SYBASE®

Using Sybase Failover in a High Availability System

Adaptive Server® Enterprise

12.5.1

DOCUMENT ID: DC31651-01-1251-02

LAST REVISED: September 2003

Sybase, Inc., One Sybase Drive, Dublin, CA 94568.

Contents

About This Book

Audience

This manual is intended for Sybase System Administrators and Database Owners.

How to use this book

This book describes how to install, configure, and use Sybase Failover in a high availability system.

- Chapter 1, “What is High Availability?” introduces the concepts of a high availability system and Sybase’s Failover.
- Chapter 2, “Failover and Failback” provides an overview of the concepts of failing over and failing back between Adaptive Servers in a high availability system.
- Chapter 3, “Asymmetric and Symmetric Setup” discusses the differences between asymmetric and symmetric configurations.
- Chapter 4, “Modes of Failover” describes the different modes in which Adaptive Server operates when configured for failover.
- Chapter 5, “Proxy Databases, User Databases, and Proxy System Tables” discusses the concepts of proxy databases, the effect of failover on user databases, and the concepts of proxy system tables.
- Chapter 6, “Running do_advisory” describes how to configure two Adaptive Servers for failover.
- Chapter 7, “Configuring Adaptive Server for Failover on HP” describes how to configure Failover on HP.
- Chapter 8, “Configuring Adaptive Server for Failover on IBM AIX” describes how to configureFailover on IBM AIX.
- Chapter 9, “Configuring Adaptive Server for Failover on HP Tru64 TruCluster Server 5.x” describes how to configure Failover on Digital Unix TruCluster.
- Chapter 10, “Active-Active Configuration for Sun Cluster 2.2” describes how to configure Failover on Sun.
- Chapter 11, “Active-Active Configuration for Sun Cluster 3.0,” describes the how to configure Failover on the Sun Cluster 3.0.

- Chapter 12, “Active-Passive Configuration for Sun Cluster 3.0,” describes how to configurean active-passive setup for Sun Cluster 3.0.
- Chapter 13, “Configuring Adaptive Server for Failover on Veritas, 1.3,” describes how to configure Failover on Veritas.
- Chapter 14, “Configuring Adaptive Server for Failover on Veritas, 3.5,” describes how to configure Failover on Veritas version 3.5 and higher.
- Chapter 15, “Configuring Adaptive Server for Failover on SGI IRIX,” describes how to configure Failover on SGI IRIX.
- Chapter 16, “Configuring Adaptive Server for Failover on Windows NT” describes how to configure Failover on Windows NT.
- Appendix A, “Troubleshooting Second Point of Failures” describes methods of troubleshooting second point of failures.
- Appendix B, “Changes to Commands, System Procedures, System Databases, and New dbcc Commands, and Functions” describes how commands, system procedures, and system databases change when Adaptive Server is configured for failover.
- Appendix C, “Open Client Functionality in a Failover Configuration” describes changes required for Open Client to work with Sybase’s ‘Failover.
- The Glossary defines terms used specifically with Sybase’s Failover.

Related documents

The Sybase® Adaptive Server® Enterprise documentation set consists of the following:

- The release bulletin for your platform – contains last-minute information that was too late to be included in the books.

  A more recent version of the release bulletin may be available on the World Wide Web. To check for critical product or document information that was added after the release of the product CD, use the Sybase Technical Library.
- The *Installation Guide* for your platform – describes installation, upgrade, and configuration procedures for all Adaptive Server and related Sybase products.
- *What’s New in Adaptive Server Enterprise?* – describes the new features in Adaptive Server version 12.5.1, the system changes added to support those features, and the changes that may affect your existing applications.

- *ASE Replicator User's Guide* – describes how to use the ASE Replicator feature of Adaptive Server to implement basic replication from a primary server to one or more remote Adaptive Servers.
- *Component Integration Services User's Guide* – explains how to use the Adaptive Server Component Integration Services feature to connect remote Sybase and non-Sybase databases.
- *Configuring Adaptive Server Enterprise* for your platform – provides instructions for performing specific configuration tasks for Adaptive Server.
- *EJB Server User's Guide* – explains how to use EJB Server to deploy and execute Enterprise JavaBeans in Adaptive Server.
- *Error Messages and Troubleshooting Guide* – explains how to resolve frequently occurring error messages and describes solutions to system problems frequently encountered by users.
- *Full-Text Search Specialty Data Store User's Guide* – describes how to use the Full-Text Search feature with Verity to search Adaptive Server Enterprise data.
- *Glossary* – defines technical terms used in the Adaptive Server documentation.
- *Historical Server User's Guide* – describes how to use Historical Server to obtain performance information for SQL Server® and Adaptive Server.
- *Java in Adaptive Server Enterprise* – describes how to install and use Java classes as data types, functions, and stored procedures in the Adaptive Server database.
- *Job Scheduler User's Guide* – provides instructions on how to install and configure, and create and schedule jobs on a local or remote Adaptive Server using the command line or a graphical user interface (GUI).
- *Monitor Client Library Programmer's Guide* – describes how to write Monitor Client Library applications that access Adaptive Server performance data.
- *Monitor Server User's Guide* – describes how to use Monitor Server to obtain performance statistics from SQL Server and Adaptive Server.
- *Performance and Tuning Guide* – is a series of four books that explains how to tune Adaptive Server for maximum performance:
  - *Basics* – the basics for understanding and investigating performance questions in Adaptive Server.

    - *Locking* – describes how the various locking schemas can be used for improving performance in Adaptive Server.
    - *Optimizer and Abstract Plans* – describes how the optimizer processes queries and how abstract plans can be used to change some of the optimizer plans.
    - *Monitoring and Analyzing* – explains how statistics are obtained and used for monitoring and optimizing performance.
- *Quick Reference Guide* – provides a comprehensive listing of the names and syntax for commands, functions, system procedures, extended system procedures, datatypes, and utilities in a pocket-sized book.
- *Reference Manual* – is a series of four books that contains the following detailed Transact-SQL® information:
    - *Building Blocks* – Transact-SQL datatypes, functions, global variables, expressions, identifiers and wildcards, and reserved words.
    - *Commands* – Transact-SQL commands.
    - *Procedures* – Transact-SQL system procedures, catalog stored procedures, system extended stored procedures, and dbcc stored procedures.
    - *Tables* – Transact-SQL system tables and dbcc tables.
- *System Administration Guide* – provides in-depth information about administering servers and databases. This manual includes instructions and guidelines for managing physical resources, security, user and system databases, and specifying character conversion, international language, and sort order settings.
- *System Tables Diagram* – illustrates system tables and their entity relationships in a poster format. Available only in print version.
- *Transact-SQL User's Guide* – documents Transact-SQL, Sybase's enhanced version of the relational database language. This manual serves as a textbook for beginning users of the database management system. This manual also contains descriptions of the pubs2 and pubs3 sample databases.
- *Using Adaptive Server Distributed Transaction Management Features* – explains how to configure, use, and troubleshoot Adaptive Server DTM features in distributed transaction processing environments.

- *Using Sybase Failover in a High Availability System* – provides instructions for using Sybase's Failover to configure an Adaptive Server as a companion server in a high availability system.
- *Utility Guide* – documents the Adaptive Server utility programs, such as isql and bcp, which are executed at the operating system level.
- *Web Services User's Guide* – explains how to configure, use, and troubleshoot Web Services for Adaptive Server.
- *XA Interface Integration Guide for CICS, Encina, and TUXEDO* – provides instructions for using the Sybase DTM XA interface with X/Open XA transaction managers.
- *XML Services in Adaptive Server Enterprise* – describes the Sybase native XML processor and the Sybase Java-based XML support, introduces XML in the database, and documents the query and mapping functions that comprise XML Services.

Other sources of information

Use the Sybase Getting Started CD, the Sybase Technical Library CD and the Technical Library Product Manuals Web site to learn more about your product:

- The Getting Started CD contains release bulletins and installation guides in PDF format, and may also contain other documents or updated information not included on the Technical Library CD. It is included with your software. To read or print documents on the Getting Started CD you need Adobe Acrobat Reader (downloadable at no charge from the Adobe Web site, using a link provided on the CD).
- The Technical Library CD contains product manuals and is included with your software. The DynaText reader (included on the Technical Library CD) allows you to access technical information about your product in an easy-to-use format.

  Refer to the *Technical Library Installation Guide* in your documentation package for instructions on installing and starting the Technical Library.
- The Technical Library Product Manuals Web site is an HTML version of the Technical Library CD that you can access using a standard Web browser. In addition to product manuals, you will find links to EBFs/Updates, Technical Documents, Case Management, Solved Cases, newsgroups, and the Sybase Developer Network.

  To access the Technical Library Product Manuals Web site, go to Product Manuals at http://www.sybase.com/support/manuals/.

Sybase certifications on the Web

Technical documentation at the Sybase Web site is updated frequently.

- **Finding the latest information on product certifications**

  1. Point your Web browser to Technical Documents at http://www.sybase.com/support/techdocs/.
  2. Select Products from the navigation bar on the left.
  3. Select a product name from the product list and click Go.
  4. Select the Certification Report filter, specify a time frame, and click Go.
  5. Click a Certification Report title to display the report.

- **Creating a personalized view of the Sybase Web site (including support pages)**

  Set up a MySybase profile. MySybase is a free service that allows you to create a personalized view of Sybase Web pages.

  1. Point your Web browser to Technical Documents at http://www.sybase.com/support/techdocs/.
  2. Click MySybase and create a MySybase profile.

Sybase EBFs and software updates

- **Finding the latest information on EBFs and software updates**

  1. Point your Web browser to the Sybase Support Page at http://www.sybase.com/support.
  2. Select EBFs/Updates. Enter user name and password information, if prompted (for existing Web accounts) or create a new account (a free service).
  3. Select a product.
  4. Specify a time frame and click Go.
  5. Click the Info icon to display the EBF/Update report, or click the product description to download the software.

Conventions

The following sections describe conventions used in this manual.

SQL is a free-form language. There are no rules about the number of words you can put on a line or where you must break a line. However, for readability, all examples and most syntax statements in this manual are formatted so that each clause of a statement begins on a new line. Clauses that have more than one part extend to additional lines, which are indented. Complex commands are formatted using modified Backus Naur Form (BNF) notation.

Table 1 shows the conventions for syntax statements that appear in this manual:

Table 1: Font and syntax conventions for this manual

Element	Example
Command names,procedure names, utility names, and other keywords display in sans serif font.	select sp_configure
Database names and datatypes are in sans serif font.	master database
Book names, file names, variables, and path names are in italics.	*System Administration Guide* *sql.ini* file *column_name* *$SYBASE/ASE* directory
Variables—or words that stand for values that you fill in—when they are part of a query or statement, are in italics in Courier font.	`select` *`column_name`* `from` *`table_name`* `where` *`search_conditions`*
Type parentheses as part of the command.	`compute` *row_aggregate* `(`*column_name*`)`
Double colon, equals sign indicates that the syntax is written in BNF notation. Do not type this symbol. Indicates “is defined as”.	`::=`
Curly braces mean that you must choose at least one of the enclosed options. Do not type the braces.	`{cash, check, credit}`
Brackets mean that to choose one or more of the enclosed options is optional. Do not type the brackets.	`[cash | check | credit]`
The comma means you may choose as many of the options shown as you want. Separate your choices with commas as part of the command.	`cash, check, credit`
The pipe or vertical bar(|) means you may select only one of the options shown.	`cash | check | credit`
An ellipsis (...) means that you can *repeat* the last unit as many times as you like.	`buy thing = price [cash | check | credit] [, thing = price [cash | check | credit]]...` You must buy at least one thing and give its price. You may choose a method of payment: one of the items enclosed in square brackets. You may also choose to buy additional things: as many of them as you like. For each thing you buy, give its name, its price, and (optionally) a method of payment.

- Syntax statements (displaying the syntax and all options for a command) appear as follows:

  sp_dropdevice [*device_name*]

  For a command with more options:

  select *column_name*
  from *table_name*
  where *search_conditions*

In syntax statements, keywords (commands) are in normal font and identifiers are in lowercase. Italic font shows user-supplied words.

- Examples showing the use of Transact-SQL commands are printed like this:

```
select * from publishers
```

- Examples of output from the computer appear as follows:

```
pub_id    pub_name                city          state
------    --------------------    -----------   -----
0736      New Age Books           Boston        MA
0877      Binnet & Hardley        Washington    DC
1389      Algodata Infosystems    Berkeley      CA

(3 rows affected)
```

In this manual, most of the examples are in lowercase. However, you can disregard case when typing Transact-SQL keywords. For example, SELECT, Select, and select are the same.

Adaptive Server's sensitivity to the case of database objects, such as table names, depends on the sort order installed on Adaptive Server. You can change case sensitivity for single-byte character sets by reconfiguring the Adaptive Server sort order. For more information, see the *System Administration Guide*.

If you need help

Each Sybase installation that has purchased a support contract has one or more designated people who are authorized to contact Sybase Technical Support. If you cannot resolve a problem using the manuals or online help, please have the designated person contact Sybase Technical Support or the Sybase subsidiary in your area.

CHAPTER 1

What is High Availability?

This chapter includes the following sections:

What is High Availability?

A high availability cluster includes two or more machines that are configured so that, if one machine (or application) is brought down, the second machine assumes the workload of both machines. Each of these machines is called one node of the high availability cluster. A high availability cluster is typically used in an environment that must always be available, for example, a banking system to which clients must connect continuously, 365 days a year.

Sybase's Failover product enables Adaptive Server to work in a high availability cluster in active-active or active-passive configuration.

Active-active configuration

An active-active setup is a two node configuration where both nodes in the cluster include Adaptive Servers managing independent workloads, and are capable of taking over each other's workload in the event of a failure.

The Adaptive Server that takes over the workload is called a secondary companion, and the Adaptive Server that fails is called the primary companion. Together they are companion servers. This movement from one node to another is called failover. After the primary companion is ready to resume its workload, it is moved back to its original node. This movement is called failback.

During **failover** clients connected to the primary companion using the failover property automatically reestablish their network connections to the secondary companion.

You must tune your operating system to successfully manage both Adaptive Servers during failover. See your operating system documentation for information about reconfiguring your system for high availability.

Note An Adaptive Server configured for failover in an *active-active* setup can be shut down using the shutdown command only after you have suspended Adaptive Server from the companion configuration, at both the server level and the platform level. For more information, see the configuration chapter of this manual for your platform.

Active-passive configuration

The active-passive configuration is a multi-node setup that involves a single Adaptive Server, a primary node on which the Adaptive Server primarily runs, and a set of secondary nodes which can potentially host the Adaptive Server and its resources.

When the Adaptive Server can no longer continue to run on the primary node, failover occurs and the Adaptive Server is relocated and restarted on a secondary node. The Adaptive Server can be moved back to the primary node after it recovers and when it can successfully host the Adaptive Server and any associated resources.

In the case of failover or failback, clients connected to the Adaptive Server re-establish their network connections and resubmit any uncommitted transactions when the Adaptive Server is restarted on the secondary node. Client connections using the failover property automatically reestablish their connections.

Sybase provides active-passive configuration support for Sun Cluster 3.0. Contact your provider for other cluster platforms. See Chapter 12, “Active-Passive Configuration for Sun Cluster 3.0,” for detailed information on configuring Adaptive Server in the active-passive mode for Sun Cluster 3.0. Other chapters of this manual pertain to the active-active configuration unless otherwise specified.

Note Adaptive Server configured for failover in an *active-passive* setup can be shut down using the shutdown command only after you disable monitoring on the Adaptive Server at the platform level.

Differences between active-active and active-passive configurations

Table 1-1 summarizes the differences between an active-active and an active-passive configuration.

Table 1-1: Difference between active-active and active-passive

Active-active	Active-passive
Setup: Two Adaptive Servers are configured as companion servers, each with independent work loads. These companions run on the primary and secondary nodes, respectively, as individual servers until failover.	*Setup*: A single Adaptive Server runs either on the primary node or on the secondary node. The Adaptive Server runs on the primary node before failover and the secondary node after failover.
Failover: When failover occurs, the secondary companion takes over the devices, client connections, and so on from the primary companion. The secondary companion services the failed-over clients, as well as any new clients, until the primary companion fails back and resumes its activities.	*Failover*: When failover occurs, the Adaptive Server and its associated resources are relocated to, and restarted on, the secondary node.
Failback: Failback is a planned event during which the primary companion takes back its devices and client connections from the secondary companion to resume its services.	*Failback*: Failback is a planned failover or relocation of the Adaptive Server and its resources to the primary node. Failback is not required, but can be done for administrative purposes.
Client Connection failover: During failover, clients connect to the secondary companion to resubmit their uncommitted transactions. During failback clients connect to the primary companion to resubmit their transactions. Clients with the failover property reestablish their connections automatically.	*Client Connection failover*: During failover and failback, clients connect to the same Adaptive Server to resubmit uncommitted transactions. Clients with the failover property reestablish their connections automatically.

Requirements for failover

You must purchase the *ASE_HA* license option to use Adaptive Server with Failover. See the Installation Guide for information about enabling the *ASE_HA* license.

The two Adaptive Servers in a high availability system must have similar, compatible configurations. Both must be

- Running Adaptive Server 12.0 or higher.
- Running the latest version of Open Client™.
- At the same release level.
- The must have a compatible configuration.
- Running Component Integration Services (CIS).
- Running a high availability subsystem (for example Sun Cluster, Windows NT running the Microsoft Server Cluster, and so on).
- Configured for parallel or non-parallel processing.

Resource requirements

Adaptive Servers configured as companions in a high availability system have different resource requirements than Adaptive Servers that function individually. These different resource requirements exists because the secondary companion must process all the work during failover. This is true even if the companions are setup asymmetrically. Consequently, an Adaptive Server in a high availability system has higher resource requirements than it would have if it was a single server. For more information, see “Single system presentation” on page 7.

The following are some of the resource requirements that you should consider when you configure Adaptive Server as a cluster companion (your site will have its own set of resource requirements that must be addressed).

- Logins, roles, and databases – You must set the number of logins, roles, and databases for the cluster equal to the total number for one Adaptive Server.
- number of user connections – Each companion must be configured for the total number of user connections required for the system.
- number of open databases – Each companion must be configured for the total number of open databases required for the system.
- srids – Each companion must be configured for the total number of srids required for the system.

- number of devices – Each must be configured for the total number of devices used by the cluster, not the number of devices used individually. That is, if one companion uses 14 devices and the second uses 23, each Adaptive Server must be configured with 37 as the number of devices.
- The sp_configure option number of open databases on an Adaptive Server configured for Failover is reduced by two to ensure successful failover. That is, if you currently have the number of open databases set to 10, you will only be able to open eight databases.
- The sp_configure option number of user connections on an Adaptive Server configured for Failover is reduced by two to ensure successful failover. That is, if you currently have the number of user connections set to 50, you will only be able to use 48 user connections.

Applications running with Sybase's failover

Client applications that connect to companion servers must re-link their libraries with the libraries included with failover software. See "CTLIB application changes" on page 357 for more information about using Open Client with failover.

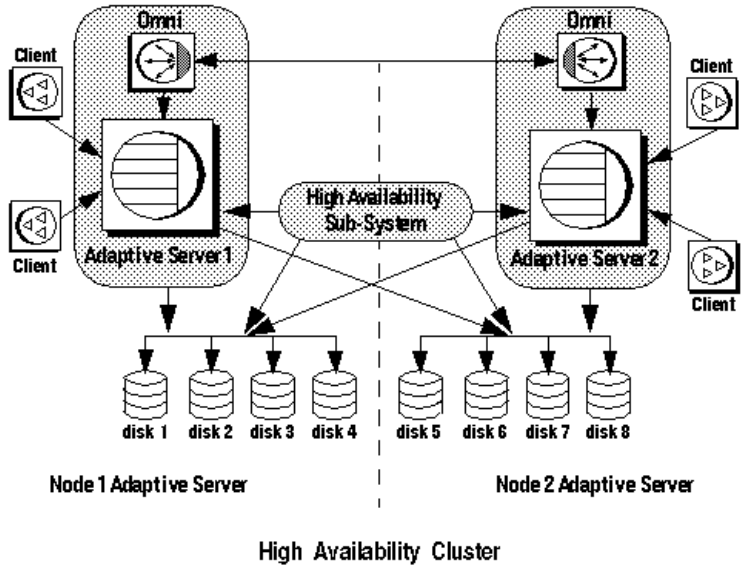
How does Sybase's failover work with high availability?

A high availability system includes the hardware and software necessary to limit the amount of downtime a system suffers. Sybase's Failover is one piece of software in this high availability system. It provides the ability for the companion to withstand a single point of failure in the cluster.

A system that uses Sybase Failover includes only two machines. Each machine is one **node** of the high availability **cluster**. Each Adaptive Server is either a **primary companion** or **secondary companion**. Each companion performs the work during operations; the secondary companion takes over the workload when the primary companion fails or is brought down. The primary companion can be brought down for any number of reasons: scheduled maintenance, system failure, power outage, and so on. The event of the second server assuming another server's workload is called **failover**. The event of moving the workload back to the original server once it is up and running again is called **failback**. Figure 1-1 describes a typical configuration consisting of two Adaptive Servers.

Included with the operating system is a high availability subsystem (for example, Sun Cluster for Sun, Microsoft Cluster Server for Windows NT, and so on) that detects and broadcasts to the cluster that part of the system is crashing or is being shut down for maintenance. When Adaptive Server goes down, the high availability subsystem tells the second machine to take over its workload. Any clients connected to the Adaptive Server going down are reconnected to the second Adaptive Server. Figure 1-1illustrates a high availability cluster is made up of two machines using Sybase's Failover:

Figure 1-1: High availability system using Sybase's Failover



The machines in Figure 1-1 are configured so that each machine can read the other machine's disks, although not at the same time (all the disks that are failed over should be shared disks).

For example, if Adaptive Server 1 is the primary companion, and it crashes, Adaptive Server 2, as the secondary companion, reads its disks (disks 1 – 4) and manages any databases on them until Adaptive Server 1 can be brought back online. Any clients connected to Adaptive Server 1 using the failover property, are connected automatically to Adaptive Server 2.

Single system presentation

One of the hallmarks of a cluster system is that users are unaware that they are logged into a system made up of of two Adaptive Servers. To the users, it appears as if they are logging into a single system with access to all the databases on the cluster. Applications, also, only see a single system. They log into either of the companions and have access to all the databases on the cluster.

However, the System Administrator must treat the system as being made up of two distinct Adaptive Servers. Both Adaptive Servers must be installed and configured individually, and their configuration may not be exactly the same. Both individual Adaptive Servers, as well as the cluster may require system maintenance.

Special considerations for Sybase failover

These Adaptive Server functions require special consideration while configuring a Sybase Failover.

Installing the monitoring tables scripts

If you add monitoring tables to your high availability configuration, you must add either of the following to the interfaces entry for both servers before you can monitor the performance of the secondary and the primary companions:

```
loopback
master tcp ether localhost port_number
```

```
query tcp ether localhost port_number
```

Or,

```
loopback
master tcp ether servername port_number
query tcp ether servername port_number
```

where *port_number* is any open port on the primary companion.

Using failover with disk mirroring

Sybase's failover and the high availability system enable users to access data while the server to which they were originally connected is down. However, neither of these systems prevent disk failures. To make sure you do not lose any data because of a disk failure, you should use Sybase's failover in conjunction with a data protection mechanism, such as disk mirroring or RAID.

Sybase disk mirroring is not supported in an Adaptive Server companion cluster, and is disabled when you issue sp_companion to configure the Adaptive Servers as companions. Use a third-party vendor mirroring system to protect your disk devices.

installhasvss script

The stored procedures required for failover are not included with the *installmaster* script. Run the *installhasvss* script to install the stored procedures and perform many of the tasks required to configure Adaptive Server for failover. *installhasvss* is located in the *$SYBASE/$SYBASE_ASE/scripts* directory.

On Windows NT, this script is *insthasv*, and is located in *%SYBASE%\ASE-12_5\scripts*

Note Do not run the *installmaster* script after running *installhasvss*. Do not use an *installhasvss* script that is a different version than Adaptive Server.

For more information, see the configuration chapter for your platform.

SYB_HACMP server entry

The *installhasvss* script creates an entry in sysservers for a server named SYB_HACMP. Before the Adaptive Server is configured as a companion, the SYB_HACMP server entry points to the local server. The SYB_HACMP sysservers entry allows the primary companion to communicate with the secondary companion using their respective entries in the *interfaces file*. The SYB_HACMP server entry should not be used to create any queries or stored procedures with the companion server.

Never drop the SYB_HACMP server entry. If this entry is inadvertently dropped, you must first re-run *installmaster* and then *installhasvss*.

Define user-defined data types in Adaptive Servers before configuring them for failover

Updates to tables that include either Java or user-defined data types are not synchronized after Adaptive Servers in high availability system are configured as primary and secondary companion servers. For example, if a table in the pubs2 database on the primary companion stores Java objects as column data, updates to this column are not propagated to the proxy table. Instead, you must manually update any changes made to columns that store user-defined data types.

And, for another example, if the pubs2 database on the primary companion includes a table that uses user-defined data types, the pubs2 proxy table on the secondary companion does not include any updates made to pubs2 on the primary companion.

Adaptive Server and two-phase commit transactions

Adaptive Servers configured as companion servers using Sybase Failover do not support SYB2PC transactions, which use the Sybase two-phase commit protocol.

CHAPTER 2

Failover and Failback

This chapter describes the characteristics of the failover and failback modes.

It includes the following sections:

What is failover?

Failover is the process of moving databases, metadata, and user connections from a failed or shut down primary companion to a secondary companion so that users can still access data. There are three sequential steps for failover:

1 System failover – The primary node fails over to the secondary node.

2 Companion failover – The primary companion fails over to the secondary node.

3 Connection failover – Connection with the failover property (for example, isql -Q) fails over to the secondary companion.

Steps two and three are described in detail below. See your high availability subsystem documentation for a description of step one.

During failover, the secondary Adaptive Server detects the primary Adaptive Server's failure through the operating system's high availability system and initiates the failover mechanism, which:

1 Performs a disk reinit to remap the master device path name to its local drive. disk reinit does not disturb the contents of the master device.

2 Mounts the master database, recovers, and brings it online.

3 Maps each of the devices listed in the primary companion's sysdevices to the secondary companion's sysdevices and performs a disk reinit on the disks.

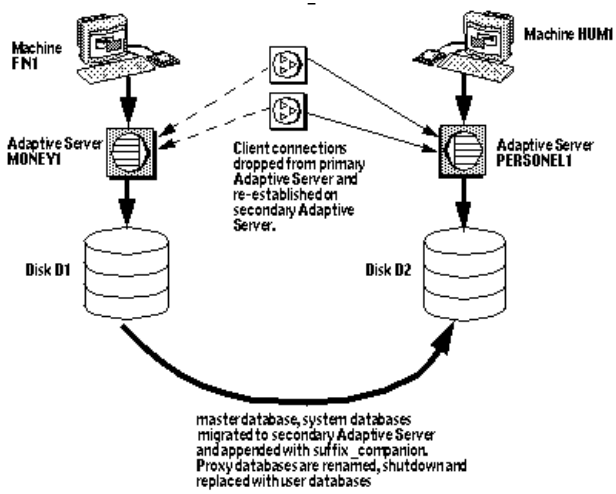
4 Mounts all the primary companion databases on the secondary companion. The secondary companion brings all databases online, after performing recovery from the logs. tempdb and model are not mounted. Proxy databases are mounted with the name comp_dbid_dbname.

   Each database the secondary companion mounts has the suffix _companion appended to its name (for example, the master database becomes master_companion, sybsystemprocs becomes sybsystemprocs_companion, and so on). The secondary Adaptive Server adds this suffix to ensure the unique identity of the databases currently on its system. The user databases do not have the _companion suffix appended to their name; they are guaranteed to be unique.

User connections with the failover property (for example, isql -Q) and clients using the CS_FAILOVER property are retained and reestablished on the secondary companion. Uncommitted transactions must be resubmitted.

Figure 2-1 describes the failover process:

Figure 2-1: The failover process



Once the secondary companion receives the failover message from the high availability system, no new transactions are started on the clients connected to the primary companion. Any transactions that are not complete at the time of failover are rolled back. After failover is complete, clients or users must resubmit rolled-back transactions.

Client connections during failover

Clients with the failover property reconnect automatically during failover. To accommodate this, you must add a line labeled "hafailover" to the *interfaces file* to provide the connection information necessary for the client to connect to the secondary companion. You can add this line using either a file editor or the dsedit utility.

The following *interfaces file* entry is for an asymmetric configuration between the primary companion PERSONNEL1 and its secondary companion MONEY1. It includes an additional *hafailover* entry that enables clients connected to PERSONNEL1 to reconnect to MONEY1 during failover:

```
PERSONNEL1

    master tli tcp /dev/tcp \x00029f7g82d63ce70000000000000000
    query tli tcp /dev/tcp \x00029f7g82d63ce70000000000000000
    hafailover MONEY1
```

On Windows NT, the connection information is included in the *sql.ini* file, which also includes an entry for *hafailover*. The following is a *sql.ini* entry for a symmetric configuration between the MONEY1 and PERSONNEL1 companions:

```
[MONEY1]
 query=TCP,FN1,9835
 master=TCP,FN1,9835
 hafailover=PERSONNEL1
[PERSONNEL1]
 query=TCP,HUM1,7586
 master=TCP,HUM1,7586
 hafailover=MONEY1
```

For more information about adding this information to the *interfaces file*, see the configuration chapter for your platform.

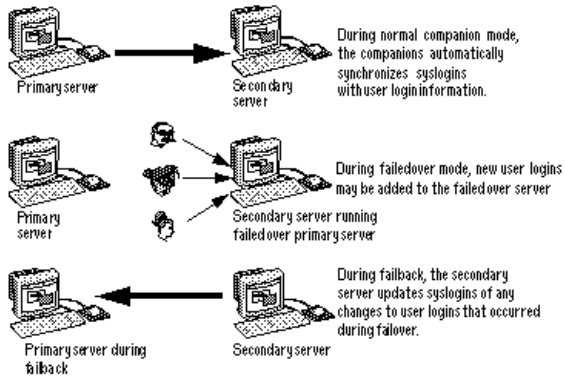
Client applications must re-send any queries that were interrupted by failover. See Appendix C, "Open Client Functionality in a Failover Configuration," for more information about client applications.

User logins in failover

During normal companion mode, companions automatically synchronize any changes to user logins, access and security information, and so on. Any logins added during failover are automatically added to the primary companion when it gets updated during failback. Any uncommitted transactions must be resubmitted and any options set at the session level must be re-established once the companion has successfully failed over.

The synchronization process is described in Figure 2-2:

Figure 2-2: Synchronizing syslogins between primary and secondary servers



All user roles and privileges are maintained after failover.

What is failback?

When the primary companion or machine is prepared to resume operation, the user with the ha_role performs a failback to return the servers to normal companion mode. Because failback temporarily shuts down the databases of the failed-over companion, you should choose a time for the failback when the application load is light. If you choose a time when the Adaptive Server is very busy, failback succeeds, but it is very slow, and the performance of the secondary companion is sacrificed. Choosing the appropriate time for failback can dramatically reduce the amount of time the clients have to wait to reconnect.

Performing failback

Failback is accomplished in four steps:

1 Prepare for failback.

Note IBM HACMP for AIX automatically fails back when the primary node is ready to resume normal companion mode. See Chapter 8, "Configuring Adaptive Server for Failover on IBM AIX," for more information.

Issuing sp_companion with the prepare_failback option forces the secondary companion to release the database devices and the databases. Issue prepare_failback from the secondary companion. The syntax is:

```
sp_companion server_name 'prepare_failback'
```

where *server_name* is the name of the secondary companion. The secondary companion issues messages similar to the following during failback:

```
Step:Access across the servers verified
Step:Primary databases are shutdown in secondary
Step:Primary databases dropped from current secondary
Step:Primary devices released from current secondary
Step:Prepare failback for primary server  complete
(return status = 0)
```

The last step of prepare_failback is to move the devices back to the primary node according to individual platform subsystem.

2 Reboot Adaptive Server on the Primary Machine.

The high availability subsystem reboots the primary companion automatically.

3 Run do_advisory.

Run sp_companion with the do_advisory option to make sure there are no attribute settings that will prevent the failback operation from succeeding. For more information about do_advisory, see Chapter 6, "Running do_advisory,"

4 Resume Normal Companion Mode

After failback is complete, issue sp_companion from the primary companion (the companion that originally failed) to return to normal companion mode. For example:

```
sp_companion PERSONNEL1, resume
Step: Checkin to See if the remote server is up
Step: Access across the servers verified
Step: User information syncup succeeded
(return status = 0)
```

See the configuration chapter for your platform for more information about sp_companion resume.

Note You cannot connect clients with the failover property (for example isql -Q) until you issue sp_companion resume. If you try to reconnect them after issuing sp_companion prepare_failback, the client hangs until you issue sp_companion resume.

Cluster locks in a high availability node

User information for the companions in a high availability cluster must be synchronized. Operations that affect the configuration of the companions are called cluster operations, and are usually initiated by sp_companion. Because the companions must be synchronized, clients performing cluster operations that affect the configuration of the node are only allowed to run serially, not in parallel. That is, only one client can perform a cluster operation at a time.

Before a client performs a cluster operation, it obtains a *cluster-wide lock*, which prevents any other client from performing a cluster operation at the same time. The cluster lock is not released until both companions are synchronized. If a client cannot obtain a cluster lock, its cluster operation fails. Even though the operations are performed in serial, there is no queue for the clients; you must resubmit the failed cluster operations.

A cluster lock may also be obtained if the cluster operation being run requires it..

Generally, you will never notice a cluster lock. They do not affect any other transactions that occur in the database, only cluster operations. However, if the client connection that holds the cluster lock fails during its cluster operation (for example, if you terminate a cluster operation using Control – c before it is finished), the client that failed leaves behind a lock that blocks the next client attempting to obtain a cluster lock.

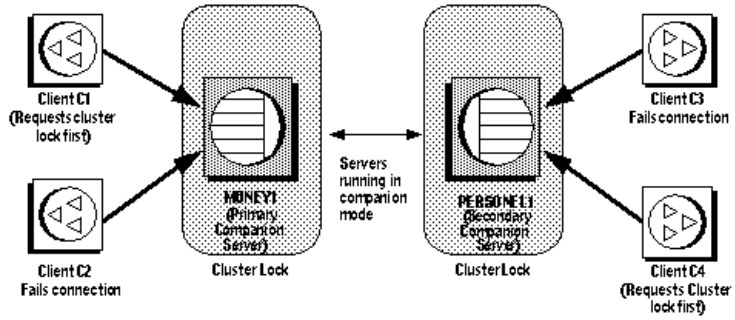
Issue dbcc ha_admin to acquire or release cluster locks:

```
dbcc ha_admin server_name clusterlock [acquire | release]
```

For more information about dbcc ha_admin, see “dbcc Options for High Availability Systems” on page 354.

Figure 2-3 describes two companion servers to which four clients are connecting. All of them are attempting to perform cluster operations.:

Figure 2-3: Clients connecting for cluster operations



1 Client connections C1 and C2 simultaneously attempt to obtain a cluster-wide lock to perform a cluster operation.

2 Client C1 connects to MONEY1 first and receives the cluster-wide lock.

3 Client C2 cannot get a cluster-wide lock, so it cannot perform a cluster operation.

4 Clients C3 and C4 attempt to obtain a cluster-wide lock from PERSONNEL1 while C1 is performing its cluster operation.

5 Clients C3 and C4 cannot obtain a cluster-wide lock because it is held by C1.

6 After client C1 is done with its cluster operation, it releases the cluster-wide lock.

7 Client connections C2, C3, and C4 inform the system administrator that they were not able obtain a cluster-wide lock. The system administrator can resubmit these client connections for their cluster operations after client C1 has released its cluster-wide lock.

CHAPTER 3

Asymmetric and Symmetric Setup

This chapter describes asymmetric and symmetric setups for Adaptive Server in a high availability system.

It includes the following sections:

Asymmetric and symmetric configuration

You can configure companion servers either asymmetrically or symmetrically. You must configure companions asymmetrically before you can configure them symmetrically.

Asymmetric companion configuration

An asymmetric configuration consists of two Adaptive Servers running on separate machines. The primary Adaptive Server performs the work during day-to-day operations, while the secondary Adaptive Server is prepared to take over the work during a system failure or a scheduled maintenance. The secondary companion is an independent Adaptive Server, and can have its own applications running. To configure for failover, the secondary companion must be a newly installed Adaptive Server, and cannot yet have any user logins or user databases. After configuration is complete, you can add user logins and databases to the secondary companion.

When you install and configure Adaptive Server for failover, Adaptive Server is in single-server mode. Use sp_companion to change it from single-server mode to a companion server in an asymmetric setup. See the *Adaptive Server Reference Manual* for information about sp_companion.

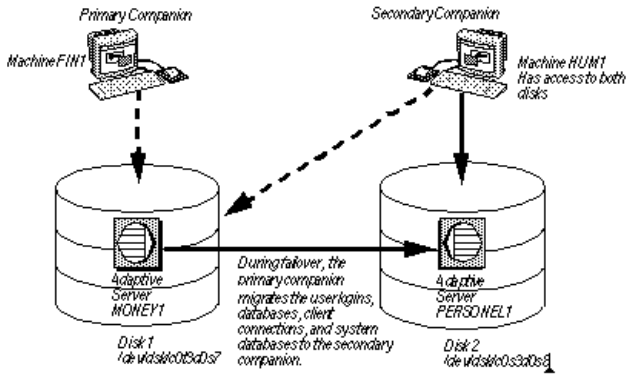
The primary companion issues messages similar to the following when you configure Adaptive Server for failover:

```
sp_companion "MONEY1", configure

 Step: Server 'PERSONNEL1' is alive and cluster aware
 Changing physical name of server 'SYB_HACMP' from 'PERSONNEL1' to 'MONEY1'
 Step: Access verified from Server:'PERSONNEL1' to Server:'MONEY1'
 Step: Server 'MONEY1' is alive and cluster aware
 Changing physical name of server 'SYB_HACMP' from 'MONEY1' to 'PERSONNEL1'
 Step: Access verified from Server:'MONEY1' to Server:'PERSONNEL1'
 Step: Companion servers configuration check succeeded
 Step: Server handshake succeeded
 Step: Master device accessible from companion
 Step: Added the servers 'PERSONNEL1' and 'MONEY1' for cluster config
 Step: Server configuration initialization succeeded
 Step: Synchronizing server logins from companion server
 Step: Synchronizing remoteserver from companion server
 Step: Synchronizing roles from companion server
 Step: Synchronizing server-wide privs from companion server
 Step: User information syncup succeeded
 Step: Server configured in normal companion mode
```

Figure 3-1 describes an asymmetric configuration:

Figure 3-1: Asymmetric configuration in a high availability system



In this setup, MONEY1 is the primary companion and fails over to PERSONNEL1, the secondary companion. Both disks are visible to machine HUM1, which connects to machine FIN1 with a dual-ported SCSI. Because this is an asymmetric setup, PERSONNEL1 cannot fail over to MONEY1. Disk 1 must be a shared disk, and disk 2 can be a local disk.

See the configuration chapter for your platform for information about configuring Adaptive Server for an asymmetric setup.

Performance of Adaptive Server in an asymmetric configuration

During normal companion mode, the performance of the system procedures that update user information (sp_addlogin, sp_addrole, and so on) and commands like create database is slightly degraded because the primary companion must perform the command locally and then synchronize this information with the secondary companion. For example, if you add user "joe" to the primary companion, both the primary companion and the secondary companion must update syslogins to include this new user.

Performance after failover depends on the configuration of the secondary companion. If the secondary server is configured similarly to the primary companion's server, the performance should be similar before and after failover. However, if the secondary server is not as robust (for example, has less memory or fewer CPUs) as the primary server, then the performance after failover will be degraded. The performance of the secondary companion can also be degraded after failover because it is now running both the primary companion and any applications it was running before the failover.

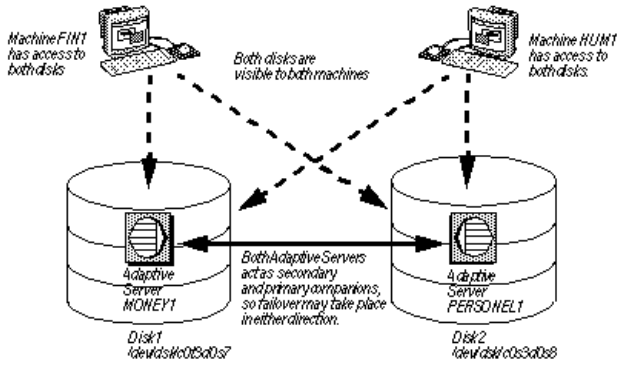
Symmetric companion configuration

Like asymmetric configuration, symmetric configuration consists of two fully functional Adaptive Servers running on separate machines, with their own system devices, system databases, user databases, and user logins. However, when failover occurs, either of the Adaptive Servers acts as a primary or secondary companion for the other Adaptive Server.

Before you configure two Adaptive Servers as symmetric companions, you must first configure them for asymmetric companions.

Figure 3-2 describes a symmetric configuration for failover between a financial department machine (FIN1 running Adaptive Server MONEY1) and a human resources machine (HUM1 running Adaptive Server PERSONNEL1):

Figure 3-2: Symmetric configuration in a high availability system



During scheduled maintenance or system failure, either MONEY1 fails over to PERSONNEL1 or PERSONNEL1 fails over to MONEY1. For this configuration, both Disk 1 and Disk 2 are shared disks.

See the configuration chapter for your platform for information about configuring the Adaptive Servers in a symmetric setup.

Performance of Adaptive Server in a symmetric configuration

During normal companion mode, do not run both Adaptive Servers in a symmetric configuration at the full capacity of their system resources (for example, they could run at 60% of the possible configuration for user connections, data cache, remote server connections, and so on). This allows the secondary companion to manage both the failed over Adaptive Server and its own Adaptive Server during failover mode with a reasonable level of performance. If the Adaptive Servers maximize their system resources, failover still succeeds, but performance may be poor.

Determining the name of the companion server with *@@hacmpservername*

Use the *@@hacmpservername* global variable to determine the name of the companion server. The syntax is:

```
select @@hacmpservername
```

For example, if you issue this command from primary companion MONEY1 you receive output similar to this:

```
select @@hacmpservername

------------------------------
PERSONE1

(1 row affected)
```

Auditing in a high availability system

This section describes the special considerations for auditing in a system configured for Sybase failover.

Configure a companion for auditing in the same way you would configure a server that does not use failover. If a primary companion is configured for auditing, the secondary companion checks to determine whether it also needs to be configured for auditing .For more information see “Setting auditing options” on page 24.

All updates to user and security information (for example, sp_addlogin, sp_addrole, and so on) are done on both the systems in transactional fashion. This keeps the user and security data identical on both the companions.

Table 3-1 describes changes to the auditing configuration parameters.

Table 3-1: Auditing configuration parameters

Configuration parameter	Functionality in Sybase failover
auditing	Both companions must be configured the same for this parameter. Checked as quorum attribute, or when explicitly listed with do_advisory. Turning this parameter on and off is not synchronized dynamically for the companions. You must manually update the remote companion if you change this parameter locally.
allow procedure grouping	Both companions must be configured the same for this parameter. Checked as quorum attribute, or when explicitly listed with do_advisory.
unified login required	Both companions must be configured the same for this parameter. Checked as quorum attribute, or when explicitly listed with do_advisory.
secure default login	Both companions must be configured the same for this parameter. Checked as quorum attribute, or when explicitly listed with do_advisory.

Configuration parameter	Functionality in Sybase failover
systemwide password expiration	Both companions must be configured the same for this parameter. Checked as quorum attribute, or when explicitly listed with do_advisory.
use security services	Both companions must be configured the same for this parameter. Checked as quorum attribute, or when explicitly listed with do_advisory.
check password for digit	Both companions must be configured the same for this parameter. Checked as quorum attribute, or when explicitly listed with do_advisory.
minimum password length	Both companions must be configured the same for this parameter. Checked as quorum attribute, or when explicitly listed with do_advisory.
maximum failed logins	Both companions must be configured the same for this parameter. Checked as quorum attribute, or when explicitly listed with do_advisory.

