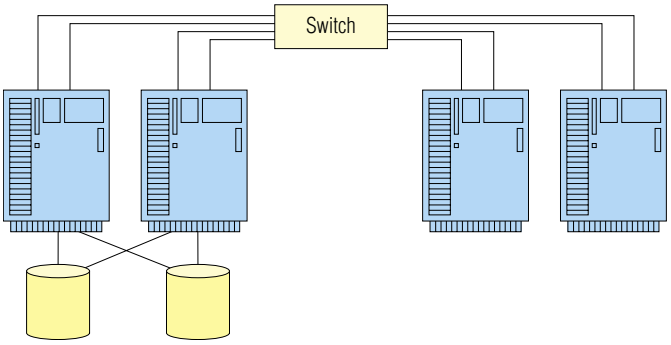

Figure 4: Diagram of Pair+M cluster topology

Optimizing cluster file system performance

With the introduction of the failover file system (HAStoragePlus) to SC3.x, it is recommended that FFS be used in place of cluster file system (CFS) for I/O-intensive applications. However, if you still prefer CFS, then follow these guidelines:

Whenever possible, collocate the IBM IDS instance with the corresponding data and use locally hosted global file systems. If a global device is dual-hosted in the cluster, let the same node be the primary resource group and its corresponding device group; the active I/O path will then be locally accessed. During a failover, the backup node (secondary) will be promoted to the status of a primary for the resource and device groups that have been switched over.

As a general guideline, Pair+M topology should be avoided because the additional M nodes will likely be configured so that access to data will be remote through one of the global I/O paths provided by the primary of the dual-hosted device. The disk I/O performance will be impacted by the overhead associated with acquiring the data remotely.

Sun HAStorage data service is an agent that can be used in your configuration to control the primary for a resource group device. It ensures that when a resource group is moved to another node, the associated device groups are switched to primary on that node. For more information on HAStorage and configuration, please see "Synchronizing the Startups Between Resource Groups and Disk Device Groups." To add this agent to our example systems perform the following:

```
scrgadm -a -t SUNW.HAStorage
scrgadm -a -j has-rs -g ids-harg -t SUNW.HAStorage -x \
ServicePaths=ids-ds1,dbspace-ds1,sbspace-ds1 -x AffinityOn=True
scswitch -e -j has-rs
scrgadm -c -j ids-hars -y Resource_Dependencies=has-rs
```

Now whenever a failover/switchover occurs, the device groups will follow the resource group to a new primary machine.

Recommended CFS mount options

The required mount options for cluster file systems are as follows:

```
global,logging
```

If a file system is being used solely for database storage and does not contain any binary files that will be executed, the following option should also be used:

```
forcedirectio
```

Typical entries in the /etc/vfstab file would be as follows for the Informix binaries file system:

```
/dev/vx/dsk/scdg2/vol05 /dev/vx/rdsk/scdg2/vol05 /global/scdg2/
    scdg2 ufs 2 yes global,logging
```

and for a database storage file system:

```
/dev/vx/dsk/scdg2/vol05 /dev/vx/rdsk/scdg2/vol05 /global/scdg2/
    scdg2 ufs 2 yes global,logging,forcedirectio
```

The global mount option simply identifies this mount point as a cluster file system. The logging option is always used implicitly for a CFS file system, regardless of whether it is directly specified on the mount line; however, it is recommended that the option be explicitly stated for clarity. The forcedirectio option only applies to CFS and essentially allows file system I/O to bypass Solaris kernel page cache. This option is especially beneficial for IBM IDS file system I/O, because IDS performs all required I/O caching and derives no further benefit from additional caching in the Solaris kernel. The reduction in CPU path length can be significant with the use of direct I/O, so its use is encouraged.

One additional CFS mount option merits mention. The syncdir option only applies to global UFS mounts. Whether it is utilized depends on the relative importance placed on error semantics versus absolute performance. Briefly, the syncdir option is relevant when a file is in process of being extended. If it is set, notification of a full file system will be returned even if a failover of the primary occurs during the write call itself. Although this is ideal error semantics behavior, it must be weighed against the performance degradation, which can be significant in the case of writes that occur on the non-primary node.

If all I/O should occur only local to the primary host, this option should be enabled. Otherwise, it is recommended that this option not be enabled because of performance penalty and the exceedingly small probability of a file system full occurring during a disk group failover.