Setting auditing options

You can configure auditing options (global, database-wide, and per-login) for each companion server on a per-node basis. That is, each companion has its own auditing setting. Global options are not synchronized between the companions.

During failover, database-wide options are audited as they are currently configured.

After failover:

- Auditing continues to enforce global options, and database-wide options run the same as before.
- Users are still allowed to set their database-wide options.
- The audit options of the local domain are used for both local and remote logins (that is, either the failed over primary companion or local secondary companion logins).

Audit trails and Sybase failover

Audit trails are logged in the audit tables of the sybsecurity database. During failover, sybsecurity for the failed server is mounted as sybsecurity_companion on the secondary companion. However, audit trails are always placed in the audit table of the current server. That is, after failover, any new audit trails are placed in the audit table of the secondary companion. Also, auditing configuration changes and auditing record changes that are made on one companion are not implemented on the other companion. For example, if you change one of the auditing configuration parameters on the primary companion, this change is not made on the secondary companion. And, if a user makes a change to a database on the primary companion that requires an audit record, this audit record is not made on the secondary companion.

On failback, no audit trails are transferred from the failed-over domain to the failed domain.

sybsecurity and Sybase failover

sybsecurity database is created by *installsecurity* as part of audit installation. If it exists in either companion during the initial configuration of Sybase failover, it must exist in both companions.

CHAPTER 4

Modes of Failover

This chapter describes the different modes that Sybase Failover moves through during its operation.

It includes the following sections:

What are modes?

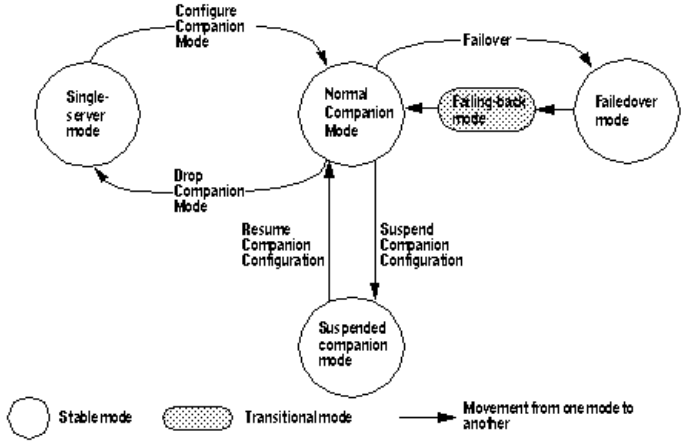
High availability consists of a series of modes in which Adaptive Server runs during its operation. There are two types of modes, stable and transitional. A **stable mode** is a system state in which Adaptive Server can exist for an extended period of time, such as the day-to-day operation of Adaptive Server.

Stable modes include:

- Single-server mode
- Normal companion mode
- Failover mode
- Suspended companion mode

The failback **transitional mode** occurs when Adaptive Server shifts from failed over mode to normal companion mode. The failback transitional mode is typically of very short duration. The different modes and the movement that the primary companion makes while changing modes are shown in Figure 4-1:

Figure 4-1: Modes of operation for high availability



Before you can configure two Adaptive Servers as companions, both must be in single-server mode, which is the default mode of a newly installed Adaptive Server after running *installhasvss*. After you configure the Adaptive Servers as companions, they are in one of three stable modes:

- Normal companion mode
- Failed-over mode
- Suspended companion mode

Determining the companion's mode

You can issue sp_companion without any options to display the mode the companion is currently in. For example:

```
sp_companion
Server 'MONEY1' is alive and cluster configured.
Server 'MONEY1' is configured for HA services.
Server 'MONEY1' is currently in 'Symmetric normal' mode.
```

Companion MONEY1 is configured for symmetric failover and is currently running in normal companion mode.

Determining the mode with @@*cmpstate*

You can also determine the mode using the *@@cmpstate* global variable. At the isql prompt, enter:

```
select @@cmpstate
```

Table 4-1 describes values that @@cmpstate returns:

Table 4-1: @@cmpstate return values

@@cmpstate	Companion Mode
0	Single server
1	Reserved
2	Secondary normal
3	Secondary suspended
4	Secondary failover
5	Secondary failback
6	Reserved
7	Primary normal
8	Primary suspended
9	Primary failback
10	Reserved
11	Symmetric normal
12	Symmetric failover
13	Symmetric suspended
14	Symmetric failback
15	Reserved

Different modes of a companion server

This section describes each mode in detail.

Single-server mode

In this mode, Adaptive Server acts as a standalone server. A newly installed Adaptive Server is in single-server mode by default.

Normal companion mode

When both companions are running and are configured for failover, they operate in normal companion mode. This is the mode in which the day-to-day operations of Adaptive Server occur. For asymmetrical systems, this means that the primary companion can fail over to the secondary companion. For a symmetric system, this means that either companion can fail over to the remaining companion.

Suspended mode

Use suspended mode to temporarily suspend the companions from normal companion mode. In suspended mode, both servers act as single servers. Suspended mode is useful for performing system maintenance because you can start and stop the Adaptive Server and associated resources without risking failover.

Even though the companions cannot fail over, the nodes upon which they are working can still fail over; you must perform some platform-specific steps to suspend node failover. Also, before you shut down a companion in suspended mode, you must perform some platform-specific tasks. See the chapter for your platform for more information.

Many utilities and commands are severely restricted in suspended mode. See Appendix B, “Changes to Commands, System Procedures, System Databases, and New dbcc Commands, and Functions,” for more information.

Note Always suspend companion mode from the secondary companion.

To suspend a companion from running in normal companion mode for any length of time (typically for maintenance), enter:

```
sp_companion 'secondary_server_name', 'suspend'
```

For example, to suspend primary companion MONEY1 from normal companion mode with its secondary companion PERSONNEL1, issue the following from PERSONNEL1:

```
sp_companion "MONEY1", suspend
```

The companion produces messages similar to the following:

```
Step: Server 'MONEY1' is alive and cluster aware
 Step: Access verified from Server:'MONEY1' to Server:'PERSONNEL1'
 Step: Server 'PERSONNEL1' is alive and cluster aware
 Step: Access verified from Server:'PERSONNEL1' to Server:'MONEY1'
 Step: Companion servers configuration check succeeded
 Step: Access across the servers verified
```

Failback mode

Adaptive Server must enter transitional failback mode to move from failover mode on the secondary companion to normal companion mode on the primary companion.

Failback mode is a planned event. That is, it is only done when the system administrator determines that the system is ready to resume normal operations. Use sp_companion prepare_failback to initiate failback and migrate the failed-over Adaptive Server to its original node. “Performing failback” on page 15 describes the steps necessary to perform failback.

Resuming normal companion mode from suspended mode

To resume normal companion mode:

```
sp_companion "primary_server_name", resume
```

For example, to resume normal companion mode between primary companion MONEY1 and its secondary companion PERSONNEL1, issue this command from MONEY1:

```
sp_companion "PERSONNEL1", resume
```

The companion from which you issued the command produces messages similar to the following:

```
Step: Server 'MONEY1' is alive and cluster aware
    Step: Access verified from Server:'MONEY1' to Server:'PERSONNEL1'
    Step: Server 'PERSONNEL1' is alive and cluster aware
    Step: Access verified from Server:'PERSONNEL1' to Server:'MONEY1'
    Step: Companion servers configuration check succeeded
    Step: Checkin to See if the remote server is up
    Step: Access across the servers verified
    Step: Synchronizing server logins from companion server
    Step: Synchronizing remoteserver from companion server
    Step: Synchronizing roles from companion server
    Step: Synchronizing server-wide privs from companion server
    Step: User information syncup succeeded
```

Drop failover mode

To permanently disable companion mode, enter:

sp_companion "*server_name*", 'drop'

The companion from which you issued the command produces messages similar to the following:

```
Step:Local server 'MONEY1' is alive  and cluster aware
    Step:HA Versions and DLL check succeeded
    Step:Access across the servers verified
    Step: Removed the servers 'MONEY1' and 'PERSONNEL1' for cluster config
 (return status = 0)
```

When this command is complete, the two Adaptive Servers are no longer companion servers and are running in single-server mode.

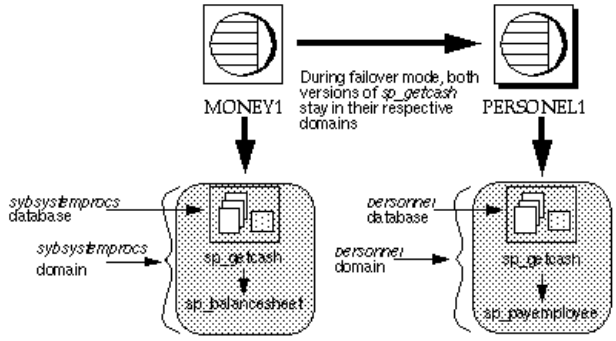
Note Drop is an irreversible operation. Once you have reverted the companion servers to single-server mode, you must dump, drop, and reload all user databases to reconfigure them as companions.

If the companion you drop is in a symmetric setup, the cluster automatically assumes an asymmetric setup between the companions.

Domains

Both the primary and the secondary companions can have stored procedures, users, and devices with the same names. Adaptive Servers configured for failover use *domains* to determine which database these objects belong to. For example, using the financial/human resources configuration outlined in Figure 3-1 and Figure 3-2, suppose both Adaptive Servers MONEY1 and PERSONNEL1 have a stored procedure named sp_getcash, as described in Figure 4-2:

Figure 4-2: Domains during failover



In MONEY1, sp_getcash (which issues a secondary stored procedure named sp_balancesheet) is defined in the domain of sybsystemprocs. In PERSONNEL1, sp_getcash (which issues a secondary stored procedure named sp_payemployee) is defined in the domain of the database personnel. During failover, even though sybsystemprocs for MONEY1 migrates to PERSONNEL1 as sybsystemprocs_companion, its domain does not change, nor do the objects that are defined for this domain. Users that issue sp_getcash in sybsystemprocs for MONEY1 during failover mode still issue the correct secondary-stored procedure, sp_balancesheet.

The concept of domains is transparent to the users; they issue the same stored procedure or use their same login and login password.

System procedures that are stored in the master database are not controlled by domains. System procedures should never have a dependency on an object that is stored in the master database.

CHAPTER 5

Proxy Databases, User Databases, and Proxy System Tables

This chapter describes how proxy databases and tables are used in a failover system.

This chapter includes the following sections:

For complete information about proxy databases and tables, see the *Component Integration Services User's Guide*.

Proxy databases

Proxy databases are not created by default when you configure the Adaptive Servers as companions. They are created in the remote server only if you configure for Failover using the with_proxydb option of sp_companion. The discussion in this chapter assumes you used sp_companion with the with_proxydb option. The companions still failover whether or not you include the with_proxydb option when you configure the Adaptive Servers as companion server; the proxy databases are created dynamically as they are needed. For more information about sp_companion with_proxydb, see the *Adaptive Server Reference Manual*.

Databases in companion servers are either primary or proxy databases. Primary database are where the data is physically located. Each proxy database corresponds to a primary database; it has the same name as the primary database, and proxy entries for all the objects in the primary database, but it contains no data.

After you configure the companions for failover and the proxy databases are created, the user databases are visible to both companions. This means that you can perform transactions on a primary database from either companion. For example, if a primary companion named PERSONNEL1 includes a database named salary, its secondary companion, MONEY1, includes a salary proxy database. You can perform inserts, updates, and deletes on salary from either MONEY1 or from PERSONNEL1. Also, sysdatabases on either companion lists the salary database. For example, the following query produces the same result on PERSONNEL1 and MONEY1:

```
1> select name from sysdatabases
 name
 ------------------------------
 master
 model
 salary
 sybsystemdb
 sybsystemprocs
 tempdb
```

Note During failover, all current connections to proxy databases on the secondary server are terminated and disconnected. During failback, the secondary companion reverses the process, mounting the primary databases and then re-creating the proxy databases.

Creating proxy databases

Adaptive Server uses Component Integration Services (CIS) to create the proxy databases. Both the primary Adaptive Server and the secondary Adaptive Server must have CIS running before they are configured for Sybase Failover. To determine if you have CIS running, enter:

```
sp_configure "enable cis"

 Parameter Name      Default    Memory Used  Config Value  Run Value
 ------------------  ---------  -----------  ------------  ---------
 enable cis               1            0            1               1
```

A Run Value of 1 indicates that CIS is running.

For information about configuring Adaptive Server for CIS, see the *Component Integration Services User's Guide*.

When it creates the proxy databases, CIS:

1 Estimates the size of the database required to contain all the proxy tables if a size or database device is not specified.

2 Creates all proxy tables. These act as placeholders for the tables and views found in the primary companion's database.

3 Imports the metadata (column names, size, indexes, and so on) from the primary companion.

4 Grants all permissions on the proxy tables to public.

5 Adds the user guest to the proxy database.

6 Sets the database status to indicate that the database is a proxy database. The status is indicated in the status3 column of sysdatabases. sp_helpdb includes information about whether a database is a proxy or primary database.

When are proxy databases created?

Proxy database are only created if you configured for Sybase Failover using the sp_companion...with_proxydb option. After the companions are configured with this option:

- Proxy databases for all the primary companions user databases created when a companion configuration is created.
- Proxy databases are created for any new user databases create in the primary companion using create database.
- During failover, the secondary companion mounts the primary databases and then drops the proxy databases. During failback, the secondary companion reverses the process, mounting the primary databases and then re-creating the proxy databases.

Size of the proxy databases

When Adaptive Server creates a proxy database, it checks the number of tables and views in the primary database and calculates the amount of space required to accommodate the same number of proxy tables in the proxy database. Each proxy table requires eight pages (one extent). Each index on a proxy table also requires eight pages. Adaptive Server also adds either an additional 10 percent or 500 pages – whichever is larger – to the database to allow for table growth.

As a result, the size of the proxy databases depends on the number of tables and views in the primary database. Proxy databases do not have a default size; the minimum size is at least the size of the model database.

Behavior of commands and system procedures in proxy databases

The behavior of some commands and system procedures changes if these are issued in proxy databases.

Changes to commands in proxy databases

For most commands, it does not matter whether you issue them from the primary database or the proxy database; only the primary database is updated. These commands cannot be issued from within the proxy database:

- create or drop procedure
- create or drop view
- create or drop trigger
- create or drop rule
- create or drop default

You must run dump and load database commands from the primary companion. If you issue these commands from the proxy database, they will update only the proxy database; they do not update the primary companion.

Changes to system procedures in proxy databases

System procedures always perform their tasks *locally*. That is, if you issue system procedure in a proxy database, any changes it makes do not appear in the primary database, and vice-versa.

System procedures begin with either the sp_ or xp_ prefix.

Issuing user-defined stored procedures in proxy databases

User-defined stored procedures always perform their tasks in the primary database. For example, whether you issue user_created_proc from the *pubs2* primary database or from the *pubs2* proxydatabase, it executes on the *pubs2* primary database.

System procedures issued from a proxy database are handled based on these criteria:

- A request to execute a user-stored procedure in a high availability system proxy database is transformed into a remote procedure call (RPC) request and sent to the server that owns the original database.
- For system procedures, search rules are invoked such that the procedure is looked for first in the current database, then in sybsystemprocs, then in master. If a procedure is not found, the request is converted to a remote procedure call (RPC) and forwarded to the server that owns the original database (as is the case with user-created stored procedures).
- CIS first looks for the system procedure in the local server. If it finds the system procedure locally, it is executed as a local stored procedure.
- If the system procedure cannot be found locally, it is forwarded to the primary companion as an RPC.
- If it is a user-defined stored procedure, it is turned into an RPC and forwarded to the primary companion.

This behavior applies only to "system" proxy databases – that is, those that are created automatically by HA configuration. User proxy databases do not exhibit this behavior.

System procedures issued in a companion configuration are processed using the same rules as a single server. For a description of how system procedures are processed, see the *Adaptive Server Reference Manual*.

sp_dboption does not update proxy databases

If you use sp_dboption to change the database options on the primary database, these changes are not automatically forwarded to the proxy databases on the secondary companion. You must set the sp_dboption on the proxy database as well.

For example, if you use sp_dboption to change the pubs2 database so that select into bulkcopy/pllsort is on the primary companion, the pubs2 proxy database on the secondary companion is not set.

Manually updating the proxy databases

alter database allows you to manually re-synchronize your proxy databases with their primary databases using the for proxy_update option. You must issue this command from the master database

```
alter database <dbname>
    [existing options]
    [for proxy_update]
```

for proxy_update is useful for synchronizing changes to the primary databases that are not automatically migrated to the proxy databases. For example, if you rename the primary database using sp_rename, the proxy database is not automatically renamed. However, if you issue the alter database... for proxy_update after renaming the database, the proxy database is rebuilt using the new database name.

If you enter for proxy_update with no other options (for example, alter database pubs2 for proxy_update), the size of the database is not extended; instead, the proxy tables are dropped from the proxy database and then re-created using the metadata from the primary companion's database.

If you use alter database to extend the size of the database, the proxy table update is performed after the size extensions are made.

for proxy_update is supported for all external data sources, not just the primary companion in a cluster environment. Also, a database does not have to be created with the for proxy_update clause for it to be manually updated. If you specify a default storage location, either through the create database command or sp_defaultloc, the primary companion's metadata can be synchronized with the metadata at the remote storage location.

For more information about alter database, see the *Adaptive Server Reference Manual*.

Proxy system tables in *master*

Proxy system tables enable a secondary companion to access the primary companion's system tables. One extent is allocated for the proxy system tables in sysobjects. You cannot drop these proxy system tables. Proxy system tables use the following naming syntax:

```
rmt_ha_system_table_name
```

Table 5-1 lists the proxy system tables in the secondary companions sysobjects:

Table 5-1: Proxy table names in secondary companion's sysobjects

Proxy System Table name	System Table Name
rmt_ha_sysalternates	sysalternates
rmt_ha_sysattributes	sysattributes
rmt_ha_sysconfigures	sysconfigures
rmt_ha_sysdatabases	sysdatabases
rmt_ha_syslanguages	syslanguages
rmt_ha_sysloginroles	sysloginroles
rmt_ha_syslogins	syslogins
rmt_ha_sysmessages	sysmessages
rmt_ha_sysobjects	sysobjects
rmt_ha_sysprotects	sysprotects
rmt_ha_sysremotelogins	sysremotelogins
rmt_ha_sysresourcelimits	sysresourcelimits
rmt_ha_sysroles	sysroles
rmt_ha_sysservers	sysservers
rmt_ha_syssessions	syssessions
rmt_ha_syssrvroles	syssrvroles
rmt_ha_systhresholds	systhresholds
rmt_ha_systypes	systypes
rmt_ha_sysusers	sysusers

CHAPTER 6

Running *do_advisory*

This chapter describes how to run sp_companion with the do_advisory option.

It includes the following sections:

What is the *do_advisory* option?

When you perform a cluster operation (for example, moving from failover mode to normal companion mode), either companion may have attribute settings that prevent the cluster operation from succeeding. For example, the secondary companion may be configured with a stack size that is too small to accommodate both companions during failover mode, or the companions may be configured for different languages.

To prevent these problems, the sp_companion command includes a do_advisory option which checks hundreds of attribute settings for each of the companions and issues warnings about any settings that will prevent a successful cluster operation. The attributes do not necessarily have to have the same values on both companions; for many attributes, the values must only be compatible between the two companions. sp_companion do_advisory does not change any of the attributes, it only advises you about any potential problems.

sp_companion...do_advisory is not triggered automatically (for example, during a sp_companion...resume). You should run sp_companion...do_advisory periodically to make sure there are no compatibility issues between your companions that will prevent a successful failover.

do_advisory allows you to specify the granularity of the attributes you want to investigate. You can either look at all the attributes, or you can specify subsets of attributes. When you specify that you want to look at all the attributes, sp_companion issues a list of all the attributes that will prevent a successful cluster operation.

The subset consists of *group*, *base*, or *quorum* attributes. A group attribute comprises a broad set of server settings (for example, all the login attributes or all the space attributes); a base attribute comprises specific settings within the group attributes (for example, user logins or CIS settings). When you specify that you want to investigate a subset of attributes, do_advisory only reports the attributes of this subset that will prevent a successful cluster operation.

Quorum attributes are configuration parameters that sp_companion checks every time it is run, regardless of whether or not you specify group or base attributes. If sp_companion finds that a quorum attribute is set such that it will prevent a successful cluster operation, the command fails. For more information, see "Quorum attributes" on page 48.

- *Application group* – Checks to make sure the configuration settings for the applications running on the local companion are compatible with the remote companion. The application group includes the following:

  Charsets – Verifies that the character sets for which the secondary companion is configured include all the character sets for which the primary companion is configured.
  Java Archives – Checks to make sure the Java archive on the primary companion has the same name and class definition on the secondary companion. If a class definition belongs to Java archive on the primary companion, it must belong to the same Java archive on the secondary companion.

  Note These are not automatically synchronized; if you configure one companion for Java, you must manually configure the other.

  Languages – Verifies that the languages for which the secondary companion is configured include all the languages for which the primary companion is configured.
  Remote servers – Checks that remote server entries used by the application on the primary companion are the same on the secondary, if they exist. This ensures that server names and the associated server IDs used by the companions are unique and consistent within the cluster.

All default server entries (including SYB_BACKUP, local server name, companion server name, SYB_HACMP, local XP server, and companion XP server) are automatically synchronized.
Sort order – Verifies that the sort orders for which the secondary companion is configured include all the sort orders for which the primary companion is configured.
Time ranges – Verifies that time range definitions defined and used by the primary companionare the same as those used by the secondary companion, if they exist.
User types – Verifies that all user-defined data type definitions in master used by an application on the primary companion are defined the same way on the secondary companion, if they exist.

- *Config group* – Checks for compatibility between configuration parameters defined in the configuration file (located in *$SYBASE/server_name.cfg*). Configuring the Adaptive Server as companions does not automatically synchronize the configuration options. The config group includes the following base attributes:

  CIS – Verifies that CIS is correctly configured for the cluster operation.
  DTM – Verifies that the Distributed Transaction Manager parameters are compatible between the companions.
  Disk i/o – Makes sure the disk configuration (disk i/o structures, allow sql server async i/o, and so on) is compatible between the companions.
  ESP – Makes sure the extended stored procedures are compatible between the companions.
  Errorlog – Makes sure that the error log information (event logging, event log computer name, and so on) is compatible between the companions.
  General config – Verifies that all the general configuration parameters (those set in the configuration file) are correctly set for the cluster operation.
  Java – Makes sure that Java is either enabled or disabled for both companions.
  Languages – Makes sure that both companions have the same language, character set, and sort order.
  Network – Makes sure that the network related parameters (allow remote access, default network packet size, and so on) are compatible between the companions.
  Parallel – Verifies that the parallel configuration parameters (max parallel degree, memory per worker process, and so on) are compatible between the companions.

Q Diag – Verifies that the Q Diagnostic attributes (autostart collector, sql text pipe active, and so on) are compatible between the companions.
Security – Verifies that the security configuration (auditing, allow procedure grouping, and so on) for the companions is compatible.

- *Database group* – Checks that database attributes are compatible between the companions. The database group includes:

  Unique Dbid – Verifies that database IDs on the primary companion are not used on the secondary companion.

  Note If a user database ID conflicts with a system database ID on the secondary companion (for example sybsystemprocs), you must drop and recreate the system database on the secondary companion.

- *Devices group* – Checks that device attributes are compatible between the companions. The devices group includes:

  Devnames – Verifies that logical device names on the primary companion are not used on the secondary companion.

- *Logins group* – Verifies that login and permissions are consistent between the primary and secondary companions.

  Logins – All user information (logins, permissions, and so on) defined on the primary companion must be defined, available, and compatible on the secondary companion, if it exists. Logins on the primary companion are checked to verify that they have unique names and suids on the secondary companions. The logins group also checks that remote logins, external logins, aliases, in master, and user names in master are compatible across the companions. do_advisory automatically corrects any issues that it finds with a value of 1 (for example, a login that exists on the primary companion that does not conflict with any logins on the secondary companion, but does not exists in secondary).

  Default login incompatibilities of probe, qcollector, qrepository, and so on are fixed automatically.

- *Roles group* – Verifies that roles are consistent between the primary and secondary companions.

  Roles – Verifies that all user-defined roles, login roles, and server wide permissions are compatible.

- *Space group* – Verifies that the secondary companion has sufficient space available for the primary companion databases during failover.

Master Space – Estimates the space required to synchronize the metadata during the initial configuration of the companion server or during sp_companion...resume.
Proxydb Space – Estimates the space required for creating the proxy databases (when you configure the companion servers with with_proxydb).

How do I run the *do_advisory* option?

The syntax for sp_companion do_advisory is:

```
sp_companion server_name, do_advisory [, all | help |
    group_attribute_name | base_attribute_name]
```

where:

- *server_name* is the name of the remote Adaptive Server.
- all indicates that you want information about both the group and the base attributes.
- help prints the sp_companion do_advisory syntax and a list of the group and base attributes
- *group_attribute_name* is the name of the group attribute upon which you want sp_companion to report.
- *base_attribute_name* is the name of the base attribute upon which you want sp_companion do_advisory to report.

sp_companion do_advisory output includes:

- Attribute name – The name of the attribute that sp_companion do_advisory is investigating.
- Attribute type – The type of attribute. For example, the type might be CIS, disk i/o, General Config (these are the configuration parameters set in the *server_name.cfg* file).
- Local value – The value of the attribute on the companion from which you entered sp_companion do_advisory.
- Remote value – The value of the attribute on the remote companion.
- Advisory – After accessing the attributes on the two companions, sp_companion do_advisory prints its findings in the Advisory column. The values in this column are:

  0 – The attributes will not affect the cluster operation.

1 – The attributes are not configured for the best configuration, but they will not prevent a cluster operation.

2 – The attributes need to be altered before proceeding with the cluster operation.

For example, the following checks the attributes between Adaptive Servers MONEY1 and PERSONNEL1:

```
sp_companion "MONEY1", do_advisory, 'all'
go
Attribute Name   Attrib Type  Local Value  Remote Value  Advisory
--------------   -----------  -----------  ------------  ------
cis connect time  CIS         1            0             2
cis rpc handling  CIS         1            0             2
max cis remote se CIS         10           25            2

(1 row affected)
(return status = 0)
```

In this example, the attributes cis connect, cis rpc handling, and max cis remote servers all have a value of 2 under the Advisory column, which indicates that these attributes will prevent a successful companion configuration between MONEY1 and PERSONNEL1. Note that the Local Values for these three attributes are different from the Remote Values. The companions must be reconfigured to have the same or compatible values.

Quorum attributes

Whether or not you include the do_advisory option, if sp_companion is issued with either the configure or resume option, sp_companion checks a select group of attributes to make sure the companions have compatible values. These are called quorum attributes. If one of the companions has a value for a quorum attribute that is not compatible with the other companion, sp_companion fails.

Note If sp_companion issues a message stating that a quorum attribute prevented it from successfully finishing, run sp_companion... do_advisory for a list of the problem attributes.

The following is a list of the quorum attributes:

- enable cis

- cis packet size
- max cis remote connections
- max cis remote servers
- number of devices
- esp execution stack size
- start mail session
- xp_cmdshell context
- default character set id
- default language id
- default sortorder id
- disable character set conversions
- enable repagent thread
- allow backward scans
- allow netsted triggers
- allow resource limits
- parition groups
- size of auto identity columns
- SQL perform integration
- cfg read committed with lock
- enable Java
- enable DTM
- number of DTX participants
- strict dtm enforcement
- allow remote access
- default network packetsize
- max network packetsize
- max parallel degree
- number of remote logins

- number of remote sites
- max parallel degree
- Charsets
- Java Archives
- Languages
- Remote Servers
- Sort order
- Time Ranges
- User Types
- Unique Dbid
- Devnames
- Logins
- Roles

CHAPTER 7

Configuring Adaptive Server for Failover on HP

This chapter lists the steps necessary to configure Adaptive Server for failover on HP.

It includes the following sections:

Configuring hardware and operating system for high availability

Sybase high availability requires the following hardware and system components:

- Two homogenous, network systems with similar configurations in terms of resources like CPU, memory, and so on.
- High availability subsystem package and the associated hardware.
- Devices that are accessible to both nodes.
- A logical volume manager (LVM) to maintain unique device pathnames across the cluster nodes.
- Vendor provided mirroring, not Sybase mirroring, for media failure protection.

See your hardware and operating system documentation for information about installing platform specific high availability software.

Preparing Adaptive Server to work with the HA subsystem

Perform the tasks in this section to prepare Adaptive Server for a high availability configuration

Install Adaptive Servers

Install the primary and the secondary servers. They must be installed in the same location on each node. The primary companion can be either a newly installed Adaptive Server, or it can be upgraded from a previous version of Adaptive Server with existing databases, users, and so on. The secondary companion must be a newly installed Adaptive Server and cannot have any user logins or user databases. This is to make sure that all user logins and database names are unique within the cluster.

After configuration for failover is complete, you can add user logins and databases to the secondary companion.

If you are installing on the local disk, make sure any databases are created on the multihost disk.

See the installation documentation for your platform for information about installing and configuring Adaptive Server.

Add entries for both Adaptive Servers to the *interfaces file*

The *interfaces file* for the primary and the secondary companion must include entries for both companions. For example, the *interfaces file* for the servers used in the examples above would have entries for both MONEY1 and PERSONNEL1. The server entry in the *interfaces file* must use the same network name that is specified in sysservers. For information about adding entries to the *interfaces file*, see the installation documentation for your platform.

Add entries to *interfaces* file for client connections during failover

To enable clients to reconnect to the failed over companion, you must add an additional line to the *interfaces file*. By default, clients connect to the port listed in the query line of the server entry. If that port is not available (because that server has failed over), the client connects to the server listed in the *hafailover* line of the server entry. Here is a sample *interfaces file* for a primary companion named MONEY1 and a secondary companion named PERSONNEL1:

```
MONEY1
    master tcp ether FIN1 4100
    query tcp ether FIN1 4100
    hafailover PERSONNEL1
```

Use dsedit to add entries to the *interfaces file*. If the interfaces entries already exist, you must modify them to work for failover.

See the Utility Guide for your platform for information about dsedit.

Set the value of *$SYBASE*

If you installed *$SYBASE* on a local file system, *$SYBASE* must point to the same directory name on both companions. You can accomplish this by:

- Making sure that the *$SYBASE* release directory on each companion is created in the same directory, or
- If the companions have the *$SYBASE* release directory in different locations, create a directory with the same path on both companions that acts as a symbolic link to the actual *$SYBASE* release directory.

  For example, even though primary companion MONEY1 has a release directory of */usr/u/sybase1* and PERSONNEL1 has uses */usr/u/sybase2* as its release directory, *$SYBASE* must point to the same path.

  Both MONEY1 and PERSONNEL1 have */SYBASE*, which they establish as a symbolic link to their respective *$SYBASE* release directories. On MONEY1, */SYBASE* is a link to */usr/u/sybase1*, and on PERSONNEL1, */SYBASE* is a link to */use/u/sybase2*.

If you installed $SYBASE on a local file system, you must also have copies of both companion RUN_<*SERVERNAME*> files in *$SYBASE/$SYBASE_ASE/install* on both nodes.

sybha executable

The *sybha* executable the Adaptive Server High Availability Basic Services library to interact with each platform's high availability cluster subsystem. The Adaptive Server High Availability Basic Services library calls *sybha*, which is located in *$SYBASE/ASE-12_5/bin*. Before *sybha* can run, you must change its ownership and permissions. You must also edit a file named *sybhauser* in *$SYBASE/ASE-12_5/install*. *sybhauser* contains a list of the users who have System Administrator privileges on the cluster. Sybase strongly recommends that you severely limit the number of users who have System Administrator privileges on the cluster.

As root, perform the following:

1 Add a new group named *sybhagrp*. You can either add this group to the */etc/group* file or you can add it to your NIS maps. Add the sybase user to this group (this is the user that owns the *$SYBASE* directory). When the server is started, the sybase user runs the data server. If you have multiple servers running and different users owning the *$SYBASE* directory for each of them, each of these users must be added to the group.

2 Change to the *$SYBASE/$SYBASE_ASE/bin* directory:

```
cd $SYBASE/$SYBASE_ASE/bin
```

3 Change the ownership of *sybha* to root:

```
chown root sybha
```

4 Change the group for the *sybha* program to *sybhagrp*:

```
chgrp sybhagrp sybha
```

5 Modify the file permissions for *sybha* to 4550:

```
chmod 4550 sybha
```

6 Change to the *$SYBASE/$SYBASE_ASE/install* directory:

```
cd $SYBASE/ASE-12_5/install
```

7 Add the sybase user to the *sybhauser* file. These logins must be in the format of UNIX login IDs, not Adaptive Server logins. For example:

```
sybase
coffeecup
spooner
venting
howe
```

8 Change the ownership of *sybhauser* to root:

```
chown root sybhauser
```

9 Modify the file permissions for *sybhauser*:

```
chmod 600 sybhauser
```

Create new default device other than master

By default, the master device is the default device in a newly installed Adaptive Server. This means that, if you create any databases (including the proxy databases used by failover), they are automatically created on the master device. However, adding user databases to master makes it more difficult to restore the master device from a system failure. To make sure that the master device contains as few extraneous user databases as possible, create a new device using disk init. Use sp_diskdefault to specify the new device as the default before you configure Adaptive Server as a companion for failover. For example, to add a new default device named money_default1 to the MONEY1 Adaptive Server, enter:

```
sp_diskdefault money1_default1, defaulton
```

The master device continues to be a default device until you specifically issue this command to suspend it as the default device:

```
sp_diskdefault master, defaultoff
```

See the *Adaptive Server Reference Manual* for more information about disk init and sp_diskdefault.

Add local server to *sysservers*

Using sp_addserver, add the local server as the local server in sysservers using the network name specified in the *interfaces file*. For example, if the companion MONEY1 uses the network name of MONEY1 in the *interfaces file*:

```
sp_addserver MONEY1, local, MONEY1
```

You must reboot Adaptive Server for this change to take effect.

Add secondary companion to *sysservers*

Add the secondary companion as a remote server in sysservers:

```
sp_addserver server_name
```

By default, Adaptive Server adds the server with an srvid of 1000. You do not need to reboot Adaptive Server for the change to take effect.

Run *installhasvss* to install HA stored procedures

Note You must perform the tasks described in "Add entries for both Adaptive Servers to the interfaces file" on page 52, above, before running *installhasvss*. If you run *installhasvss* before performing these tasks, you must re-run *installmaster* to re-install all the system stored procedures.

The *installhasvss* script performs the following tasks to configure Adaptive Server for failover:

- Installs the stored procedures required for failover (for example, sp_companion).
- Installs the SYB_HACMP server in sysservers.

You must have System Administrator privileges to run the *installhasvss* script.

installhasvss is located in the *$SYBASE/$SYBASE_ASE/scripts* directory. To execute the *installhasvss* script, enter:

```
$SYBASE/$SYBASE_OCS/bin/isql -Usa -Ppassword -Sservername <
$SYBASE/$SYBASE_ASE/scripts/installhasvss
```

installhasvss prints messages as it creates stored procedures and creates the SYB_HACMP server.

Assign *ha_role* to system administrator

You must have the ha_role on both Adaptive Servers to run sp_companion. To assign the ha_role, issue the following from isql:

```
sp_role "grant", ha_role, sa
```

You must log out and then log back in to the Adaptive Server for the change to take effect.

Verify configuration parameters

You must enable the following configuration parameters before you configure Adaptive Server for failover:

- enable CIS – Enables Component Integration Services (CIS). This configuration parameter is enabled by default.
- enable xact coordination – Enables Distributed Transaction Management (DTM). This configuration parameter is enabled by default.
- enable HA – Enables Adaptive Server to function as a companion in a high availability system. enable HA is off by default. This configuration is static, so you must reboot Adaptive Server for it to take effect. This parameter causes a message to be written to your error log stating that you have started the Adaptive Server in a high availability system.

See the *System Administration Guide* for information about enabling configuration parameters.

Configuring HP for failover

This section describes the steps for preparing your HP MC/ServiceGuard high availability subsystem for Sybase's failover. This section assume that:

- You are familiar with HP MC/ServiceGuard.
- You have configured a two-node cluster hardware for MC/ServiceGuard
- You have installed HP MC/ServiceGuard version 11.05 on both nodes running under HPUX 11.0.
- The cluster system has been installed and configured.
- You have set up volume groups to contain all the database devices in the cluster on the shared disk devices.
- All the shared volume groups are already part of the cluster configuration.

See your HP documentation *Managing MC/ServiceGuard* for more information about installing, configuring, and managing MC/ServiceGuard.

Create the package configuration

The package configuration process defines the Adaptive Server and associated resources run by the package manager when a package starts on a node in the cluster. The package configuration also includes a prioritized list of cluster nodes on which the package runs and defines the different types of failover the package allows. You must define a package for each companion server.

Note The name of the Adaptive Server specified in the *interfaces file* must be the same as the name of the HP MC/ServiceGuard package.

For example, for the companion servers described in this manual, you would create a package named MONEY1 for primary companion MONEY1 and another package named PERSONNEL1 for secondary companion PERSONNEL1.

Note You can use either SAM or MC/ServiceGuard commands to create and customize your package configuration file. See the *HP MC/ServiceGuard* document for information on how to use SAM to perform these operations. This document describes the steps using MC/ServiceGuard commands.

As root, perform the following steps for both the primary and secondary companions:

1 Create a subdirectory on the primary node in the */etc/cmcluster* directory to contain the package information for your primary companion. For example, to create a directory for primary companion MONEY1:

```
mkdir /etc/cmcluster/MONEY1
```

2 Change the permissions for this directory so it is only accessible by root:

```
chmod 700 /etc/cmcluster/MONEY1
```

3 Create the same subdirectory on the secondary node. For example, to create this directory on machine FIN1 for primary companion MONEY1:

```
rsh FIN1 "mkdir /etc/cmcluster/MONEY1"
```

4 Change the permissions for this directory so it is only accessible by root:

```
rsh FIN1 chmod 700 /etc/cmcluster/MONEY1
```

5 Generate a package configuration template for the primary companion using the cmmakepkg command. This command uses the following syntax:

```
/usr/sbin/cmmakepkg -p
/etc/cmcluster/subdirectory_name/companion_name.ascii
```

where *subdirectory_name* is the name of the subdirectory you created in step 1, and *companion_name* is the name of the companion for which you are configuring the package. For example, to create a package configuration template for primary companion, MONEY1:

```
/usr/sbin/cmmakepkg -p
/etc/cmcluster/MONEY1/MONEY1.ascii
```

6 Edit the configuration template file you just created so it specifies the package name, a prioritized list of nodes, the location of the control script, and the failover parameters for each package.

The following are the edits made to the MONEY1.ascii configuration file (your edits will be different):

```
PACKAGE_NAME                    MONEY1
FAILOVER_POLICY                 CONFIGURED_NODE
FAILBACK_POLICY                 MANUAL
NODE_NAME                       FIN1
NODE_NAME                       HUM1
RUN_SCRIPT                      /etc/cmcluster/MONEY1/MONEY1.cntl
HALT_SCRIPT                     /etc/cmcluster/MONEY1/MONEY1.cntl
SERVICE_NAME                    MONEY1
SERVICE_FAIL_FAST_ENABLED       NO
SERVICE_HALT_TIMEOUT            300
```

Copy this file to the subdirectory on the second node you created in step 3. For example, to copy the *MONEY1.ascii* file using rcp:

```
rcp /etc/cmcluster/MONEY1/MONEY1.ascii HUM1:/etc/cmcluster/MONEY1/MONEY1.ascii
```

Edit the *ASE_HA.sh* script

The *ASE_HA.sh* template script configures the high availability subsystem to start, stop, and monitor Adaptive Server for failover. The *ASE_HA.sh* template script is included in the *$SYBASE/$SYBASE_ASE/install* directory. Make a copy of this script in the package subdirectory you created in step 1, above, and modify it to include the environment variables for your cluster environment. Both the primary and secondary companions require a copy of this script. As root, perform the following steps:

1 If you are currently using a script to configure Adaptive Server applications to run in your high availability system, make a backup copy of this file. For example, if you have a script named *SYBASE1.sh*, copy it to *SYBASE1.sh.backup*. Otherwise proceed to step 2.

2 On the primary node, change to the package subdirectory under */etc/cmcluster*. For example, if you are configuring the primary companion MONEY1:

```
cd /etc/cmcluster/MONEY1
```

3 Copy the *ASE_HA.sh* template script from the *$SYBASE/$SYBASE_ASE/install* directory to the primary companion's package subdirectory. Use the following syntax for the package template name:

```
<package_name>.sh
```

where *package_name* is the name of the companion server you are configuring. For example, to make a copy of the *ASE_HA.sh* file for MONEY1:

```
cp ASE_HA.sh /etc/cmcluster/MONEY1/MONEY1.sh
```

4 Edit the *server_name.sh* file for your environment. Edit the lines that include "__FILL_IN__" (and any other lines that require editing for your site). This is a list of these lines:

- ASE_12_0 – specifies the version of Adaptive Server. Set this to:
  - "yes" if both servers are using Sybase Adaptive Server version 12.0 or greater,
  - "no" if you are using earlier versions of Adaptive Server.
- ASE_HAFAILOVER – specifies whether you are using Sybase failover. Set this to:
  - "yes" if you are using Sybase failover,
  - "no" if you are using mode 0 failover.
- BASIC_FAILOVER – is set to either "yes" or "no:"
  - "yes" - Use the failover mechanisms provided by the HP MC/ServiceGuard high availability subsystem if it determines the servers are running in modes that allow failover. When a failover occurs, the script first checks if the companions are in a correct mode to perform a failover. If the companions are not enabled for Sybase's failover (that is, they are running in single-server mode), the script attempts to start up the primary companion on the secondary node.

  - "no" - Do not revert to mode 0 failover. That is, if BASIC_FAILOVER is set to no, failover does not happen at either the node or the companion level.

- PACKAGE_NAME - the name of the package as specified in the MC/ServiceGuard package configuration script.

Note The value of the PACKAGE_NAME must be the same as the companion name. For example, if the PRIM_SERVER value is MONEY1, the value of the PACKAGE_NAME must be MONEY1 as well.

- MONITOR_INTERVAL – The amount of time – in seconds – this scripts waits between checks to see if the Adaptive Server process is alive.
- SHUTDOWN_TIMEOUT - The maximum amount of time – in seconds – to wait for a companion server abort to complete before killing the SYBASE Adaptive Server process. SHUTDOWN_TIMEOUT protects a hung companion server that prevents the halt script from completing. The value of SHUTDOWN_TIMEOUT must be less than the time out variable set in the package configuration file
- RECOVERY_TIMEOUT – is the maximum amount of time the high availability subsystem waits, in seconds, before determining that the companion failed to start. Set this number long enough for a loaded companion to reboot. RECOVERY_TIMEOUT is also used as the maximum amount of time the subsystem waits for failover and failback to complete.
- SYBASE - The location in which the Sybase products are installed. This value is automatically set to PRIM_SYBASE if you are on primary host and to SEC_SYBASE if you are on the secondary host

- *SYBASE_ASE* – is the installation directory of Sybase Adaptive Server products. The default is ASE-12_0, change it to ASE-12_5.

  Note The original lines are:

  ```
  export SYBASE_ASE = ASE-12_0
  export SYBASE_OCS = ASE-12_0
  ```

  You must change these lines to:

  ```
  export SYBASE_ASE = ASE-12_5
  export SYBASE_OCS = ASE-12_5
  ```

- *SYBASE_OCS* – The installation directory for Sybase Open Client products. The default is OCS-12_0, change it to ASE-12_5.
- *SYBUSER* - The name of the user who starts the Adaptive Server session.
- *HALOGIN* – The login of the user with the sa_role and ha_role. This must be the same on both the primary and secondary companion.
- *HAPWD* – The password for the HA_LOGIN. This must be the same on both the primary and secondary companion.

  Note The HA_LOGIN and HA_PWD must be the same name and password used when configuring Adaptive Server as a companion server (that is when running sp_companion).

- *PRIM_SYBASE* – The path to the directory in the primary node in which the Adaptive Server products are installed. If you are using local devices, the location must be the same on both nodes. If you are using a shared device, this location must be different on both nodes.
- *PRIM_ASE_HOME* - The path to the directory in which the Adaptive Server products are installed on the primary node. The default is *$SYBASE/$SYBASE_ASE*
- *PRIM_SERVER* – The name of the primary companion.
- *PRIM_HOSTNAME* – The name of the primary node.
- *PRIM_CONSOLE_LOG* – The full path to the error log for the current primary companion session. This can be any file that has sufficient space and is writable by SYBUSER. The default is *$SYBASE/$SYBASE_ASE/install/server_name.cs_log*.

- *PRIM_RUNSCRIPT* – The name of the RUNSERVER file that is used to bring up the primary companion. The default is *$SYBASE/$SYBASE_ASE/install/RUN_server_name.*
- *SEC_SYBASE* – The directory in which the Adaptive Server products are installed on the secondary node. If you are using local devices, the location must be the same on both nodes. If you are using a shared device, this location must be different on both nodes.
- *SEC_ASE_HOME* - The path to the directory in which the Adaptive Server products are installed on the secondary node. The default is *$SYBASE/$SYBASE_ASE.*
- *SEC_SERVER* – The name of the secondary companion.
- *SEC_HOSTNAME* – The name of the secondary node.
- *SEC_CONSOLE_LOG* – The full path to the error log for the current secondary companion session. This can be any file that has sufficient space and is writable by SYBUSER. The default is *$SYBASE/$SYBASE_ASE/install/server_name.cs_log.*
- *ISQL* – The path to the isql binary. The default is *$SYBASE/$SYBASE_OCS/bin/isql.*

Table 7-1 shows the settings in MONEY1.sh for the primary companion MONEY1 running on host FIN1, and for the secondary companion PERSONNEL1, running on host HUM1. Both use a local file system. During failover, MONEY1 restarts on HUM1 if PERSONNEL1 is down or not in companion mode.

Table 7-1: Settings for MONEY1 in the ASE_HA.sh script

Variable	Setting
ASE_12_0 or higher	yes
ASE_HAFAILOVER	yes
BASIC_FAILOVER	yes
PACKAGE_NAME	MONEY1
MONITOR_INTERVAL	5
SHUTDOWN_TIMEOUT	60
RECOVERY_TIMEOUT	300
SYBASE_ASE	ASE-12_5
SYBASE_OCS	OCS-12_5
HALOGIN	“SA”
HAPASSWD	“Odd2Think
PRIM_SYBASE	*/opt/sybase*

Variable	Setting
PRIM_SERVER	MONEY1
PRIM_HOSTNAME	FIN1
PRIM_CONSOLE_LOG	*$PRIM_SYBASE/$SYBASE_ASE/install/MONEY1.cs_log*
SEC_SYBASE	*/opt/sybase*
SEC_SERVER	PERSONNEL1
PRIM_HOSTNAME	HUM1
SEC_CONSOLE_LOG	*$PRIM_SYBASE/$SYBASE_ASE/install/PERSONNEL1.cs_log*

5 Change the permission on the file to 700 so it is only readable, writable, and executable by root. For example, to change permissions for *MONEY1.sh*:

```
chmod 700 MONEY1.sh
```

6 Distribute the script to the secondary node. For example, to distribute the file to the secondary node HUM1:

```
rcp /etc/cmcluster/MONEY1/MONEY1.sh
HUM1:/etc/cmcluster/MONEY1/MONEY1.sh
```

7 Repeat the above steps for the secondary companion.

The secondary companion package script uses values for PRIM_SERVER, PRIM_HOST, PRIM_SYBASE, SEC_SERVER, SEC_HOST, and SEC_SYBASE that are the opposite of the primary companion package script. Table 7-2 shows values for *PERSONNEL1.sh*.

Table 7-2: Settings for PERSONNEL1 in the ASE_HA.sh script

Variable	Setting
ASE_12_0 or higher	yes
ASE_HAFAILOVER	yes
BASIC_FAILOVER	yes
PACKAGE_NAME	MONEY1
MONITOR_INTERVAL	5
SHUTDOWN_TIMEOUT	60
RECOVERY_TIMEOUT	300
SYBASE_ASE	ASE-12_5
SYBASE_OCS	OCS-12_5
HALOGIN	“SA”
HAPASSWD	“Odd2Think
PRIM_SYBASE	*/opt/sybase*

Variable	Setting
PRIM_SERVER	PERSONNEL1
PRIM_HOSTNAME	HUM1
PRIM_CONSOLE_LOG	*$PRIM_SYBASE/$SYBASE_ASE/install/MONEY1.cs_log*
SEC_SYBASE	*/opt/sybase*
SEC_SERVER	MONEY1
PRIM_HOSTNAME	FIN1
SEC_CONSOLE_LOG	*$PRIM_SYBASE/$SYBASE_ASE/install/PERSONNEL1.cs_log*

Create the package control script

The package control script contains the information necessary to:

- Run the companion servers in the package
- Monitor the companion servers
- Respond to failure
- Halt the package

For security reasons, the control script must reside in a directory that includes *cmcluster* in its path.

Each package requires a separate control script. The control script placed in the package subdirectory under */etc/cmcluster* is given the same name that it has in the package configuration file. It must be executable.

Perform the following as root:

1 Use the cmmakepkg utility to generate a package control script template for the primary companion in the same directory you created. The cmmakepkg utility uses the following syntax:

```
/usr/sbin/cmmakepkg -s /etc/cmcluster/package_name/companion_name.cntl
```

where *package_name* is the name of the directory you created and *companion_name* is the name of the companion you are configuring.

For example, to create a package control script for primary companion MONEY1:

```
/usr/sbin/cmmakepkg -s
/etc/cmcluster/MONEY1/MONEY1.cntl
```

2 Edit the package control script to reflect your cluster environment

Follow the steps below to edit your package control script:

1. Define the volume groups that are used by this companion server package:

   ```
   VG[0]=""
   ```

   For example, if primary companion MONEY1 uses volume group *ha_vg1*, enter the following:

   ```
   VG[0]="ha_vg1"
   ```

2. If you are using a shared file system, define the logical volumes and file system in the following line in the FILESYSTEMS section of the script:

```
LV[0]="";FS[0]="", FS_MOUNT_OPT[0]="-Fvxfs -o rw, suid, log, mincache,
dync, blkclear, detainlog, largefiles"
```

   For example, if primary companion MONEY1 has data on a *ha_fs1* file system on logical volume *ha_lv1*:

```
LV[0]="ha_lv1";FS[0]="/ha_fs1", FS_MOUNT_OPT[0]=""
```

3. Enter the command to halt the companion service inside the customer_defined_halt_cmds function. This command includes the location of the *ASE_HA.sh* file (described in "Edit the ASE_HA.sh script" on page 59). Before editing, this section looks similar to:

```
function customer_defined_halt_cmds
{
# ADD customer defined run commands.
: # do nothing instruction, because a function must contain some command.

test_return 52
}
```

   Edit the function to include the halt command. For example, to include the halt command for companion MONEY1:

```
function customer_defined_halt_cmds
{
# ADD customer defined run commands.
: # do nothing instruction, because a function must contain some command.

/etc/cmcluster/MONEY1/MONEY1.sh halt
test_return 52
}
```

4 Move to the START OF CUSTOMER DEFINED FUNCTIONS section of *companion_name.cntl* and enter the command to start the companion service. Enter this command inside the customer_defined_run_cmds function. This command includes the location of the *ASE_HA.sh* file (described in "Edit the ASE_HA.sh script" on page 59). Before editing this section looks similar to:

```
function customer_defined_run_cmds
{
# ADD customer defined run commands.
: # do nothing instruction, because a function must contain some command.

test_return 51
}
```

Edit the function to include the start command. For example, to include the start command for companion MONEY1:

```
function customer_defined_run_cmds
{
# ADD customer defined run commands.
: # do nothing instruction, because a function must contain some command.

/etc/cmcluster/MONEY1/MONEY1.sh start
test_return 51
}
```

5 Define the script that monitors the server process as a service in the SERVICE NAMES AND COMMANDS section of the script:

```
SERVICE_NAME[0]=""
SERVICE_CMD[0]=""
SERVICE_RESTART[0]=""
```

For example, configure monitoring for primary companion MONEY1:

```
SERVICE_NAME[0]="MONEY1"
SERVICE_CMD[0]="/etc/cmcluster/MONEY1/MONEY1.sh monitor"
SERVICE_RESTART[0]="-R"
```

6 Distribute the script to each node in the cluster. For example, to distribute the script to the secondary node HUM1:

```
# rcp /etc/cmcluster/MONEY1/MONEY1.cntl
HUM1:/etc/cmcluster/MONEY1/MONEY1.cntl
```

7 Repeat these steps for the secondary companion.

Verify and distribute the configuration

Perform the following steps to verify and distribute the configuration.

1. Use the cmcheckconf utility to verify that the package configuration file is correct. cmcheckconf uses the following syntax:

```
cmcheckconf -C /etc/cmcluster/cmclconfig.ascii -P
/etc/cmcluster/package_name/primary_companion_name.ascii
-p /etc/cmcluster/secondary_package_name/secondary_companion_name.ascii
```

   where *package_name* is the name of the directory you created, *primary_companion_name* is the name of the companion you are configuring, and *secondary_companion_name* is the name of its secondary companion. For example, to verify the package configuration file for MONEY1:

```
cmcheckconf -C /etc/cmcluster/cmclconfig.ascii -P
/etc/cmcluster/MONEY1/MONEY1.ascii
-p /etc/cmcluster/PERSONNEL1/PERSONNEL1.ascii
```

2. Perform the following steps to distribute the binary cluster configuration file:

   a. Issue the vgchange command to activate the cluster lock volume group so that the lock disk can be initialized:

```
/usr/sbin/vgchange -a y /dev/vglock
```

   b. Use the cmapplyconf utility to generate the binary configuration file and distribute it across the nodes. cmapplyconf uses the following syntax:

```
/usr/sbin/cmapplyconf -v -C
/etc/cmcluster/cmclconf.ascii -P
/etc/cmcluster/primary_package_name/primary_com
panion_name.ascii
-p
/etc/cmcluster/secondary_package_name/secondary
_companion_name.ascii
```

   where *primary_package_name* is the name of the directory you created, *primary_companion_name* is the name of the companion you are configuring, and similar definitions for *secondary_package_name* and *secondary_companion_name*. For example, to generate a binary configuration file for MONEY1:

```
 # cmapplyconf -v -C /etc/cmcluster/cmclconf.ascii -P
/etc/cmcluster/MONEY1/MONEY1.ascii
-p /etc/cmcluster/PERSONNEL1/PERSONNEL1.ascii
```

c Issue the vgchange command to deactivate the cluster lock volume group:

```
/etc/sbin/vgchange -a n /dev/vglock
```

Note The cluster lock volume group must be activated only on the node from which you issue the cmapplyconf command so that the lock disk can be initialized. When you configure the cluster, the cluster lock volume group must be active only on the configuration node and deactivated on all other nodes. Make sure you deactivate the cluster lock volume group on the configuration node after cmapplyconf is executed.

Note You must run cmcheckconf and cmapplyconf any time you make changes to the cluster and package configuration files.

Start up both the primary and secondary companions

At this point, you are ready to start the package which starts and monitors the Adaptive Server. As root, start the primary companion using the following syntax:

```
/usr/sbin/cmrunpkg -n node_name primary_companion_name
```

For example, to start primary companion MONEY1 on node FIN1:

```
/usr/sbin/cmrunpkg -n FIN1 MONEY1
```

Start the secondary companion using the same command and syntax.

Configuring companion servers for failover

Perform the tasks in this section to configure the Adaptive Servers as primary and secondary companions in a high availability system.

Run *sp_companion* with *do_advisory* option

You must configure the secondary companion with sufficient resources to perform the work of both servers during failover. The secondary companion may have attributes that will prevent a successful cluster operation. For example, if both the primary and secondary companions are configured for 250 user logins, during failover, the secondary companion only has the resources for half the number of potential user logins necessary. Instead, both MONEY1 and PERSONNEL1 should be configured for 500 user logins.

The sp_companion do_advisory option checks the configuration options on both the primary and the secondary companion to make sure a cluster operation (such as configuring an Adaptive Server as a secondary companion) will be successful. sp_companion do_advisory advises you of any configuration options that should be changed.

See Chapter 6, “Running do_advisory” for a complete description of the sp_companion do_advisory option.

Configure for asymmetric configuration

Configure the primary companion for asymmetric configuration. Issue the following command from the secondary companion:

```
sp_companion "primary_server_name", configure, with_proxydb, login_name,
password
```

- *primary_server_name* is the name of the primary Adaptive Server as defined in the *interfaces file* entry and in sysservers.
- *with_proxydb* indicates that proxy databases are created on the secondary companion for all databases other than system databases. Any subsequent databases that are added also create proxy databases.
- *login_name* is the name of the user performing this cluster operation (this person must have the ha_role).
- *password* is the password of the person performing this cluster operation

Note You must execute the above command ONLY from the secondary companion.

This example configures an Adaptive Server named MONEY1 as a primary companion. Issue the following command from the secondary server PERSONNEL1:

```
sp_companion "MONEY1", configure, with_proxydb, sa, Odd2Think
Server 'PERSONNEL1' is alive and cluster configured.
Step: Access verified from Server:'PERSONNEL1' to Server:'MONEY1'
Server 'MONEY1' is alive and cluster configured.
Step: Access verified from Server:'MONEY1' to Server:'PERSONNEL1'
(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)
.....
Step: Companion servers configuration check succeeded
Step: Server handshake succeeded
Step: Master device accessible from companion
Step: Added the servers 'PERSONNEL1' and 'MONEY1' for cluster config
Step: Server configuration initialization succeeded
Step: Synchronizing server logins from companion server
Step: Synchronizing remoteserver from companion server
Step: Synchronizing roles from companion server
Step: Synchronizing server-wide privs from companion server
Step: User information syncup succeeded
Step: Server configured in normal companion mode
```

If user databases are created during the sp_companion configuration, you see messages similar to these:

```
Step: Created proxy database 'pubs2'
 Step: Proxy status for database has been set. Please Checkpoint the database
'pubs2'
 Step: Server configured in normal companion mode"
 Starting companion watch thread
```

Before you configure the companions for symmetric configuration, you must first configure them for asymmetric configuration.

See "Asymmetric companion configuration" on page 19 for more information about asymmetric configuration.

Configure for symmetric configuration

After you configure your companions for asymmetric failover, you can configure them for symmetric configuration. In a symmetric configuration, both servers act as primary and secondary companions. See Figure 3-2 on page 22 for a description of symmetric configuration.

Issue sp_companion from the secondary companion to configure it for symmetric configuration. Use the same syntax as for asymmetric configuration. See "Configure for asymmetric configuration," above, for a description of the syntax for sp_companion.

The following example adds an Adaptive Server named MONEY1 as the secondary companion to the Adaptive Server named PERSONNEL1 described in "Configure for asymmetric configuration" on page 70. Issue the following command from the MONEY1 server:

```
sp_companion 'PERSONNEL1', configure, with_proxydb, sa, Think2Odd
Server 'MONEY1' is alive and cluster configured.
Step: Access verified from Server:'MONEY1' to Server:'PERSONNEL1'
Server 'PERSONNEL1' is alive and cluster configured.
Step: Access verified from Server:'PERSONNEL1' to Server:'MONEY1'
(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)
.....
Step: Companion servers configuration check succeeded
Step: Server handshake succeeded
Step: Master device accessible from companion
Step: Added the servers 'MONEY1' and 'PERSONNEL1' for cluster config
Step: Server configuration initialization succeeded
Step: Synchronizing server logins from companion server
Step: Synchronizing remoteserver from companion server
Step: Synchronizing roles from companion server
Step: Synchronizing server-wide privs from companion server
Step: User information syncup succeeded
Step: Server configured in normal companion mode
```

Administering Sybase failover

This section describes information about:

- Failing back to the primary companion
- Resuming normal companion mode
- Suspending normal companion mode

- Troubleshooting common problems with Sybase's failover.

Failing back to the primary companion and resuming normal companion mode

Failback moves the primary companion's shared disks from the secondary node back to the primary node and starts the primary companion on the primary node. Failback is a planned event. To failback to the primary companion:

1 Issue sp_companion from the secondary companion to verify that it is in failover mode.

Note The high availability subsystem automatically restarts the primary companion.

2 Issue the following from the secondary companion:

```
sp_companion primary_companion_name,
prepare_failback
```

where *primary_companion_name* is the name of primary companion server.

For example, to fail back the primary companion MONEY1, issue the following from the secondary companion PERSONNEL1:

```
sp_companion MONEY1, prepare_failback
```

3 From the primary companion, issue:

```
sp_companion secondary_companion_name, resume
```

Where *secondary_companion_name* is the name of the secondary companion server. For example, to resume normal companion mode between primary companion MONEY1 and secondary companion PERSONNEL1:

```
sp_companion PERSONNEL1, 'resume'
```

4 Issue sp_companion without any options from either companion to make sure you are in normal companion mode.

Note You cannot connect clients with the failover property (for example isql -Q) until you issue sp_companion resume. If you do try to reconnect them after issuing sp_companion prepare_failback, the client hangs until you issue sp_companion resume.

Suspending companion mode

Suspended mode temporarily disables the ability of the primary companion to failover to the secondary companion. When you move the companions to suspended mode, synchronization between the companions does not occur, and the primary companion cannot failover to the secondary companion. However, suspended mode is very useful for performing such maintenance tasks as changing configuration parameters. Perform the following steps to switch from normal companion mode to suspended mode:

1 As root, issue cmhaltserv to disable the monitoring process so that it does not trigger a failover when you shut down the companion server:

```
cmhaltserv -v primary_package_name
```

where *primary_package_name* is the name of the primary package, which is also the same as the name of the primary companion. For example, to disable the monitoring process for primary companion MONEY1:

```
cmhaltserv -v MONEY1
```

2 Move the companions from normal companion mode to suspended mode. Issue the following from the secondary companion:

```
sp_companion primary_server_name, suspend
```

For example, To suspend primary companion MONEY1, issue the following from secondary companion PERSONNEL1:

```
sp_companion MONEY1, suspend
```

You can now shut down the primary companion as necessary and it will not failover to the secondary companion.

Resuming normal companion mode from suspended mode

To resume normal companion mode between two companions that have been moved to suspended mode:

1 As root, issue cmhaltpkg from the primary node to shut down down the primary companion:

```
cmhaltpkg primary_package_name
```

where *primary_package_name* is the name of the primary package, which is the same as the name of the primary companion server. For example, to halt the MONEY1 package:

```
cmhaltpkg MONEY1
```

2 As root, issue cmmodpkg and cmrunpkg from the primary companion to run the package which restarts the primary companion:

```
cmmodpkg -e primary_package_name
cmrunpkg primary_package_name
```

where *primary_package_name* is the name of the primary package, which is the same as the name of the primary companion server. For example to run the MONEY1 package to restart the MONEY1 primary companion:

```
cmmodpkg -e MONEY1
cmrunpkg MONEY1
```

Dropping companion mode

To drop companion mode, issue:

```
sp_companion companion_name, "drop"
```

Dropping companion mode is an irreversible process; you must reconfigure the Adaptive Servers companion servers before they will failover in a high availability system and retain the functionality that Sybase's failover provides. However, the nodes upon which the Adaptive Servers are running are still monitored by the high availability subsystem.

If you drop the companion mode while the monitor script is running, the script continues to monitor the server for any down or hung instances. If you plan to shut down the server and do not want the node to failover, kill the monitor process by issuing:

```
/usr/sbin/cmhaltsrv service_name
```

For example, to halt the service for primary companion MONEY1:

```
/usr/sbin/cmhaltsrv MONEY1
```

Alternatively, you can halt the package, reactivate the volume group, and then restart the companion only.

If you do not kill the monitor process and it detects that the companion has gone down, it triggers a failover to the secondary node. It restarts the primary companion on the secondary node depending on your settings for BASIC_FAILOVER.

Troubleshooting Sybase failover on HP

This section includes troubleshooting information about common errors.

Error message 18750

If a companion server issues error message 18750, check the *@@cmpstate* of the servers. If the primary companion is in normal companion mode, but the secondary companion is in secondary failover mode, the cluster is in an inconsistent state, and you need to manually recover. This inconsistent state may be caused by an sp_companion 'prepare_failback' command failing on the secondary companion. You can determine whether this happened by examining the log on the secondary node. To recover, perform the following steps:

1. Shut down both the primary and the secondary companions by halting both their packages.
2. Reboot the secondary companion by starting the package for the secondary companion.
3. Repair all databases marked “suspect.” To determine which databases are suspect, issue:

   ```
   select name, status from sysdatabases
   ```

   Databases marked suspect have a status value of 320.
4. Allow updates to system tables:

   ```
   sp_configure “allow updates”, 1
   ```
5. For each suspect, failed-over database, perform the following:

```
1> update sysdatabase set status=status-256 where name='database_name'
```

```
2> go
1> dbcc traceon(3604)
2> go
1> dbcc dbrecover(database_name)
2>go
```

6 From the secondary companion, issue:

```
sp_companion primary_companion_name, prepare_failback
```

For example, from secondary companion PERSONNEL1:

```
sp_companion MONEY1, prepare_failback
```

Make sure that this command executes successfully.

7 Resume normal companion mode. From the primary companion, issue:

```
sp_companion secondary_companion, resume
```

For example, from the primary companion MONEY1:

```
sp_companion PERSONNEL1, resume
```

Recovering from a failed *prepare_failback*

During a failback, if prepare_failback executed successfully on the secondary companion but the primary companion fails to boot, perform the following to rollback and then reissue the prepare_failback command:

1 Check the primary companion's error log, the HP MC/ServiceGuard package log, or the system log to find the reason the server failed to boot, and correct the problems.

2 If the package for the primary companion is running on the primary node, halt the package.

3 Login to the secondary companion and issue:

```
dbcc ha_admin ("", "rollback_failback")
dbcc ha_admin ("", "rollback_failover")
```

4 Verify that the secondary companion is in normal companion mode.

5 As root, start up the package for the primary companion to run on secondary node.

```
/usr/sbin/cmrunpkg -n secondary_node primary_companion_package_name
```

The secondary companion is now in failover mode. Once you verify that everything is ready for the primary companion to failback to normal companion mode, issue sp_companion...prepare_failback.

Location of error logs

Sybase's failover and HP MC/ServiceGuard includes the following error logs:

- */var/adm/syslogs/syslog.log* – contains the output of HP MC/ServiceGuard cluster-level activities as well as operating system level activities.
- */etc/cmcluster/<package_name>/<package_name>.cntl.log* – contains the output of the HP MC/ServiceGuard package activities and Sybase's failover activities from the companion start, stop, and monitor script.

  For output from the companion start, stop, and monitor script, search for "SYBASE HA".

  For MC/ServiceGuard package failure, search for the string "ERROR".
- *$PRIMARY_CONSOLE_LOG* - The location of this log is defined in */etc/cmcluster/<package_name>/<package_name>.sh*. This error log includes information from the last execution of Adaptive Server from the *ASE_HA.sh* script.

CHAPTER 8

Configuring Adaptive Server for Failover on IBM AIX

Perform the tasks in this chapter to configure Adaptive Server for failover on IBM AIX.

It includes the following sections:

Configuring hardware and operating system for high availability

Sybase high availability requires the following hardware and system components:

- Two homogenous, network systems with similar configurations in terms of resources like CPU, memory, and so on.
- High availability subsystem package and the associated hardware.
- Devices that are accessible to both nodes.
- A logical volume manager (LVM) to maintain unique device pathnames across the cluster nodes.
- Vendor provided mirroring, not Sybase mirroring, used for media failure protection.

See your hardware and operating system documentation for information about installing platform specific high availability software.

Requirements for running Sybase's failover on IBM AIX

Configuring for high availability on IBM HACMP requires:

- 2 hardware-compatible nodes running HACMP for AIX, Version 4.2.2, that are configured in the same cluster.
- Each node must have 3 IP addresses, one for service, one for boot, and one for standby. The standby IP address should be on a different subnet from the other two.
- Shared disk devices set up for the high availability system between the nodes.
- Shared logical volume groups set up to contain all the database devices in the cluster. Make sure that both nodes have the same major number for each of the shared volume groups defined in the cluster. In this chapter, these resources are referred to as:
  - *shared_vg1* for the primary node
  - *shared_vg2* for the secondary node

  See the *HACMP for AIX Installation or Administration Guide* for information about installing the high availability subsystem.

  Sybase also recommends that you identify the following resources in advance.
- A shared volume group name for the primary node (for example, *shared_vg1*).
- A shared volume group name for the secondary node (for example, *shared_vg2*).
- A resource group name for the primary companion (for example, *resgrp1*).
- A resource group name for the secondary companion (for example, *resgrp2*).
- The name of the primary companion.
- The name of the secondary Adaptive Server companion.

Special considerations for running Adaptive Server on HACMP for AIX

When the primary companion fails over on HACMP 4.2.2, the entire node fails over, not just the primary companion. During this node failover, the IP address of the servicing host (the primary node) is swapped with another standby address. In some networking environments, this may cause all the processes on the initial IP address to freeze and eventually time out. Because of this, when you use Sybase's Failover with HACMP on AIX:

- Do not allow clients to log in directly to the primary node
- Limit the primary node to running only one high availability application at a time

Preparing Adaptive Server to work with the HA subsystem

Perform the tasks in this section to prepare Adaptive Server for a high availability configuration.

Install Adaptive Servers

Before you install Adaptive Server, start the HACMP services on the same node on which you are installing Adaptive Server. Make sure that the HACMP node is running on its service IP address and not the boot or standby IP address.

Install the primary and the secondary servers. You can install the companions on either local or shared file systems. If they are installed on shared file systems, the file system must not be the same. This is to prevent the file systems from overwriting each other during a device failover. For example, you can install the primary companion on *node1_sybase*, but install the secondary companion on */node2_sybase*.

If the servers are installed on local file system, the name of the file systems must be the same. For example, both the primary and the secondary companion could be installed in */sybase*.

The file systems that contain *$SYBASE* must be either local or shared; you cannot mix local and shared file systems for *$SYBASE* in the cluster.

The database devices for the primary companion must be devices in the shared volume group on the primary node (for example, *shared_vg1*), so the volume group for this node must be "varied on."

If you are creating an asymmetric configuration, you can use any device (shared or local) for the database device. If you are creating a symmetric configuration, you must use a device in the shared volume group on the secondary node (for example, *shared_vg2*) for its database devices, so the volume group for this node must be "varied on."

The primary companion can be either a newly installed Adaptive Server, or it can be upgraded from a previous version of Adaptive Server with existing databases, users, and so on.

The secondary companion must be a newly installed Adaptive Server without any user logins or user databases. This ensures that all user logins and database names are unique within the cluster. After configuration for failover is complete, you can add user logins and databases to the secondary companion.

See the installation documentation for your platform for information about installing and configuring Adaptive Server.

Add entries for both Adaptive Servers to the *interfaces file*

The *interfaces file* for the primary and the secondary companion must include entries for both companions. For example, the *interfaces file* for the servers used in the examples above would have entries for both MONEY1 and PERSONNEL1. The server entry in the *interfaces file* must use the same network name that is specified in sysservers. For information about adding entries to the *interfaces file*, see the installation documentation for your platform.

Add entries to *interfaces* file for client connections during failover

To enable clients to reconnect to the failed over companion, you must add an additional line to the *interfaces file*. By default, clients connect to the port listed in the query line of the server entry. If that port is not available (because that server has failed over), the client connects to the server listed in the *hafailover* line of the server entry. Here is a sample *interfaces file* for a primary companion named MONEY1 and a secondary companion named PERSONNEL1:

```
MONEY1
        master tcp ether FIN1 4100
```

```
query tcp ether FIN1 4100
hafailover PERSONNEL1
```

Use dsedit to add entries to the *interfaces file*. If the interfaces entries already exist, you must modify them to work for failover.

See the Utility Guide for your platform for information about dsedit.

Set the value of *$SYBASE*

If you installed *$SYBASE* on a local file system, *$SYBASE* must point to the same directory name on both companions. You can accomplish this by either:

- Making sure that the *$SYBASE* release directory on each companion is created in the same directory, or
- If the companions have the *$SYBASE* release directory in different locations, create a directory with the same path on both companions that acts as a symbolic link to the actual *$SYBASE* release directory.

  For example, even though primary companion MONEY1 has a release directory of */usr/u/sybase1* and PERSONNEL1 has uses */usr/u/sybase2* as its release directory, *$SYBASE* must point to the same path.

  Both MONEY1 and PERSONNEL1 have */SYBASE*, which they establish as a symbolic link to their respective *$SYBASE* release directories. On MONEY1, */SYBASE* is a link to */usr/u/sybase1*, and on PERSONNEL1, */SYBASE* is a link to */use/u/sybase2*.

If you installed *$SYBASE* on a local file system, you must also have copies of both companion *RUNSERVER* files in *$SYBASE/$SYBASE_ASE/install* on both nodes.

sybha executable

The *sybha* executable provides the ability for the Adaptive Server High Availability Basic Services library to interact with each platform's high availability cluster subsystem. The Adaptive Server High Availability Basic Services library calls *sybha*, which is located in *$SYBASE/ASE-12_5/bin*. Before *sybha* can run, you must change its ownership and permissions. You must also edit a file named *sybhauser* in *$SYBASE/ASE-12_5/install*. *sybhauser* contains a list of the users who have System Administrator privileges on the cluster. Sybase strongly recommends that you severely limit the number of users who have System Administrator privileges on the cluster.

As root, perform the following:

1 Add a new group named *sybhagrp*. You can either add this group to the */etc/group* file, or you can add it to your NIS maps. Add the sybase user to this group (this is the user that owns the *$SYBASE* directory). When the server is started, the sybase user runs the data server. If you have multiple servers running and different users owning the *$SYBASE* directory for each of them, each of these users must be added to the group.

2 Change to the *$SYBASE/$SYBASE_ASE/bin* directory:

```
cd $SYBASE/$SYBASE_ASE/bin
```

3 Change the ownership of *sybha* to root:

```
chown root sybha
```

4 Change the group for the *sybha* program to *sybhagrp*:

```
chgrp sybhagrp sybha
```

5 Modify the file permissions for *sybha* to 4550:

```
chmod 4550 sybha
```

6 Change to the *$SYBASE/$SYBASE_ASE/install* directory:

```
cd $SYBASE/ASE-12_5/install
```

7 Add the sybase user to the *sybhauser* file. These logins must be in the format of UNIX login IDs, not Adaptive Server logins. For example:

```
sybase
coffeecup
spooner
venting
howe
```

8 Change the ownership of *sybhauser* to root:

```
chown root sybhauser
```

9 Modify the file permissions for *sybhauser*:

```
chmod 600 sybhauser
```

Verify configuration parameters

You must enable the following configuration parameters before you configure Adaptive Server for failover:

- enable CIS – Enables Component Integration Services (CIS). This configuration parameter is enabled by default.
- enable xact coordination – Enables Distributed Transaction Management (DTM). This configuration parameter is enabled by default.
- enable HA – Enables Adaptive Server to function as a companion in a high availability system. enable HA is off by default. This configuration is static, so you must reboot Adaptive Server for it to take effect. This parameter causes a message to be written to your error log stating that you have started the Adaptive Server in a high availability system.

See the *System Administration Guide* for information about enabling configuration parameters.

Add thresholds to the master log

If you have not already done so, you must add a threshold to the master log.

1 Define and execute sp_thresholdaction on the master database's log to set a threshold on the number of pages left before a dump transaction occurs. Sybase does not supply sp_thresholdaction. See the *Adaptive Server Reference Manual* for information about creating this system procedure.

2 Place thresholds on the master and sybsystemprocs log segments so they do not fill up:

```
sp_addthreshold "master", "logsegment", 250, sp_thresholdaction
sp_addthreshold "sybsystemprocs", "logsegment", 250, sp_thresholdaction
```

3 You must reboot the primary companion for this static parameter to take effect.

Create new default device other than master

By default, the master device is the default device in a newly installed Adaptive Server. This means that, if you create any databases (including the proxy databases used by failover), they are automatically created on the master device. However, adding user databases to master makes it more difficult to restore the master device from a system failure. To make sure that the master device contains as few extraneous user databases as possible, create a new device using disk init. Use sp_diskdefault to specify the new device as the default before you configure Adaptive Server as a companion for failover.

For example, to add a new default device named money_default1 to the MONEY1 Adaptive Server, enter:

```
sp_diskdefault money1_default1, defaulton
```

The master device continues to be a default device until you specifically issue this command to suspend it as the default device:

```
sp_diskdefault master, defaultoff
```

See the *Adaptive Server Reference Manual* for more information about disk init and sp_diskdefault.

Add local server to *sysservers*

Using sp_addserver, add the local server as the local server in sysservers using the network name specified in the *interfaces file*. For example, if the companion MONEY1 uses the network name of MONEY1 in the *interfaces file*:

```
sp_addserver MONEY1, local, MONEY1
```

You must reboot Adaptive Server for this change to take effect.

Add secondary companion to *sysservers*

Add the secondary companion as a remote server in sysservers:

```
sp_addserver server_name
```

By default, Adaptive Server adds the server with an srvid of 1000. You do not need to reboot Adaptive Server for the change to take effect.

Run *installhasvss* to install HA stored procedures

Note You must perform the tasks described in "Add entries for both Adaptive Servers to the interfaces file" on page 82, before running *installhasvss*. If you run *installhasvss* before performing these tasks, you must re-run *installmaster* to re-install all the system stored procedures.

The *installhasvss* script performs the following tasks to configure Adaptive Server for failover:

- Installs the stored procedures required for failover (for example, sp_companion).
- Installs the SYB_HACMP server in sysservers.

You must have System Administrator privileges to run the *installhasvss* script.

installhasvss is located in the *$SYBASE/$SYBASE_ASE/scripts* directory. To execute the *installhasvss* script, enter:

```
$SYBASE/$SYBASE_OCS/bin/isql -Usa -Ppassword -Sservername <
../scripts/installhasvss
```

installhasvss prints messages as it creates stored procedures and creates the SYB_HACMP server.

Assign *ha_role* to system administrator

You must have the ha_role on both Adaptive Servers to run sp_companion. To assign the ha_role, issue the following from isql:

```
sp_role "grant", ha_role, sa
```

You must log out and then log back in to the Adaptive Server for the change to take effect.

Configure the IBM AIX Subsystem for Sybase Failover

Perform the steps in this section to configure IBM AIX for Failover.

Modify the *ASE_HA.sh* Script

The *ASE_HA.sh* script is used to start, stop, and monitor Adaptive Server in a high availability environment. Adaptive Server includes this script in the *$SYBASE/$SYBASE_ASE/install* directory. You must make a copy of this script and modify it for your environment for both Adaptive Servers in the cluster. The modifications you make to the script will differ, depending on whether the script is for the primary or secondary companion. Each node has a copy of this script at the same location (for example, both nodes have a copy of the script in */usr/u/sybase*), and both copies have read, write, and execute permissions for root. An easy way to do this is to first modify both scripts on the same node, copy both the scripts to the other node, and then set the appropriate permissions for the scripts on both nodes.

To modify the script for your environment:

1 Change to the *$SYBASE/$SYBASE_ASE/install* directory.

2 As root, copy *ASE_HA.sh* to the HACMP event handler script directory, usually */usr/sbin/cluster/events*, and name it:

```
RUNHA_<server_name>.sh
```

where *server_name* is the Adaptive Server to be monitored.

For example, to copy *ASE_HA.sh* for a server named MONEY1 to the */usr/sbin/cluster/events* directory, enter:

```
cp ASE_HA.sh /usr/sbin/cluster/events/RUNHA_MONEY1.sh
```

3 You must edit the *RUNHA_server_name.sh* script for your environment. The original *ASE_HA.sh* script contains the variables listed below. Edit the lines that include "__FILL_IN__" (and any other lines that require editing) with the values for your site:

- MONITOR_INTERVAL – The interval of time, in seconds, *RUNHA_server_name.sh* waits between checks to see if the data server process is alive.
- RECOVERY_TIMEOUT – The maximum amount of time the high availability subsystem waits, in seconds, before determining that the companion failed to start. Set this number long enough for a loaded companion to reboot. RECOVERY_TIMEOUT is also used as the maximum amount of time the subsystem waits for failover and failback to complete.

- SHUTDOWN_TIMEOUT – The maximum time the high availability subsystem waits for the companion to shut down before killing it.

  Note This value should always be less than the amount of time it takes for the HACMP wait time parameter to go into a config_too_long state. This is 360 seconds by default. If your server takes longer than this to boot up, reconfigure this value by executing:

  ```
  chssys -s clstrmgr -a "-u milliseconds_to_wait"
  ```

- RESPONSE_TIMEOUT – The maximum amount of time the subsystem allows for a simple query to return a result set is used to diagnose whether the companion server is hung. For example, if isql fails to establish a connection in 60 seconds, it automatically times out and exits. However, if isql successfully connects, but does not return a result set, RESPONSE_TIMEOUT may determine that the companion server is hung. By default, RESPONSE_TIMEOUT is set to 999999.
- ASE_FAILOVER – Set to either yes or no.
  - If yes - Monitors the companion server for hung or dead processes and stops HACMP services on this node so the devices failover to the secondary node. If set to “yes”, one must run sp_companion configure on the server as well to keep the high availability consistent.
  - If no - Do not bring down the HACMP subsystem on this node even if the primary companion fails over. This setting is useful if you need to bring down a companion for maintenance or re configuration.

    If you are configuring an asymmetric setup, set ASE_FAILOVER to “no.”

    Warning! This should only be set to “yes” if BOTH the servers are running Adaptive Server version 12.0 or later. Adaptive Servers from previous versions set to “no”.
- BASIC_FAILOVER – Set to either “yes” or “no”

  - If yes - Use the failover mechanisms provided by the HACMP subsystem if it determines the servers are running in modes that allow failover. When a failover occurs, the HACMP subsystem monitor first checks if the companions are in a correct mode to perform a failover. If the companions are not enabled for Sybase's failover (that is, they have enable ha set to 1), or they are running in single-server mode, or if the secondary companion is down, the HACMP subsystem monitor checks if BASIC_FAILOVER is set. If it is, the monitor attempts to start up the primary companion on the secondary node.
  - If no - Do not revert to mode 0 failover even if Sybase's failover criteria is not met. That is, if BASIC_FAILOVER is set to no, failover does not happen at either the node or the companion level.

- retry – The number of times the HACMP subsystem attempts to reboot on the local node before failing over. Set this to a high number for an asymmetric setup so the secondary companion is more likely to reboot itself if it ever goes down. The default is 0, which means that the companion will not reboot on the same node if it goes down.
- SYBASE_ASE – The installation directory of Sybase Adaptive Server products. The default is ASE-12_0; change this to ASE-12_5.
- SYBASE_OCS – The installation directory of Sybase Open Client products. The default is OCS-12_0; change this to OCS-12_5.

Note The original lines are:

```
export SYBASE_ASE = ASE-12_0
export SYBASE_OCS = ASE-12_0
```

You must change these lines to:

```
export SYBASE_ASE = ASE-12_5
export SYBASE_OCS = ASE-12_5
```

- PRIM_SERVER – The name of the primary companion.
- SEC_SERVER – The name of the secondary companion.
- PRIM_HOST – The name of the primary host or service interface name.
- SEC_HOST – The name of the secondary host or service interface.

- PRIM_SYBASE – The directory to which the $SYBASE environment variable should be set on the primary host. If you are using local devices, the location must be the same on both nodes. If you are using a shared device, this location must be different on both nodes.
- SEC_SYBASE – The directory to which the $SYBASE environment variable should be set on the secondary host. If you are using local devices, the location must be the same on both nodes. If you are using a shared device, this location must be different on both nodes.
- PRIM_SYBASE_HOME – The path to the directory in the secondary host in which the Adaptive Server products are installed. Usually this is *$SYBASE/$SYBASE_ASE*.
- PRIM_ISQL – The path to the isql binary on the primary host.
- SEC_ISQL – The path to the isql binary on the secondary host.
- HA_LOGIN – The login of the user with the sa_role and ha_role. This must be the same on the primary and secondary companion.
- HA_PWD – The password for the HA_LOGIN. This must be the same on the primary and secondary companion.
- PRIM_RUNSCRIPT – The name of the RUNSERVER file that is used to bring up the primary companion.
- PRIM_CONSOLE_LOG – The full path to the error log for the current primary companion session. This can be any file that has sufficient space and is writable by root. The default is *$SYBASE/install*.
- SEC_CONSOLE_LOG – The full path to the error log for the current secondary companion session. This can be any file that has sufficient space and is writable by root. The default is *$SYBASE/install*.

4 Edit the script for the primary companion. The example below shows the settings in the *RUNHA_MONEY1.sh* script for primary companion MONEY1 running on host FIN1, and for secondary companion PERSONNEL1 running on host HUM1. In this example, when the primary companion shuts down, the monitor script triesonce to reboot the primary companion on node FIN1. If this fails, the script shuts down the HACMP services on FIN1 and moves the database devices for MONEY1 to PERSONNEL1 on HUM1. If PERSONNEL1 is down or in an inconsistent state, the script starts MONEY1 on HUM1.

Variable	Primary Companion
ASE_FAILOVER	yes
BASIC_FAILOVER	yes
RETRY	1
SYBASE_ASE	ASE-12_5
SYBASE_OCS	OCS-12_5
PRIM_SERVER	MONEY1
PRIM_HOST	FIN1
HA_LOGIN	“sa”
HA_PWD	“Odd1Think
PRIM_CONSOLE_LOG	*$SYBASE/$SYBASE_ASE/install/MONEY1.CS_log*
SEC_SERVER	PERSONNEL1
SEC_HOST	HUM1
SEC_CONSOL_LOG	*$SYBASE/$SYBASE_ASE/install/PERSONNEL.CS_log*

5 Edit the script for the secondary companion. These values differ depending on whether you are using an asymmetric or a symmetric setup.

If this is an asymmetric setup, the values for PRIM_SERVER should be the same as SEC_SERVER (the name of the secondary companion). PRIM_HOST should be the same as SEC_HOST, and PRIM_SYBASE should be the same as SEC_SYBASE.

If this is a symmetric setup, the values for the PRIM_SERVER, PRIM_HOST, PRIM_SYBASE, SEC_SERVER, SEC_HOST, and SEC_SYBASE in the secondary companion script are the opposite of what is set in the primary companion script.

Table 8-1 describes the values for the variables for both an asymmetric setup and a symmetric setup on primary companion MONEY1 and secondary companion PERSONNEL1:

Table 8-1: Values for the secondary companion

Variables	Asymmetric secondary companion	Symmetric secondary companion
RETRY	10	1
ASE_FAILOVER	no	yes
BASIC_FAILOVER	no	yes
PRIM_SERVER	PERSONNEL1	PERSONNEL1
PRIM_HOST	HUM1	HUM1
SEC_SERVER	PERSONNEL1	MONEY1
SEC_HOST	HUM1	FIN1

Configure the resource groups in HACMP

Note You can perform the steps described in this section from the command line or through the configuration utility SMIT. This document explains how to use SMIT to configure resource groups. See HACMP for AIX documentation for information about configuring resource groups at the command line.

Shut down the cluster services on both nodes in 'graceful' mode, and then log in to the boot IP addresses of the primary node as root and perform the following tasks.

1 Start SMIT.

2 From the Cluster Resources screen, select Add a Resource Group if you are creating a new resources group, or select Change/Show a Resource Group if you are changing an existing resources group.

3 Enter "cascading" in the Node Relationship field, as described below:

```
Define the Resource Group for the Primary Companion:
Resource Group Name                 [<resgrp1>]
Node Relationship                   [cascading]
Participating Node Names            [<primary_node> <secondary_node>]

Define the Resource Group for the Secondary Companion:
(For Asymmetric Failover Configuration)

Resource Group Name                 [<resgrp2>]
Node Relationship                   [cascading]
Participating Node Names            [<secondary_node>]
```

```
(For Symmetric Failover Configuration)
Resource Group Name                 [<resgrp2>]
Node Relationship                   [cascading]
Participating Node Names              [<secondary_node> <primary_node>]
```

4 Configure each of the resource groups you defined. For the Application Server field, enter the name of the primary companion. Enter information in all the required fields, such as IP Label, Volume Groups, and File systems. Repeat this step for each of the companions.