Finally, for customers desiring the maximum performance possible with a globally available device, the use of raw partitions is recommended.

Failover time and how to minimize it

To reduce the service interruption time, it is important to understand the discrete components which make up failover:

- *Time for SC3.x to detect and respond to failure*
- *Time for IBM IDS to recover from failure*
- *Time for TCP/IP clients to detect hostname switchover and reconnect*

In particular, you can influence and reduce the amount of time required for IBM IDS failure recovery with the following methods. An example of setting this parameter to allow this time window to be no longer than 10 seconds follows.

```
ndd -set /dev/tcp tcp_ip_abort_cinterval 10000
```

Also, you can reduce the time required for IBM IDS database restart recovery by employing standard techniques discussed in the *Administrator's Guide for Informix Dynamic Server* and the *Performance Guide for Informix Dynamic Server.*

Client issues

Applications that rely on a highly available IBM IDS instance must be able to reconnect in the event of a failover. Since the hostname and IP address of a logical host remain the same, there is no need to connect to a different hostname or re-catalog the database.

Consider a cluster with two machines and one IBM IDS instance. The IBM IDS instance will normally reside on one of the machines in the cluster. Clients of the HA instance will connect to the logical IP address (or hostname) of the logical host associated with the HA instance.

According to an HA client, there are two types of failovers. One type occurs if the machine hosting the HA instance crashes. The other type is when the HA instance is given an opportunity to shut down properly.

If a machine crashes and takes down the HA instance, existing connections and new connections to the database will hang. The connections hang because no machines on the network have the IP address that clients were using for the database. If the database is shut down properly, an `onmode -uky` command forces off the connections to the database and then shuts it down. In this case, existing database connections will receive an error because they were forced off.

During the failover, the logical IP address associated with the database is offline, either because the SC3.x software took it offline or because the machine that was hosting the logical host crashed. At this point, any new database connections will hang for a short period of time.

The logical IP address associated with the database is eventually brought up on another machine before IBM IDS is started. At this stage, a connection to the database will not hang, but will receive a communication error because IBM IDS has not yet been started on the system. Clients that were still connected to the database will also begin receiving communication errors at this stage. Although the clients still believe they are connected, the new machine just started hosting the logical IP address and has no knowledge of any existing connections. The connections are simply reset and the client receives a communication error. After a short time, IBM IDS will be started on the machine and a connection to the database will succeed. At this point, the database may be inconsistent and the clients may have to wait for it to be recovered.

Even though this sounds like a complicated and time-consuming set of events, it is actually quite simple. When designing an application for a highly available environment, it is not necessary to write special code for the stages at which database connections hang. The connections only hang for a short period of time while the Sun Cluster software moves the logical IP address. Any data service running on Sun Cluster will experience the same hanging connections during this stage. No matter how the database comes down, the clients will receive an error and will simply need to try to reconnect until successful.

From the client's perspective, it is as if the HA instance went down and was then brought back up on the same machine. In a controlled failover, it would appear to the client that it was forced off and that it could later reconnect to the database on the same machine. In an uncontrolled failover, it would appear to the client that the database server crashed and was soon brought back up on the same machine.

The use of keep-alives

On the server side, using TCP keep-alives protects the server from wasting system resources for a down (or network-partitioned) client. If those resources are not cleaned up (in a server that stays up long enough), eventually the wasted resources grow without bound as clients crash and reboot.

On the client side, using TCP keep-alives enables the client to be notified when a network address resource has failed over or switched over from one physical host to another. That transfer of the network address resource breaks the TCP connection. However, unless the client has enabled the keep-alive, it would not necessarily learn of the connection break if the connection happens to be quiescent at the time.

In Solaris, the tcp_keepalive feature institutes a series of probe messages from the host to the client. The host sends a series of messages at specified intervals to the client. If no response returns, the host closes the channel. As in many other implementations, the default interval is 7.2 million milliseconds (2 hours). You can use `ndd(1M)` to change the value of tcp_keepalive_interval. In order for its value to be set every time the system boots, you can add the following entry in /etc/init.d/inetinit script. The entry for setting tcp_keepalive_interval for one hour is as follows:

```
/usr/sbin/ndd -set /dev/tcp tcp_keepalive_interval 3600000
```

It is recommended that the value of tcp_keepalive_interval should not be set below 15 minutes. In Microsoft® Windows NT® and Windows® 2000 environments, the loss of a client/server connection is also determined by using keep-alive probes. The following registry entries control keep-alive probe packet parameters on computers running Windows environments. Changing registry parameters affects all TCP/IP stream connections on the system.

```
KeepAliveInterval
KeepAliveTime
TcpMaxDataRetransmissions
The above entries can be found in following registry location:
\HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\TCPIP\
    Parameters
```

Client retry

From the client's perspective, a failover appears to be a crash of the logical host followed by a fast reboot. In a controlled failover, it would appear to the client that it was forced off and that it could later reconnect to the database on the same machine.

In an uncontrolled failover, it would appear to the client that the database server crashed and was soon brought back up on the same machine. During a failover, it is recommended that client applications check the connection status and try to reconnect. Applications also can be implemented to notify the user that a long retry is in progress and enable the user to choose whether to continue.

Summary

We have discussed the design, implementation and verification of a highly available IBM Informix Dynamic Server cluster in a Sun Cluster 3.x environment. This combination produces an outstanding database management system with a high degree of availability, reliability and performance.

Appendix A: Example sqlhosts

```
#************************************************************
#
# Licensed Material - Property Of IBM
#
# "Restricted Materials of IBM"
#
# IBM Informix Dynamic Server
# (c) Copyright IBM Corporation 1996, 2004 All rights reserved.
#
# Title:   sqlhosts.demo
# Description:
#      sqlhosts file for IBM IDS documentation.
#
#************************************************************

dbservershm onipcshm dbserver no-op
dbservertcp ontlitcp dbserver online940
dbserverimc ontliimc dbserver online940mc
```

Appendix B: Example onconfig

```
#**************************************************************
#
# Licensed Material - Property Of IBM
#
# "Restricted Materials of IBM"
#
# IBM Informix Dynamic Server
# (c) Copyright IBM Corporation 1996, 2004 All rights reserved.
#
# Title:   onconfig.std
# Description: IBM Informix Dynamic Server Configuration
     Parameters
#
#**************************************************************
# Root Dbspace Configuration

ROOTNAME rootdbs940       # Root dbspace name
ROOTPATH /global/dbspace/rootdbs940
                          # Path for device containing root dbspace
ROOTOFFSET 0              # Offset of root dbspace into device (Kbytes)
ROOTSIZE 30000            # Size of root dbspace (Kbytes)
# Disk Mirroring Configuration Parameters
MIRROR 0                  # Mirroring flag (Yes = 1, No = 0)
MIRRORPATH                # Path for device containing mirrored root
MIRROROFFSET 0            # Offset into mirrored device (Kbytes)
# Physical Log Configuration
PHYSDBS rootdbs940        # Location (dbspace) of physical log
PHYSFILE 2000             # Physical log file size (Kbytes)
# Logical Log Configuration
LOGFILES 6                # Number of logical log files
LOGSIZE 2000              # Logical log size (Kbytes)
# Diagnostics
MSGPATH /global/informix/ids940/online.log
                          # System message log file path
CONSOLE /dev/console      # System console message path
ALARMPROGRAM /global/informix/ids940/etc/log_full.sh
                          # Alarm program path
TBLSPACE_STATS 1          # Maintain tblspace statistics
```

```
# System Archive Tape Device
TAPEDEV /dev/null          # Tape device path
TAPEBLK 16                 # Tape block size (Kbytes)
TAPESIZE 10240             # Maximum amount of data to put on tape (Kbytes)
# Log Archive Tape Device
LTAPEDEV /dev/null         # Log tape device path
LTAPEBLK 16                # Log tape block size (Kbytes)
LTAPESIZE 10240            # Max amount of data to put on log tape (Kbytes)
# Optical
STAGEBLOB                   # Informix Dynamic Server staging area
# System Configuration
SERVERNUM 1                 # Unique id corresponding to a OnLine instance
DBSERVERNAME dbservershm  # Name of default database server
DBSERVERALIASES dbservertcp,dbserverimc
                           # List of alternate dbservernames
NETTYPE tlitcp,1,,NET      # Configure poll thread(s) for nettype
NETTYPE tliimc,1,,NET      # Configure poll thread(s) for nettype
NETTYPE ipcshm,1,,CPU      # Configure poll thread(s) for nettype
DEADLOCK_TIMEOUT 60        # Max time to wait of lock in distributed env.
RESIDENT 0 # Forced residency flag (Yes = 1, No = 0)