5 Define the primary and secondary companions as application servers in HACMP Cluster Resources. Select either Add Application Server or Change Application Server, and enter these values:

- For the Start/Stop Scripts, enter the name of the scripts you created.
- For primary and symmetric secondary companions, enter “monitor” and “failover” as the arguments for the start and stop scripts, respectively.
- For an asymmetric secondary companion, use “monitor” and “stop” as the arguments for the start and stop script, respectively.

For example:

```
Define the Primary Companion Server
Server Name   [<primary_ase>]
Start Script  [/usr/sbin/cluster/events/RUN_<primary_ase>_ha monitor]
Stop Script   [/usr/sbin/cluster/events/RUN_<primary_ase>_ha failover]

Define the Secondary Companion Server:
(For Asymmetric Failover Configuration)
Server Name   [<secondary_ase>]
Start Script  [/usr/sbin/cluster/events/RUN_<secondary_ase>_ha monitor]
Stop Script   [/usr/sbin/cluster/events/RUN_<secondary_ase>_ha stop]

(For Symmetric Failover Configuration)
Server Name  [<secondary_ase>]
Start Script [/usr/sbin/cluster/events/RUN_<secondary_ase>_ha monitor]
Stop Script  [/usr/sbin/cluster/events/RUN_<secondary_ase>_ha failover]
```

6 Synchronize the cluster resources. Using SMIT on the node on which you have performed steps 1 through 5, go to the Cluster Resources screen and select Synchronize Cluster Resources. This propagates the changes you made to all the other nodes within the same cluster. In some cases, you may need to stop the HACMP services and reboot both nodes before performing the synchronization. Make sure the synchronization does not produce any errors.

Configuring companion servers for failover

Perform the tasks in this section to configure the Adaptive Servers as primary and secondary companions in a high availability system.

Run *sp_companion* with *do_advisory* option

You must configure the secondary companion with sufficient resources to perform the work of both servers during failover. The secondary companion may have attributes that will prevent a successful cluster operation. For example, if both the primary and secondary companions are configured for 250 user logins, during failover, the secondary companion only has the resources for half the number of potential user logins necessary. Instead, both MONEY1 and PERSONNEL1 should be configured for 500 user logins.

The sp_companion do_advisory option checks the configuration options on the primary and the secondary companion to make sure a cluster operation (such as configuring an Adaptive Server as a secondary companion) will be successful. sp_companion do_advisory advises you of any configuration options that should be changed.

See Chapter 6, “Running do_advisory” for a complete description of the sp_companion do_advisor option.

Configure for asymmetric configuration

Configure the primary companion for asymmetric configuration. Issue the following command from the secondary companion:

```
sp_companion "primary_server_name", configure, proxy_device_name, login_name,
    password, cluster_login, cluster_login_password.
```

where:

- *primary_server_name* is the name of the primary Adaptive Server as defined in the *interfaces file* entry and in sysservers.
- The *proxy_device_name* is the device where the proxy user databases are created (if no proxy device is specified, the default devices for the server will be used for proxy databases). For more information, see "sp_companion" on page 183.
- *login_name* is the name of the user performing this cluster operation (the user must have both the sa_role and the ha_role).
- *password* is the password of the person performing this cluster operation.

Note You must execute the above command ONLY from the secondary companion.

This example configures an Adaptive Server named MONEY1 as a primary companion. Issue the following command from the secondary server PERSONNEL1:

```
 sp_companion "MONEY1", configure, sa, "Odd2Think"
Server 'PERSONNEL1' is alive and cluster configured.
Step: Access verified from Server:'PERSONNEL1' to Server:'MONEY1'
Server 'MONEY1' is alive and cluster configured.
Step: Access verified from Server:'MONEY1' to Server:'PERSONNEL1'
(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)
.....
Step: Companion servers configuration check succeeded
Step: Server handshake succeeded
Step: Master device accessible from companion
Step: Added the servers 'PERSONNEL1' and 'MONEY1' for cluster config
Step: Server configuration initialization succeeded
Step: Synchronizing server logins from companion server
Step: Synchronizing remoteserver from companion server
Step: Synchronizing roles from companion server
Step: Synchronizing server-wide privs from companion server
Step: User information syncup succeeded
Step: Server configured in normal companion mode
```

If user databases are created during the sp_companion configuration, you see messages similar to these:

```
Step: Created proxy database ‘pubs2’
 Step: Proxy status for database has been set. Please Checkpoint the database
'pubs2'
 Step: Server configured in normal companion mode”
 Starting companion watch thread
```

Before you configure the companions for symmetric configuration, you must first configure them for asymmetric configuration.

See “Asymmetric companion configuration” on page 19 for more information.

Configure for symmetric configuration

After you configure the companions for asymmetric failover, you can configure them for symmetric configuration. In a symmetric configuration, both servers act as primary and secondary companions. See Figure 3-2 on page 22 for a description of symmetric configuration.

Issue sp_companion from the secondary companion to configure it for symmetric configuration. Use the same syntax as for asymmetric configuration. See ““Configure for asymmetric configuration” on page 95,” for a description of the syntax for sp_companion.

The following example adds an Adaptive Server named MONEY1 as the secondary companion to the Adaptive Server named PERSONNEL1 described in “Configure for asymmetric configuration” on page 95. Issue the following command from server MONEY1:

```
sp_companion ’PERSONNEL1’, configure, sa, MyPassword, sa_cluster_login,
MyClusterPassword
Server ’MONEY1’ is alive and cluster configured.
Step: Access verified from Server:’MONEY1’ to Server:’PERSONNEL1’
Server ’PERSONNEL1’ is alive and cluster configured.
Step: Access verified from Server:’PERSONNEL1’ to Server:’MONEY1’
(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)
.....
Step: Companion servers configuration check succeeded
Step: Server handshake succeeded
```

```
Step: Master device accessible from companion
Step: Added the servers 'MONEY1' and 'PERSONNEL1' for cluster config
Step: Server configuration initialization succeeded
Step: Synchronizing server logins from companion server
Step: Synchronizing remoteserver from companion server
Step: Synchronizing roles from companion server
Step: Synchronizing server-wide privs from companion server
Step: User information syncup succeeded
Step: Server configured in normal companion mode
```

Bring up primary companion as a monitored resource

Follow the instructions in this section to start the primary companion as a service monitored by the high availability subsystem.

Note Before monitoring begins on the primary companion, make sure that there is no need to shut sown the primary server for maintenance or any other purpose. Once monitoring begins, the primary companion must be moved to suspended mode to bring it down. If you are unsure, start the primary server using the startserver script in *$SYBASE/$SYBASE_ASE/install*, finish configuring the companion, then reboot the server using the steps described here.

To start the primary companion as a resource monitored for failover:

1 Stop the HACMP services on the primary node.

2 Check */tmp/hacmp.out* to make sure the node_down event completed, then restart the HACMP services either by using SMIT or by executing this command as root at the command line:

```
/usr/sbin/cluster/etc/rc.cluster -boot '-N' '-b'  '-i'
```

This automatically executes the *RUNHA_<server_name>.sh* monitor script, which brings up the primary companion and monitors it during crash or hang situations.

Repeat this process on the secondary node to bring up the secondary companion.

Administering Sybase failover

This section describes information about:

- Failing back to the primary companion
- Resuming normal companion mode
- Suspending normal companion mode
- Troubleshooting common problems with Sybase failover

Failing back to the primary node

Failback automatically occurs on HACMP. When HACMP is brought up on the primary node, the *stop_server* event on the secondary node triggers the monitoring script to execute sp_companion 'prepare_failback'.

To fail back to the primary node, make sure that the secondary companion is in secondary failover mode, and bring up HACMP services on the primary node. To make sure that sp_companion 'prepare_failback' was executed successfully, search for this string in */tmp/hacmp.out*:

```
SYBASE HA MONITOR: Prepare_failback was successful.
```

Note Before you start the HACMP services on the primary node, make sure that the secondary node is up and the secondary companion is running in secondary failover mode. If the secondary companion or secondary node is not up and running, do not bring up the primary companion. If both nodes are down, or the HACMP services has stopped on both nodes, always restart the secondary node and its HACMP services before restarting the primary node.

Manually failing back

Note If the automatic failback failed, examine the logs to make sure that the high availability system performed the following steps. If it did not, you can perform them manually. You must perform them in the sequence described below.

1 Stop the HACMP subsystem with the takeover mode on the primary node. This shuts down the primary companion and fails over its resources to the secondary companion.

2 Shutdown and then restart your secondary companion.The *RUNHA_servername.sh* restarts the companion automatically after you shut it down if RETRY is set to a value greater than 0.

3 Log in as LOGIN_NAME to the secondary companion through isql and make sure that it is running in secondary failover mode.

4 Issue sp_companion 'prepare_failback'. For example, to fail back from the secondary companion PERSONNEL1:

```
sp_companion MONEY1, 'prepare_failback'
```

5 Restart HACMP on the primary node.

6 Log in to the primary companion using isql and make sure that it is running in primary failback mode.

7 Issue sp_companion 'resume'. For example, to resume companion mode for primary companion MONEY1:

```
sp_companion PERSONNEL1, 'resume'
```

Note You cannot connect clients with the failover property (for example isql -Q) until you issue sp_companion resume. If you do try to reconnect them after issuing sp_companion prepare_failback, the client hangs until you issue sp_companion resume.

Suspending companion mode

If you must shut down the primary companion for maintenance but do not want to fail over to the secondary companion, you must temporarily suspend companion mode. When the companion mode is suspended, synchronization between the companions does not occur, and the primary companion cannot fail over to the secondary companion. However, suspended mode is very useful for performing such maintenance tasks as changing configuration parameters.

1 To move to suspended mode, issue:

```
sp_companion <primary_server_name>, suspend
```

For example, to suspend primary companion MONEY1:

```
sp_companion MONEY1, suspend
```

2 Kill the monitoring process so it does not trigger a failover when the companion server goes down. As root, enter:

```
ps -ef|grep "RUNHA_<server_name>.sh monitor"
kill -9 <pid>
```

For example, to kill the monitoring process for MONEY1 which has a pid of 2509:

```
ps -ef|grep "RUNHA_MONEY1.sh monitor"
kill -9 2509
```

3 Shut down the primary companion.

After killing the monitoring process, you can bring the companion server down as many times as necessary and it will not failover.

Restarting shutdown companion during suspended mode

Use the startup script in *$SYBASE/$SYBASE_ASE/install* to restart the primary companion without it being monitored:

```
startserver -f ./RUN_<server_name>
```

For example, to start the MONEY1 companion:

```
startserver -f ./RUN_MONEY1
```

If you use this script to start a companion server, it will not fail over when the server goes down, even if it is configured to do so. Use this method only if you are doing maintenance and you are certain that you do not want the server databases to be accessible when the server is down.

Resuming normal companion mode

The steps for resuming normal companion mode are slightly different depending on whether you are moving from suspended mode or from failover mode.

Resuming normal companion mode from suspended mode

To resume normal companion mode between two companions that have been moved to suspended mode:

1 Shut down the primary companion.

2 Stop the HACMP services on the primary node in “graceful” mode.

3 Restart the HACMP services on the primary node.

Resuming normal companion mode

To resume normal companion mode between two companions that are in failover mode, restart the HACMP services on the primary node, and perform:

1 Check that both companions are in failback mode by issuing sp_companion with no parameters.

2 Resume normal companion mode by issuing:

```
sp_companion secondary_server_name, resume
```

For example, to issue normal companion mode for primary companion PERSONNEL1:

```
sp_companion PERSONNEL1, resume
Server 'MONEY1' is alive and cluster configured.
Step: Access verified from Server:'MONEY1' to Server:'PERSONNEL1'
Server 'PERSONNEL1' is alive and cluster configured.
Step: Access verified from Server:'PERSONNEL1' to Server:'MONEY1'
Step: Checkin to See if the remote server is up
Step: Synchronizing server logins from companion server
Step: Synchronizing remoteserver from companion server
Step: Synchronizing roles from companion server
Step: Synchronizing server-wide privs from companion server
Step: User information syncup succeeded
sys_id ses_id      ses_id2     ses_status Purged from s1.
------ ---------- ----------- ---------- ------------------------------

(0 rows affected)
sys_id ses_id      ses_id2     ses_status Copied to s1.
------ ---------- ----------- ---------- ------------------------------

(0 rows affected)
sys_id ses_id      ses_id2     ses_status Purged from s2.
------ ---------- ----------- ---------- ------------------------------
(0 rows affected)
Step: Syssession information syncup succeeded
```

Dropping companion mode

To drop companion mode, issue:

```
sp_companion companion_name, "drop"
```

Dropping companion mode is an irreversible process; you must reconfigure the Adaptive Servers companion servers before they will failover in a high availability system and retain all the functionality that Sybase's failover provides. However, the nodes upon which the Adaptive Servers are running are still monitored by the high availability subsystem.

If you drop the companion mode while the *RUNHA_<servername>.sh* script is running, the script continues to monitor the server for any down or hung instances. If you plan to shut down the server and do not want the node to failover, kill the monitor process by issuing:

```
kill -9 ‘ps -ef | grep "RUNHA_<servername>.sh monitor" | grep -v grep | awk
’{print $2}’‘
```

If you do not kill the monitor process, it triggers a failover of the resources when it detects that the companion has gone down, and tries to restart the companion from either the primary or secondary node, depending on your settings for RETRY and BASIC_FAILOVER.

Troubleshooting failover on HACMP for AIX

This section includes troubleshooting information about common errors.

Error message 18750

If a companion server issues error message 18750, check the @@cmpstate of the servers. If the primary companion is in normal companion mode, but the secondary companion is in secondary failover mode, the cluster is in an inconsistent state requiring manual recovery. The inconsistent state may be caused by an sp_companion 'prepare_failback' command failing on the secondary companion. You can determine whether this happened by examining the HACMP log on the secondary node, (located in */tmp/hacmp.out*). To recover, perform the following steps:

1 Shut down both the primary and the secondary companions.

2 Reboot the secondary companion.

3 Repair all databases marked “suspect.” To determine which databases are suspect, issue:

```
select name, status from sysdatabases
```

Databases marked suspect have a status value of 320.

4 Allow updates to system tables:

```
sp_configure “allow updates”, 1
```

5 For each suspect failed over database, perform the following:

```
1> update sysdatabase set status=status-256 where name='database_name'
2> go
1> dbcc traceon(3604)
2> go
1> dbcc dbrecover(database_name)
2> go
```

6 From the secondary companion, issue:

```
sp_companion primary_companion_name, prepare_failback
```

For example, from primary companion MONEY1:

```
sp_companion MONEY1, prepare_failback
```

Make sure that this command executes successfully.

7 Restart the HACMP services on the primary node.

Recovering from a failed *prepare_failback*

During a failback, if prepare_failback executed successfully on the secondary companion but the primary companion fails to boot, perform the following to rollback and then reissue the prepare_failback command:

1 Check the primary companion’s error log and the HACMP error log to find the reason the server failed to boot, and correct the problems.

2 Stop the HACMP services on the primary node with takeover.

3 Login to the secondary companion as LOGIN_NAME, and issue:

```
dbcc ha_admin ("", "rollback_failback")
dbcc ha_admin ("", "rollback_failover")
```

Both companion servers should both be back in the failover mode.

4 Restart HACMP on the primary node.

Location of failover logs

Sybase's failover includes the following logs. These logs are helpful for investigating and diagnosing errors encountered during HACMP failover:

- */tmp/hacmp.out* – contains output of the HACMP activities, as well as the output from the *RUNHA_server_name.sh* monitoring script. For general HACMP failure, search for the string "ERROR". For output of the *RUNHA_server_name.sh* script, search for "SYBASE HA MONITOR".

  After determining the reason for the failure, correct it, then go to the Cluster Recovery Aids screen of SMIT and do a Recover From Script Failure, before continuing.

  If a node does not include a sufficient amount of space in a particular file system, HACMP hangs in the middle of a failover or failback process, which results in a config_too_long lock. If this occurs, you must clean up the full directories, then start SMIT and move to the Cluster Recovery Aids screen and perform a Recover From Script Failure before continuing.

- $PRIM_CONSOLE_LOG – the location of this log is defined in the *RUNHA_server_name.sh* monitoring script. This error log includes the Adaptive Server information from the last execution of the *RUNHA_server_name.sh* script.

CHAPTER 9

Configuring Adaptive Server for Failover on HP Tru64 TruCluster Server 5.x

Perform the tasks in this chapter to configure Adaptive Server for Failover on HP TruCluster Server 5.x.

Configuring hardware and operating system for high availability

Sybase high availability requires the following hardware and system components:

- Two homogenous, network systems with similar configurations in terms of resources like CPU, memory, and so on.
- High availability subsystem package and the associated hardware.
- Devices that are accessible to both nodes.
- Vendor provided mirroring, not Sybase mirroring, should be used for media failure protection.

  See your hardware and operating system documentation for information about installing platform specific high availability software.

Requirements for running Sybase's failover on HP TruCluster

Configuring for high availability on HP TruCluster requires:

- 2 hardware-compatible nodes running HP Tru64 version 5.0A with HP TruCluster Server version 5.0A.
- A TruCluster resource name for the primary companion (for example, *primary*).
- A TruCluster resource name for the secondary companion (for example, *secondary*).
- The name of the primary Adaptive Server companion name must be the same its TruCluster resource name.
- The name of the secondary Adaptive Server companion must be the same as its TruCluster resource name.

Preparing Adaptive Server to work with The HA subsystem

Perform the tasks in this section to prepare Adaptive Server for a high availability configuration.

Install Adaptive Servers

The primary companion can be either a newly installed Adaptive Server, or it can be upgraded from a previous version of Adaptive Server with existing databases, users, and so on.

The secondary companion must be a newly installed Adaptive Server without any user logins or user databases. This ensures that all user logins and database names are unique within the cluster. After configuration for failover is complete, you can add user logins and databases to the secondary companion.

See the installation documentation for your platform for information about installing and configuring Adaptive Server.

Add entries for both Adaptive Servers to the *interfaces file*

The *interfaces file* must contain entries for both primary and secondary companions. For example, the *interfaces file* for the servers used in the examples above would have entries for MONEY1 and PERSONNEL1. The server entry in the *interfaces file* must use the network name specified in sysservers. For information about adding entries to the *interfaces file*, see the installation documentation for your platform.

Add entries to *interfaces* File for client connections during failover

To enable clients to reconnect to the failed over companion, you must add an additional line to the *interfaces file*. By default, clients connect to the port listed in the query line of the server entry. If that port is not available (because that server has failed over), the client connects to the server listed in the *hafailover* line of the server entry. Here is a sample *interfaces file* for a primary companion named MONEY1 and a secondary companion named PERSONNEL1:

```
MONEY1
    master tcp ether FIN1 4100
    query tcp ether FIN1 4100
    hafailover PERSONNEL1
```

If this is a symmetric configuration, make the failover entry for the secondary server:

```
PERSONNEL1
master tcp ether HUM1 4100
query tcp ether HUM1 4100
hafailover MONEY1
```

Use dsedit to add entries to the *interfaces file*. If the interfaces entries already exist, you must modify them to work for failover.

See the Utility Guides for your platform for information about dsedit.

sybha executable

The *sybha* executable allows the Adaptive Server High Availability Basics Services library to interact with each platform's high availability cluster subsystem. The Adaptive Server High Availability Basics Services library calls *sybha*. which is located in *$SYBASE/ASE-12_5/bin*. Before *sybha* can run, you must change its ownership and permissions. You must also edit a file named *sybhauser* in *$SYBASE/ASE-12_5/install*. *sybhauser* contains a list of the users who have System Administrator privileges on the cluster. Sybase strongly recommends that you severely limit the number of users who have System Administrator privileges on the cluster.

As root, perform the following:

1 Add a new group named *sybhagrp*. You can either add this group to the */etc/group* file, or you can add it to your NIS maps. Add the sybase user to this group (this is the user that owns the *$SYBASE* directory). When the server is started, the sybase user runs the data server. If you have multiple servers running and different users owning the *$SYBASE* directory for each of them, each of these users must be added to the group

2 Change to the *$SYBASE/$SYBASE_ASE/bin* directory:

```
cd $SYBASE/$SYBASE_ASE/bin
```

3 Change the ownership of *sybha* to root:

```
chown root sybha
```

4 Change the group for the *sybha* program to *sybhagrp*:

```
chgrp sybhagrp sybha
```

5 Modify the file permissions for *sybha* to 4550:

```
chmod 4550 sybha
```

6 Change to the *$SYBASE/$SYBASE_ASE/install* directory:

```
cd $SYBASE/ASE-12_5/install
```

7 Add the sybase user to the *sybhauser* file. These logins must be in the format of UNIX login IDs, not Adaptive Server logins. For example:

```
sybase
coffeecup
spooner
venting
howe
```

8 Change the ownership of *sybhauser* to root:

```
chown root sybhauser
```

9 Modify the file permissions for *sybhauser*:

```
chmod 600 sybhauser
```

Verify configuration parameters

You must enable the following configuration parameters before you configure Adaptive Server for failover:

- enable CIS – Enables Component Integration Services (CIS). This configuration parameter is enabled by default.
- enable xact coordination – Enables Distributed Transaction Management (DTM). This configuration parameter is enabled by default.
- enable HA – Enables Adaptive Server to function as a companion in a high availability system. enable HA is off by default. This configuration is static, so you must reboot Adaptive Server for it to take effect. This parameter causes a message to be written to your error log stating that you have started the Adaptive Server in a high availability system.

See the *System Administration Guide* for information about enabling configuration parameters.

Add thresholds to the master log

If you have not already done so, you must add a threshold to the master log.

1 Define and execute sp_thresholdaction on the master database's log to set a threshold on the number of pages left before a dump transaction occurs. Sybase does not supply sp_thresholdaction. See the *Adaptive Server Reference Manual* for information about creating this system procedure.

2 Place thresholds on the master and sybsystemprocs log segments so they do not fill up:

```
sp_addthreshold "master", "logsegment", 250, sp_thresholdaction
sp_addthreshold "sybsystemprocs", "logsegment", 250, sp_thresholdaction
```

3 Do this on both the primary and secondary servers. After you have completed these tasks, reboot both servers to allow the static configuration parameters to take effect.

Create new default device other than master

By default, the master device is the default device in a newly installed Adaptive Server. This means that, if you create any databases (including the proxy databases used by failover), they are automatically created on the master device. However, adding user databases to master makes it more difficult to restore the master device from a system failure. To make sure that the master device contains as few extraneous user databases as possible, create a new device using disk init. Use sp_diskdefault to specify the new device as the default before you configure Adaptive Server as a companion for failover.

For example, to add a new default device named money_default1 to the MONEY1 Adaptive Server, enter:

```
disk init name="money_default1",
physname="/home/moneyHA/money_default1.dat",
size=40000, vdevno=2
go
```

At the next ">", enter:

```
sp_diskdefault money_default1, defaulton
go
```

Then turn off disk default on master:

```
sp_diskdefault master, defaultoff
go
```

See the *Adaptive Server Reference Manual* for more information about disk init and sp_diskdefault.

Add the local server to *sysservers*

On the primary server, log in through isql, and add both servers with the primary server as the local server. For example:

```
sp_addserver MONEY1, local
go
sp_addserver PERSONNEL1
go
```

Add secondary companion to *sysservers*

On the secondary server, login through isql and add both servers, with the secondary as the local server. For example:

```
sp_addserver PERSONNEL1, local
go
sp_addserver MONEY1
```

By default, Adaptive Server adds the server with an srvid of 1000. You do not need to reboot Adaptive Server for the change to take effect.

Assign *ha_role* to the system administrator

The system administrator must have the ha_role on both Adaptive Servers to run sp_companion. To assign the ha_role, issue the following from isql:

```
sp_role "grant", ha_role, sa
```

You must log out and then log back in to the Adaptive Server for the change to take effect.

Run *installhasvss* to install HA stored procedures

Note You must perform the tasks described "Add entries for both Adaptive Servers to the interfaces file" on page 109, above, before running *installhasvss*. If you run *installhasvss* before performing these tasks, you will have to re-run *installmaster* to re-install all the system stored procedures.

The *installhasvss* script performs the following tasks to configure Adaptive Server for failover:

- Installs the stored procedures required for failover (for example, sp_companion).
- Installs the SYB_HACMP server entry in sysservers.

You must have System Administrator privileges to run the *installhasvss* script.

installhasvss islocated in the *$SYBASE/$SYBASE_ASE/scripts* directory. To execute the *installhasvss* script, enter:

```
$SYBASE/$SYBASE_OCS/bin/isql -Usa -P<password> -S<servername>
-i$SYBASE/$SYBASE_ASE/scripts/installhasvss
```

installhasvss prints messages as it creates stored procedures and creates the SYB_HACMP server.

Configure the HP Tru64 subsystem for Sybase's failover

Perform the steps in this section to configure HP Tru64 TruCluster Server for Failover.

Modify the *ASE_HA.sh* script

The *ASE_HA.sh* script is used to start, stop, and monitor an Adaptive Server in a high availability environment. Adaptive Server includes this script in the *$SYBASE/$SYBASE_ASE/install* directory. You must make a copy of this script and modify it for your environment for both Adaptive Servers running in the cluster. The modifications you make to the script will differ slightly depending on whether the script is for the primary or secondary companion. You must install the *ASE_HA.sh* script in */var/cluster/caa/script/ASE_server_name.scr*.

To modify the script for your environment:

1 Change to the *$SYBASE/$SYBASE_ASE/install* directory.

2 As root, copy *ASE_HA.sh* to a file named */var/cluster/caa/script/server_name.scr.*

where *server_name* is the Adaptive Server to be monitored.

For example, to copy a *ASE_HA.sh* script for a server named MONEY1:

```
cp ASE_HA.sh /var/cluster/caa/script/MONEY1.scr
```

3 You must edit the *server_name.scr* script for your environment. The original *ASE_HA.sh* script contains the variables listed below. Edit the lines that include "__FILL_IN__" (and any other lines that require editing) with the values for your site:

- RECOVERY_TIMEOUT – The maximum amount of time the high availability subsystem waits, in seconds, before determining that the companion failed to start. Make sure you set this number long enough for a loaded companion to reboot. RECOVERY_TIMEOUT is also used as the maximum amount of time the subsystem waits for failover and failback to complete.

- SHUTDOWN_TIMEOUT – The maximum time the high availability subsystem waits for the companion to shut down before killing it.

  Note This value should always be less than the amount of time configured for the action script time_out in the profile.

- RESPONSE_TIMEOUT – The maximum amount of time the subsystem allows for a simple query to return a result set. Its used to diagnose whether or not the companion server is hung. For example, if isql fails to establish a connection in 60 seconds, it automatically times out and exits. However, if isql successfully connects, but does not return a result set, RESPONSE_TIMEOUT may determine that the companion server is hung. By default, RESPONSE_TIMEOUT is set to 999999.
- ASE_FAILOVER – can be set to either yes or no.
  - If
  - yes - Monitors the companion server for hung or dead processes and relocates the TruCluster service to the secondary node. If set to “yes”, sp_companion...configure must be run on the server as well to keep the high availability consistent.
  - If no - Do not relocate the TruCluster subsystem on this node even if the primary companion fails over. This setting is useful if you need to bring down a companion for maintenance or re configuration.

  Note If you are configuring an asymmetric setup, set ASE_FAILOVER to “no” on the secondary server.

  Warning! This should only be set to “yes” if BOTH the servers are running Adaptive Server version 12.0 or later. Adaptive Servers for previous versions should be set to “no”.

- BASIC_FAILOVER – Set to either “yes” or “no”.
  - If

  - yes - Use the failover mechanisms provided by the TruCluster subsystem if it determines the servers are running in modes that allow failover. When a failover occurs, the TruCluster subsystem monitor first checks if the companions are in a correct mode to perform a failover. If the companions are not enabled for Sybase's Failover (that is, have enable ha set to 1), or they are running in single-server mode, or if the secondary companion is down, the TruCluster subsystem monitor checks whether BASIC_FAILOVER is set. If it is, the monitor attempts to start up the primary companion on the secondary node.
  - If no - Do not revert to mode0 failover (mode 0 restarts the primary companion on the secondary node, and does not involve Sybase's Failover) even if Sybase's failover criteria is not met. That is, if BASIC_FAILOVER is set to no, failover does not happen at either the node or the companion level.
- SYBASE – The $SYBASE environment variable.
- retry – The number of times the TruCluster subsystem attempts to reboot the local node before failing over. Set this to a high number for an asymmetric setup so the secondary companion is more likely to reboot itself if it goes down. The default is 0, which means that the companion will not reboot on the same node if it goes down.
- SYBASE_ASE – The installation directory of Sybase Adaptive Server products. The default is ASE-12_0.
- SYBASE_OCS – The installation directory of Sybase Open Client products. The default is OCS-12_0.
- PRIM_SERVER – The name of the primary companion.
- SEC_SERVER – The name of the secondary companion.
- PRIM_HOST – The name of the primary host or service interface.
- SEC_HOST – The name of the secondary host or service interface name.
- HA_LOGIN – The login of the user with the sa_role and ha_role. This must be the same on the primary and the secondary companion.
- HA_PWD – The password for the HA_LOGIN. This has to be the same on both the primary and secondary companion.

4 Edit the script for the primary companion. The example below shows the settings in the *server_name.scr* script for primary companion MONEY1 running on host FIN1, and for secondary companion PERSONNEL1 running on host HUM1. In this example, when the primary companion shuts down, the monitor script tries once to reboot the primary companion on node FIN1. If this fails, the script shuts down the TruCluster services on FIN1 and moves the database devices for MONEY1 to PERSONNEL1 on HUM1. If PERSONNEL1 is down or in an inconsistent state, the script starts MONEY1 on HUM1.

Variable	Primary Companion
ASE_FAILOVER	yes
BASIC_FAILOVER	yes
RETRY	1
PRIM_SERVER	MONEY1
PRIM_HOST	FIN1
HA_LOGIN	“sa”
HA_PWD	“Odd1Think
SEC_SERVER	PERSONNEL1
SEC_HOST	HUM1

5 Edit the script for the secondary companion. These values will differ depending on whether you are using an asymmetric or a symmetric setup.

If this is an asymmetric setup, the values for PRIM_SERVER should be the same as SEC_SERVER (the name of the secondary companion). PRIM_HOST should be the same as SEC_HOST.

If this is a symmetric setup, the values for the PRIM_SERVER, PRIM_HOST, SEC_SERVER, and SEC_HOST in the secondary companion script are the opposite of what is set in the primary companion script.

Table 9-1 describes the values for the variables for both an asymmetric setup and a symmetric setup on primary companion MONEY1 and secondary companion PERSONNEL1:

Table 9-1: Values for the secondary companion

Variables	Asymmetric secondary companion	Symmetric secondary companion
RETRY	10	1
ASE_FAILOVER	no	yes
BASIC_FAILOVER	no	yes
PRIM_SERVER	PERSONNEL1	PERSONNEL1
PRIM_HOST	HUM1	HUM1
SEC_SERVER	PERSONNEL1	MONEY1
SEC_HOST	HUM1	FIN1

6 Register your *server_name.cap* file. The syntax is

```
caa_register PRIM_SERVER
```

and

```
caa_register SEC_SERVER
```

7 Start the *server_name.cap* file. The syntax is:

```
caa_start PRIM_SERVER
```

and

```
caa_start SEC_SERVER
```

Modify the ASE_HA.cap profile

The *ASE_HA.cap* profile specifies some parameters and the name of the action script for the cluster. You must install the modified profile as */var/cluster/caa/profile/ASE_server_name.cap*.

1 Edit the <*servername*>.*cap* files located in */var/cluster/caa/profile* to include your server names. These scripts contain the string FILL_IN for the values you must change.

2 Validate the *server_name.cap* files with the caa_profile command. The syntax is:

```
caa_profile -validate PRIM_SERVER
```

and,

```
caa_profile -validate SEC_SERVER
```

if caa_profile does not return an error message, your files are valid.

Configure companion servers for failover

Perform the tasks in this section to configure the Adaptive Servers as primary and secondary companions in a high availability system.

Run *sp_companion* with *do_advisory* Option

You must configure the secondary companion with sufficient resources to perform the work of both servers during failover. The secondary companion may have attributes that will prevent a successful cluster operation. For example, if both the primary and secondary companions are configured for 250 user logins, during failover the secondary companion only has the resources for half the number of potential user logins necessary. Instead, both MONEY1 and PERSONNEL1 should be configured for 500 user logins.

The sp_companion do_advisory option checks the configuration options on both the primary and the secondary companion to make sure a cluster operation (such as configuring an Adaptive Server as a secondary companion) will be successful. sp_companion do_advisory advises you of any configuration options that should be changed.

See Chapter 6, “Running do_advisory” for a complete description of the sp_companion do_advisor option.

Configure for asymmetric configuration

Configure the primary companion for asymmetric configuration. Issue the following from the secondary companion:

```
sp_companion "primary_server_name", configure, with_proxydb, login_name,
    password
```

where:

- *primary_server_name* is the name of the primary Adaptive Server as defined in the *interfaces file* entry and in sysservers.
- *with_proxydb* indicates that proxy databases are created on the secondary companion for all databases other than system databases. Any subsequent databases that are added also create proxy databases.
- *login_name* is the name of the user performing this cluster operation (this user must have both the sa_role and the ha_role).

- *password* is the password of the person performing this cluster operation.

This example configures an Adaptive Server named PERSONNEL1 as a secondary companion:

```
 sp_companion "MONEY1", configure, with_proxydb, sa, "Odd2Think"
Server 'MONEY1' is alive and cluster configured.
Step: Access verified from Server:'MONEY1' to Server:'PERSONNEL1'
Server 'PERSONNEL1' is alive and cluster configured.
Step: Access verified from Server:'PERSONNEL1' to Server:'MONEY1'
(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)
.....
Step: Companion servers configuration check succeeded
Step: Server handshake succeeded
Step: Master device accessible from companion
Step: Added the servers 'MONEY1' and 'PERSONNEL1' for cluster config
Step: Server configuration initialization succeeded
Step: Synchronizing server logins from companion server
Step: Synchronizing remoteserver from companion server
Step: Synchronizing roles from companion server
Step: Synchronizing server-wide privs from companion server
Step: User information syncup succeeded
Step: Server configured in normal companion mode
```

If user databases are created during the sp_companion configuration, you see messages similar to these:

```
Step: Created proxy database 'pubs2'
 Step: Proxy status for database has been set. Please Checkpoint the database
'pubs2'
 Step: Server configured in normal companion mode"
 Starting companion watch thread
```

Before you configure the companions for symmetric configuration, you must first configure them for asymmetric configuration.

See "Asymmetric companion configuration" on page 19 for more information.

Configure for symmetric configuration

After you configure the companions for asymmetric failover, you can configure them for symmetric configuration. In a symmetric configuration, both servers act as primary and secondary companions. See Figure 3-2 on page 22 for a description of symmetric configuration.

Issue sp_companion from the secondary companion to configure it for symmetric configuration. Use the same syntax as for asymmetric configuration. See "Configure for asymmetric configuration," above, for a description of the syntax for sp_companion.

The following example adds an Adaptive Server named MONEY1 as the secondary companion to the Adaptive Server named PERSONNEL1 described in "Configure for asymmetric configuration" on page 119. Log in to the primary server MONEY1, and enter:

```
sp_companion 'PERSONNEL1', configure,
go
```

Login to the secondary server <PERSONNEL1>, and enter:

```
sp_companion "MONEY1", configure
go
Server 'MONEY1' is alive and cluster configured.
Step: Access verified from Server:'MONEY1' to Server:'PERSONNEL1'
Server 'PERSONNEL1' is alive and cluster configured.
Step: Access verified from Server:'PERSONNEL1' to Server:'MONEY1'
(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)
.....
Step: Companion servers configuration check succeeded
Step: Server handshake succeeded
Step: Master device accessible from companion
Step: Added the servers 'MONEY1' and 'PERSONNEL1' for cluster config
Step: Server configuration initialization succeeded
Step: Synchronizing server logins from companion server
Step: Synchronizing remoteserver from companion server
Step: Synchronizing roles from companion server
Step: Synchronizing server-wide privs from companion server
Step: User information syncup succeeded
Step: Server configured in normal companion mode
```

Bring up primary companion as a monitored resource

Follow the instructions in this section to start the primary companion as a service monitored by the high availability subsystem.

Note Before monitoring begins on the primary companion, make sure that there is no need to shutdown the primary server for maintenance or any other purpose. Once monitoring begins, the primary companion must be moved to suspended mode to bring it down. If you are unsure, start the primary server using the startserver script in *$SYBASE/$SYBASE_ASE/install*, finish configuring the companion, then reboot the server using the steps described here.

To start the primary companion as a resource monitoring for failover:

Restart the TruCluster resource by executing this command as root at the command line:

```
caa_start server_name
```

This automatically executes the <*server_name*>.*scr* monitor script, which brings up the primary companion and monitors it during crash or hang situations.

Repeat this process on the secondary node to bring up the secondary companion.

Administering Sybase failover

This section describes information about:

- Failing back to the primary companion
- Resuming normal companion mode
- Suspending normal companion mode
- Troubleshooting common problems with Sybase Failover.

Failing back to the primary node

The monitor scripts executes sp_companion...prepare_failback when you are failing back the TruCluster resource to the primary node.

To fail back to the primary node, make sure that the secondary companion is in secondary failover mode, and relocate the TruCluster resource on the primary node using:

```
caa_relocate server_name
```

where *server_name* is the name of the primary companion, and *host_name* is the name of the host on which it is running.

To make sure that sp_companion 'prepare_failback' was executed successfully, search for this string in *$SYBASE/$SYBASE_ASE/install/server_name_na.log*:

```
SYBASE HA MONITOR: Prepare_failback was successful.
```

Note Before you relocate the TruCluster resource on the primary node, make sure that the secondary node is up and the secondary companion is running in secondary failover mode. If the secondary companion or secondary node is not up and running, do not relocate the TruCluster resource. If both nodes are down, restart the secondary node before restarting the primary node.

Manually failing back

Note If the automatic failback failed, examine the logs to make sure that the high availability system performed the following steps. If it did not, perform them manually. You must perform them in the sequence described below.

1. Relocate the primary node's TruCluster resource to the secondary node. This shuts down the primary companion and fails over its resources to the secondary companion.
2. Shut down and then restart your secondary companion.The *server_name.scr* restarts the companion automatically after you shut it down if RETRY is set to a value greater than 0.
3. Log in as LOGIN_NAME to the secondary companion through isql and make sure that it is running in secondary failover mode.
4. Issue sp_companion 'prepare_failback'. For example, to fail back from the secondary companion PERSONNEL1:

   ```
   sp_companion MONEY1, 'prepare_failback'
   ```

5. Relocate the TruCluster resource to the primary node.

6 Log in to the primary companion using isql and make sure that it is running in primary failback mode.

7 Issue sp_companion 'resume'. For example, to resume companion mode for primary companion MONEY1:

```
sp_companion PERSONNEL1, 'resume'
```

Note You cannot connect clients with the failover property (for example isql -Q) until you issue sp_companion resume. If you do try to reconnect them after issuing sp_companion prepare_failback, the client hangs until you issue sp_companion resume.

Suspending companion mode

If you must shut down the primary companion for maintenance but do not want to fail over to the secondary companion, you must temporarily suspend companion mode. When companion mode is suspended, synchronization between the companions does not occur, and the primary companion cannot fail over to the secondary companion. However, suspended mode is very useful for performing such maintenance tasks as changing configuration parameters.

To move to suspended mode, issue:

```
sp_companion primary_server_name, suspend
```

For example, to suspend primary companion MONEY1,

stop the synchronization between the two companion servers:

```
sp_companion MONEY1, suspend
```

Stop the TruCluster resource:

```
/usr/sbin/caa_stop MONEY1
```

Restarting shutdown companion during suspended mode

Use the start up script in *$SYBASE/$SYBASE_ASE/install* to restart the primary companion without it being monitored:

```
startserver -f ./RUN_server_name
```

For example, to start the MONEY1 companion:

```
startserver -f ./RUN_MONEY1
```

If you use this script to start a companion server, it will not fail over when the server goes down, even if it is configured to do so. Use this method only if you are doing maintenance, and you are certain that you do not want the server databases to be accessible when the server is down.

Resuming normal companion mode

The steps for resuming normal companion mode are slightly different depending on whether you are moving from suspended mode or from failover mode.

Resuming normal companion mode from suspended mode

To resume normal companion mode between two companions that have been moved to suspended mode:

1 Shut down the primary companion if it is not already shut down.

2 Issue the following command to restart the TruCluster resource on the primary node:

```
caa_start server_name
```

Resuming normal companion mode from failover

To resume normal companion mode between two companions that are in failover mode, restart the TruCluster services on the primary node. Issue `/usr/sbin/caa_relocate <primary server>`, which puts the server in failback mode. Now perform the following:

1 Check that both companions are in failback mode by issuing sp_companion with no parameters.

2 Resume normal companion mode by issuing:

```
sp_companion secondary_server_name, resume
```

For example, to issue normal companion mode for primary companion MONEY1:

```
sp_companion PERSONNEL1, resume
Server 'MONEY1' is alive and cluster configured.
Step: Access verified from Server:'MONEY1' to Server:'PERSONNEL1'
Server 'PERSONNEL1' is alive and cluster configured.
Step: Access verified from Server:'PERSONNEL1' to Server:'MONEY1'
```

```
Step: Checkin to See if the remote server is up
Step: Synchronizing server logins from companion server
Step: Synchronizing remoteserver from companion server
Step: Synchronizing roles from companion server
Step: Synchronizing server-wide privs from companion server
Step: User information syncup succeeded
sys_id ses_id      ses_id2     ses_status Purged from s1.
------ ----------- ----------- ---------- ------------------------------

(0 rows affected)
sys_id ses_id      ses_id2     ses_status Copied to s1.
------ ----------- ----------- ---------- ------------------------------

(0 rows affected)
sys_id ses_id      ses_id2     ses_status Purged from s2.
------ ----------- ----------- ---------- ------------------------------
(0 rows affected)
Step: Syssession information syncup succeeded
```

Dropping companion mode

To drop companion mode, issue:

```
sp_companion companion_name, "drop"
```

Dropping companion mode is an irreversible process; you must reconfigure the Adaptive Servers companion servers before they will failover in a high availability system and retain all the functionality that Sybase's failover provides. However, the nodes upon which the Adaptive Servers are running are still monitored by the high availability subsystem.

In asymmetric configuration, log in to the secondary (PERSONNEL1), and enter:

```
sp_companion MONEY1, "drop"
go
```

In symmetric configuration, do the above and then log in to the primary (MONEY1), and enter:

```
sp_companion PERSONNEL1, "drop"
go
```

Troubleshooting failover on TruCluster Server for HP Tru64

This section includes troubleshooting information about common errors.

Error message 18750

If a companion server issues error message 18750, check the @@cmpstate of the servers. If the primary companion is in normal companion mode, but the secondary companion is in secondary failover mode, the cluster is in an inconsistent state, and you need to manually recover. The inconsistent state may be caused by an sp_companion 'prepare_failback' command failing on the secondary companion. You can determine if this happened by examining the log on the secondary node (located in *$SYBASE/SYBASE_ASE/install/server_name_ha.log*). To recover, perform the following steps :

1 Shut down both the primary and the secondary companions.

2 Reboot the secondary companion.

3 Repair all databases marked “suspect.” To determine which databases are suspect, issue:

```
select name, status from sysdatabases
```

Databases marked suspect have a status value of 320.

4 Allow updates to system tables:

```
sp_configure “allow updates”, 1
```

5 For each suspect failed over database, perform the following:

```
1> update sysdatabase set status=status-256 where name='database_name'
2> go
1> dbcc traceon(3604)
2> go
1> dbcc dbrecover(database_name)
2> go
```

6 From the secondary companion, issue:

```
sp_companion primary_companion_name, prepare_failback
```

For example, from primary companion MONEY1:

```
sp_companion MONEY1, prepare_failback
```

Make sure that this command executes successfully.