MULTIPROCESSOR 0           # 0 for single-processor, 1 for multiprocessor
NUMCPUVPS 1                # Number of user (cpu) vps
SINGLE_CPU_VP 0            # If non-zero, limit number of cpu vps to one
NOAGE 0                    # Process aging
AFF_SPROC 0                # Affinity start processor
AFF_NPROCS 0               # Affinity number of processors
# Shared Memory Parameters
LOCKS 2000                 # Maximum number of locks
BUFFERS 5000               # Maximum number of shared buffers
NUMAIOVPS                  # Number of IO vps
PHYSBUFF 32                # Physical log buffer size (Kbytes)
LOGBUFF 32                 # Logical log buffer size (Kbytes)
CLEANERS 1                 # Number of buffer cleaner processes
SHMBASE 0x0A000000L        # Shared memory base address
SHMVIRTSIZE 8000           # initial virtual shared memory segment size
SHMADD 8192                # Size of new shared memory segments (Kbytes)
SHMTOTAL 0                 # Total shared memory (Kbytes). 0=>unlimited
CKPTINTVL 300              # Check point interval (in sec)
LRUS 8                     # Number of LRU queues
```

```
LRU_MAX_DIRTY 60    # LRU percent dirty begin cleaning limit
LRU_MIN_DIRTY 50    # LRU percent dirty end cleaning limit
TXTIMEOUT 300     # Transaction timeout (in sec)
STACKSIZE 32              # Stack size (Kbytes)
# System Page Size
# BUFFSIZE - OnLine no longer supports this configuration
# parameter.
# To determine the page size used by OnLine on your platform
# see the last line of output from the command, 'onstat -b'.
# Recovery Variables
# OFF_RECVRY_THREADS:
# Number of parallel worker threads during fast recovery or an
# offline restore.
# ON_RECVRY_THREADS:
# Number of parallel worker threads during an online restore.
OFF_RECVRY_THREADS 10       # Default number of offline worker threads
ON_RECVRY_THREADS 1         # Default number of online worker threads
# Data Replication Variables
DRINTERVAL 30               # DR max time between DR buffer flushes (in sec)
DRTIMEOUT 30                # DR network timeout (in sec)
DRLOSTFOUND /global/informix/ids940/etc/dr.lostfound
                            # DR lost+found file path
# CDR Variables
CDR_EVALTHREADS 1,2         # evaluator threads (per-cpu-vp,additional)
CDR_DSLOCKWAIT 5            # DS lockwait timeout (seconds)
CDR_QUEUEMEM 4096           # Maximum amount of memory for any CDR queue
     (Kbytes)
CDR_NIFCOMPRESS 0           # Link level compression (-1 never, 0 none,
     9 max)
CDR_SERIAL 0                # Serial Column Sequence
CDR_DBSPACE                 # dbspace for syscdr database
CDR_QHDR_DBSPACE            # CDR queue dbspace (default same as catalog)
CDR_QDATA_SBSPACE           # CDR queue smart blob space
CDR_QDATA_SBFLAGS 0         # Log/no-log (default no log)
# Backup/Restore variables
BAR_ACT_LOG /global/informix/ids940/bar_act.log
                            # ON-Bar Log file - not in /tmp please
BAR_DEBUG_LOG /global/informix/ids940/bar_dbug.log
                            # ON-Bar Debug Log - not in /tmp please
```

```
BAR_MAX_BACKUP 0
BAR_RETRY 1
BAR_NB_XPORT_COUNT 10
BAR_XFER_BUF_SIZE 31
RESTARTABLE_RESTORE ON
BAR_PROGRESS_FREQ 0
# Informix Storage Manager variables
ISM_DATA_POOL ISMData
ISM_LOG_POOL ISMLogs
# Read Ahead Variables
RA_PAGES                    # Number of pages to attempt to read ahead
RA_THRESHOLD                # Number of pages left before next group
# DBSPACETEMP:
# OnLine equivalent of DBTEMP for SE. This is the list of dbspaces
# that the OnLine SQL Engine will use to create temp tables etc.
# If specified it must be a colon separated list of dbspaces that
# exist when the OnLine system is brought online. If not specified,
# or if all dbspaces specified are invalid, various ad hoc queries
# will create temporary files in /tmp instead.
DBSPACETEMP # Default temp dbspaces