7 Restart the TruCluster resource on the primary node.

Recovering from a failed *prepare_failback*

During a failback, if prepare_failback executed successfully on the secondary companion but the primary companion fails to boot, perform the following to rollback and then reissue the prepare_failback command:

1 Check the primary companion's error log and the TruCluster error log to find the reason the server failed to boot, and correct the problems.

2 Stop the TruCluster service on the primary node with caa_stop server_name

3 Log in to the secondary companion as LOGIN_NAME, and issue:

```
dbcc ha_admin ("", "rollback_failback")
dbcc ha_admin ("", "rollback_failover")
```

The companion servers should both be back in the failover mode

4 Restart the TruCluster resource on the primary node.

Location of failover logs

Sybase's failover includes the following logs. These logs are helpful for investigating and diagnosing errors encountered during TruCluster failover:

- *$SYBASE/SYBASE_ASE/install/server_name.ha_log* – contains output of the TruCluster activities, as well as the output from the *server_name.scr* monitoring script. For general TruCluster failure, search for the string "ERROR". For output of the *server_name.scr* script, search for "SYBASE HA MONITOR".

  After determining the reason for the failure, correct it, and restart the TruCluster resource.

CHAPTER 10

Active-Active Configuration for Sun Cluster 2.2

This chapter lists the steps necessary to configure Adaptive Server on Sun Cluster 2.2 in active-active setup.

It includes the following sections:

Configuring hardware and operating system requirements

Sybase high availability requires the following hardware and system components:

- Two homogenous, network systems with similar configurations in terms of resources like CPU, memory, and so on.
- High availability subsystem package and the associated hardware.
- Devices must be accessible to both nodes.
- The system must have a logical volume manager (LVM) to maintain unique device pathnames across the cluster nodes.
- Configure both public and private networks on both the nodes.
- Create volumes and volume groups on the multihost disks.
- Create one logical host on both the primary and secondary hosts.

- Register one or more volume groups on each logical host.
- Use third-party vendor mirroring rather than Sybase mirroring for media failure protection.

For more information about commands for running Sun Cluster 2.2, see the Sun Cluster 2.2 documentation.

See your hardware and operating system documentation for information about installing platform specific high availability software.

Prepare Adaptive Server to work with the HA Subsystem

Perform the tasks in this section to prepare Adaptive Server for high availability configuration.

Install Adaptive Servers

Install the primary and the secondary servers. They must be installed in the same location on each node. The primary companion can be either a newly installed Adaptive Server, or it can be upgraded from a previous version of Adaptive Server with existing databases, users, and so on. The secondary companion must be a newly installed Adaptive Server and cannot have any user logins or user databases. This is to make sure that all user logins and database names are unique within the cluster.

After configuration for failover is complete, you can add user logins and databases to the secondary companion.

If you are installing on the local disk, make sure any databases are created on the multihost disk.

See the installation documentation for your platform for information about installing and configuring Adaptive Server.

Add entries for both Adaptive Servers to the *interfaces file*

The *interfaces file* for both primary and secondary companion must include entries for both companions. For example, the *interfaces file* for the servers used in the setups described in this manual would have entries for both MONEY1 and PERSONNEL1. The server entry in the *interfaces file* must use the same network name that is specified in sysservers. For information about adding entries to the *interfaces file*, see the installation documentation for your platform.

Note *Interfaces file* entries for Sun must use a network type of TLI in hexidecimal format. For example:

```
master tli tcp /dev/tcp /x000224b782f650950000000000000000
query tli tcp /dev/tcp /x000224b782f650950000000000000000
hafailover PERSONNEL1
```

If you use other network types, the monitor for Adaptive Server fails and the high availability services do not function.

Add entries to *interfaces* file for client connections during failover

To enable clients to reconnect to the failed over companion, you must add an additional line to the *interfaces file*. By default, clients connect to the port listed in the query line of the server entry. If that port is not available (because that server has failed over), the client connects to the server listed in the *hafailover* line of the server entry. Here is a sample *interfaces file* for a primary companion named MONEY1 and a secondary companion named PERSONNEL1:

```
MONEY1
    master tli tcp /dev/tcp /x000224b782f650950000000000000000
    query tli tcp /dev/tcp /x000224b782f650950000000000000000
    hafailover PERSONNEL1
```

Use dsedit to add entries to the *interfaces file*. If the interfaces entries already exist, you must modify them to work for failover.

See the Utility Guide for your platform for information about dsedit.

Make the value of *$SYBASE* the same for both companions

If *$SYBASE* is installed on the local disk, then *$SYBASE* on both companions must point to the same directory path name. This is not necessary if *$SYBASE* is installed on the shared disk. You can accomplish this by either:

- Making sure that the *$SYBASE* release directory on each companion is created in the same directory: or
- If the companions have the *$SYBASE* release directory in different locations, create a directory with the same path on both companions that acts as a symbolic link to the actual *$SYBASE* release directory.

  For example, even though primary companion MONEY1 has a release directory of */usr/u/sybase1* and PERSONNEL1 has uses */usr/u/sybase2* as its release directory, *$SYBASE* must point to the same path.

  Both MONEY1 and PERSONNEL1 have */SYBASE*, which they establish as a symbolic link to their respective *$SYBASE* release directories. On MONEY1, */SYBASE* is a link to */usr/u/sybase1*, and on PERSONNEL1, */SYBASE* is a link to */use/u/sybase2*.

Note The remote monitors for either companion do not function if the value of *$SYBASE* is not the same for both companions.

The *sybha* executable

The *sybha* executable allows the Adaptive Server High Availability Basics Services library to interact with each platform's high availability cluster subsystem. The Adaptive Server High Availability Basics Services library calls *sybha*. which is located in *$SYBASE/ASE-12_5/bin*. Before *sybha* can run, you must change its ownership and permissions. You must also edit a file named *sybhauser* in *$SYBASE/ASE-12_5/install*. *sybhauser* contains a list of the users who have System Administrator privileges on the cluster. Sybase strongly recommends that you severely limit the number of users who have System Administrator privileges on the cluster.

As root, perform the following:

1. Add a new group, named *sybhagrp*. You can either add this group to the */etc/group* file, or you can add it to your NIS maps. Add the sybase user to this group (this is the user that owns the *$SYBASE* directory). When the server is started, the sybase user runs the data server. If you have multiple servers running and different users owning the *$SYBASE* directory for each of them, each of these users must be added to the group.
2. Change to the *$SYBASE/$SYBASE_ASE/bin* directory:

   ```
   cd $SYBASE/$SYBASE_ASE/bin
   ```

3. Change the ownership of *sybha* to root:

   ```
   chown root sybha
   ```

4. Change the group for the *sybha* program to *sybhagrp*:

   ```
   chgrp sybhagrp sybha
   ```

5. Modify the file permissions for *sybha* to 4550:

   ```
   chmod 4550 sybha
   ```

6. Change to the *$SYBASE/$SYBASE_ASE/install* directory:

   ```
   cd $SYBASE/ASE-12_5/install
   ```

7. Add the sybase user to the *sybhauser* file. These logins must be in the format of UNIX login IDs, not Adaptive Server logins. For example:

   ```
   sybase
   coffeecup
   spooner
   venting
   howe
   ```

8. Change the ownership of *sybhauser* to root:

   ```
   chown root sybhauser
   ```

9. Modify the file permissions for *sybhauser*:

   ```
   chmod 600 sybhauser
   ```

Create new default device other than master

By default, master is the default device in a newly installed Adaptive Server. This means that, if you create any databases (including the proxy databases used by failover) they are automatically created on the master device. However, adding user databases to master makes it more difficult to restore the master device from a system failure. To make sure that the master device contains as few extraneous user databases as possible, create a new device using disk init. Use sp_diskdefault to specify the new device as the default before you configure Adaptive Server as a companion for failover.

For example, to add a new default device named money_default1 to the MONEY1 Adaptive Server, enter:

```
sp_diskdefault money1_default1, defaulton
```

The master device continues to be a default device until you specifically issue this command to suspend it as the default device:

```
sp_diskdefault master, defaultoff
```

See the *Adaptive Server Reference Manual* for more information about disk init and sp_diskdefault.

Add the local server to *sysservers*

Using sp_addserver, add the local server as the local server in sysservers using the network name specified in the *interfaces file*. For example, if the companion MONEY1 uses the network name of MONEY1 in the *interfaces file*:

```
sp_addserver MONEY1, local, MONEY1
```

You must reboot Adaptive Server for this change to take effect.

Add secondary companion to *sysservers*

Add the secondary companion as a remote server in *sysservers*:

```
sp_addserver server_name
```

By default, Adaptive Server adds the server with an srvid of 1000. You do not need to reboot Adaptive Server for the change to take effect.

Run *installhasvss* to install HA stored procedures

Note You must perform the tasks described in Add entries for both Adaptive Servers to the interfaces file, above, before running *installhasvss*. If you run *installhasvss* before performing these tasks you must re-run *installmaster* to re-install all the system stored procedures.

The *installhasvss* script performs the following tasks to configure Adaptive Server for failover:

- Installs the stored procedures required for failover (for example, sp_companion).
- Installs the SYB_HACMP server in sysservers.

You must have System Administrator privileges to run the *installhasvss* script.

installhasvss in located in the *$SYBASE/$SYBASE_ASE/scripts* directory. To execute the *installhasvss* script, enter:

```
$SYBASE/$SYBASE_OCS/bin/isql -Usa -Ppassword
    -Sservername < ../scripts/installhasvss
```

installhasvss prints messages as it creates stored procedures and creates the SYB_HACMP server.

Assign *ha_role* to system administrator

You must have the ha_role on both Adaptive Servers to run sp_companion. To assign the ha_role, issue the following from isql:

```
sp_role "grant", ha_role, sa
```

You must log out and then log back in to the Adaptive Server for the change to take effect.

Verify configuration parameters

You must enable the following configuration parameters before you configure Adaptive Server for failover:

- enable CIS – Enables Component Integration Services (CIS). This configuration parameter is enabled by default.

- enable xact coordination – Enables Distributed Transaction Management (DTM). This configuration parameter is enabled by default.
- enable HA – Enables Adaptive Server to function as a companion in a high availability system. enable HA is off by default. This configuration is static, so you must reboot Adaptive Server for it to take effect. This parameter causes a message to be written to your error log stating that you have started the Adaptive Server in a high availability system.

See the *System Administration Guide* for information about enabling configuration parameters.

Add thresholds to the master log

If you have not already done so, you must add a threshold to the master log.

1. Define and execute sp_thresholdaction on the master database's log to set a threshold on the number of pages left before a dump transaction occurs. Sybase does not supply sp_thresholdaction. See the *Adaptive Server Reference Manual* for information about creating this system procedure.
2. Place thresholds on the master log segment so it does not fill up:

```
sp_addthreshold "master", "logsegment", 250, sp_thresholdaction
```

3. You must reboot the primary companion for this static parameter to take effect.

Configuring the Sun Cluster subsystem for Sybase failover

See the Sun Cluster high availability subsystem manuals for information about installing the high availability subsystem.

This section assumes that the high availability subsystem is already installed.

Note The *$SYBASE/$SYBASE_ASE/install* directories for each companion must include RUNSERVER files after installing Adaptive Server on the local disks.

---

inst_ha_script sets up the environment for Sybase failover to run with the Sun Cluster high availability subsystem. *inst_ha_scripts* is located in *$SYBASE/%SYBASE_ASE/install*. Before you run this script, you must edit it so that:

- The *$SYBASE* environment variable points to the correct directory.
- The SC_DIR, SYB_DIR, BIN_DIR, and SCSYB_DIR variables are set correctly for your site.

After you have modified *inst_ha_scripts* for your site, as root, run it to:

1 Copy the following scripts to */opt/SUNWcluster/ha/sybase:*

- *hasybase_fmon*
- *hasybase_fmon_start*
- *sybase_ccd_toggles*
- *sybase_db_restart*
- *sybase_db_shutdown*
- *sybase_fm_check*
- *sybase_fm_init*
- *sybase_fm_start*
- *sybase_fm_stop*
- *sybase_get_lh*
- *sybase_get_version*
- *sybase_shutdown*
- *sybase_status*
- *sybase_status_svcs*
- *sybase_svc_abort*
- *sybase_svc_abort_net*
- *sybase_svc_start*
- *sybase_svc_start_net*
- *sybase_svc_stop*
- *sybase_svc_stop_net*

2 Copy the following scripts to */opt/SUNWcluster/bin*

- *hasybase*
- *dbms_utilities*

3 Change the permissions for the files listed in steps 1 and 2 so the owner and group is bin, and have their permissions set to 755. For example, to change the permissions for *sybase_svc_stop*, move to */opt/SUNWcluster/bin* and issue:

```
chmod 755 sybase_svc_stop
chown bin sybase_svc_stop
chgrp bin sybase_svc_stop
```

4 Copy the following scripts to */etc/opt/SUNWscsyb*

- *hasybase_support*
- *hasybase_config_V1*

5 Change the permissions for all these files so the owner is root and group is *sys*, and have their permissions set to 444. For example, to change the permissions for *hasybase_support*, move to */opt/SUNWcluster/bin* and issue:

```
chmod 444 hasybase_support
chown root hasybase_support
chgrp sys hasybase_support
```

Note This ends the tasks *inst_ha_scripts* performs. You must manually perform the rest of the steps in this section.

6 Create a file named *sybtab* in the */var/opt/sybase* directories for both nodes. This file must be identical on both nodes. Edit *sybtab* to contain:

- The name and release directory location of the primary and secondary companion
- The name and release directories of Backup Server for the primary and secondary companion
- The name of *$SYBASE_ASE* and *$SYBASE_OCS* directories

Use the following syntax for each entry:

```
server_name:$SYBASE path
```

where *server_name* is the name of the Adaptive Server or Backup Server.

For example, the *sybtab* file for MONEY1 and PERSONNEL1 would look similar to:

```
MONEY1:/SYBASE12_5
MONEY1_back:/SYBASE12_5
PERSONNEL1:/SYBASE12_5
PERSONNEL1_back:/SYBASE12_5
SYBASE_ASE:ASE-12_5
SYBASE_OCS:OCS-12_5
```

7 Run the following command to make sure the logical hosts are running on both nodes:

```
haget -f mastered
```

haget returns the name of the logical host it is mastering. For example, if this command is run on FIN1, it returns:

```
loghost-MONEY1
```

8 If you have installed the *$SYBASE* directory on a multihost disk, create the setup files for the fault monitor. Copy the following directories (with their subdirectories) and files from *$SYBASE* to */var/opt/sybase*:

- *ctlib.loc*
- *interfaces*
- *charsets/iso_1/*
- *locales/locales.dat*
- *locales/us_english/*

The ctlib.loc file appears in */var/opt/sybase* and in */var/opt/sybase/locales/us_english/iso_1*.

9 Register the Sybase service using the hareg command. Run hareg on only one node of the cluster. As root, enter:

```
hareg -s -r sybase -h
    loghost-primary_companion,loghost-secondary_companion
```

where loghost-*primary_companion* and loghost-*secondary_companion* are the two logical hosts defined on the primary and secondary nodes. For example, to register the Sybase service for primary companion MONEY1 and secondary companion PERSONNEL1:

```
hareg -s -r sybase -h loghost-MONEY1,loghost-PERSONNEL1
```

For more information about creating logical hosts and the hareg command, see the Sun documentation.

10 Check the status of the Sybase service. As root, issue:

```
hareg
```

hareg should return:

```
sybase  off
```

If the output shows that Sybase service is off, then, still as root, activate the Sybase service:

```
hareg -y sybase
```

hareg returns:

```
sybase  on
```

11 Register the primary and secondary companions with the logical hosts by issuing the hasybase command on either node of the cluster:

```
hasybase insert server_name loghost_name 60 10 120 300 srvlogin/srvpasswd
/$SYBASE/$SYBASE_ASE/install/RUNSERVER_file_name
```

where:

- *server_name* – The name of the companion server.
- *loghost-loghost_name* – The name of the logical host on which the companion server is registered.
- 60,10,120,300 – Indicates the probe cycle time, connectivity probe cycle count, probe time-out, and restart delay respectively.
- *srvlogin/srvpasswd* – The login name and password the cluster subsystem uses to monitor and shut down the Adaptive Server.
- *RUNSERVER_file_name* – The run server file for the companion server.

For example to register primary companion MONEY1 on loghost *loghost-MONEY1*:

```
hasybase insert MONEY1 loghost-MONEY1 60 10 120 300 sa/
/SYBASE120/$SYBASE_ASE/install/RUN_MONEY1
```

To register secondary companion PERSONNEL1 on logical *loghost-PERSONNEL1*:

```
hasybase insert PERSONNEL1 loghost-PERSONNEL1 60 10 120 300 sa/
/SYBASE120/$SYBASE_ASE/install/RUN_PERSONNEL1
```

See your Sun documentation for more information about the hasybase command.

12 Issue the hasybase command to start the primary and secondary companions and invoke the monitors for both companion servers:

```
hasybase start companion_name
```

where *companion_name* is the name of the companion you want to start monitoring. For example, to begin monitoring MONEY1:

```
hasybase start MONEY1
```

Note hasybase starts the companions automatically if they are not running when the command is issued.

When two adaptive servers are configured as asymmetric companions, you must start the monitor for the primary companion server and set it to on, and you must stop the monitor for the secondary companion and set it to 'off'. The secondary companion server must be started with its *RUN_server* file, otherwise the failover from primary server to secondary server will not succeed when something goes wrong on the primary server. For example, to configure MONEY1 and PERSONNEL1 as asymmetric companions with MONEY1 as the primary companion:

1 On MONEY1, start monitoring MONEY1 (if MONEY1 is not running, it is started):

```
hasybase start MONEY1
```

2 On PERSONNEL1, start PERSONNEL1

```
SYBASE/$SYBASE_ASE/install/RUN_PERSONNEL1 &
```

When two adaptive servers are configured as symmetric companions, the monitors for both companion servers should be started, otherwise the failover will not succeed. For example, to configure MONEY1 and PERSONNEL1 as symmetric companions:

1 On MONEY1, start monitoring MONEY1 (if MONEY1 is not running, it is started):

```
hasybase start MONEY1
```

2 On PERSONNEL1, start monitoring PERSONNEL1 (if PERSONNEL1 is not running, it is started):

```
hasybase start PERSONNEL1
```

For more information about configuring Adaptive Server for failover, see "Configure companion servers for failover" on page 142.

Configure companion servers for failover

Perform the tasks in this section to configure the Adaptive Servers as primary and secondary companions in a high availability system.

Run *sp_companion* with *do_advisory* option

You must configure the secondary companion with sufficient resources to perform the work of both servers during failover. The secondary companion may have attributes that will prevent a successful cluster operation. For example, if both the primary and secondary companions are configured for 250 user logins, during failover, the secondary companion only has the resources for half the number of potential user logins necessary. Instead, both MONEY1 and PERSONNEL1 should be configured for 500 user logins.

The sp_companion do_advisory option checks the configuration options on both the primary and the secondary companion to make sure a cluster operation (such as configuring an Adaptive Server as a secondary companion) will be successful. sp_companion do_advisory advises you of any configuration options that should be changed.

See Chapter 6, “Running do_advisory” for a complete description of the sp_companion do_advisory option.

Configure for asymmetric configuration

Configure the primary companion for asymmetric configuration. Issue the following from the secondary companion:

```
sp_companion "primary_server_name", configure, with_proxydb, login_name,
password
```

where:

- *primary_server_name* is the name of the primary Adaptive Server as defined in the *interfaces file* entry and in sysservers.
- *with_proxydb* indicates that proxy databases are created on the secondary companion for all databases other than system databases. Any subsequent databases that are added also create proxy databases.
- *login_name* is the name of the user performing this cluster operation (this user must have the ha_role).

- *password* is the password of the person performing this cluster operation.

This example configures an Adaptive Server named PERSONNEL1 as a secondary companion:

```
sp_companion "PERSONNEL1", configure, with_proxydb, sa, Odd2Think
Server 'MONEY1' is alive and cluster configured.
Step: Access verified from Server:'MONEY1' to Server:'PERSONNEL1'
Server 'PERSONNEL1' is alive and cluster configured.
Step: Access verified from Server:'PERSONNEL1' to Server:'MONEY1'
(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)
.....
Step: Companion servers configuration check succeeded
Step: Server handshake succeeded
Step: Master device accessible from companion
Step: Added the servers 'MONEY1' and 'PERSONNEL1' for cluster config
Step: Server configuration initialization succeeded
Step: Synchronizing server logins from companion server
Step: Synchronizing remoteserver from companion server
Step: Synchronizing roles from companion server
Step: Synchronizing server-wide privs from companion server
Step: User information syncup succeeded
Step: Server configured in normal companion mode
```

If user databases already exist while you are using sp_companion, you see messages similar to these:

```
Step: Created proxy database ‘pubs2’
 Step: Proxy status for database has been set. Please Checkpoint the database
'pubs2'
 Step: Server configured in normal companion mode”
 Starting companion watch thread
```

See “Asymmetric companion configuration” on page 19 for more information about asymmetric configuration.

Configure for symmetric configuration

After you configure the companions for asymmetric failover, you can configure them for symmetric configuration. In a symmetric configuration, both servers act as primary and secondary companions. See Figure 3-2 on page 22 for a description of symmetric configuration.

Issue sp_companion from the secondary companion to configure it for symmetric configuration. Use the same syntax as for asymmetric configuration. See "Configure for asymmetric configuration" on page 142," above, for a description of the syntax for sp_companion.

The following example adds an Adaptive Server named MONEY1 as the secondary companion to the Adaptive Server named PERSONNEL1 described in "Configure for asymmetric configuration" on page 142:

```
sp_companion 'MONEY1', configure, with_proxydb, sa, Think2Odd
Server 'MONEY1' is alive and cluster configured.
Step: Access verified from Server:'MONEY1' to Server:'PERSONNEL1'
Server 'PERSONNEL1' is alive and cluster configured.
Step: Access verified from Server:'PERSONNEL1' to Server:'MONEY1'
(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)
.....
Step: Companion servers configuration check succeeded
Step: Server handshake succeeded
Step: Master device accessible from companion
Step: Added the servers 'MONEY1' and 'PERSONNEL1' for cluster config
Step: Server configuration initialization succeeded
Step: Synchronizing server logins from companion server
Step: Synchronizing remoteserver from companion server
Step: Synchronizing roles from companion server
Step: Synchronizing server-wide privs from companion server
Step: User information syncup succeeded
Step: Server configured in normal companion mode
```

Administrating Sybase's failover

This section includes information about using Sybase Failover.

Failing back to the primary companion

Note When you register the logical hosts, disable the automatic failback option. Failback should be a planned event.

Failback moves the primary companion's shared disks from the secondary node back to the primary node and starts the primary companion on the primary node.

1 After the primary host is ready to take over the primary companion, issue this command from the secondary companion:

```
sp_companion primary_companion_name, prepare_failback
```

where *primary_companion_name* is the name of primary companion server.

This command moves the primary companion's logical host to the primary host.

For example, to fail back the primary companion MONEY1, issue this command from the secondary companion PERSONNEL1:

```
sp_companion MONEY1, prepare_failback
```

2 Make sure the primary companion's logical host is moved successfully to the primary host by issuing this command:

```
haget -f mastered
```

The output shows the primary host monitoring the logical host of the primary companion.

3 Start the primary companion:

```
hasybase start primary_companion_name
```

For example, to start the primary companion MONEY1:

```
hasybase start MONEY1
```

4 To resume normal companion mode, issue the following from the primary companion.

```
sp_companion secondary_companion_name, resume
```

where *secondary_companion_name* is the name of the secondary companion server. For example, to resume normal companion mode for primary companion MONEY1:

```
sp_companion PERSONNEL1, resume
```

Note You cannot connect clients with the failover property (for example isql -Q) until you issue sp_companion resume. If you do try to reconnect them after issuing sp_companion prepare_failback, the client hangs until you issue sp_companion resume.

Suspending normal companion mode

Suspended mode temporarily disables the ability of the primary companion to fail over to the secondary companion. To switch from normal companion mode to suspended mode:

1 Stop the high availability subsystem from monitoring the primary and secondary companion as resources. As root, issue:

```
hasybase stop primary_companion_name
hasybase stop secondary_companion_name
```

For example, to stop monitoring primary and secondary companions MONEY1 and PERSONNEL1:

```
hasybase stop MONEY1
hasybase stop PERSONNEL1
```

2 Suspend normal companion mode. From the secondary companion, issue:

```
sp_companion companion_name, suspend
```

For example, to suspend primary companion MONEY1 for maintenance, connect to secondary companion PERSONNEL1 and issue:

```
sp_companion MONEY1, suspend
```

To put the entire logical host in maintenance mode, refer to *Sun Cluster System Administration Guide* for details.

Resuming normal companion mode

To move from suspended mode to normal companion mode:

1 Make sure both companions are running.

2 Begin monitoring the primary and secondary companion as resources. Issue the following as root:

```
hasybase start primary_companion_name
hasybase start secondary_companion_name
```

For example, to begin monitoring primary and secondary companions MONEY1 and PERSONNEL1:

```
hasybase start MONEY1
hasybase start PERSONNEL1
```

3 Resume normal companion mode. From the secondary companion, issue:

```
sp_companion primary_companion_name, resume
```

For example, to resume normal companion mode for primary companion MONEY1:

```
sp_companion MONEY1, resume
```

Note You cannot connect clients with the failover property (for example isql -Q) until you issue sp_companion resume. If you do try to reconnect them after issuing sp_companion prepare_failback, the client hangs until you issue sp_companion resume.

Dropping companion mode

To drop companion mode, issue:

```
sp_companion companion_name, "drop"
```

Dropping companion mode is an irreversible process; you must reconfigure the Adaptive Servers companion servers before they will failover in a high availability system and retain all the functionality that Sybase's failover provides. However, the companion server are still monitored by the high availability subsystem. To stop the high availability subsystem from monitoring the companions, issue:

```
hasybase stop companion_server_name
```

Troubleshooting failover for Sun Cluster

This section includes troubleshooting information about common errors.

- When you shut down a companion, it is restarted on the same node instead of failing over the first time. It fails over on the second shutdown. This is an issue with the Sun Monitor.

  As a work around, set *restart_delay* to a large value (say, 50000) when you issue *hasybase* insert so the companion always fails over within the time specified by the *restart_delay* value if the companion is shut down. To use this work around, you must start the companion using the hasybase start command; you cannot start the companion using the Sybase RUNSERVER file.

- Sybase has not analyzed the *hasybase_config_V1* file for Adaptive Server version 12.5.

- If any of the nodes have a large number of remote NFS mounts, you may see NFS errors, and the response time from this node may be slow when the logical host is deported from this node. Specifically, when you issue sp_companion...prepare_failback from the secondary node, and the primary companions logical host is being deported to the primary host, you will see a slow response from the secondary node. This is temporary, and should revert to the normal response time in a few minutes. To avoid this, make sure your secondary host is working with a normal response time before you issue sp_companion...resume from the primary host.

- If your cluster includes only two nodes and does not include any quorum disks, and a node in your cluster fails, split-brain partitions occur and failover does not proceed without user intervention. Every 10 seconds, the system displays:

```
*** ISSUE ABORTPARTITION OR CONTINUEPARTITION ***
```

  along with the commands you must issue to either abort or continue. To continue, issue:

```
scadmin continuepartition <localnode> <clustername>
```

  To avoid this situation, make sure you have quorum disks defined on both nodes.

- Error message 18750. If a companion server issues error message 18750, check the @@cmpstate of your servers. If the primary companion is in normal companion mode, but the secondary companion is in secondary failover mode, the cluster is in an inconsistent state, and manual recovery is necessary. This inconsistent state may be caused by an sp_companion 'prepare_failback' command failing on the secondary companion. To recover , perform the following steps:

  a Issue the following to stop monitoring both companion servers:

```
hasybase stop companion_name
```

b Shut down both the primary and the secondary companions.

c As root, issue the following to move the primary logical host back to the secondary node:

```
haswitch secondary_host_name primary_log_host
```

d Restart the secondary companion.

e Repair all databases marked “suspect.” To determine which databases are suspect, issue:

```
select name, status from sysdatabases
```

Databases marked suspect have a status value of 320.

f Allow updates to system tables:

```
sp_configure “allow updates”, 1
```

g For each suspect failed over database, perform the following:

```
1> update sysdatabase set status=status-256 where
name='database_name'
2> go
1> dbcc traceon(3604)
2> go
1> dbcc dbrecover(database_name)
2> go
```

h From the secondary companion, issue:

```
sp_companion primary_companion_name, prepare_failback
```

For example, from primary companion MONEY1:

```
sp_companion MONEY1, prepare_failback
```

Make sure that this command executes successfully.

i Issue the following to resume monitoring the primary companion:

```
hasybase start primary_companion_name
```

Recovering from a failed *prepare_failback*

During a failback, if prepare_failback executed successfully on the secondary companion but the primary companion fails to boot, perform the following to rollback and then reissue the prepare_failback command:

1 Check the primary companion's error log and the HADBMS error log to find the reason the server failed to boot, and correct the problems.

2 Issue the following to stop monitoring the primary companion:

```
hasybase stop primary_companion_name
```

3 As root, issue the following to move the primary logical host back to the secondary node:

```
haswitch seconary_host_name primary_log_host
```

4 Log in to the secondary companion and issue:

```
dbcc ha_admin ("", "rollback_failback")
dbcc ha_admin ("", "rollback_failover")
```

The companion servers should be back in the failover mode. For more information about dbcc ha_admin, see "dbcc Options for High Availability Systems" on page 354.

5 Reissue sp_companion...prepare_failback on the secondary companion.

Location of the logs

Use this information for debugging your high availability subsystem:

- Adaptive Server error log (the location is defined in the RUNSERVER file)
- Messages from HASYBASE layer (located in */var/opt/SUNWscsyb/hadbms.log*)
- Console log located in */var/adm/messages*
- CCD logs are located in */var/opt/SUNWCluster/ccd/ccd.log*

CHAPTER 11

Active-Active Configuration for Sun Cluster 3.0

This chapter lists the steps necessary to configure Adaptive Server Enterprise on Sun Cluster 3.0 in the active-active setup.

Configuring hardware and operating system requirements

High availability requires:

- Two homogenous, network systems with similar configurations in terms of resources such as CPU, memory, and so on.
- The high availability package and the associated hardware.
- Devices that are accessible to both nodes.
- A logical volume manager (LVM) to maintain unique device path names across the cluster nodes.
- Volumes or disk suite objects on the multihost disks.
- Third-party vendor mirroring for media failure protection.

- Logical hostname or floating IP address which can be bound to the primary or secondary node. In a symmetric configuration, you need two logical host names, each corresponding to a primary companion.

For more information about requirements for running Sun Cluster 3.0, see the Sun Cluster 3.0 documentation.

See your hardware and operating system documentation for information about installing platform-specific high availability software.

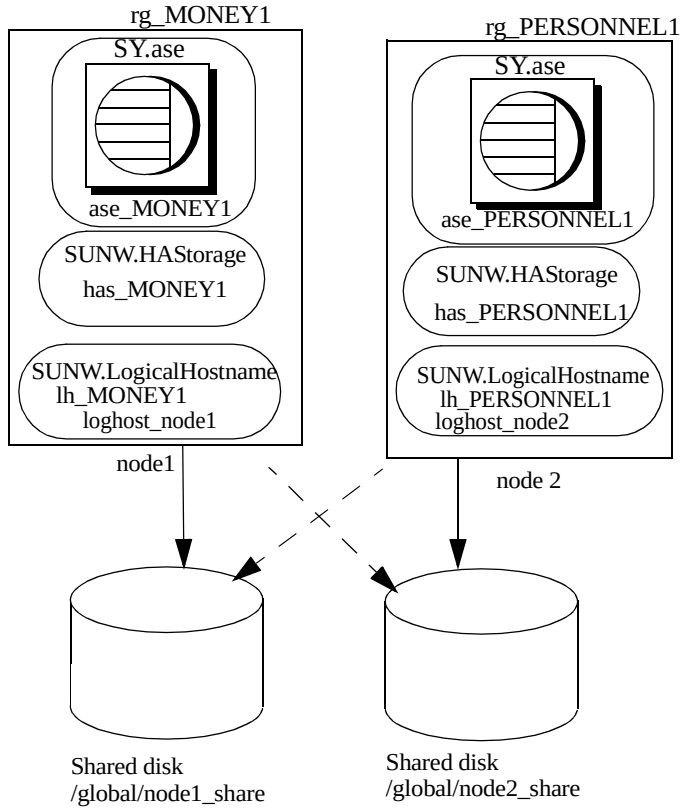
Active-active setup in Sun Cluster 3.0

Figure 11-1 on page 153 depicts active-active configuration in Sun Cluster 3.0.

In Sun Cluster 3.0, Adaptive Server runs as a **data service** and is managed by the Sun Cluster's Resource Group Manager (RGM). Adaptive Server is associated with a resource group which contains the Adaptive Sever resource and all other resources it requires, such as the *SUNW.HAStorage*, *SUNW.HAStoragePlus*, and *SUNW.LogicalHostname* resources.

SY.ase is the resource type of the Adaptive Server resource and it defines various extension properties for the resources of type SY.ase. See "Adaptive Server Resource extension properties" on page 164 for more information. See Sun Cluster 3.0 documentation for more information on standard resource properties.

Figure 11-1: Sample Sun Cluster resource group configuration



In this figure, there are two resource groups, *rg_MONEY1* and *rg_PERSONNEL1*, corresponding to the companion servers MONEY1 and PERSONNEL1 in symmetric configuration.

rg_MONEY1 consists of three resources: *ase_MONEY1* of resource type SY.ase, *has_MONEY1* of resource type SUNW.HAStorage, and *lh_MONEY1* of resource type SUNW.LogicalHostname. The storage resource *has_MONEY1* manages the global file system */global/node1_share* on the shared disk and the logical host resource *lh_MONEY1* manages the logical host name or floating IP address *loghost_node1*. The Adaptive Server resource *ase_MONEY1* depends on *has_MONEY1* and *lh_MONEY1*.

rg_PERSONNEL1 consists of three resources: *ase_PERSONNEL1* of resource type SY.ase, *has_PERSONNEL1* of resource type SUNW.HAStorage, and *lh_PERSONNEL1* of resource type SUNW.LogicalHostname. The storage resource *has_PERSONNEL1* manages the global file system */global/node2_share* on the shared disk and the logical host resource *lh_PERSONNEL1* manages the logical host name or floating IP address *loghost_node2*. The Adaptive Server resource *ase_PERSONNEL1* depends on *has_PERSONNEL1* and *lh_PERSONNEL1*.

Preparing Adaptive Server for active-active setup

Perform the tasks in this section to set up Adaptive Server for active-active high availability configuration.

Installing Adaptive Servers

Install the primary and the secondary servers in the same directory path, but on separate disks. The primary companion can be either a newly installed Adaptive Server, or it can be upgraded from an earlier version of Adaptive Server with existing databases, users, and so on.

The secondary companion must be a newly installed Adaptive Server and cannot have any user logins or user databases. This is to ensure that all user logins and database names are unique within the cluster. After you have completed the configuration you can add user logins and databases to the secondary companion.

See the installation documentation for your platform for information about installing and configuring Adaptive Server.

Adding entries for both Adaptive Servers to the *interfaces file*

The *interfaces file* is for the primary and secondary companions must include entries for both companions. For example, the *interfaces file* for the servers used in the examples in this manual must have entries for both MONEY1 and PERSONNEL1. The server entry in the *interfaces file* must use the network name that is specified in sysservers. For information about adding entries to the *interfaces file*, see the installation documentation for your platform.

For each entry added to the *interfaces file*, the host name must be a logical host. You must create an entry for the logical host in */etc/hosts*, NIS hosts map, or in directory services, whichever is appropriate for your system. The logical host name in the *interfaces file* must be the same as the name used with the -l (lower case L) parameter of the scrgadm command used to add a SUNW.LogicalHostname resource when you configured Adaptive Server to work with the Sun Cluster 3.0 subsystem.

Here is a sample *interfaces file* for a primary companion named MONEY1 and a secondary companion named PERSONNEL1:

```
MONEY1

        query tcp ether loghost_node1 9865
        master tcp ether loghost_node1 9865
        hafailover PERSONNEL1

PERSONNEL1

        query tcp ether loghost_node2 9866
        master tcp ether loghost_node2 9866
        hafailover MONEY1
```

This *interfaces file* is also used by Adaptive Server clients.

Here is a sample */etc/hosts* file with proper entries for the logical host names used in the above *interfaces file*:

```
#
# Internet host table on machine node1
#
127.0.0.1      localhost
10.22.98.43     node1
10.22.98.44     node2
10.22.98.165    loghost_node1
10.22.98.166    loghost_node2
```

Use dsedit to add entries to the *interfaces file*. If the interfaces entries already exist, you must modify them to work for failover.

See the *Adaptive Server Enterprise Utility Guide* for information about dsedit.

Making the value of *$SYBASE* the same for both companions

$SYBASE on both companions must point to the same directory path name. You can accomplish this by using a local release directory:

- Making sure the *$SYBASE* release directory on each companion is created in the same directory.
- Creating a directory with the same path on both companions that acts as a symbolic link to the actual *$SYBASE* release directory, if the companions have the *$SYBASE* release directory in different locations.

  For example, even though primary companion MONEY1 has a release directory of */usr/u/sybase1* and PERSONNEL1 uses */usr/u/sybase2* as its release directory, their *$SYBASE* must point to the same path.

  Both MONEY1 and PERSONNEL1 have */SYBASE*, which they establish as a symbolic link to their respective *$SYBASE* release directories. On MONEY1, */SYBASE* is a link to */usr/u/sybase1*, and on PERSONNEL1, */SYBASE* is a link to */use/u/sybase2*.

Executing *sybha*

The Adaptive Server high availability services library calls sybha, which allows the library to interact with each platform's high availability cluster subsystem. sybha is located in *$SYBASE/$SYBASE_ASE/bin*. Before sybha can run, you must change its ownership and permissions.

You must also edit a file named *sybhauser* in *$SYBASE/$SYBASE_ASE/install*. This file contains a list of the users who have System Administrator privileges on the cluster. Sybase strongly recommends that you limit the number of users who have System Administrator privileges on the cluster.

As root, perform the following:

1 Add a new group called sybhagrp either in the */etc/group* file or to your NIS maps.

2 Add the sybase user to sybhagrp. This is the user who owns the *$SYBASE* directory, and when the server is started, this user runs the dataserver. If you have multiple servers running with different users owning the *$SYBASE* directory, then all of these users must be added to sybhagrp.

3 Change to the *$SYBASE/$SYBASE_ASE/bin* directory.

4 Change the ownership of the sybha program to root:

```
chown root sybha
```

5 Change the group of the sybha program to sybhagrp:

```
chgrp sybhagrp sybha
```

6 Modify the file permissions for sybha to 4550:

```
chmod 4550 sybha
```

7 Change to the *$SYBASE/$SYBASE_ASE/install* directory.

8 Add the sybase user to the *sybhauser* file.

9 Change the permissions of *sybhauser* to root:

```
chown root sybhauser
```

10 Modify the file permissions for *sybhauser* so it can be modified only by root:

```
chmod 600 sybhauser
```

Creating new default devices

By default, master is the default device in a newly installed Adaptive Server. This means that any databases (including the proxy databases used by failover) are automatically created on the master device. However, having user databases on the master device makes it more difficult to restore from a system failure.

To make sure that the master device contains as few user databases as possible, create a new device using disk init. Use sp_diskdefault to specify the new device as the default before you configure Adaptive Server as a companion for failover.

For example, to add a new default device named money_default_1 to the MONEY1 Adaptive Server, enter:

```
sp_diskdefault money1_default1, defaulton
```

The master device continues to be a default device until you specifically make it a non default device:

```
sp_diskdefault master, defaultoff
```

See the *Adaptive Server Enterprise Reference Manual* for more information about disk init and sp_diskdefault.

Adding the local server to *sysservers*

Use sp_addserver to add the local server as the local server in sysservers using the network name specified in the *interfaces file*. For example, if the companion MONEY1 uses the network name of MONEY1 in the *interfaces file* enter:

```
sp_addserver MONEY1, local, MONEY1
```

You must reboot Adaptive Server for this change to take effect.

Adding secondary companion to *sysservers*

Add the secondary companion as a remote server in sysservers:

```
sp_addserver server_name
```

By default, Adaptive Server adds the server with srvid of 1000. You do not need to reboot Adaptive Server for the change to take effect.

Assigning *ha_role* to system administrator

You must have the ha_role on both Adaptive Servers to run sp_companion. To assign the ha_role, issue the following from isql:

```
sp_role "grant", ha_role, sa
```

Log out and then log back in to the Adaptive Server for this change to take effect.

Installing HA stored procedures

Note You must perform the tasks described in “Adding entries for both Adaptive Servers to the interfaces file” on page 154 before you run *installhasvss*. If you run *installhasvss* before performing these tasks, you will have to re-run *installmaster* to reinstall all of the system stored procedures.

The *installhasvss* script:

- Installs the stored procedures required for failover (for example, sp_companion)
- Installs the *SYB_HACMP* server in sysservers

You must have System Administrator privileges to run the *installhasvss* script.

installhasvss is located in *$SYBASE/$SYBASE_ASE/scripts*. To execute *installhasvss*, enter:

```
$SYBASE/$SYBASE_OCS/bin/isql -Usa -Ppassword
-Sservername
< $SYBASE/$SYBASE_ASE/scripts/installhasvss
```

installhasvss prints messages as it creates stored procedures and creates the *SYB_HACMP* server.

Verifying configuration parameters

You must enable the following configuration parameters before you configure Adaptive Server for failover:

- enable CIS – enables Component Integration Services (CIS). This configuration parameter is enabled by default.
- enable xact coordination – enables Distributed Transaction Management (DTM). This configuration parameter is enabled by default.
- enable HA – enables Adaptive Server to function as a companion in a high availability system. enable HA is off by default. Reboot Adaptive Server for this parameter to take effect. This parameter writes a message to the error log stating that you have started the Adaptive Server in a high availability system.

See the *Adaptive Server Enterprise System Administration Guide* for information about enabling configuration parameters.

Adding thresholds to the master log

If you have not already done so, you must add a threshold to the master log.

1. Define and execute sp_thresholdaction in the master database's log to set a threshold on the number of pages left before a dump transaction occurs. Sybase does not supply sp_thresholdaction. See the *Adaptive Server Reference Manual* for information about creating this system procedure.
2. Place thresholds on the master log segment so it does not fill up:

```
sp_addthreshold "master", "logsegment", 250, sp_thresholdaction
```

3. Reboot the primary companion for this static parameter to take effect.

Add user and login for fault monitor

When the HA agent fault monitor, *ase_monitor*, runs thorough probe, it:

1 Connects to the Adaptive Server.

2 Creates a table, inserts an entry into the table, updates the table, and deletes the table.

3 Disconnects from Adaptive Server after the cycle count reaches the value specified by the Adaptive Server resource property *Connect_cycle_count*.

Create or specify a special user and login for the monitor to perform a thorough probe operation. Use isql to connect to the data servers and issue:

sp_addlogin <*user for monitoring ase*>, <*password*>
sp_adduser <*user for monitoring ase*>

For example:

```
sp_addlogin ase_monitor_user,ase_monitor_user_password
sp_adduser ase_monitor_user
```

Note During Adaptive Server configuration, the System Administrator should take into account that the user and login used for probe actually reduces by one the total number of connections available for other purposes. That is, if the total number of connections is 25, the effective number of connections available for other purposes will be 24, as one is used by the fault monitor probe.

Configuring the Sun Cluster 3.0 subsystem

See the *Sun Cluster 3.0 Installation Guide* for information about installing the high availability subsystem.

This section assumes that you have:

- Set up your PATH environment variable to contain */usr/cluster/bin* when you run the cluster system command.
- Installed the Sun Cluster 3.0 High Availability subsystem.
- Installed Adaptive Server and created the required database device files on the shared disk.