# DUMP*:
# The following parameters control the type of diagnostics
# information which is preserved when an unanticipated error
# condition (assertion failure) occurs during OnLine operations.
# For DUMPSHMEM, DUMPGCORE and DUMPCORE 1 means Yes, 0 means No.
DUMPDIR /tmp                # Preserve diagnostics in this directory
DUMPSHMEM 1                 # Dump a copy of shared memory
DUMPGCORE 0                 # Dump a core image using 'gcore'
DUMPCORE 0                  # Dump a core image (Warning: this aborts
OnLine)
DUMPCNT 1                   # Number of shared memory or gcore dumps for
# a single user's session
FILLFACTOR 90               # Fill factor for building indexes
# method for OnLine to use when determining current time
USEOSTIME 0                 # 0: use internal time(fast),
                            # 1: get time from OS(slow)
```

```
# Parallel Database Queries (pdq)
MAX_PDQPRIORITY 100         # Maximum allowed pdqpriority
DS_MAX_QUERIES              # Maximum number of decision support queries
DS_TOTAL_MEMORY             # Decision support memory (Kbytes)
DS_MAX_SCANS 1048576        # Maximum number of decision support scans
DATASKIP                    # List of dbspaces to skip
# OPTCOMPIND
# 0 => Nested loop joins will be preferred (where possible) over
# sortmerge joins and hash joins.
# 1 => If the transaction isolation mode is not "repeatable read",
# optimizer behaves as in (2) below. Otherwise it behaves as in
# (0) above.
# 2 => Use costs regardless of the transaction isolation mode.
# Nested loop joins are not necessarily preferred. Optimizer bases
# its decision purely on costs.
OPTCOMPIND 2                # To hint the optimizer
DIRECTIVES 1                # Optimizer DIRECTIVES ON (1/Default) or OFF (0)
ONDBSPACEDOWN 2             # Dbspace down option: 0 = CONTINUE,
                            # 1 = ABORT, 2 = WAIT
OPCACHEMAX 0                # Maximum optical cache size (Kbytes)
# HETERO_COMMIT (Gateway participation in distributed transactions)
# 1 => Heterogeneous Commit is enabled
# 0 (or any other value) => Heterogeneous Commit is disabled
HETERO_COMMIT 0
SBSPACENAME                 # Default smartblob space name - this is
# where blobs go if no sbspace is specified when the smartblob is
# created. It is also used by some datablades as the location to
# put their smartblobs.
SYSSBSPACENAME              # Default smartblob space for use by the
# Informix Server. This is used primarily for Informix Server
# system statistics collection.
BLOCKTIMEOUT 3600           # Default timeout for system block
SYSALARMPROGRAM /global/informix/ids940/evidence.sh
                            # System Alarm program path
# Optimization goal: -1 = ALL_ROWS(Default), 0 = FIRST_ROWS
    OPT_GOAL -1
ALLOW_NEWLINE 0             # embedded newlines(Yes = 1, No = 0 or
# anything but 1)
#
```

```
# The following are default settings for enabling Java in the
# database.
# Replace all occurrences of /usr/informix with the value of
# $INFORMIXDIR.
#VPCLASS jvp,num=1          # Number of JVPs to start with
JVPJAVAHOME /global/informix/ids940/extend/krakatoa/jre/ # JRE
installation root directory
JVPHOME /global/informix/ids940/extend/krakatoa # Krakatoa
installation directory
JVPPROPFILE /global/informix/ids940/extend/krakatoa/.jvpprops #
     JVP
property file
JVPLOGFILE /global/informix/ids940/jvp.log # JVP log file.
JDKVERSION 1.3 # JDK version supported by this server
# The path to the JRE libraries relative to JVPJAVAHOME
JVPJAVALIB /lib/sparc/
# The JRE libraries to use for the Java VM
JVPJAVAVM hpi:server:verify:java:net:zip:jpeg
# use JVPARGS to change Java VM configuration
#To display jni call
#JVPARGS -verbose:jni
# Classpath to use upon Java VM start-up (use _g version for
     debugging)
# JVPCLASSPATH
/usr/informix/extend/krakatoa/krakatoa_g.jar:/usr/informix/
     extend/krakatoa/jdbc_g.jar
JVPCLASSPATH
/global/informix/ids940/extend/krakatoa/krakatoa.jar:/global/
     informix/ids940/extend/krakatoa/jdbc.jar
```