- Configured Adaptive Server according to the steps in “Preparing Adaptive Server for active-active setup” on page 154.
- Created *$SYBASE/SYBASE.sh* and edited the file with the required environment for Adaptive Server.

  As the file is executed in the HA agent scripts, protect the file from unauthorized access and make sure only root has read and executable permissions.
- Created *$SYBASE/$SYBASE_ASE/install/RUN_<Dataserver_name>* file. You must specify the Adaptive Server error log with the -e option in this file.

  If -s is specified, it must be the same as the Adaptive Server resource property *Dataserver_name*.
- Installed *$SYBASE/$SYBASE_ASE/SC-3_0* properly (automatically installed with Adaptive Server). This directory must contain all the required files for the Adaptive Server HA agent.

  The default *$SYBASE/$SYBASE_ASE/SC-3_0/* contains these directories:

  - *bin*
  - *etc*
  - *log*

  $SYBASE/$SYBASE_ASE/SC-3_0/bin contains these files:

  - *ase_start*
  - *ase_stop*
  - *ase_monitor_start*
  - *ase_monitor_stop*
  - *ase_update*
  - *ase_validate*
  - *utils.ksh*
  - *ase_monitor*
  - *syscadm*

  $SYBASE/$SYBASE_ASE/SC-3_0/etc contains these files:

  - *SY.ase*

- *ase_monitor_action*
- *ase_login_file*
- *sysc_input_file*

$SYBASE/$SYBASE_ASE/SC-3_0/log contains no files initially, but will eventually contain *Callback_log* and *Monitor_log* files once the Adaptive Server resource is created.

The syscadm script

Use the *syscadm* script to configure and administer Adaptive Server resource groups and their associated resources in Sun Cluster 3.0. You can use *syscadm* to create, remove, and unmanage the Adaptive Server resource group and its resources for both active-active and active-passive configurations. The *syscadm* script is located in *$SYBASE/$SYBASE_ASE/SC-3_0/bin/*.

The create option of the script:

- Registers required resource types with the Resource Group Manager
- For each specified resource group, creates the resource group, creates the specified resources and adds them to the resource group
- Establishes resource dependencies for Adaptive Server resource on the storage and logical host resources

The remove option in the script removes specified resource groups and their resources.

The unmanage option:

- Disables all the resources in the resource group
- Brings the resource group to an offline state
- Then brings the resource group to the unmanaged state

Note You must be logged in as root to run the syscadm script.

The *syscadm* script works with an input file called *sysc_input_file*, which you edit to provide the correct input values for *syscadm* for your configuration. The *sysc_input_file* is located in *$SYBASE/$SYBASE_ASE/SC-3_0/etc/*.

Note Make sure the file is not tampered with when you finish editing the *sysc_input_file*. If erroneous values are included in this file, they may affect your installation. You can change the permissions on this file so only system administrators can edit it.

When editing the *sysc_input_file*, make sure that:

- There are no spaces around "=" in the "<name>=<value>" entries.
- Comments start with #.
- Names ending with 1 correspond to the primary.
- Names ending with 2 correspond to the secondary.

See "Sample sysc_input_file" on page 187 for a sample of the *sysc_input_file*.

The input file is divided into three sections.

- Section 1 – Enter the right-side values for all entries. This section includes entries for the Adaptive Server installation directory, the high availability setup, the Dataserver name, the Nodelist, and so on.
- Section 2 – Enter right-side values for the required entries. For example, if you are using only SUNW.HAStoragePlus resource, you must enter values for SUNW.HAStoragePlus related entries. Do not enter values for the entries you are not using.
- Section 3 – All the entries in this section are assigned default values. You do not need to provide the right-side values unless you want to override the defaults.

For example, to edit the file for the Adaptive Server resource name, change this line:

```
ASE_RNAME="ase_$Dataserver_name"
```

To:

```
ASE_RNAME="my_ase_name"
```

Or, to specify the *RUN_SERVER* file and to set *Debug_callback* flag, change the entry for *OTHER_PROPERTIES*, whose value is a space separated list of <*name*>=<*value*> strings, to:

```
OTHER_PROPERTIES="RUN_server_file=/mypath/RUN_my_ase Debug_callback=TRUE"
```

The syntax for *syscadm* is:

```
syscadm [-v] -c|r|u [primary|secondary|both] -f <sysc_input_file>
syscadm [-v] -r|u <rg1,rg2,...> [-t <ASE_resource_type>]
```

where -c creates resource groups, -r removes resource groups, -u un-manages the resource groups, -f specifies the input file, -v is verbose (shows the Sun Cluster commands as they are being run), -t specifies the Adaptive Server resource type name, if it is not *SY.ase* (useful for -r and -u commands when the input file is not specified).

SUNW.HAStoragePlus resources are created with `AffinityOn=True`.

Note For the active-passive configuration, only *primary* should be used with -c option to create the Adaptive Server resource group.

Adaptive Server Resource extension properties

The Table 11-1 summarizes all extension properties for the Adaptive Server Resource. Refer to the Sun Cluster 3.0 manuals for the details on standard resource properties.

Table 11-1: Extension properties for the SY.ase resource

Property	Default	Description
Sybase_home	None	The home directory of the Adaptive Server installation, and the same as the value for the *$SYBASE* environment variable in an Adaptive Server installation. This property is required to create Adaptive Server resource.
Environment_file	*Sybase_home/SYBASE.sh*	Absolute path to the environment file where you specify the environment you want to pass to the Adaptive Server. This file must be available for proper functioning of the HA agent.
Dataserver_name	None	Name of the Adaptive Server dataserver. This property is required to create Adaptive Server resource.
Backup_server_name	None	Name of the Backup Server.
Monitor_server_name	None	Name of the Monitor Server.
Text_server_name	None	Name of the full-text search server.

Property	Default	Description
Secondary_companion_name	None	Name of the secondary companion server, which is automatically set or unset by sp_companion commands configure or drop respectively. Reserved for active-active setup. Do not set this property manually.
Dataserver_login_file	*Sybase_home/$SYBASE-ASE/SC-3_0/etc/ase_login_file*	Absolute path to a file containing login information for the dataserver. The file consists of two lines; the first line is the login and password for system administrator, the second line is the user login and password for the thorough probe used by the fault monitor program *ase_monitor*.
Action_file	*Sybase_home/$SYBASE_ASE/SC-3_0/etc/ase_monitor_action*	Absolute path to a file that associates error codes with actions to be taken by the fault monitor program *ase_monitor*.
RUN_server_file	*Sybase_home/$SYBASE_ASE/install/RUN_<Dataserver_name>*	Absolute path to the *RUN_SERVER* file for the Adaptive Server specified by the property *Dataserver_name*. Do not include environment variables in this file.
Thorough_probe_script	Ignored. Reserved for future use.	Absolute path to a file containing SQL scripts for the fault monitoring program to perform through probe.
Monitor_log	*Sybase_home/$SYBASE_ASE/SC-3_0/log/ase_monitor_<Dataserver_name>.log*	Absolute path to the log file for the fault monitor program, *ase_monitor*.
Callback_log	*Sybase_home/$SYBASE_ASE/SC-3_0/log/ase_callback_<Dataserver_name>.log*	Absolute path to the log file used by Adaptive Server HA agent callback scripts in *$SYBASE/$SYBASE_ASE/SC-3_0/bin*.
Callback_log_max_size	5000000	Maximum size for the callback log file. If the log size exceeds this limit, the callback log is renamed with current date and time as extension. Any new log messages are written to the *Callback_log*.
Monitor_log_max_size	Ignored. Reserved for future use.	Maximum size for the monitor log file.
Probe_timeout	30	Time, in seconds, after which the fault monitoring probe times out and registers an error.

Property	Default	Description
Restart_delay	30	Time, in seconds, to delay the next probe after a restart.
Debug_monitor	FALSE	If TRUE, the fault monitor program *ase_monitor* logs debugging message to the file specified by property *Monitor_log*.
Debug_callback	FALSE	If TRUE, the Adaptive Server HA agent scripts log debugging messages to the file specified by property *Callback_log*.
Connect_cycle_count	5	The number of thorough probe cycles that an existing connection to Adaptive Server reuses before the connection is dropped and a new one is established.
Failback_strategy	Ignored. Reserved for future use.	Reserved for future versions of Adaptive Server.

Configuring Adaptive Server resource groups

Perform these steps to configure Adaptive Server resource groups on Sun Cluster 3.0:

1 Modify the Adaptive Server resource type registration file *SY.ase*. This file is located in *$SYBASE/$SYBASE_ASE/SC-3_0/etc/*. Find the line for resource type property, *RT_BASEDIR*, which specifies the location of the Adaptive Server HA agent. Change the value to point to the installation location of *$SYBASE/$SYBASE_ASE/SC-3_0/bin*.

For example:

```
RT_BASEDIR=/sybase/ASE-12_5/SC-3_0/bin/
```

Note You cannot use environment variables in *SY.ase*. Use the full path for this value. Substitute the value for *SYBASE*, *SYBASE_ASE* in *$SYBASE/$SYBASE_ASE/SC-3_0/bin.*

2 Create or edit a file that contains Adaptive Server login information for system administrator and the user you added for the fault monitor. The default file is *$SYBASE/$SYBASE_ASE/SC-3_0/etc/ase_login_file*.

If you use another file at a different location, specify the full path for the resource extension property *Dataserver_login_file* when configuring the *SY.ase* resource. The file consists of two lines. The first line is for the login and password, the second line is for the monitor user login and password.

```
login_type <tab> login string
login_type <tab> login_string
```

The only valid value for login type is normal. Values for login string are in the form login-name/password. An example of *$SYBASE/$SYBASE_ASE/SC-3_0/etc/ase_login_file* follows:

```
normal <tab> sa/sa-password
normal <tab> ase_monitor_user/ase_monitor_user_password
```

Note The *ase_login_file* should be protected. After editing the file with the proper values, make the file readable only to the root user. Perform the following:

```
chmod 400 ase_login_file
chown root ase_login_file
chgrp sys ase_login_file
```

3 Create or edit the *sysc_input_file* and run the following syscadm command, it registers the resource type, creates the resource group, adds resources to the resource group, and establishes resource dependencies.

For example, to run the *syscadm* script with an input file named *sysc_input_file*:

```
syscadm -c both -f sysc_input_file
```

For more information on the syscadm script, see “The syscadm script” on page 162.

You can also run these steps manually. See “Configuring the resource groups manually” on page 180 for more information.

Note For a list of the extension properties see Table 11-1 on page 164.

4 For the primary Adaptive Server resource group, run the scswitch command to complete the following tasks:

- Move the resource group to managed state.
- Enable all resources and their monitors.
- Bring the resource group online on the primary node:

```
scswitch -Z -g resource_group_name
```

For example:

```
scswitch -Z -g rg_MONEY1
```

5 For the secondary Adaptive Server resource group, run scswitch command to complete the following tasks:

- Move the resource group to managed state.
- Enable all resources and their monitors.
- Bring the resource group online on the secondary node, that is the primary node of the secondary companion resource group.

```
scswitch -Z -g resource_group_name
```

For example:

```
scswitch -Z -g rg_PERSONNEL1
```

Using SUNW.HAStoragePlus

If you are running Sun Cluster 3.0 with Update2 or later, you can use the *SUNW.HAStoragePlus* resource in the Adaptive Server resource group. You can use *SUNW.HAStoragePlus* resource in place of *SUNW.HAStorage* resource, or you can have both *SUNW.HAStorage* and *SUNW.HAStoragePlus* resources in your resource group.

To add a *SUNW.HAStoragePlus* resource to the Adaptive Server resource group, you must set the *SUNW.HAStoragePlus* resource properties *GlobalDevicePaths* and *FilesystemMountPoints* as required. If you are using *syscadm* script, this can be done by specifying values for corresponding entries in the *sysc_input_file*. To enable collocation, the *SUNW.HAStoragePlus* resource property AffinityOn must be set to *TRUE*, which is done by the *syscadm* script.

To manually add a *SUNW.HAStoragePlus* resource:

1 Register the resource type *SUNW.HAStoragePlus*:

```
scrgadm -a -t SUNW.HAStoragePlus
```

2 Add the *SUNW.HAStoragePlus* resource to the Adaptive Server resource group.

```
scrgadm -a -j hasp_resource_name
-t SUNW.HAStoragePlus
-g resource_group
```

```
-x FilesystemMountPoints=shared_disk_filesystem
-x AffinityOn=TRUE
```

For example:

```
scrgadm -a -j hasp_MONEY1
-t SUNW.HAStoragePlus
-g rg_MONEY1
-x fileSystemMountPoints=/global/node1_share
-x Affinityon=TRUE
```

When using *SUNW.HAStoragePlus* resources, Adaptive Server database devices can be created either on global file system or on a Failover File System (FFS) managed by the *SUNW.HAStoragePlus* resource. In either case, data must reside on shared disk. Specify all corresponding file system and device paths when creating the *SUNW.HAStoragePlus* resource.

3 Enable the *SUNW.HAStoragePlus* resource:

```
scswitch -e -j hasp_resource_name
```

For example:

```
scswitch -e -j hasp_MONEY1
```

4 Establish a resource dependency between *SY.ase* resource and the *SUNW.HAStoragePlus* resource:

```
scrgadm -c -j ase_resource_name
-y Resource_dependencies=hasp_resource_name
```

For example:

```
scrgadm -c -j ase_MONEY1
-y Resource_dependencies=hasp_MONEY1
```

If you are using both *SUNW.HAStorage* and *SUNW.HAStoragePlus* resources, you must specify all the storage resource names as a comma separated list.

```
scrgadm -c -j ase_resource_name
-y Resource_dependencies=hasp_resource_name,hastorage_name
```

For example:

```
scrgadm -c -j ase_MONEY1
-y Resource_dependencies=hasp_MONEY1,has_MONEY1
```

Refer to your Sun Cluster 3.0 documentation for more information about *SUNW.HAStoragePlus* resource type.

Configuring companion servers for failover

Perform the tasks in this section to configure the Adaptive Servers as primary and secondary companions in a high availability system.

Setting the HA services library within Adaptive Server

Since Adaptive Server supports both Solaris SC2.2 and SC3.0, you must load the high availability services library for SC3.0 since the high availability services library for SC2.2 is the default.

First, verify that the high availability services library is available. Use isql to connect to any Adaptive Server:

```
sp_companion "MONEY1", show_cluster
```

Returns message:

```
The default cluster is: SC-2.2.
The current cluster is set to SC-3.0.
Supported cluster systems for SunOS are:
SC-2.2
VCS-1.3.0
SC-3.0
```

Set the high availability services library for SC3.0. For example, from PERSONNEL1, enter:

```
sp_companion "MONEY1", set_cluster, "SC-3.0"
The current cluster is set to SC-3.0.
```

Check the interaction of Adaptive Server with the underlying cluster system. From PERSONNEL1, enter:

```
sp_companion
Server 'PERSONNEL1' is alive and cluster configured.
Server 'PERSONNEL1' is configured for HA services.
Server 'PERSONNEL1' is currently in 'Single server'
mode.
```

Note Perform these steps from only one of the servers in the cluster. The high availability services library is automatically populated to another Adaptive Server in the cluster. If the high availability services library has been loaded on another Adaptive Server, you see the following when you issue sp_companion on server MONEY1:

```
Server 'MONEY1' is alive and cluster configured.
```

```
Server 'MONEY1' is configured for HA services.
Server 'MONEY1' is currently in 'Single server' mode.
```

Since the two companion servers synchronize user information to remove any potential conflict, the user login and password used for thorough probe on secondary companion server should not exist. If they do, during the user information synchronization process, both sp_companion configure and sp_companion do_advisory fail.

To drop the user and login of user probe in the secondary companion server, use sp_droplogin and sp_dropuser.

Running *sp_companion* with *do_advisory*

Before initiating *sp_companion*

Before doing sp_companion do_advisory and sp_companion configure:

1 Disable the monitoring of the secondary ASE server:

```
scswitch -n -M -j secondary-resource
```

2 Drop the user and login for monitor for the secondary ASE server:

```
sp_droplogin secondary_probe_ase
sp_dropuser secondary_probe_ase
```

where *secondary_probe_ase* is the login and user created in the previous section “Add user and login for fault monitor” on page 160.

After successfully executing commands sp_companion do_advisory and sp_companion configure for asymmetric configuration (see the two sections below for detail), perform the following steps:

1 Add the user and login for monitor for secondary ASE server:

```
sp_addlogin secondary_probe_ase, secondary_probe_passwd
sp_adduser secondary_probe_ase
```

where *secondary_probe_ase* is the login and user created in “Add user and login for fault monitor” on page 160.

2 Enable monitoring of secondary ASE server:

```
scswitch -e -M -j secondary-resource
```

You must configure the secondary companion with sufficient resources to perform the work of both servers during failover. The secondary companion may have attributes that will prevent a successful cluster operation. For example, if both the primary and secondary companions are configured for 250 user logins, during failover, the secondary companion has only the resources for half the number of potential user logins necessary. Instead, both MONEY1 and PERSONNEL1 should be configured for 500 user logins.

The sp_companion do_advisory option checks the configuration options on both companions to verify that a cluster operation (such as configuring an Adaptive Server as a secondary companion) will be successful. sp_companion do_advisory advises you of any configuration options that should be changed.

See Chapter 6, "Running do_advisory," for a complete description of the sp_companion do_advisory option.

Configuring for asymmetric configuration

Before you configure for an asymmetric set up, you must first use the scswitch utility to disable the monitoring of the primary and secondary resources.

```
scswitch -n -M -j primary_resource
scswitch -n -M -j secondary_resource
```

Use sp_companion to configure the primary companion for asymmetric configuration:

```
sp_companion "primary_server_name", configure, with_proxydb, login_name,
password
```

where:

- *primary_server_name* is the name of the primary Adaptive Server as defined in the *interfaces file* entry and in sysservers.
- *with_proxydb* indicates that proxy databases are created on the secondary companion for all databases other than system databases. Any subsequent databases that are added also create proxy databases.
- *login_name* is the name of the user performing this cluster operation (user must have ha_role permission).
- *password* is the password of the person performing this cluster operation.

This example configures an Adaptive Server named MONEY1 as a primary companion:

```
sp_companion "MONEY1", configure, with_proxydb, sa, Odd2Think
Server 'MONEY1' is alive and cluster configured.
Step: Access verified from Server:'MONEY1' to Server:'PERSONNEL1'
Server 'PERSONNEL1' is alive and cluster configured.
Step: Access verified from Server:'PERSONNEL1' to Server:'MONEY1'
(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)
.....
Step: Companion servers configuration check succeeded
Step: Server handshake succeeded
Step: Master device accessible from companion
Step: Added the servers 'MONEY1' and 'PERSONNEL1' for cluster config
Step: Server configuration initialization succeeded
Step: Synchronizing server logins from companion server
Step: Synchronizing remoteserver from companion server
Step: Synchronizing roles from companion server
Step: Synchronizing server-wide privs from companion server
Step: User information syncup succeeded
Step: Server configured in normal companion mode
```

If user databases already exist while you are using sp_companion, you see messages similar to these:

```
Step: Created proxy database ‘pubs2’
 Step: Proxy status for database has been set. Please Checkpoint the database
'pubs2'
 Step: Server configured in normal companion mode”
```

Use the scswitch utility to enable the monitoring of the primary resource:

```
scswitch -e -M -j primary_resource
```

To prevent the failover of the secondary companion server in asymmetric configuration, you must disable the monitoring of the secondary resource after failover.

See “Asymmetric companion configuration” on page 19 for more information about asymmetric configuration.

Configuring for symmetric configuration

After you configure the companions for asymmetric failover, you can configure them for symmetric configuration. In a symmetric configuration, both servers act as primary and secondary companions. See Figure 3-2 for a description of symmetric configuration.

Before you configure for a symmetric set up, you must first use the scswitch utility to disable the monitoring of the primary and secondary resources:

```
scswitch -n -M -j primary_resource
scswitch -n -M -j secondary_resource
```

Issue sp_companion from the secondary companion to configure it for symmetric configuration. Use the same syntax as for asymmetric configuration. See “Configuring for asymmetric configuration” on page 172” for a description of the syntax for sp_companion.

The following example adds an Adaptive Server named PERSONNEL1 as the primary companion to the Adaptive Server named MONEY1.

```
sp_companion 'PERSONNEL1', configure, with_proxydb, sa, Think2Odd
Server 'PERSONNEL1' is alive and cluster configured.
Step: Access verified from Server:'MONEY1' to Server:'PERSONNEL1'
Server 'MONEY1' is alive and cluster configured.
Step: Access verified from Server:'PERSONNEL1' to Server:'MONEY1'
(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)
.....
Step: Companion servers configuration check succeeded
Step: Server handshake succeeded
Step: Master device accessible from companion
Step: Added the servers 'MONEY1' and 'PERSONNEL1' for cluster config
Step: Server configuration initialization succeeded
Step: Synchronizing server logins from companion server
Step: Synchronizing remoteserver from companion server
Step: Synchronizing roles from companion server
Step: Synchronizing server-wide privs from companion server
Step: User information syncup succeeded
Step: Server configured in normal companion mode
```

Change the Nodelist property of the secondary resource group to include both nodes:

```
scrgadm -c -g secondary_group -y NodeList=secondary_node,primary_node
```

The following example changes the Nodelist property of the resource group *rg_PERSONNEL1* which contains the Adaptive Server PERSONNEL1:

```
scrgadm -c -g rg_PERSONNEL1 -y NodeList=node2,node1
```

Use the scswitch utility to enable the monitoring of the primary and secondary resources:

```
scswitch -e -M -j primary_resource
scswitch -e -M -j secondary_resource
```

Administering Sybase Failover

This section includes information about using Sybase Failover.

Failing back to the primary companion

Failback moves the primary companion's resource group from the secondary node back to the primary node and starts the primary companion on the primary node.

1. After the primary host is ready to take over the primary companion, disable the monitoring of the secondary resource with the scswitch utility, if you have already not done so:

   ```
   scswitch -n -M -j secondary_resource
   ```

2. Issue the following from the secondary companion:

```
sp_companion primary_companion_name, prepare_failback
```

This command moves the primary companion's resource group back to the primary host.

For example, to failback the primary companion MONEY1, issue this command from the secondary companion, PERSONNEL1:

```
sp_companion MONEY1, prepare_failback
```

Note Alternatively, you can use the following Sun Cluster scswitch utility to fail back the resource group:

```
scswitch -z -h primary_host -g failed_over_group
```

For example, to failback the primary companion MONEY1 on *node1*, issue the following from either the secondary or primary host (if it is running normally under cluster control):

```
scswitch -z -h node1 -g rg_MONEY1
```

3 To resume normal companion mode, disable monitoring of the primary resource with the scswitch utility:

```
scswitch -n -M -j primary_resource
```

4 Issue the following from the primary companion:

```
sp_companion secondary_companion_name, resume
```

For example, to resume normal companion mode for primary companion MONEY1:

```
sp_companion PERSONNEL1, resume
```

5 Enable the monitoring of the primary resource with the scswitch utility:

```
scswitch -e -M -j primary_resource
```

6 If you are in symmetric mode, enable monitoring of the secondary resource with the scswitch utility.

Note You cannot connect clients with the failover property to an Adaptive Server configured for high availability until you issue sp_companion resume. If you try to connect them after issuing sp_companion prepare_failback, the client hangs until you issue sp_companion resume.

Suspending normal companion mode

Suspended mode temporarily disables the ability of the primary companion to fail over to the secondary companion. To switch from normal companion mode to suspended mode:

1 Stop the high availability subsystem from monitoring the primary and secondary companion as resources. As root, issue:

```
scswitch -n -M -j primary-resource-name
scswitch -n -M -j secondary-resource-name
```

2 Suspend normal companion mode. From the secondary companion, issue:

```
sp_companion companion_name, suspend
```

Resuming normal companion mode

To move from suspended mode to normal companion mode:

1 Make sure both companions are running.

2 Resume normal companion mode. From the secondary companion, issue:

```
sp_companion primary_companion_name, resume
```

3 Begin monitoring the primary and secondary companion as resources. Issue the following as root:

```
scswitch -e -M -j primary-resource-name
scswitch -e -M -j secondary-resource-name
```

Dropping companion mode

1 Stop the high availability subsystem from monitoring the companions. Issue:

```
scwitch -n -M -j primary-resource-name
scswitch -n -M -j secondary-resource-name
```

2 Then, to drop companion mode, issue:

```
sp_companion companion_name, "drop"
```

Dropping companion mode is an irreversible process; you must reconfigure the Adaptive Servers companion servers before they fail over in a high availability system.

Verifying high availability on Sun Cluster 3.0

Perform the following verification tests to make sure that you have correctly installed and configured the Adaptive Server high availability (HA) on the Sun Cluster 3.0.

- Clients such as isql can access Adaptive Server data server through logical host.
- Adaptive Server data service and its associated resource group can fail over and fail back.

The following steps assume that two Adaptive Server resource groups have been configured in asymmetric mode.

1 Log in to the primary node for Adaptive Server resource group.

2 Set the Adaptive Server environment variables: SYBASE, S*YBASE_ASE*, *SYBASE_OCS*, and so on. Environment variables are specified with the *Environment_file* extension property.

3 Verify the Adaptive Server resource group is online:

```
scstat -g
```

4 Use isql to connect to the primary data server:

```
isql -Usa -Ppassword -Sprimary-server-name
>select name from sysdatabases
>go
>quit
```

For example:

```
 isql -Usa -Ppassword -SMONEY1
>select name from sysdatabases
>go
name
------------------------------
master
model
sybsystemdb
sybsystemprocs
tempdb

(5 rows affected)
> quit
```

5 Switch the primary resource group to the secondary node. This is a simulated failover.

```
scswitch -z -g primary-resource-group -h secondary-host
```

For example:

```
scswitch -z -g rg_MONEY1 -h node2
```

6 Use isql to connect to the secondary data server and confirm that the databases in the primary data server have been taken over by the secondary data server and can be accessed.

```
isql -Usa -Ppassword -Ssecondary-server-name
>select name from sysdatabases
>go
>quit
```

For example:

```
isql -Usa -Ppassword -SPERSONNEL1
>select name from sysdatabases
>go
    name
    ------------------------------
    master
    master_companion
    model
    sybsystemdb
    sybsystemdb_companion
    sybsystemprocs
    sybsystemprocs_companion
    tempdb

    (8 rows affected)
> quit
```

7 Follow the instructions in the subsection “Failing back to the primary companion” on page 175 to fail back the primary resource group.

8 Use isql to connect to the primary data server and verify that databases in the primary data server have been taken over by the primary data server and can be accessed.

```
isql -Usa -Ppassword -Sprimary-server-name
>select name from sysdatabases
>go
>quit
```

For example:

```
isql -Usa -Ppassword -SMONEY1
>select name from sysdatabases
```

```
>go
     name
    ------------------------------
    master
    model
    sybsystemdb
    sybsystemprocs
    tempdb

    (5 rows affected)
> quit
```

Configuring the resource groups manually

This section describes the commands executed by the *syscadm* script to create and configure the Adaptive Server resource groups.

If necessary, these steps can be done manually, for example to configure, reconfigure, or troubleshoot the Adaptive Server resource groups. Make sure you have properly modified the files *SY.ase* and *ase_login_file* as explained in Steps 1 and 2 in “Configuring Adaptive Server resource groups” on page 166.

You must be logged on as root to run these Sun Cluster 3.0 commands.

Primary companion resource group

1 Register the *SY.ase* resource type.

```
scrgadm -a -t SY.ase -f full-path-of-SY.ase-file
```

For example:

```
scrgadm -a -t SY.ase
-f /sybase/ASE-12_5/SC-3_0/etc/SY.ase
```

Note Install the *SY.ase* resource type only once per cluster. An error message displays if the resource type is already installed.

2 Create a resource group for the primary companion server. Specify the primary and secondary nodes for the resource group property Nodelist.

```
scrgadm -a -g resource_group
-y Nodelist=primary-node,secondary-node
```

For example:

```
scrgadm -a -g rg_MONEY1 -y Nodelist=node1,node2
```

3 Register the SUNW.HAStorage resource type.

```
scrgadm -a -t SUNW.HAStorage
```

4 Create and add the *SUNW.HAStorage* resource to the Adaptive Server resource group. Specify the file system and device paths on the shared disk that must be relocated to the secondary node in case of failover:

```
scrgadm -a -j hastorage_resource_name
-t SUNW.HAStorage
-g resource_group
-x ServicePaths=shared-disk-storage-path
```

For example:

```
scrgadm -a -j has_MONEY1 -g rg_MONEY1
-t SUNW.HAStorage
-x ServicePaths=/global/node1_share
```

5 Create and add the *SUNW.LogicalHostname* resource to the Adaptive Server resource group. Specify a logical hostname or floating IP address that can be relocated to the secondary node in case of failover.

```
scrgadm -a -L -j loghost_resource
-g resource_group
-l logical_hostname
```

For example:

```
scrgadm -a -L -j lh_MONEY1
-g rg_MONEY1
-l loghost_node1
```

6 Create and add the *SY.ase* resource to the Adaptive Server resource group.

Specify any standard resource property values and extension property values for the Adaptive Server resource of type *SY.ase*.

You must specify the following three extension property values; otherwise, the command fails: *Sybase_home*, *Dataserver_name*, and *Dataserver_login_file*. You may let other extension properties use default values.

You may configure the following standard resource properties which are used by the HA agent fault monitor: *Cheap_probe_interval*, *Thorough_probe_interval*, *Retry_count*, and *Retry_interval*.

The following command creates the Adaptive Server resource and adds it to the resource group:

```
scrgadm -a -j ase_resource_name -g resource_group \
   -t SY.ase \
   -x Sybase_home=sybase_home_value \
   -x Environment_file=environment_file_path \
   -x Dataserver_name=dataserver_name_value \
   -x Dataserver_login_file=login_file_path \
   -x RUN_server_file=run_server_file_path
```

For example:

```
scrgadm -a -j ase_MONEY1 -g rg_MONEY1 \
    -t SY.ase \
    -x Sybase_home=/sybase \
    -x Environment_file=/sybase/SYBASE.sh \
    -x Dataserver_name=MONEY1 \
    -x Dataserver_login_file=/sybase/ASE-12_5/SC-3_0/etc/ase_login_file
    -x RUN_server_file=/sybase/ASE-12_5/install/RUN_MONEY1
```

For more information about the standard resource properties, see the Sun Cluster 3.0 documentation. Table 11-1 on page 164 describes the extension properties for the Adaptive Server resource.

7 Establish resource dependency between the *SY.ase* resource and the *SUNW.HAStorage* resource. This means the *SY.ase* resource is online only if the *SUNW.HAStorage* resource is online, and the *SY.ase* resource is offline before the *SUNW.HAStorage* resource is offline:

```
scrgadm -c -j ase_resource_name
-y Resource_dependencies=hastorage_resource_name
```

For example:

```
scrgadm -c -j ase_MONEY1
-y Resource_dependencies=has_MONEY1
```

Note All resources in a resource group implicitly depend on the *SUNW.LogocalHostname* resource if one is added to the resource group.

8 For the primary Adaptive Server resource group, run the scswitch command to complete the following tasks:

- Move the resource group to managed state.
- Enable all resources and their monitors.
- Bring the resource group online on the primary node:

  ```
  scswitch -Z -g resource_group_name
  ```

  For example:

```
scswitch -Z -g rg_MONEY1
```

Note See "Using SUNW.HAStoragePlus" on page 168 to create and add the *SUNW.HAStoragePlus* resource to the Adaptive Server resource group.

Secondary companion resource group

1 Create a resource group for the secondary companion server. Assuming symmetric configuration, specify both primary and secondary nodes for the resource group property Nodelist.

```
scrgadm -a -g resource_group
-y Nodelist=secondary-node, primary-node
```

For example:

```
scrgadm -a -g rg_PERSONNEL1
-y Nodelist=node2,node1
```

Note the order of the nodes in the Nodelist, node2 is the primary node and node1 is the secondary node for the secondary companion server resource group.

For asymmetric configuration use:

```
scrgadm -a -g rg_PERSONNEL1
-y Nodelist=node2
```

2 Create and add the SUNW.HAStorage resource to the Adaptive Server resource group:

```
scrgadm -a -j hastorage_resource_name
-g resource_group
-t SUNW.HAStorage
-x ServicePaths=shared-disk-storage-path
```

For example:

```
scrgadm -a -j has_PERSONNEL1
-g rg_PERSONNEL1
-t SUNW.HAStorage
-x ServicePaths=/global/node2_share
```

3 Create and add *SUNW.LogicalHostname* to the Adaptive Server resource group:

```
scrgadm -a -L
-j loghost_resource
```

```
-g resource_group
-l logical_hostname
```

For example:

```
scrgadm -a -L
-j lh_PERSONNEL1
-g rg_PERSONNEL1
-l loghost_node2
```

4 Create and add the *SY.ase* resource to the Adaptive Server resource group:

```
scrgadm -a -j ase_resource_name
-g resource_group \
-t SY.ase \
-x Sybase_home=sybase_home_value \
-x Environment_file=environment_file_path \
-x Dataserver_name=dataserver_name_value \
-x Dataserver_login_file=login_file_path \
-x RUN_server_file=run_server_file_path
```

For example:

```
scrgadm -a -j ase_PERSONNEL1
-g rg_PERSONNEL1 \
-t SY.ase \
-x Sybase_home=/sybase \
-x Environment_file=/sybase/SYBASE.sh \
-x Dataserver_name=PERSONNEL1 \
-x Dataserver_login_file=/sybase/ASE-12_5/SC-3_0/etc/ase_login_file \
-x RUN_server_file=/sybase/ASE-12_5/install/RUN_PERSONNEL1
```

5 Establish resource dependency between *SY.ase* and *SUNW.HAStorage* so the *SY.ase* resource always depends on *SUNW.HAStorage* resource:

```
scrgadm -c -j ase_resource_name
-y Resource_dependencies=hastorage_resource_name
```

For example:

```
scrgadm -c -j ase_PERSONNEL1
-y Resource_dependencies=has_PERSONNEL1
```

6 For the secondary Adaptive Server resource group, run scswitch command to complete the following tasks:

- Move the resource group to managed state.
- Enable all resources and their monitors.
- Bring the resource group online on the secondary node, that is the primary node of the secondary companion resource group :

```
scswitch -Z -g resource_group_name
```

For example:

```
scswitch -Z -g rg_PERSONNEL1
```

Troubleshooting

This section includes troubleshooting information about common errors.

Recovering from a failed *prepare_failback*

During failback, if prepare_failback is successfully executed on the secondary companion but the primary companion fails to boo, roll back and then reissue the prepare_failback command:

1 Check the cluster system error logs, callback error logs, high availability agent fault monitor error logs, and Adaptive Server error logs to find the reason the failback failed, and correct any problems.

2 Clear any error states in the resource group. To determine the states of resource group, enter:

scha_resourcegroup_get -O RG_STATE -G resource_group_name

For example:

```
scha_resourcegroup_get -O RG_STATE -G rg_MONEY1
```

To determine the states of resource group, enter:

scha_resource_get
-O RESOURCE_STATE_NODE
-R resource_name node_name

For example, to find the state of the resource ase_MONEY1 on node2:

```
scha_resource_get
-O RESOURCE_STATE_NODE -R ase_MONEY1 node2
```

Issue the following command to clear the STOP_FAILED state:

scswitch -c -h node_name -j resource_name -f STOP_FAILED

3 Log in to the secondary companion and issue:

```
dbcc ha_admin("", "rollback_failback")
```

Recovering from a secondary failover on the secondary companion

If the primary companion is in normal companion mode, but the secondary companion is in secondary failover mode, the cluster is in an inconsistent state, and you need to manually recover. The inconsistent state may be caused by sp_companion 'prepare_failback' failing on the secondary companion. To recover, perform the following steps manually:

1 Issue sp_helpdb on secondary companion to see if any primary companion databases (for example, the master_companion) are mounted on the secondary companion.

2 Make sure the primary databases are accessible from the secondary node. To do this, move the primary SUNW.HAStorage resource to the secondary node, which can be done by disabling the primary Adaptive Server resource and starting the primary resource group on the secondary node. For example, the following starts the primary resource group *rg_MONEY1* on the secondary node:

```
scswitch -z -h node2  -g  rg_MONEY1
```

3 Issue ha_admin:

```
dbcc ha_admin("",  "rollback_failover")
```

Prevent failover of secondary companion

You must disable monitoring after a failover.

Changing resource and resource group state

Bring all resources in the Adaptive Server resource group offline and move Adaptive Server resource group to an unmanaged state when doing maintenance on the cluster. Issue the following commands:

```
scswitch -F -g primary-resource-group
scswitch -F -g secondary-resource-group

scswitch -u -g primary-resource-group
scswitch -u -g secondary-resource-group
```

Sample sysc_input_file

The following is the *sysc_input_file* used to create and configure the Adaptive Server resource group rg_MONEY and its resources as shown in Figure 11-1:

```
#############################################################################
##NOTE:                                                                    ##
##    1. This file will be executed by ksh to set environment of syscadm ##
##      You will be responsible for executing anything in this file      ##
##       So, make sure THERE ARE NO DANGEROUS COMMANDS IN THIS FILE      ##
##                                                                       ##
##    2. No spaces around = in the <Variable_name>=<value> pairs         ##
##                                                                       ##
##    3. Comments should start with #, like ksh comments                 ##
##                                                                       ##
##    4. Names ending with 1 correspond to primary, and 2 to secondary   ##
#############################################################################

#############################################################
##  Section1: Must specify right hand side values        ##
#############################################################
# Sybase home directory
SYBASE=”/sybase”

# Valid HA Setups are "ACTIVE_PASSIVE" or "ASYMMETRIC" or "SYMMETRIC"
HA_SETUP=”SYMMETRIC”

# Comma separated list of nodes, Ex: "node1,node2"
Nodelist=”node1,node2”

# ASE Dataserver name and Dataserver login file
Dataserver_name1=”MONEY1”
Dataserver_login_file1=”/sybase/ASE-12_5/SC-3_0/etc/ase_login_file”

Dataserver_name2=”PERSONNEL1”
Dataserver_login_file2=”/sybase/ASE-12_5/SC-3_0/etc/ase_login_file”

#############################################################################
##  Section2: Must specify right hand side values, if required           ##
#############################################################################

# if using Logical Hostname or Virtual/Floating IP address
LOGHOST_NAME_OR_FLOATING_IP1=”loghost_node1”
LOGHOST_NAME_OR_FLOATING_IP2=”loghost_node2”

# if using HAStorage resource
ServicePaths1=”/global/node1_share”
```

```
ServicePaths2=”/global/node2_share”

# if using HAStoragePlus resource
GlobalDevicePaths1=
FilesystemMountPoints1=

GlobalDevicePaths2=
FilesystemMountPoints2=

################################################################################
## Section3: May specify right hand side values to override defaults      ##
################################################################################

# bin of the cluster commands
CLUSTER_BIN="/usr/cluster/bin"

# ASE Resource Type and corresponding registration file
RT_NAME="SY.ase"
RT_FILE="$SYBASE/ASE-12_5/SC-3_0/etc/$RT_NAME"

# Resource Group names
RG_NAME1=”rg_$Dtatserver_name1”
RG_NAME2="rg_$Dataserver_name2"

# ASE Resource names and space separated extended properties
ASE_RNAME1="ase_$Dataserver_name1"
ASE_RNAME2="ase_$Dataserver_name2"

OTHER_PROPERTIES1="RUN_server_file=  Callback_log=  Monitor_log="
OTHER_PROPERTIES2="RUN_server_file=  Callback_log=  Monitor_log="

# Logical Host Resource names
LOGHOST_RNAME1="lh_$Dataserver_name1"
LOGHOST_RNAME2="lh_$Dataserver_name2"

# HA Storage Resource names
HASTORAGE_RNAME1="has_$Dataserver_name1"
HASTORAGE_RNAME2="has_$Dataserver_name2"

# HA Storage Plus Resource names
HASTORAGE_PLUS_RNAME1="hasp_$Dataserver_name1"
HASTORAGE_PLUS_RNAME2="hasp_$Dataserver_name2"
```

Location of the logs

Use the information in these logs to debug the high availability subsystem:

- Adaptive Server error log – the location is defined in the RUNSERVER file. For example:

  ```
  /sybase/ASE-12_5/install/MONEY1.log
  ```

- Adaptive Server HA agent callback scripts log:

```
$SYBASE/$SYBASE_ASE/SC-3_0/log/ase_callback_<server-name>.log
```

  or as specified by the Adaptive Server resource property *Callback_log*.

- Adaptive Server agent fault monitor log:

```
$SYBASE/$SYBASE_ASE/SC-3_0/log/ase_monitor_<server-name>.log
```

  or as specified by the Adaptive Server resource property *Monitor_log*.

- Sun Cluster system log:

  ```
  /var/adm/messages
  ```

CHAPTER 12

Active-Passive Configuration for Sun Cluster 3.0

This chapter lists the steps necessary to configure Adaptive Server on Sun Cluster 3.0 in active-passive setup.

An active-passive configuration is a High Availability configuration that involves two or more nodes and a single Adaptive Server. The node that primarily hosts the Adaptive Server is called the primary nodes; the set of nodes that can potentially host the Adaptive Server are called the secondary nodes.

When the Adaptive Server or any of the resources it depends on, such as disk or the node itself crashes, the Adaptive Server, along with all required resources, is relocated and restarted on a secondary node. This movement from the primary node to the secondary node is called **failover**.

After failover, the node hosting Adaptive Server is considered the primary node until the system administrator performs planned failover, or until the Adaptive Server on the new primary node fails, causing another failover.

After failover all existing client connections are lost. The clients must re-establish their connections and resubmit any uncommitted transactions as soon as the Adaptive Server is started on the secondary node. The client **connection failover** can be done automatically by using HA connections and **self-referencing** the `hafailover` entry in the *interfaces file.* See “Configuring the interfaces file on the client side” on page 200 for information.

You can configure the active-passive setup with multiple secondary nodes so that Adaptive Server can survive multiple failures. With a multi-node setup, Adaptive Server is available to service requests from clients as long as at least one of the primary and secondary nodes is available to host the Adaptive Server and its resources. See “Working with a multi node cluster” on page 214 for more information.

Hardware and Operating System requirements

High availability requires:

- Two homogenous, network systems with similar configurations in terms of resources such as CPU, memory, and so on.
- The high availability package and the associated hardware.
- Devices that are accessible to both nodes.
- A logical volume manager (LVM) to maintain unique device path names across the cluster nodes.
- Volumes or disk suite objects on the multi host disks.
- Third-party vendor mirroring for media failure protection.
- Logical hostname or floating IP address which can be bound to any of the primary and secondary nodes.

For more information about requirements for running Sun Cluster 3.0, see the Sun Cluster 3.0 documentation.

See your hardware and operating system documentation for information about installing platform-specific high availability software.

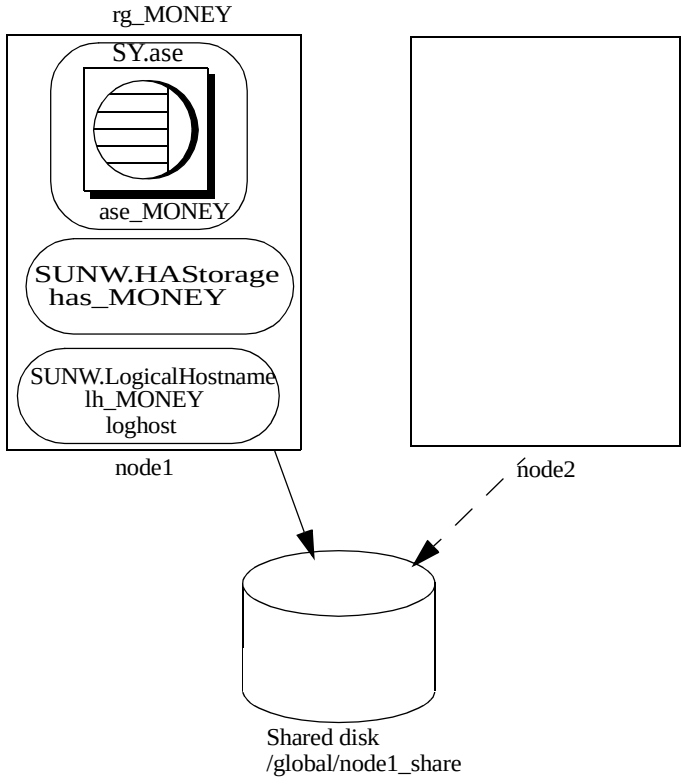
Active-passive setup in Sun Cluster 3.0

A two node active-passive configuration is described in Figure 12-1.