Appendix C: Error messages

Environment Error messages:
4010: No oninit executable found in "INFORMIXDIR/bin"!
4020: INFORMIXDIR is not set correctly or does not point to the correct or mounted directory.
4030: This does not appear to be a valid onconfig file.
4040: "INFORMIXSERVER" not found in "INFORMIXDIR/etc/ONCONFIG"!
4050: "INFORMIXSERVER" not found in "INFORMIXSQLHOSTS"!
4060: File "ONCONFIG" not found in "INFORMIXDIR/etc"!
4070: File "INFORMIXSQLHOSTS" not found!
Database start Notice messages:
2100: Starting database instance "INFORMIXSERVER".
2110: Instance already started for "INFORMIXSERVER".
2120: Removing remnant IPC facilities for a previously failed "INFORMIXSERVER", instance.
2130: Waiting for On-Line mode, current status is "status".
2140: Database instance "INFORMIXSERVER" successfully started.
Database start Error messages:
4101: Failed to get resource group name for ids.
4102: Failed to get resource name for ids.
4103: Failed to form the start command for ids.
4104: Failed to start ids, because "cmd" failed.
4110: Initialization failed for "INFORMIXSERVER", check online log for details.
4120: Database instance "INFORMIXSERVER" did not come ON-LINE in allotted time, last status = "status". Contact your DBA.
Database stop Notice messages:
2200: Shutting down database instance "INFORMIXSERVER".
2210: Database instance "INFORMIXSERVER", already shutdown.
2220: Database instance "INFORMIXSERVER", successfully shutdown.
Database stop Error messages:
4201: Failed to get resource group name for ids.
4202: Failed to get resource name for ids.
4203: Failed to take the resource out of PMF control.
4204: Failed to form the stop command for ids.
4205: The stop command "cmd" failed to stop ids.
4206: Failed to stop ids
4210: Database instance "INFORMIXSERVER", did not shutdown in allotted time. Contact your DBA.

```
Database probe Notice messages:
2300: Probing database instance "INFORMIXSERVER".
2310: Database instance "INFORMIXSERVER" blocked on "blocked"
     event.
Values for blocked: CKPT, LONGTX, DBS_DROP, DDR, CKPT ARCHIVE,
     ARCHIVE, LBU
2320: Database instance "INFORMIXSERVER" in "status" mode.
Database probe Error messages:
4310: Database instance "INFORMIXSERVER" is down!
4320: Database instance "INFORMIXSERVER" blocked on "blocked"
     event. Contact Your DBA!
Values for blocked: MEDIA_FAILURE
4330: Database instance "INFORMIXSERVER" blocked on "blocked"
     event. Contact your DBA!"
Values for blocked: HANG_SYSTEM
4399: Database instance "INFORMIXSERVER" in unknown state
     "statusline". Contact your DBA!
Agent validation Error messages:
5000: Failed to retrieve the extension property "property":
     "errmsg".
5001: Cannot access ids start command : "errmsg"
5002: ids start command does not have execute permissions: "errmsg"
5003: Cannot access ids stop command : "errmsg"
5004: ids stop command does not have execute permissions: "errmsg"
5005: Cannot access ids probe command : "errmsg"
5006: ids probe command does not have execute permissions: "errmsg"
5007: Please bring all HAStoragePlus resources online before
     creating this resource.
5008: HAStoragePlus Configuration error.
5009: scds_hasp_check returned an unexpected status code.
5010: Extension variable INFORMIXDIR is not set
5020: Cannot access oninit in INFORMIXDIR/bin: "errmsg"
5030: Extension variable INFORMIXSERVER is not set
5040: "INFORMIXSERVER" not found in "ONCONFIG"!
5050: "INFORMIXSERVER" not found in "INFORMIXSQLHOSTS"!
5060: Extension variable ONCONFIG is not set
5070: Cannot access "ONCONFIG" in "INFORMIXDIR"/etc: "errmsg"
5080: Extension variable INFORMIXSQLHOSTS is not set
5090: Cannot access sqlhosts file "INFORMIXSQLHOSTS": "errmsg"
6000: Call to scds_hasp_check failed: "errmsg"
```

IBM Corporation
Software Group
Route 100
Somers, NY 10589

Produced in the United States of America

12-04

GC34-2484-00