In Sun Cluster 3.0, Adaptive Server runs as a **data service** and is managed by the Sun Cluster's Resource Group Manager (RGM). Adaptive Server is associated with a resource group which contains the Adaptive Sever resource and all other resources it requires, such as the *SUNW.HAStorage*, *SUNW.HAStoragePlus*, and *SUNW.LogicalHostname* resources.

SY.ase is the resource type of the Adaptive Sever resource and it defines various extension properties for the resources of type *SY.ase*. See "Adaptive Server Resource extension properties" on page 164 for more information. See Sun Cluster 3.0 documentation for more information on standard resource properties.

Figure 12-1: Active-passive setup in Sun Cluster 3.0 before failover



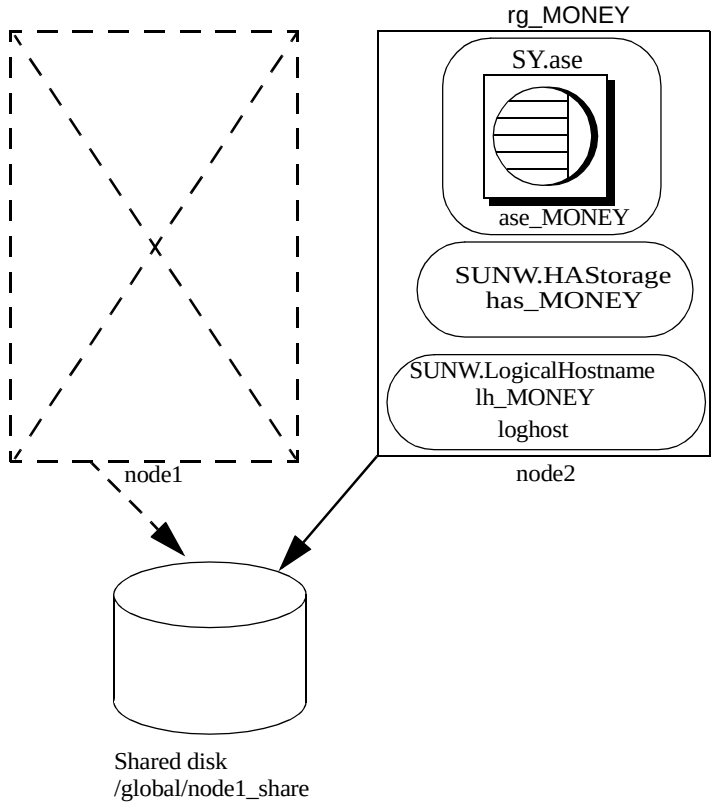
In Figure 12-1, the Adaptive Server *MONEY* is associated with the resource group, *rg_MONEY*, which consists of three resources: the Adaptive Server resource, *ase_MONEY,* of resource type *SY.ase*, the storage resource, *has_MONEY*, of resource type *SUNW.HAStorage*, and the logical host resource, *lh_MONEY*, of resource type *SUNW.LogicalHostname*. The storage resource *has_MONEY* manages the global file system */global/node1_share* on the shared disk. The logical host resource is associated with the logical hostname or floating IP address *loghost*. The Adaptive Server resource *ase_MONEY* depends on *has_MONEY* and *lh_MONEY*.

Initially, the Adaptive Server resource group, *rg_MONEY*, is hosted by the primary node, `node1`, and Adaptive Server *MONEY* serves its clients through the logical hostname *loghost* associated with *lh_MONEY.*

When node1 crashes the resource group *rg_MONEY* along with all its resources is relocated and restarted on the secondary node as shown in Figure 12-2.

After failover, the Adaptive Server runs on node2 and continues to serve its clients using the same logical hostname *loghost*.

Figure 12-2: Active-passive setup on Sun Cluster 3.0 after failover



Note The resource group properties Pingpong_interval and Global_resources_used may affect the failover. For example, in Sun Cluster 3.0 update1, if the Adaptive Server resource group, *rg_MONEY*, is moving between the primary and secondary nodes too frequently (within about 300 seconds), the RGM may fail the failover of the Adaptive Server resource group with the following error:

```
608202 :scha_control: resource group ase_MONEY was
frozen on Global_resources_used within the past 300
```

```
seconds; exiting
```

Refer to the Sun Cluster documentation for more details.

Failing back in an active-passive configuration

You can relocate the Adaptive Server resource group back to the primary node when the node recovers and can successfully host the Adaptive Server resources group. Failback in an active-passive configuration is the same as failing over to the primary node; stopping Adaptive Server and its resources on the current node and then relocating and starting them on the primary node. Failback is not required, but can be done solely for administration purposes. If failback is not done, the recovered primary node acts as secondary node until another failover.

Clients in an active-passive configuration

When failover or failback occurs, all existing client connections are lost. As failback is the same as failover, clients do not see any difference between the two events and the connection failover happens the same way in both cases. However, the client connection failover happens differently depending on the type of connection the client has established with Adaptive Server. Client connections are either HA connections or non-HA connections.

HA connections have the *CS_HAFAILOVER* property set in the connection handle, and the *hafailover* entry in the *interfaces file.* For clients that use the HA connection, failover is transparent. That means that the broken connections are automatically re-established when the Adaptive Server restarts on the secondary node. However, the client must resubmit any uncommitted transactions.

Non-HA connections are regular connections and do not reconnect automatically. With non-HA connections, the clients must first re-establish their connections to Adaptive Server, then resubmit uncommitted transactions.

For more information see, “Configuring the interfaces file on the client side” on page 200.

Preparing Adaptive Server for active-passive setup

Perform the tasks in this section to set up Adaptive Server for an active-passive high availability configuration.

Installing Adaptive Server

You can install Adaptive Server on a global file system or on the local files systems of all the primary and secondary nodes.

Installing on a global file system

If you install Adaptive Server on a global file system, the advantage is that you need only maintain a single server installation. However, you must install the Adaptive Server on a global file system that is managed by the SUNW.HAStorage or SUNW.HAStoragePlus resource in the Adaptive Server resource group, so that the installation directory *$SYBASE* also moves to the secondary node in the case of failover.

Note Do not install *$SYBASE* on a failover file system managed by a SUNW.HAStoragePlus resource.

Installing on local file system

You must maintain the following criteria:

- The installation directory *$SYBASE* must have the same directory path on all the primary and secondary nodes. If different nodes have the *$SYBASE* release directory in different locations, create a directory with the same path on all the primary and secondary nodes that acts as a symbolic link to the respective actual *$SYBASE* release directory paths.

  For example, if the release directory on node1 is */usr/sybase1* and that on node2 is */usr/sybase2*, you can create a symbolic link */sybase* on both the nodes to their respective *$SYBASE* release directories.

  On node1, */sybase* is a link to */usr/sybase1*, and on node2 */sybase* is a link to */usr/sybase2*. Thus, the value of *$SYBASE* points to the same path on both primary and secondary nodes.

- The contents of *$SYBASE* on all the primary and secondary nodes must be consistent:
  - Contents of files such as RUNSERVER, interfaces, *SYBASE.sh*, server configuration file, *<servername>.cfg*, and so on, must be consistent.
  - The contents of *$SYBASE/$SYBASE-ASE/SC-3_0*, especially the files in *etc* and *bin* directories must be consistent.

    - The upgrades and patches must be consistently applied.
- Various log files are created on all the nodes whenever a node hosts the Adaptive Server resource group. For example, *Callback_log*, *Monitor_log*, Adaptive Server and auxiliary server *error logs*, and so on.

You must maintain consistency of the these and any related files whether they are in default directories or you have specified different directory paths for any files using the corresponding Adaptive Server resource properties.

Passing environment to Adaptive Server

Specify the environment you want to pass to the Adaptive Server in the *SYBASE.sh* file. This file is run in the HA agent scripts, so you must protect the file from unauthorized access and make sure only root has read and execute permissions.

The HA agent looks for the file in *$SYBASE* or as specified in the Adaptive Server resource property *Environment_file*. The HA agent may not behave as expected if *SYBASE.sh* is not available.

Note *SYBASE.csh* file is not supported.

Running License Manager in the Cluster

To run the license manager in the cluster, you must run it on all the primary and secondary nodes. This does not require additional steps if *$SYBASE* is installed on the local file systems.

If *$SYBASE* is installed on the global file system, follow the steps below to run the license manager on all nodes using the same *license.dat* file.

- Create the same alias in the */etc/hosts* file for the respective physical hostnames of all the primary and secondary nodes.

  For example, if `node1` and `node2` are the physical hostnames of the primary and secondary nodes, add an alias, such as *license_host*, for both nodes in their */etc/hosts* files.

  For example:

  On node1, */etc/hosts* looks like:

```
10.22.98.43     node1   license_host
10.22.98.44     node2
```

On node2, */etc/hosts* looks like:

```
10.22.98.43     node1
10.22.98.44     node2    license_host
```

- Edit the *license.dat* file in *$SYBASE/$SYBASE_SYSAM/licenses* or as specified by the environment variable *LM_LICENSE_FILE*.

  Change the hostname in the SERVER line to the alias hostname defined in the */etc/hosts* file. Following the above example, the SERVER line e changes from:

  ```
  SERVER node1 any 1700
  ```

  To:

  ```
  SERVER license_host any 1700
  ```

For complete details about Sybase Software Asset Management, refer to the *Installation Guide for Adaptive Server Enterprise*.

Add entry for Adaptive Server to the *interfaces file*

You must maintain an *interfaces file* on both the server side and on the client side. The host name you specify in the *interfaces file* for the Adaptive Server entry must be a logical host name or a floating IP address that can be moved between the primary and secondary nodes.

Configuring the *interfaces file* on the server-side

Modify the *interfaces file* for the server entry to use a floating IP address or logical host name. Do not include the *retry* and *timeout* options for the server entry on the server-side *interfaces file*. The following is an example of the server-side *interfaces file* using the logical hostname loghost.

```
MONEY
    master tcp ether loghost 4010
    query tcp ether loghost 4010
```

Make sure the logical hostname is accessible on all primary and secondary nodes by properly updating the */etc/hosts* or NIS hosts map and */etc/nsswitch.conf* files.

Note Using local */etc/hosts* is recommended over the NIS hosts map in a cluster environment to avoid unnecessary dependency on the NIS server. Modify the */etc/nsswitch.conf* file appropriately.

For example, the */etc/hosts* file for the setup in Figure 12-1 looks like this:

```
#
#internet host table
#
10.22.98.43          node1
10.22.98.44         node2
10.22.98.165         loghost
```

Hosts entry in */etc/nsswitch.conf* file looks like:

```
hosts:    files nis dns
```

Configuring the *interfaces file* on the client side

Client connections can be either HA connections or non-HA connections. Regardless, client connections require that:

- You specify large enough values for the *retry* and *timeout* options in the *interfaces file*. When you determine these values, allow for failover delays, such as starting Adaptive Server on the secondary node, recovery time, and for multiple node failures. Note that the Adaptive Server resource group tries to fail over until it finds a secondary node that can successfully host the resource group.
- The logical hostname is accessible form the client machine.

Note The version of ctlib that is shipped with Adaptive Server version 12.5.0.3 or later contains a corrected implementation of the *retry* and *timeout* features. Re-link your applications to take advantage of the *retry* and *timeout* features.

Non-HA connections

Non-HA connections are regular connections that do not include either the *hafailover* entry in the *interfaces file* or the CS_HAFAILOVER property set in the client connection. When these connections are lost, clients must be able to reconnect to the Adaptive Server after failure. To re-establish the connections, clients must retry enough times, or wait long enough between retries, until failover completes and Adaptive Server is running on the secondary node.

To reconnect to the server, clients can use the *retry* and *timeout* options in the *interfaces file* or the corresponding connection properties. In the following *interfaces file* example, the retry count is 10 and the timeout delay between each retry is 20 seconds:

```
MONEY    10    20
    master tcp ether loghost 4010
    query tcp ether loghost 4010
```

HA connections

HA connections are made with:

- The CS_HAFAILOVER property set at the connection or context level (equivalent to the -Q option of isql).
- The *hafailover* entry in the *interfaces file*, which must point to the Adaptive Server entry to be contacted in case of failover.

In an active-passive configuration, clients must self-reference the *hafailover* entry because they reconnect to the same Adaptive Server after failover. That is, they must set the same server name as the *hafailover* server in the *interfaces file* because the same Adaptive Server is rebooted on the secondary node.

For example, the Adaptive Server entry in the example above can be self-referenced as:

```
MONEY    10    20
    master tli tcp loghost 4010
    query tli tcp loghost 4010
    hafailover MONEY
```

For more information about configuring client connections with the failover property, see Appendix C, “Open Client Functionality in a Failover Configuration.”.

Verify configuration parameters

To set up Adaptive Server for the active-passive configuration you must set the enable HA configuration parameter to 2. By default, enable HA is set to 0.

To set enable HA to 2:

```
sp_configure “enable HA”, 2
```

You must restart Adaptive Server for this parameter to take effect.

See the *System Administration Guide* for information about enabling configuration parameters.

Add thresholds to the master log

If you have not already done so, you must add a threshold to the master log.

1 Define and execute sp_thresholdaction on the master database’s log to set a threshold on the number of pages left before a dump transaction occurs. Sybase does not supply sp_thresholdaction. See the *System Administration Guide* and the *Adaptive Server Reference Manual* for information about creating this system procedure.

2 Place thresholds on the master log segment so it does not fill up:

```
sp_addthreshold "master", "logsegment", 250, sp_thresholdaction
```

3 Restart Adaptive Server for this static parameter to take effect.

Add user and login for fault monitor

When the HA agent fault monitor, *ase_monitor*, runs the thorough_probe, it performs the following transactions:

1 Connects to the Adaptive Server.

2 Creates a table, inserts an entry into the table, updates the table, and deletes the table.

3 Disconnects from Adaptive Server after the thorough_probe runs the number of times as specified by the *Connect_cycle_count*. Next, thorough_probe establishes a new connection.

Create or specify a special user and login for the monitor to perform the thorough_probe operation. Use isql to connect to the dataserver and issue:

sp_addlogin *<user for monitoring ase>*, *<password>*
sp_adduser *<user for monitoring ase>*

For example:

```
sp_addlogin ase_monitor_user,ase_monitor_user_password
sp_adduser ase_monitor_user
```

Note During Adaptive Server configuration, the System Administrator should take into account that the user and login used for probe actually reduces by one the total number of connections available for other purposes. That is, if the total number of connections is 25, the effective number of connections available for other purposes will be 24, as one is used by the fault monitor probe.

Configuring the Sun Cluster 3.0 subsystem

See the *Sun Cluster 3.0 Installation Guide* for information about installing the high availability subsystem.

This section assumes that you have:

- Set up the PATH environment variable to contain */usr/cluster/bin* when the cluster system command is run.
- Installed the Sun Cluster 3.0 High Availability subsystem.
- Installed Adaptive Server and created the required database device files on the shared disk.
- Configured Adaptive Server according to the steps in “Preparing Adaptive Server for active-passive setup” on page 197.
- Created *$SYBASE/SYBASE.sh* and edited the file with the required environment for Adaptive Server. As the file is executed in the HA agent scripts, protect the file from unauthorized access and make sure that only the root user has read and execute permissions.
- Created the *$SYBASE/$SYBASE_ASE/install/RUN_<Dataserver_name>* file. You must specify Adaptive Server error log with the -e option in this file. If -s is specified, it must be the same as the Adaptive Server resource property *Dataserver_name*.

- Installed *$SYBASE/$SYBASE_ASE/SC-3_0* properly (automatically installed with Adaptive Server). This directory must contain all the required files for the Adaptive Server HA agent.

  The default *$SYBASE/$SYBASE_ASE/SC-3_0/* contains these directories:

  - *bin*
  - *etc*
  - *log*

  $SYBASE/$SYBASE_ASE/SC-3_0/bin contains these files:

  - *ase_start*
  - *ase_stop*
  - *ase_monitor_start*
  - *ase_monitor_stop*
  - *ase_update*
  - *ase_validate*
  - *utils.ksh*
  - *ase_monitor*
  - *syscadm*

  $SYBASE/$SYBASE_ASE/SC-3_0/etc contains these files:

  - *SY.ase*
  - *ase_monitor_action*
  - *ase_login_file*
  - *sysc_input_file*

  $SYBASE/$SYBASE_ASE/SC-3_0/log contains no files initially, but will eventually contain *Callback_log* and *Monitor_log* files once the Adaptive Server resource is created.

The syscadm script

Use the *syscadm* script to configure and administer Adaptive Server resource groups, and their associated resources in Sun Cluster 3.0. You can use *syscadm* to create, remove, and un-manage the Adaptive Server resource group and its resources for both active-active and active-passive configurations. The *syscadm* script is located in *$SYBASE/$SYBASE_ASE/SC-3_0/bin/*.

The create option of the script:

- Registers required resource types with the Resource Group Manager,
- For each specified resource group, creates the resource group, creates the specified resources and adds them to the resource group,
- Establishes resource dependencies for Adaptive Server resource on the storage and logical host resources.

Remove option in the script removes specified resource groups and their resources.

Unmanage option:

- Disables all the resources in the resource group,
- Brings the resource group to an offline state,
- Then brings the resource group to the unmanaged state.

Note You must be logged in as root to run the syscadm script.

The *syscadm* script works with an input file called *sysc_input_file*, which you edit to provide the correct input values for *syscadm* for your configuration. The *sysc_input_file* is located in *$SYBASE/$SYBASE_ASE/SC-3_0/etc/*.

Note Make sure the file is not tampered with when you finish editing the *sysc_input_file*. If erroneous values are included in this file, they may affect your installation. You can change the permissions on this file so only system administrators can edit it.

When editing the *sysc_input_file*, make sure that:

- You do not include any spaces around "=" in the "`<name>=<value>`" entries.
- Comments start with #.

- Names ending with 1 correspond to the primary, and 2 to the secondary.

See “Sample sysc_input_file” on page 218 for a sample of the *sysc_input_file*.

The input file is divided into three sections.

- Section 1 – in this section, you must enter the right-side values for all entries. This section includes entries for the Adaptive Server installation directory, the high availability setup, the Dataserver name, the Nodelist, and so on
- Section 2 – in this section, you must enter right-side values for the required entries. For example, if you are using only SUNW.HAStoragePlus resource, you must enter values for SUNW.HAStoragePlus related entries. Do not enter values for the entries you are not using.
- Section 3 – all the entries in this section are assigned default values. You do not need to provide the right-side values unless you want to override the defaults.

For example, to edit the file for the Adaptive Server resource name, change this line:

```
ASE_RNAME="ase_$Dataserver_name"
```

To:

```
ASE_RNAME="MONEY_RNAME"
OTHER_PROPERTIES="RUN_server_file=/mypath/RUN_MONEY
Debug_callback=TRUE"
```

Or, to specify the *RUN_SERVER* file and to set *Debug_callback* flag, change the entry for *OTHER_PROPERTIES*, whose value is a space separated list of <*name*>=<*value*> strings.

The syntax for *syscadm* is:

```
syscadm [-v] -c|r|u [primary|secondary|both] -f <sysc_input_file>
syscadm [-v] -r|u <rg1,rg2,...> [-t <ASE_resource_type>]
```

Where -c creates resource groups, -r removes resource groups, -u un-manages the resource groups, -f specifies the input file, -v is verbose (shows the Sun Cluster commands as they are being run), -t specifies the Adaptive Server resource type name, if it is not *SY.ase* (useful for -r and -u commands when the input file is not specified).

SUNW.HAStoragePlus resources are created with `AffinityOn=True`.

Note For the active-passive configuration, only *primary* should be used with -c option to create the Adaptive Server resource group.

Configuring Adaptive Server resource group

Perform these steps to configure Adaptive Server for high availability on Sun Cluster 3.0:

1 Modify the Adaptive Server resource type registration file *SY.ase*. This file is located in *$SYBASE/$SYBASE_ASE/SC-3_0/etc/*. Find the line for the resource type property, *RT_BASEDIR*, which specifies the location of the Adaptive Server HA agent. Change the value to point to the installation location of *$SYBASE/$SYBASE_ASE/SC-3_0/bin*.

For example:

```
RT_BASEDIR=/sybase/ASE-12_5/SC-3_0/bin/
```

Note You cannot use environment variables in *SY.ase*. For example, substitute the values for *SYBASE*, *SYBASE_ASE* in *$SYBASE/$SYBASE_ASE/SC-3_0/bin*.

2 Create or edit a file that contains Adaptive Server login information for system administrator and the user you added for the fault monitor. The default file is *$SYBASE/$SYBASE_ASE/SC-3_0/etc/ase_login_file*. If you use another file at a different location, specify the full path for the resource extension property *Dataserver_login_file* when configuring the *SY.ase* resource. The file consists of two lines. The first line is for the system administrator login and password, the second line is for the monitor user login and password.

```
login_type <tab> login_string
login_type <tab> login_string
```

The only valid value for login type is normal. Value for login string is in the form login-name/password. An example of *$SYBASE/$SYBASE_ASE/SC-3_0/etc/ase_login_file* follows:

```
normal <tab> sa/sa-password
normal <tab> ase_monitor_user/ase_monitor_user_password
```

Note The *ase_login_file* should be protected. After editing the file with proper values, make the file only readable to the root user. Perform the following:

```
chmod 400 ase_login_file
chown root ase_login_file
chgrp sys ase_login_file
```

3 Create or edit the *sysc_input_file* and run the syscadm script which registers the resource type, creates the resource group, adds resources to the resource group, and establishes resource dependencies. For example:

```
syscadm -c primary
-f $SYBASE/$SYBASE_ASE/SC-3_0/etc/sysc_input_file
```

For more information, see “The syscadm script” on page 162.

You can also run the steps performed by the syscadm command manually. See “Configuring the resource group manually” on page 215 for more information.

Note For a list of the extension properties see Table 11-1 on page 164.

4 Run the scswitch command to complete the following tasks:

- Move the resource group to managed state.
- Enable all resources and their monitors.
- Bring the resource group online on the primary node.

  scswitch -Z -g *resource_group_name*

For example:

```
scswitch -Z -g rg_MONEY
```

Adaptive Server Resource extension properties

The Table 12-1 summarizes all extension properties for the Adaptive Server Resource. Refer to the Sun Cluster 3.0 manuals for the details on standard resource properties.

Table 12-1: Extension properties for the SY.ase resource

Property	Default	Description
Sybase_home	None	The home directory of the Adaptive Server installation, and the same as the value for the *$SYBASE* environment variable in an Adaptive Server installation. This property is required to create Adaptive Server resource.
Environment_file	*Sybase_home/SYBASE.sh*	Absolute path to the environment file where you specify all the environment you want to pass to the Adaptive Server.This file must be available for proper functioning of the HA agent.
Dataserver_name	None	Name of the Adaptive Server dataserver. This property is required to create Adaptive Server resource.
Backup_server_name	None	Name of the Backup Server.
Monitor_server_name	None	Name of the Monitor Server.
Text_server_name	None	Name of the full-text search server.
Secondary_companion_name	None	Name of the secondary companion server, which is automatically set or unset by sp_companion commands configure or drop respectively. Reserved for active-active setup. Do not set this property manually.
Dataserver_login_file	*Sybase_home/$SYBASE-ASE/SC-3_0/etc/ase_login_file*	Absolute path to a file containing login information for the dataserver. The file consists of two lines; the first line is the login and password for sa, the second line is the user login and password for thorough probe used by the fault monitor program *ase_monitor*.
Action_file	*Sybase_home/$SYBASE_ASE/SC-3_0/etc/ase_monitor_action*	Absolute path to a file that associates error codes with actions to be taken by fault monitor program *ase_monitor*.
RUN_server_file	*Sybase_home/$SYBASE_ASE/install/RUN_<Dataserver_name>*	Absolute path to the *RUN_SERVER* file, for the Adaptive Server specified by the property *Dataserver_name*. Do not include environment variables in this file.
Thorough_probe_script	Ignored. Reserved for future use.	Absolute path to a file containing SQL scripts for the fault monitoring program to perform through probe.

Property	Default	Description
Monitor_log	*Sybase_home/$SYBASE_ASE/SC-3_0/log/ase_monitor_<Dataserver_name>.log*	Absolute path to the log file for the fault monitor program, *ase_monitor*.
Callback_log	*Sybase_home/$SYBASE_ASE/SC-3_0/log/ase_callback_<Dataserver_name>.log*	Absolute path to the log file used by Adaptive Server HA agent callback scripts in *$SYBASE/$SYBASE_ASE/SC-3_0/bin.*
Callback_log_max_size	5000000	Maximum size for the callback log file. If the log size exceeds this limit the callback log is renamed with current date and time as an extension. Any new log messages are written to the *callback_log*.
Monitor_log_max_size	Ignored. Reserved for future use.	Maximum size for the monitor log file.
Probe_timeout	30	Time, in seconds, after which the fault monitoring probe times out and registers an error.
Restart_delay	30	Time, in seconds, to delay the next probe after a restart.
Debug_monitor	FALSE	If TRUE, the fault monitor program *ase_monitor* logs debugging message to the file specified by property *Monitor_log*.
Debug_callback	FALSE	If TRUE, the Adaptive Server HA agent scripts log debugging messages to the file specified by property *Callback_log*.
Connect_cycle_count	5	The number of thorough probe cycles that an existing connection to Adaptive Server reuses before the connection is dropped and a new one is established.
Failback_strategy	Ignored. Reserved for future use.	Reserved for future versions of Adaptive Server.

Using SUNW.HAStoragePlus

If you are running Sun Cluster 3.0 with Update2 or later, you can use the *SUNW.HAStoragePlus* resource in the Adaptive Server resource group. You can use *SUNW.HAStoragePlus* resource in place of *SUNW.HAStorage* resource, or you can have both *SUNW.HAStorage* and *SUNW.HAStoragePlus* resources in your resource group.

To add a *SUNW.HAStoragePlus* resource to the Adaptive Server resource group, you must set the *SUNW.HAStoragePlus* resource properties *GlobalDevicePaths* and *FilesystemMountPoints* as required. If using *syscadm* script, this can be done by specifying values for corresponding entries in the *sysc_input_file*. To enable collocation, the *SUNW.HAStoragePlus* resource property AffinityOn must be set to *TRUE*, which is done by the *syscadm* script.

To manually add a *SUNW.HAStoragePlus* resource:

1 Register the resource type *SUNW.HAStoragePlus*:

```
scrgadm -a -t SUNW.HAStoragePlus
```

2 Add the *SUNW.HAStoragePlus* resource to the Adaptive Server resource group.

```
scrgadm -a -j hasp_resource_name
-t SUNW.HAStoragePlus
-g resource_type
-x FilesystemMountPoints=shared_disk_filesystem
-x AffinityOn=TRUE
```

For example:

```
scrgadm -a -j hasp_MONEY
-t SUNW.HAStoragePlus
-g rg_MONEY
-x fileSystemMountPoints=\global\node1_share
-x Affinityon=TRUE
```

When using *SUNW.HAStoragePlus* resources Adaptive Server database devices can be created either on global file system or on Failover File System (FFS) managed by the *SUNW.HAStoragePlus* resource. In either case data must reside on shared disk. Specify all corresponding file system and device paths when creating the *SUNW.HAStoragePlus* resource.

- Enable the *SUNW.HAStoragePlus* resource:

  ```
  scswitch -e -j hastorageplus_name
  ```

  For example:

  ```
  scswitch -e -j hasp_MONEY
  ```

- Establish a resource dependency between *SY.ase* resource and the *SUNW.HAStoragePlus* resource:

  ```
  scrgadm -c -j ase_resource_name
  -y Resource_dependencies=hastorageplus_name
  ```

  For example:

  ```
  scrgadm -c -j ase_MONEY
  ```

```
-y Resource_dependencies=hasp_MONEY
```

If you are using both *SUNW.HAStorage* and *SUNW.HAStoragePlus* resources, you must specify all the storage resource names as a comma separated list.

```
scrgadm -c -j ase_resource_name
-y Resource_dependencies=hastorageplus-name, hastorage-name
```

For example:

```
scrgadm -c -j MONEY
-y Resource_dependencies=has_MONEY,hasp_MONEY
```

Refer to your Sun Cluster documentation for more information about *SUNW.HAStoragePlus* resource type.

Verifying the active-passive configuration

Perform the following tests to make sure you have correctly installed and configured the Adaptive Server for active-passive high availability on Sun Cluster 3.0.

1 Bring the resource group online on its primary node and enable all resources and their fault monitors in the resource group.

For example:

```
scswitch -Z -g rg_MONEY
```

2 Make sure clients such as isql connect to Adaptive Server using the logical host. To verify client connection failover, connect to Adaptive Server. Use the isql to establish the HA connection (modify the *interfaces file* and self-reference hafailover entry, if necessary).

```
isql -Usa -Ppassword -SMONEY -Q
1> select @@servername
2> go

------------------------------
MONEY

(1 row affected)
```

3 Simulate failover, either by:

Shutting down the server

```
isql -Usa -Ppassword -SMONEY
```

```
1> shutdown with nowait
2> go
```

Or by relocating the Adaptive Server resource group to the secondary node:

```
scswitch -z -h node2 -g rg_MONEY
```

4 Check connection failover by issuing a command in the isql session started in step 2.

Enter:

```
1> select @@servername
2> go
CT-LIBRARY error:
         ct_results(): user api layer: internal
Client Library error:
HAFAILOVER:Trying to connect to MONEY server.
1> select @@servername
2> go
------------------------------
MONEY

(1 row affected)
```

5 Simulate failback by relocating the resource group back to the primary node.

```
scswitch -z -h node1 -g rg_MONEY
```

6 Check connection failover by issuing a command in the isql session started in step 2.

Enter:

```
1> select @@servername
2> go
CT-LIBRARY error:
         ct_results(): user api layer: internal
Client Library error:
HAFAILOVER:Trying to connect to MONEY server.
1> select @@servername
2> go
------------------------------
MONEY

(1 row affected)
```

Working with a multi node cluster

This section describes how to configure Adaptive Server in active-passive setup in a Sun Cluster with more than two nodes.

Multi node setup

You can configure an Adaptive Server resource group to withstand multiple node failures by configuring with multiple secondary nodes. All the nodes that can potentially host the Adaptive Server resource group are specified in the resource group property Nodelist.

As examples:

To specify multiple nodes as in Figure 12-3, for the existing Adaptive Server resource group *rg_MONEY*, use:

```
scrgadm -c -g rg_MONEY -y Nodelist=node1,node2,node3
```

To create a resource group with multiple nodes, use:

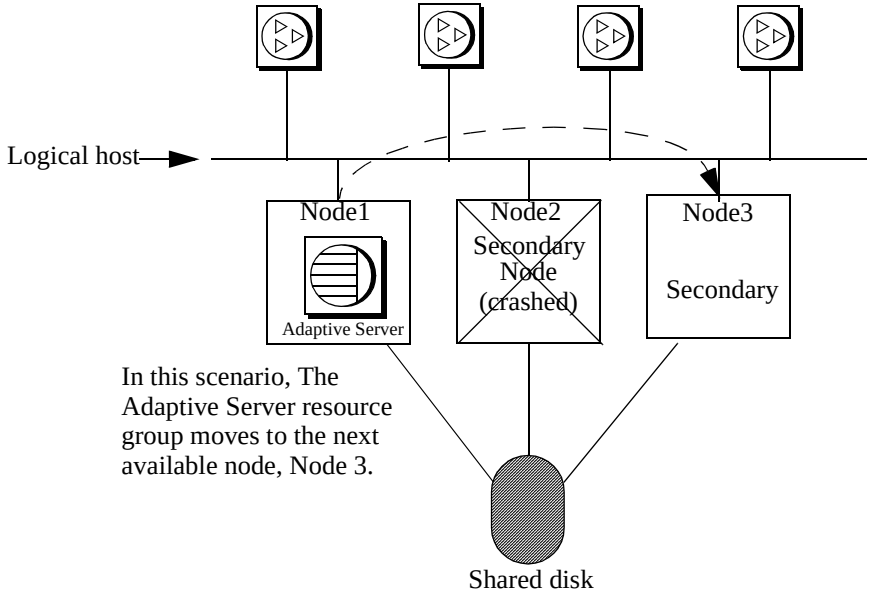
```
scrgadm -a -g rg_MONEY -y Nodelist=node1,node2,node3
```

The order of the node names in the Nodelist is the order of preference in which the Resource Group Manager selects a node to host the Adaptive Server resource group. When Adaptive Server fails over, it does so to the next available secondary node, as determined by the Sun Cluster Resource Group Manager.

As long as at least one of the potential primary nodes is available, the Adaptive Server resource group is available, regardless of the number of crashes.

Figure 12-3 describes a three node setup.

Figure 12-3: Multi-node setup



In this example, the Adaptive Server running on node1 could fail over to either node2 or node3, depending on the order of the node list. Because node2 crashed, Adaptive Server fails over to node3. If the other nodes in this configuration are included in the node list as candidates for secondary nodes, Adaptive Server could fail over to any of them as well.

After the primary node is brought back online, you can either fail back to this node, or keep it available as a candidate for a secondary node.

Configuring the resource group manually

This section describes the commands executed by the *syscadm* script to create and configure the Adaptive Server resource group.

If necessary, these steps can be done manually, for example to configure, reconfigure, or troubleshoot the Adaptive Server resource group. Make sure you have properly modified the files *SY.ase* and *ase_login_file* as explained in Steps 1 and 2 in “Configuring Adaptive Server resource group” on page 207.

You must be logged on as *root* to run these Sun Cluster 3.0 commands.

1 Register the *SY.ase* resource type.

```
scrgadm -a -t SY.ase -f full-path-of-SY.ase-file
```

For example:

```
scrgadm -a -t SY.ase
-f /sybase/ASE-12_5/SC-3_0/etc/SY.ase
```

2 Create the Adaptive Server resource group. Specify the primary and secondary nodes for the resource group property Nodelist:

```
scrgadm -a -g resource_group
-y Nodelist=primary-node,secondary-node
```

For example:

```
scrgadm -a -g rg_MONEY -y Nodelist=node1,node2
```

3 Register the *SUNW.HAStorage* resource type.

```
scrgadm -a -t SUNW.HAStorage
```

4 Create and add the *SUNW.HAStorage* resource to the Adaptive Server resource group. Specify the file system and device paths on the shared disk that must be relocated to the secondary node in case of failover:

```
scrgadm -a -j hastorage_resource_name
-t SUNW.HAStorage
-g resource_group
-x ServicePaths=shared-disk-storage-path
```

For example:

```
scrgadm -a -j has_MONEY -g rg_MONEY
-t SUNW.HAStorage
-x ServicePaths=/global/node1_share
```

5 Create and add the *SUNW.LogicalHostname* resource to the Adaptive Server resource group. Specify a logical hostname or floating IP address that can be relocated to the secondary node in case of failover.

```
scrgadm -a -L -j loghost_resource_name
-g resource_group
-l logicalhostname
```

For example:

```
scrgadm -a -L -j lh_MONEY -g rg_MONEY -l loghost
```

6 Create and add the *SY.ase* resource to the Adaptive Server resource group. Specify any standard resource property values and extension property values for the Adaptive Server resource.

You must specify these three extension property values; otherwise, the command fails: *Sybase_home*, *Dataserver_name*, and *Dataserver_login_file*.

You may let other extension properties use default values. Configure the following standard resource properties which are used by the HA agent fault monitor: *Cheap_probe_interval*, *Thorough_probe_interval*, *Retry_count*, and *Retry_interval*.

The following command creates the Adaptive Server resource and adds it to the resource group:

```
scrgadm -a -j ase_resource_name -g resource_group \
   -t SY.ase \
   -x Sybase_home=sybase_home_value \
   -x Environment_file=environment_file_path \
   -x Dataserver_name=dataserver_name_value \
   -x Dataserver_login_file=login_file_path \
   -x RUN_server_file=run_server_file_path
```

For example:

```
scrgadm -a -j ase_MONEY -g rg_MONEY \
    -t SY.ase \
    -x Sybase_home=/sybase \
    -x Environment_file=/sybase/SYBASE.sh \
    -x Dataserver_name=MONEY \
    -x Dataserver_login_file=/sybase/ASE-12_5/SC-3_0/etc/ase_login_file
    -x RUN_server_file=/sybase/ASE-12_5/install/RUN_MONEY
```

For more information about the standard resource properties, see the Sun Cluster 3.0 documentation. Table 11-1 on page 164 describes the extension properties for the Adaptive Server resource.

7 Establish resource dependency between the *SY.ase* resource and the *SUNW.HAStorage* resource. This means the *SY.ase* resource is online only after the *SUNW.HAStorage* resource is online, and the *SY.ase* resource is offline before the *SUNW.HAStorage* resource is offline:

```
scrgadm -c -j ase_resource_name
-y Resource_dependencies=hastorage_resource_name
```

For example:

```
scrgadm -c -j ase_MONEY
```

```
-y Resource_dependencies=has_MONEY
```

Note All resources in a resource group implicitly depend on the *SUNW.LogocalHostname* resource if one is added to the resource group.

8 Run scswitch to complete the following tasks:

- Move the resource group to managed state.
- Enable all resources and their monitors.
- Bring the resource group online on the primary node.

scswitch -Z -g *resource_group_name*

For example:

```
scswitch -Z -g rg_MONEY
```

Note See "Using SUNW.HAStoragePlus" on page 168 to create and add the *SUNW.HAStoragePlus* resource to the Adaptive Server resource group.

Sample sysc_input_file

The following is the *sysc_input_file* used to create and configure the Adaptive Server resource group rg_MONEY and its resources as shown in Figure 12-1:

```
#############################################################################
##NOTE:                                                                    ##
##    1. This file will be executed by ksh to set environment of syscadm ##
##      You will be responsible for executing anything in this file      ##
##       So, make sure THERE ARE NO DANGEROUS COMMANDS IN THIS FILE      ##
##                                                                       ##
##    2. No spaces around = in the <Variable_name>=<value> pairs         ##
##                                                                       ##
##    3. Comments should start with #, like ksh comments                 ##
##                                                                       ##
##    4. Names ending with 1 correspond to primary, and 2 to secondary   ##
#############################################################################

#############################################################
##  Section1: Must specify right hand side values        ##
#############################################################
```

```
# Sybase home directory
SYBASE=”/sybase”

# Valid HA Setups are "ACTIVE_PASSIVE" or "ASYMMETRIC" or "SYMMETRIC"
HA_SETUP=”ACTIVE_PASSIVE”

# Comma separated list of nodes, Ex: "node1,node2"
Nodelist=”node1,node2”

# ASE Dataserver name and Dataserver login file
Dataserver_name1=”MONEY”
Dataserver_login_file1=”/sybase/ASE-12_5/SC-3_0/etc/ase_login_file”

Dataserver_name2=
Dataserver_login_file2=

##############################################################################
##  Section2: Must specify right hand side values, if required            ##
##############################################################################

# if using Logical Hostname or Virtual/Floating IP address
LOGHOST_NAME_OR_FLOATING_IP1=”loghost”
LOGHOST_NAME_OR_FLOATING_IP2=

# if using HAStorage resource
ServicePaths1=”/global/node1_share”
ServicePaths2=

# if using HAStoragePlus resource
GlobalDevicePaths1=
FilesystemMountPoints1=

GlobalDevicePaths2=
FilesystemMountPoints2=

##############################################################################
## Section3: May specify right hand side values to override defaults      ##
##############################################################################

# bin of the cluster commands
CLUSTER_BIN="/usr/cluster/bin"

# ASE Resource Type and corresponding registration file
RT_NAME="SY.ase"
RT_FILE="$SYBASE/ASE-12_5/SC-3_0/etc/$RT_NAME"
```

```
# Resource Group names
RG_NAME1=”rg_$Dtatserver_name1”
RG_NAME2="rg_$Dataserver_name2"

# ASE Resource names and space separated extended properties
ASE_RNAME1="ase_$Dataserver_name1"
ASE_RNAME2="ase_$Dataserver_name2"

OTHER_PROPERTIES1="RUN_server_file=/sybase/ASE-12_5/install/RUN_MONEY”
OTHER_PROPERTIES2="RUN_server_file=  Callback_log=  Monitor_log="

# Logical Host Resource names
LOGHOST_RNAME1="lh_$Dataserver_name1"
LOGHOST_RNAME2="lh_$Dataserver_name2"

# HA Storage Resource names
HASTORAGE_RNAME1="has_$Dataserver_name1"
HASTORAGE_RNAME2="has_$Dataserver_name2"

# HA Storage Plus Resource names
HASTORAGE_PLUS_RNAME1="hasp_$Dataserver_name1"
HASTORAGE_PLUS_RNAME2="hasp_$Dataserver_name2"
```

Location of the logs

Use this information to debug your high availability subsystem:

- Adaptive Server error log – the location is specified in the RUNSERVER file. For example:

  ```
  /sybase/ASE-12_5/install/MONEY.log
  ```

- Adaptive Server HA agent callback scripts log:

```
$SYBASE/$SYBASE_ASE/SC-3_0/log/ase_callback_<server-name>.log
```

  or as specified by the Adaptive Server resource property *Callback_log.*

- Adaptive Server agent fault monitor log:

```
$SYBASE/$SYBASE_ASE/SC-3_0/log/ase_monitor_<server-name>.log
```

  or as specified by the Adaptive Server resource property *Monitor_log*:

- Sun Cluster 3.0 system log:

  ```
  /var/adm/messages
  ```

CHAPTER 13

Configuring Adaptive Server for Failover on Veritas, 1.3

This chapter discusses how to configure Adaptive Server for Failover on Veritas Cluster Server (VCS).

Make sure you read the Veritas user manuals and familiarize yourself with the Veritas cluster before you implement the steps in this chapter.

Note If you are upgrading from a previous version of Veritas, first review "Upgrading from an agent of resource type Sybase" on page 245 before performing the tasks in this chapter.

Hardware and operating system requirements

Sybase high availability requires the following hardware and system components:

- Two homogenous, networked systems, preferably with similar configurations in terms of resources like CPU, memory, and so on. The two servers should be installed and configured with Solaris 2.8 and VCS version 1.3.0 or later. You should also install the VCS GUI (graphical user interface) to facilitate configuration and administration.
- Import the new resource type *HAase* into the VCS.

- The two systems must have access to shared multihost disks, which store the databases for the Adaptive Server configured for high availability.
- You should install Veritas Volume Manager 3.1 or later to manage disks and create resources like DiskGroup and Volume.
- Use third-party vendor mirroring for media failure protection.
- Create a service group on each system. A service group is a set of resources that provides a specific service. To provide a service for an Adaptive Server that is configured for high availability, the service group should include such resources as DiskGroup, Volume, Mount, IP, NIC, and *HAase* for Sybase Adaptive Server. A sample service group and the resource dependency graph is shown in Figure 13-1. See the *Veritas Cluster Server User's Guide* for more information on how to create a service group and how to add resources to a service group.

  Note Each service group must contain at least two resources with one resource of type HAase. Use the cluster command to establish resource dependency so that the resource of type HAase depends on the other resources.

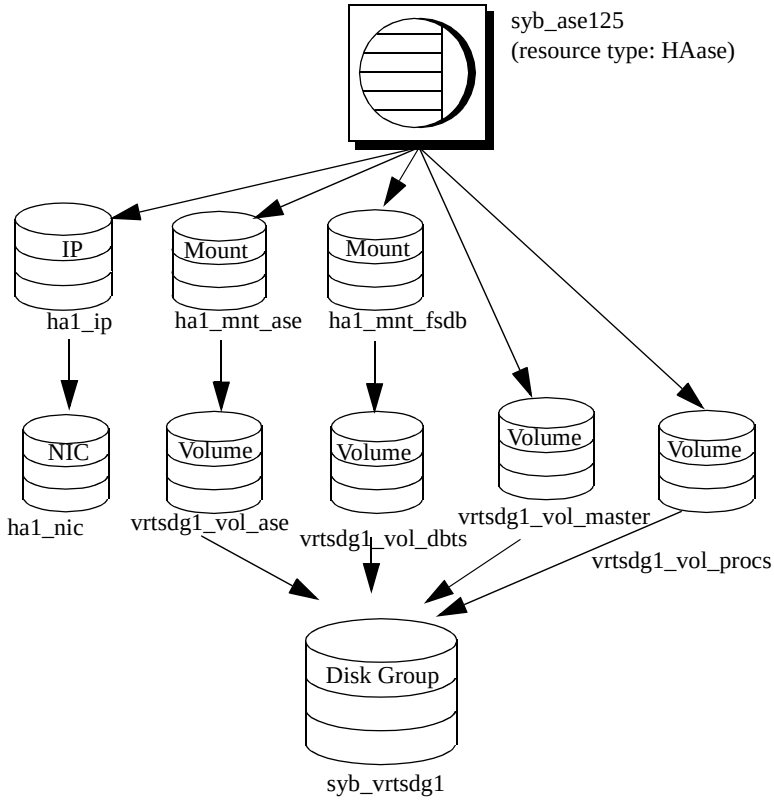
- Configure both public and private networks on both the nodes.

See your hardware and operating system documentation for information about installing platform-specific high availability software.

In Figure 13-1, the configuration of the service group has one DiskGroup, *syb_vrtsdg1*, on which four volumes are created. One volume is for the Adaptive Server installation, one is for databases that are created on the file system, and the other two are for databases created on raw devices. The two mount resources are created for file system of type ufs layering on the volume resources. The resource, *syb_ase125* of type *HAase* is the Adaptive Server installation, which sits on top of the mount resources. *syb_ase125* also requires resource IP, which also requires resource NIC for public network access.

The service group *SybASE* runs on the primary node and another service group, *SybASE2* (not included in Figure 13-1) runs on the secondary node, with a similar configuration:

Figure 13-1: Sample service group running on Veritas Cluster Server



Preparing Adaptive Server to work with the HA subsystem

Perform the tasks in this section to prepare Adaptive Server for a high availability configuration.

Installing Adaptive Servers

Install both the primary and the secondary servers. They can be installed either on shared or local disks. The primary companion can be either a newly installed Adaptive Server, or it can be upgraded from an earlier version of Adaptive Server with existing databases, users, and so on. The secondary companion must be a newly installed Adaptive Server and cannot have any user logins or user databases, to ensure that all user logins and database names are unique within the cluster. After configuration for failover is complete, you can add user logins and databases to the secondary companion.

If you are installing on the local disk, make sure any databases are created on the multihost disk.

See the installation documentation for your platform for information about installing and configuring Adaptive Server.

Adding entries for both Adaptive Servers to the *interfaces file*

The *interfaces file* for both primary and secondary companion must include entries for both companions. For example, the *interfaces file* for the servers used in the setups described in this manual would have entries for both MONEY1 and PERSONNEL1. The server entry in the *interfaces file* must use the same network name that is specified in sysservers. For information about adding entries to the *interfaces file*, see the installation documentation for your platform.

Adding entries to the *interfaces file* for client connections during failover

To enable clients to reconnect to the failed over companion, you must add a line to the *interfaces file*. By default, clients connect to the port listed in the query line of the server entry. If that port is not available (because that server has failed over), the client connects to the server listed in the *hafailover* line of the server entry. Here is a sample *interfaces file* for a primary companion named MONEY1 and a secondary companion named PERSONNEL1:

```
MONEY1
        master tli tcp MONEY 9678
        query tli tcp MONEY 9678
        hafailover PERSONNEL1

PERSONNEL1
        master tli tcp PERSONNEL 9679
```

```
query tli tcp PERSONNEL 9679
```

Use dsedit to add entries to the *interfaces file*. If the interfaces entries already exist, you must modify them to work for failover.

See the *Adaptive Server Enterprise Utility Guide* for information about dsedit.

The sybha executable

The *sybha* executable provides the ability for the Adaptive Server High Availability Basis Services library to interact with each platform's high availability cluster subsystem. The Adaptive Server High Availability Basis Services library calls *sybha*, which is located in *$SYBASE/ASE-12_5/bin*. Before *sybha* can run, you must change its ownership and permissions. You must also edit a file named *sybhauser* in *$SYBASE/ASE-12_5/install*. *sybhauser* contains a list of the users who have System Administrator privileges on the cluster. Sybase strongly recommends that you limit the number of users who have System Administrator privileges on the cluster.

As root, perform the following:

1 Add a new group named *sybhagrp*. You can either add this group to the */etc/group* file, or you can add it to your NIS maps. Add the sybase user to this group (this is the user that owns the *$SYBASE* directory). When the server is started, the sybase user runs the data server. If you have multiple servers running and different users owning the *$SYBASE* directory for each of them, each of these users must be added to the group

2 Change to the *$SYBASE/$SYBASE_ASE/bin* directory:

```
cd $SYBASE/$SYBASE_ASE/bin
```

3 Change the ownership of *sybha* to root:

```
chown root sybha
```

4 Change the group for the *sybha* program to *sybhagrp*:

```
chgrp sybhagrp sybha
```

5 Modify the file permissions for *sybha* to 4550:

```
chmod 4550 sybha
```

6 Change to the *$SYBASE/$SYBASE_ASE/install* directory:

```
cd $SYBASE/ASE-12_5/install
```

7 Add the sybase user to the *sybhauser* file. These logins must be in the format of UNIX login IDs, not Adaptive Server logins. For example:

```
sybase
coffeecup
spooner
venting
howe
```

8 Change the ownership of *sybhauser* to root:

```
chown root sybhauser
```

9 Modify the file permissions for *sybhauser*:

```
chmod 600 sybhauser
```

Creating a new default device

By default, master is the default device in a newly installed Adaptive Server. This means that, if you create any databases (including the proxy databases used by failover), they are automatically created on the master device. Adding user databases to the master device makes it difficult to restore the master device from a system failure. To make sure that the master device contains as few extraneous user databases as possible, create a new device using disk init. Use sp_diskdefault to specify the new device as the default before you configure Adaptive Server as a companion for failover.

For example, to add a new default device named money_default1 to the MONEY1 Adaptive Server, enter:

```
sp_diskdefault money1_default1, defaulton
```

The master device continues to also be a default device until you specifically issue the following to suspend it as the default device:

```
sp_diskdefault master, defaultoff
```

See the *Adaptive Server Reference Manual* for more information about disk init and sp_diskdefault.

Adding the local server to *sysservers*

Use sp_addserver to add the local server in sysservers using the network name specified in the *interfaces file*. For example, if the companion MONEY1 uses the network name of MONEY1 in the *interfaces file*:

```
sp_addserver MONEY1, local, MONEY1
```

You must reboot Adaptive Server for this change to take effect.

Adding secondary companion to *sysservers*

Add the secondary companion as a remote server in *sysservers*:

```
sp_addserver server_name
```

By default, Adaptive Server adds the server with a srvid of 1000. You do not need to reboot Adaptive Server for the change to take effect.

Assigning *ha_role*

You must have the ha_role on both Adaptive Servers to run sp_companion. To assign the ha_role, issue the following from isql:

```
sp_role "grant", ha_role, sa
```

You can use set role to turn the role on or off for this session.

Installing HA stored procedures

Note You must already have added both servers to the *interfaces file* before you can install the high-availability stored procedures. If you run *installhasvss* before performing these tasks, you will have to reinstall all the system stored procedures.

The *installhasvss* script:

- Installs the stored procedures required for failover (for example, sp_companion).
- Installs the SYB_HACMP server in sysservers.

You must have System Administrator privileges to run *installhasvss*.

installhasvss is located in *$SYBASE/ASE-12_5/scripts*. To execute *installhasvss*, enter:

```
$SYBASE/OCS-12_5/bin/isql -Usa -Ppassword -Sservername
< ../scripts/installhasvss
```

installhasvss prints messages as it creates stored procedures and creates the SYB_HACMP server.

Verifying configuration parameters

You must enable the following configuration parameters before you configure Adaptive Server for failover:

- enable CIS – enables Component Integration Services (CIS). This configuration parameter is enabled by default.
- enable xact coordination – enables Distributed Transaction Management (DTM). This configuration parameter is enabled by default.
- enable HA – enables Adaptive Server to function as a companion in a high availability system. enable HA is off by default. You must reboot Adaptive Server for it to take effect. This parameter causes a message to be written to your error log stating that you have started the Adaptive Server in a high availability system. You need to purchase the ASE_HA license option to use Adaptive Server with Failover. See the installation guide for your platform for information about enabling the ASE_HA license.

See the *System Administration Guide* for information about enabling configuration parameters.

Adding thresholds to the master log

Failing over, failing back, creating proxy databases and so on are log-intensive activities. If you do not have adequate log space, any of these activities can fail. If you have not already done so, you must add a threshold to the master log.

1. Define and execute sp_thresholdaction on the master database's log to set a threshold on the number of pages left before a dump transaction occurs. Sybase does not supply sp_thresholdaction. See the *Adaptive Server Reference Manual* for information about creating this system procedure.
2. Place thresholds on the master log segment so it does not fill up:

```
sp_addthreshold "master", "logsegment", 250, sp_thresholdaction
```

3. You must reboot the primary companion for this static parameter to take effect.

Configuring the Veritas subsystem for Sybase Failover

This section assumes that you have already installed the high availability subsystem. See the *VCS 3.5 Installation Guide* and *VCS 3.5 User's Guide* for information about installing and using the Veritas Cluster Server high availability subsystem.

Installing the HAase agent

Perform the following steps to install the *HAase* agent on each node of the cluster (you must have root permission to run the following commands):

1 Change to the *$SYBASE/$SYBASE_ASE/install/veritas/HAase* directory:

```
cd $SYBASE/$SYBASE_ASE/install/veritas/HAase
```

2 Execute the installation script:

```
perl installHAase.pl
```

The installation script:

- Copies the *HAase* resource type file *HaaseTypes.cf* to */etc/VRTSvcs/conf/config/* on local system
- Makes a new directory, */opt/VRTSvcs/bin/HAase*, if it does not already exist.
- Copies the following agent binary and scripts to */opt/VRTSvcs/bin/HAase/* on the local system:
  - *HAaseAgent*
  - *online*
  - *offline*
  - *clean*
  - *sybhautil.pm*
  - *attr_changed*

Creating an Adaptive Server login file

Create a file that contains the Adaptive Server login information for system administrator and for the user you added for the fault monitor. A sample file containing a template for this information is located in: *$SYBASE/$SYBASE_ASE/install/veritas/HAase/ase_login_file*.

This file consists of two lines. The first line is the login and password for system administrator; the second line is the monitor user login and password.

```
login-type<tab>login string
login-type<tab>login string
```

The *login-type* and the *login string* must be separated by a `tab` character.

Note If you use another file at a different location, specify the full path for the resource extension property *Dataserver_login_file* when configuring the *HAase* resource.

The default value for *login-type* is `normal`. Values for *login string* are in the form *login-name/password*. For example:

```
normal     sa/sa-password
normal     probe-user/probe-password
```

For security reasons, the *ase_login_file* should be securely protected so that `read` and `write` access permissions are restricted to root. To maintain this security, perform the following:

```
chmod 400 ase_login_file
chown root ase_login_file
chgrp  sys  ase_login_file
```

Note Sybase strongly recommends that you use a non-empty password. If you use an empty password, the agent scripts generate a warning message.

Importing the HAase resource type

There are two methods to import the HAase resource type:

- Use the cluster GUI tool to import the new resource type, *HAase*. See your VCS User Guide for more information.

- Use cluster commands hatype and haattr to manually import the new resource type from the command line. See your VCS User Guide for more information.

Starting the HAase agent

You can start the *HAase* agent by either:

- Restarting the Veritas Cluster, or
- Using the cluster commands to manually start the *HAase* agent

The second method is more desirable since it causes no disruption. The steps to manually start the *HAase* agent are:

1 Check the status of *HAase* agent with the *haagent* utility:

```
#haagent -display HAase
#Agent   Attribute  Value
HAase     AgentFile
HAase     Faults     0
HAase     Running    No
HAase      Started      No
```

2 Start the *HAase* agent on the host *myhost* with the *haagent* utility:

```
# haagent -start HAase -sys myhost
VCS:10001:Please look for messages in the log file
```

3 Check the status of *HAase* agent using the *haagent* utility:

```
# haagent -display HAase
#Agent    Attribute   Value
HAase     AgentFile
HAase     Faults      0
HAase     Running     Yes
HAase      Started       Yes
```

Adding the *HAase* resource

Each service group should contain an *HAase* resource. The following table shows the attributes of an *HAase* resource.

Property	Datatype, dimension, and default	Description
Sybase_home	string, scalar, null	The home directory of the Adaptive Server installation, and the same as the value for the environment variable SYBASE in an Adaptive Server installation.
Dataserver_name	string, scalar, null	Name of the Adaptive Server that is supplied at the time of configuration.
Backup_server_name	string, scalar, null	Name of the Backup Server that is supplied at the time of configuration.
Monserver_name	string, scalar, null	Name of the Monitor Server that is supplied at the time of configuration.
Textserver_name	string, scalar, null	Name of the full-text search server that is supplied at the time of configuration.
Secondary_companion_name	string, scalar, null	Name of secondary companion server which is set when you run the 'sp_companion configure' command.
Dataserver_login_file	string, scalar, null	Absolute path to a file containing login information for current data server. The file consists of two lines; the first line is the login and password for system administrator, the second line is the user login and password used for thorough probe used by the HA agent monitor.
RUN_server_file	string, scalar, null	Absolute path to an alternative *RUN_server* file, which overwrites the default *$SYBASE/$SYBASE_ASE/install/RUN_SERVER*.
Thorough_probe_cycle	int, scalar, 3	The number of "shallow" probes before a thorough probe is performed.

Property	Datatype, dimension, and default	Description
Thorough_probe_script	string, scalar, null	Absolute path to an alternative file containing SQL scripts for the fault monitoring program to perform a thorough probe. If it is set to null, the agent uses the default SQL commands. For security reasons, this file should only restrict write access to owner of *$SYBASE* directory). **Note** This value is ignored by the *HAase* resource.
Debug	Boolean, scalar, 0	If set to 1 (true), the monitor logs debugging messages to *$VCS_LOG/log/HAase_A.log*; other scripts log debugging messages to *$VCS_LOG/log/engine_A.log*. The message number range is 2,000,001 and greater.
Log_max_size	int, scalar, 5000000	Maximum size for the *$VCS_LOG/log/HAase_A.log* file.
Failback_strategy	string, scalar, null	Reserved for future use.
HA_config	boolean, scalar, 0	Reserved for future use.
Cmpstate	boolean, scalar, 0	Reserved for future use.

Note The default value for *$VCS_LOG* is */var/VRTSvcs*.

The following table shows a sample configuration of an *HAase* instance:

Attribute	Value
Sybase_home	*/release/rel125*
Dataserver_name	Money1
Backup_server_name	
Monserver_name	
Textserver_name	
Secondary_companion_name	
Dataserver_login_file	*/release/rel125/ASE-12_5/install/MONEY1_login*

Attribute	Value
RUN_server_file	*/release/rel125/ASE-12_5/install/RUN_MONEY1*
Thorough_probe_cycle	3
Thorough_probe_script	
Debug	
Log_max_size	5000000
Failback_strategy	
HA_config	0
Cmpstate	0

Configuring an instance of HAase resource for each service group

You can configure an instance of the *HAase* resource by either:

- Using the cluster GUI tool to configure an instance of *HAase*. (see your VCS User Guide for more information), or,
- Using cluster commands to manually add a new resource and configure its attributes. This is described below.

  The following are the cluster commands used to add resource *syb_ase125* that uses the configuration described in the table above. (The configuration of service group *SybASE* is shown in Figure 13-1 on page 223):

  - Add the *HAase* resource:

```
#hares -add syb_ase125 HAase SybASE
VCS:10245:Resource added
NameRule and Enabled attributes must be set before agent monitors
# hares -modify syb_ase125 Dataserver_name MONEY1
# hares -modify syb_ase125 RUN_server_file  /release/rel125/ASE-
12_5/install/RUN_MONEY1
# hares -modify syb_ase125 Log_max_size 5000000
# hares -modify syb_ase125 Dataserver_login_file /release/rel125/ASE-
12_5/install/MONEY1_login
# hares -modify syb_ase125 Sybase_home  /release/rel125
# hares -modify syb_ase125 Thorough_probe_cycle 3
```

  - Configure the agent to monitor the status of resource *syb_ase125*:

```
# hares -modify syb_ase125 Enabled 1
```

Note After you add the new resource to service group, you must establish the resource dependency between the *HAase* resource and other storage and network access resources access.

Use the following cluster commands to establish a resource dependency between *syb_ase125* and resources of types *Mount*, *Volume*, and *IP* (refer to Figure 13-1 for more details):

```
# hares -link syb_ase125 ha1_mnt_ase
# hares -link syb_ase125 ha1_mnt_fsdb
# hares -link syb_ase125 vrtsdg1_vol_master
# hares -link syb_ase125 vrtsdg1_vol_procs
#hares -link syb_ase125 ha1_ip
```

Configuring companion servers for Failover

Perform the tasks in this section to configure the Adaptive Servers as primary and secondary companions in a high availability system.

Adding user and login for HA monitor

Create a special user and login for the monitor for each data server associated with the SY.ase resource. Use isql to connect to the data servers and issue:

```
sp_addlogin probe_ase, password

sp_adduser probe_ase
```

For example:

```
sp_addlogin joe, joe_password

sp_adduser joe
```

Note During Adaptive Server configuration, the System Administrator should take into account that the user and login used for probe actually reduces by one the total number of connections available for other purposes.

For more information about storing the monitor login information, see “Creating an Adaptive Server login file” on page 230.

Running *sp_companion* with *do_advisory*

You must configure the secondary companion with sufficient resources to perform the work of both servers during failover. The secondary companion may have attributes that will prevent a successful cluster operation. For example, if both the primary and secondary companions are configured for 250 user logins, during failover, the secondary companion only has the resources for half the number of potential user logins necessary. Instead, both MONEY1 and PERSONNEL1 should be configured for 500 user logins.

The sp_companion do_advisory option checks the configuration options on both the primary and the secondary companion to make sure a cluster operation will be successful. sp_companion do_advisory advises you of any configuration options that should be changed.

See Chapter 6, “Running do_advisory” for a complete description of the sp_companion do_advisory option.

Verifying the HA agent

Because machines running the Solaris operating system can support different cluster software, sp_companion includes the show_cluster option to query the HA agent currently running and the set_cluster option to set the HA agent.

If you are running the Veritas Cluster Server subsystem, you must specify this with sp_companion. Adaptive Server assumes it is running the Sun Cluster software unless you specify otherwise.

The syntax is:

sp_companion *companion_server_name*, [show_cluster]
sp_companion *companion_server_name*, [set_cluster ["SC-2.2" | "SC-3.0" | "VCS-Sybase" | "VCS-HAase"]]

In the following example, Adaptive Server is using the default HA agent for Sun Cluster 2.2:

```
sp_companion MONEY1, show_cluster

The default cluster is: SC-2.2.
The current cluster is set to default.
```

```
Supported cluster systems for SunOS are:
SC-2.2
VCS-Sybase
SC-3.0
VCS-HAase
```

To change the Adaptive Server to use the *HAase* agent for the Veritas Cluster:

```
sp_companion MONEY1, set_cluster, "VCS-HAase"

The current cluster is set to VCS-HAase
```

The Adaptive Server now uses the *HAase* agent for the VCS subsystem.

Note You must not change to another HA agent type when Adaptive Server is configured for normal companion mode on your VCS system

Configuring for asymmetric configuration

Configure the primary companion for asymmetric configuration. From the secondary companion, issue:

```
sp_companion "primary_server_name", configure, with_proxydb, login_name,
password
```

Where:

- *primary_server_name* is the name of the primary Adaptive Server as defined in the *interfaces file* entry and in sysservers.
- *with_proxydb* indicates that proxy databases are created on the secondary companion for all databases other than system databases. Any subsequent databases that are added also create proxy databases.
- *login_name* is the name of the user performing this cluster operation (they must have the ha_role).
- *password* is the password of the person performing this cluster operation.

This example configures an Adaptive Server named MONEY1 as a primary companion (issue the command from the secondary companion, PERSONNEL1):

```
sp_companion "MONEY1", configure, null, "Think2Odd", "password"
Server 'PERSONNEL1' is alive and cluster configured.
Step: Access verified from Server:'PERSONNEL1' to Server:'MONEY1'.
Server 'MONEY1' is alive and cluster configured.
```

```
Step: Access verified from Server:'MONEY1' to Server:'PERSONNEL1'.
(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)
...
(1 row affected)
(1 row affected)
(1 row affected)
Step: Companion server's configuration check succeeded.
Step: Server handshake succeeded.
Step: Master device accessible from companion.
Step: Added the servers 'PERSONNEL1' and 'MONEY1' for cluster configuration.
Step: Server configuration initialization succeeded.
Step: Synchronizing Application Specific information from companion server
Step: Synchronizing Roles from companion server
Step: Synchronizing Login Roles from companion server
Step: Synchronizing Remote Logins from companion server
Step: Synchronizing Groups in sysusers from companion server
Step: Synchronizing Sysattributes from companion server
Step: Synchronizing server logins from companion server
Step: Synchronizing server-wide privs from companion server
Step: User information synchronization succeeded.
Step: Server configured in normal companion mode.
```

If user databases already exist while you are using sp_companion, you see messages similar to these:

```
Step: Created proxy database 'pubs2'
 Step: Proxy status for database has been set. Please Checkpoint the database
'pubs2'
 Step: Server configured in normal companion mode"
 Starting companion watch thread
```

See Chapter 3, "Asymmetric and Symmetric Setup" for more information about asymmetric configuration.

Note The *login_name* and *password* in the above sp_companion configure command cannot be null. After you successfully execute sp_companion configure, the operating system creates a new file, */etc/VRTSvcs/conf/config/ha_companion.remote_server_name*. Make sure this file has read and write access only for the user who runs the server, otherwise security may be compromised.

Configuring for symmetric configuration

After you configure your companions for asymmetric failover, you can configure them for symmetric configuration. In a symmetric configuration, both servers act as primary and secondary companions. See Figure 3-2 on page 22 for a description of symmetric configuration.

Issue sp_companion from the primary companion to set it up for symmetric configuration. Use the same syntax as for asymmetric configuration. See "Configuring for asymmetric configuration" on page 237, above, for a description of the syntax for sp_companion.

The following example adds an Adaptive Server named MONEY1 as the secondary companion to the Adaptive Server named PERSONNEL1 described in "Configuring for asymmetric configuration" on page 237 (issue this command from primary companion MONEY1):

```
sp_companion 'PERSONNEL1', configure, with_proxydb, null, sa, Think2Odd
Server 'MONEY1' is alive and cluster configured.
Step: Access verified from Server:'MONEY1' to Server:'PERSONNEL1'
Server 'PERSONNEL1' is alive and cluster configured.
Step: Access verified from Server:'PERSONNEL1' to Server:'MONEY1'
(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)
.....
Step: Companion servers configuration check succeeded
Step: Server handshake succeeded
Step: Master device accessible from companion
Step: Added the servers 'MONEY1' and 'PERSONNEL1' for cluster config
Step: Server configuration initialization succeeded
Step: Synchronizing server logins from companion server
Step: Synchronizing remoteserver from companion server
Step: Synchronizing roles from companion server
Step: Synchronizing server-wide privs from companion server
Step: User information syncup succeeded
```

```
Step: Server configured in normal companion mode
```

Note The *login_name* and *password* in the above sp_companion configure command cannot be null. After you successfully execute sp_companion configure, the operating system creates a new file, */etc/VRTSvcs/conf/config/ha_companion.remote_server_name*. Make sure this file has read and write access only for the user who runs the server, otherwise security may be compromised.

Administering Sybase Failover

This section includes information about using Sybase Failover.

During failover

When the primary node fails over to the secondary node, the service group that is online on the primary node is switched to the secondary node. At this point, all the resources except the Adaptive Server binary are online on the secondary node. The Adaptive Server on the secondary node takes over these resources.

Note When one service group fails over from the primary host to the secondary host, the Adaptive Server on the secondary host takes over all its resources, but the Adaptive Server on the failed-over group is not started.

Failing back to the primary companion

Failback switches the service group that originally belonged to the primary node from the secondary node back to the primary node and brings it online.

To initiate failback:

- After your primary node is ready to take back the service group, issue the following from the secondary companion:

  ```
  sp_companion primary_companion_name,
  prepare_failback
  ```

where *primary_companion_name* is the name of primary companion. This command switches the primary node's service group from secondary node back to primary node. For example, to fail back the primary companion MONEY1, issue this command from the secondary companion PERSONNEL1:

```
sp_companion "MONEY1", prepare_failback
Step: Primary databases are shutdown in secondary.
Step: Primary databases dropped from current secondary.
Step: Primary devices released from current secondary.
Step: Prepare failback for primary server completed successfully.
```

- Make sure the primary nodes service group is successfully switched to primary node by issuing this command from the command line:

  ```
  hastatus -group service_group_name
  ```

  This command displays the status of the primary nodes service group.

- To resume normal companion mode, issue the following from the primary companion:

  ```
  sp_companion secondary_companion_name, resume
  ```

  where *secondary_companion_name* is the name of the secondary companion server. For example, to resume normal companion mode for primary companion MONEY1:

  ```
  sp_companion "PERSONNEL1", resume
  ```

Note You cannot connect clients with the failover property (for example isql -Q) to Adaptive Server until you issue sp_companion resume. If you do try to reconnect them after issuing sp_companion prepare_failback, the client hangs until you issue sp_companion resume.

Suspending normal companion mode

Suspended mode temporarily disables the ability of the primary companion to fail over to the secondary companion. To switch from normal companion mode to suspended mode:

1 As root, use hares to change the attribute *Critical* for the *Sybase* resource on primary node to 0. The syntax is:

```
hares -modify name_of_Sybase_resource Critical 0
```

For example, to modify the attribute *Critical* of the *Sybase* resource, *syb_ase125* for primary companion, MONEY1:

```
hares -modify syb_ase125 Critical 0
```

(See the *Veritas Cluster Server User's Guide* for more information about the hares command.)

2 Suspend normal companion mode. From the secondary companion, issue:

```
sp_companion companion_name, suspend
```

For example, to suspend primary companion MONEY1 for maintenance, connect to secondary companion PERSONNEL1 and issue:

```
sp_companion MONEY1, suspend
```

Resuming normal companion mode

To move from suspended mode to normal companion mode:

1 Make sure both companions are running. As root, issue:

```
hastatus
```

2 Change the *Critical* attribute of the *Sybase* resource on the primary node to 1. As root, issue:

```
hares -modify name_of_Sybase_resource Critical 1
```

For example, to modify the *Critical* attribute of the *Sybase syb_ase125* resource for primary companion MONEY1:

```
hares -modify syb_ase125 Critical 1
```

3 Resume normal companion mode. From the secondary companion, issue:

```
sp_companion primary_companion_name, resume
```

For example, to resume normal companion mode for primary companion MONEY1:

```
sp_companion MONEY1, resume
```

Note You cannot connect clients with the failover property (for example isql -Q) until you issue sp_companion resume. If you do try to reconnect them after issuing sp_companion prepare_failback, the client hangs until you issue sp_companion resume.

Dropping companion mode

To drop companion mode, issue:

```
sp_companion companion_name, "drop"
```

Dropping companion mode is irreversible; you must reconfigure the companion servers before they will fail over in a high availability system and retain all the functionality that Sybase Failover provides. However, the companion server is still monitored by the HA agent. Before dropping companion mode, you must first disable the agent to monitor Adaptive Server. Issue the following command:

```
hares -modify Sybase_resource_name Enabled 0
```

To drop the companion mode, issue sp_companion ... drop.

For example, to drop the companionship with primary companion MONEY1, connect to secondary companion, PERSONNEL1 and issue:

```
sp_companion "MONEY1", "drop"
```

Troubleshooting Failover for Veritas Cluster

This section includes troubleshooting information about common errors.

- Turn on the debugging for Adaptive Server. Use the trace flag 2205 to get high availability-related debugging information. The following isql session turns on the debugging and redirects its messages to the console:

  ```
  dbcc traceon(2205)
  dbcc traceon(3604)
  ```

- When your system reports errors, first check the error log. In the VCS system log, */var/VRTSvcs/log/engine_A.log*, any error message with a message ID greater than 2,000,000 is an error message from *HAase* agent.
- The VCS error logs are located in:

  /var/VRTSvcs/log/log_name.log

  Among them, the *engine_A.log* is an important source of information.

  The system error log is located in */var/log/syslog*.
- Sybase recommends that you use the following monitoring tools to find information about your system:
  - *hagui* – a GUI tool
  - *hastatus* – a command line tool.
  - The following trigger scripts, which alert you of events on the VCS system: *injeopardy*, *preonline*, *postonline*, *postoffline*, *resnotoff*, *resfault*, *sysoffline*, *violation*.
- When one service group fails over from the primary host to the secondary host, the Adaptive Server on the secondary host takes over all its resources, but the Adaptive Server on the failed-over group is not started, and VCS may indicate that the HAase resource is "faulted" on the secondary host. Use the following command on the secondary host to clear the state after failover:

  ```
  hares -clear sybase_res_name -sys
  secondary_host_name
  ```

Recovering from a failed *prepare_failback*

During a failback, if prepare_failback was executed successfully on the secondary companion but the primary companion does not boot:

1 Check the primary companion's errorlog and the cluster error log to identify why the server did not boot, and correct the problems.

2 To clear the FAULTED state of the *HAase* resource, issue:

```
hasybase clear HAase_res_name
```

3 As root, issue the following to move the primary logical host back to the secondary node:

```
hagrp -switch primary_service_group -to
```

```
secondary_host_name
```

4 Log in to the secondary companion and issue:

```
dbcc ha_admin ("", "rollback_failback")
```

Your companion servers should both be back in failover mode. For more information about dbcc ha_admin, see “dbcc Options for High Availability Systems” on page 354.

5 Reissue sp_companion...prepare_failback on the secondary companion.

Location of the logs

Use the information in these logs to debug your high availability subsystem:

- Adaptive Server error log (defined in the RUNSERVER file).
- Veritas cluster log, located in */var/VRTSvcs/log/engine_A.log*.
- Operating system messages are in */var/log/syslog*.
- *HAase* agent log, located in */var/VRTSvcs/log/HAase_A.log.*

Upgrading from an agent of resource type *Sybase*

If you are using an earlier version of the resource type *Sybase* on VCS 1.3.0, you can continue to use the HA agent from this version on an upgraded VCS version that is certified by Sybase. If you are using an HA agent from an earlier release of VCS for resource type *Sybase* and you want to use the new agent for the resource type *HAase*, perform the following steps to switch from the old to the new agent:

1 Install the new agent for resource type *HAase*. See “Installing the HAase agent” on page 229.

2 Import the new resource type, *HAase*. See “Importing the HAase resource type” on page 230.

3 Start the new agent for resource type, HAase. See “Starting the HAase agent” on page 231.

4 Disable the Sybase resource monitoring:

```
haconf -makerw
```

```
hares -modify Sybase_resource_name Enabled 0
haconf -dump -makero
```

5 Drop the existing resource instances of *Sybase* from the service group.

```
haconf -makerw
hares -delete sybase_resource_name
haconf -dump -makero
```

6 Configure a new resource instance of resource type *HAase*. See “Adding the HAase resource” on page 231“.

7 Enable the new *HAase* resource with the following cluster command:

```
hares -modify HAase_resurce_name Enabled 1
```

CHAPTER 14

Configuring Adaptive Server for Failover on Veritas, 3.5

This chapter discusses how to configure Adaptive Server for failover on Veritas Cluster Server (VCS), version 3.5, with the HA agent and resource type HAase. For information on how to configure Adaptive Server for failover on Veritas Cluster Server, version 1.3, with the old HA agent and resource type Sybase, refer to.

Make sure you read the Veritas user manuals and familiarize yourself with the Veritas cluster before you implement the steps in this chapter.

Note If you are upgrading from a previous version of Veritas, first review "Upgrading from an agent of resource type Sybase" on page 271 before performing the tasks in this chapter.

Hardware and operating system requirements

Sybase high availability requires the following hardware and system components:

- Two homogenous, network systems, preferably with similar configurations in terms of resources like CPU, memory, and so on. The two servers should be installed and configured with Solaris 2.8 and VCS version 3.5 or later. You should also install the VCS GUI (graphical user interface) to facilitate configuration and administration.
- Import the new resource type *HAase* into the VCS.
- The two systems must have access to shared multihost disks, which store the databases for the Adaptive Server configured for high availability.
- You should install Veritas Volume Manager 3.1 or later to manage disks and create resources like DiskGroup and Volume.
- Use third-party vendor mirroring for media failure protection.
- Create a service group on each system. A service group is a set of resources that provides a specific service. To provide a service for an Adaptive Server that is configured for high availability, the service group should include such resources as DiskGroup, Volume, Mount, IP, NIC, and *HAase* for Sybase Adaptive Server. A sample service group and the resource dependency graph is shown in Figure 14-1. See the *Veritas Cluster Server User's Guide* for more information on how to create a service group and how to add resources to a service group.

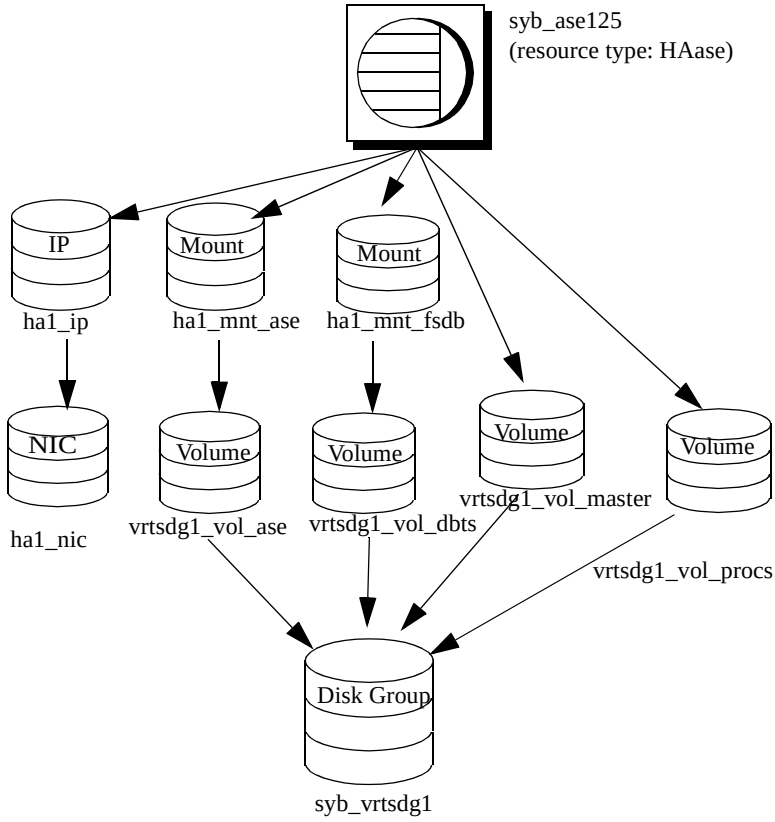
  Note each service group must contain at least two resources with one resource of type HAase. Use the cluster command to establish resource dependency so that the resource of type HAase depends on the other resources.
- Configure both public and private networks on both the nodes.

See your hardware and operating system documentation for information about installing platform-specific high availability software.

In Figure 14-1, the configuration of the service group has one DiskGroup, *syb_vrtsdg1*, on which four volumes are created. One volume is for the Adaptive Server installation, one is for databases that are created on the file system, and the other two are for databases created on raw devices. The two mount resources are created for file system of type ufs layering on the volume resources. The resource, *syb_ase125* of type *HAase* is the Adaptive Server installation, which sits on top of the mount resources. *syb_ase125* also requires resource IP, which also requires resource NIC for public network access.

The service group *SybASE* runs on the primary node and another service group, *SybASE2* (not included in Figure 14-1) runs on the secondary node, with a similar configuration:

Figure 14-1: Sample service group running on Veritas Cluster Server



Preparing Adaptive Server to work with the HA subsystem

Perform the tasks in this section to prepare Adaptive Server for a high availability configuration.

Installing Adaptive Servers

Install both the primary and the secondary servers. They can be installed either on shared or local disks. The primary companion can be either a newly installed Adaptive Server, or it can be upgraded from an earlier version of Adaptive Server with existing databases, users, and so on. The secondary companion must be a newly installed Adaptive Server and cannot have any user logins or user databases, to ensure that all user logins and database names are unique within the cluster. After configuration for failover is complete, you can add user logins and databases to the secondary companion.

If you are installing on the local disk, make sure all databases are created on the multihost disk.

See the installation documentation for your platform for information about installing and configuring Adaptive Server.

Adding entries for both Adaptive Servers to the *interfaces file*

The *interfaces file* for both primary and secondary companion must include entries for both companions. For example, the *interfaces file* for the servers used in the setups described in this manual would have entries for both MONEY1 and PERSONNEL1. The server entry in the *interfaces file* must use the same network name that is specified in sysservers. For information about adding entries to the *interfaces file*, see the installation documentation for your platform.

Adding entries to the *interfaces file* for client connections during failover

To enable clients to reconnect to the failed over companion, you must add a line to the *interfaces file*. By default, clients connect to the port listed in the query line of the server entry. If that port is not available (because that server has failed over), the client connects to the server listed in the *hafailover* line of the server entry. Here is a sample *interfaces file* for a primary companion named MONEY1 and a secondary companion named PERSONNEL1:

```
MONEY1
        master tli tcp MONEY 9678
        query tli tcp MONEY 9678
        hafailover PERSONNEL1

PERSONNEL1
        master tli tcp PERSONNEL 9679
```

```
query tli tcp PERSONNEL 9679
```

Use dsedit to add entries to the *interfaces file*. If the interfaces entries already exist, you must modify them to work for failover.

See the *Adaptive Server Enterprise Utility Guide* for information about dsedit.

The sybha executable

The *sybha* executable provides the ability for the Adaptive Server High Availability Basis Services library to interact with each platform's high availability cluster subsystem. The Adaptive Server High Availability Basis Services library calls *sybha*, which is located in *$SYBASE/ASE-12_5/bin*. Before *sybha* can run, you must change its ownership and permissions. You must also edit a file named *sybhauser* in *$SYBASE/ASE-12_5/install*. *sybhauser* contains a list of the users who have System Administrator privileges on the cluster. Sybase strongly recommends that you limit the number of users who have System Administrator privileges on the cluster.

As root, perform the following:

1 Add a new group named *sybhagrp*. You can either add this group to the */etc/group* file, or you can add it to your NIS maps. Add the sybase user to this group (this is the user that owns the *$SYBASE* directory). When the server is started, the sybase user runs the data server. If you have multiple servers running and different users owning the *$SYBASE* directory for each of them, each of these users must be added to the group

2 Change to the *$SYBASE/$SYBASE_ASE/bin* directory:

```
cd $SYBASE/$SYBASE_ASE/bin
```

3 Change the ownership of *sybha* to root:

```
chown root sybha
```

4 Change the group for the *sybha* program to *sybhagrp*:

```
chgrp sybhagrp sybha
```

5 Modify the file permissions for *sybha* to 4550:

```
chmod 4550 sybha
```

6 Change to the *$SYBASE/$SYBASE_ASE/install* directory:

```
cd $SYBASE/ASE-12_5/install
```

7 Add the sybase user to the *sybhauser* file. These logins must be in the format of UNIX login IDs, not Adaptive Server logins. For example:

```
sybase
coffeecup
spooner
venting
howe
```

8 Change the ownership of *sybhauser* to root:

```
chown root sybhauser
```

9 Modify the file permissions for *sybhauser*:

```
chmod 600 sybhauser
```

Creating a new default device

By default, master is the default device in a newly installed Adaptive Server. This means that, if you create any databases (including the proxy databases used by failover), they are automatically created on the master device. Adding user databases to the master device makes it difficult to restore the master device from a system failure. To make sure that the master device contains as few extraneous user databases as possible, create a new device using disk init. Use sp_diskdefault to specify the new device as the default before you configure Adaptive Server as a companion for failover.

For example, to add a new default device named money_default1 to the MONEY1 Adaptive Server, enter:

```
sp_diskdefault money1_default1, defaulton
```

The master device continues to also be a default device until you specifically issue the following to suspend it as the default device:

```
sp_diskdefault master, defaultoff
```

See the *Adaptive Server Reference Manual* for more information about disk init and sp_diskdefault.

Adding the local server to *sysservers*

Use sp_addserver to add the local server in sysservers using the network name specified in the *interfaces file*. For example, if the companion MONEY1 uses the network name of MONEY1 in the *interfaces file*:

```
sp_addserver MONEY1, local, MONEY1
```

You must reboot Adaptive Server for this change to take effect.

Adding secondary companion to *sysservers*

Add the secondary companion as a remote server in *sysservers*:

```
sp_addserver server_name
```

By default, Adaptive Server adds the server with a srvid of 1000. You do not need to reboot Adaptive Server for the change to take effect.

Assigning *ha_role*

You must have the ha_role on both Adaptive Servers to run sp_companion. To assign the ha_role, issue the following from isql:

```
sp_role "grant", ha_role, sa
```

You must logout and then login back in for the changes to Adaptive Server to take effect.

Installing HA stored procedures

Note You must already have added both servers to the *interfaces file* before you can install the high-availability stored procedures. If you run *installhasvss* before performing these tasks, you will have to reinstall all the system stored procedures.

The *installhasvss* script:

- Installs the stored procedures required for failover (for example, sp_companion).
- Installs the *SYB_HACMP* server in sysservers.

You must have System Administrator privileges to run *installhasvss*.

installhasvss is located in *$SYBASE/ASE-12_5/scripts*. To execute *installhasvss*, enter:

```
$SYBASE/$SYBASE_ocs/bin/isql -Usa -Ppassword -
Sservername < ../scripts/installhasvss
```

installhasvss prints messages as it creates stored procedures and creates the SYB_HACMP server.

Verifying configuration parameters

You must enable the following configuration parameters before you configure Adaptive Server for failover:

- enable CIS – enables Component Integration Services (CIS). This configuration parameter is enabled by default.
- enable xact coordination – enables Distributed Transaction Management (DTM). This configuration parameter is enabled by default.
- enable HA – enables Adaptive Server to function as a companion in a high availability system. enable HA is off by default. You must reboot Adaptive Server for it to take effect. This parameter causes a message to be written to your error log stating that you have started the Adaptive Server in a high availability system. You need to purchase the ASE_HA license option to use Adaptive Server with Failover. See the installation guide for your platform for information about enabling the ASE_HA license.

See the *System Administration Guide* for information about enabling configuration parameters.

Adding thresholds to the master log

Failing over, failing back, creating proxy databases and so on are log-intensive activities. If you do not have adequate log space, any of these activities can fail. If you have not already done so, you must add a threshold to the master log.

1. Define and execute sp_thresholdaction on the master database's log to set a threshold on the number of pages left before a dump transaction occurs. Sybase does not supply sp_thresholdaction. See the *Adaptive Server Reference Manual* for information about creating this system procedure.
2. Place thresholds on the master log segment so it does not fill up:

```
sp_addthreshold "master", "logsegment", 250, sp_thresholdaction
```

3. You must reboot the primary companion for this static parameter to take effect.

Configuring the Veritas subsystem for Sybase Failover

This section assumes that you have already installed the high availability subsystem. See the *VCS 3.5 Installation Guide* and *VCS 3.5 User's Guide* for information about installing and using the Veritas Cluster Server high availability subsystem.

Installing the HAase agent

Perform the following steps to install the *HAase* agent on each node of the cluster (you must have root permission to run the following commands):

1 Change to the *$SYBASE/$SYBASE_ASE/install/veritas/HAase* directory:

```
cd $SYBASE/$SYBASE_ASE/install/veritas/HAase
```

2 Execute the installation script:

```
perl installHAase.pl
```

The installation script:

- Copies the *HAase* resource type file *HaaseTypes.cf* to */etc/VRTSvcs/conf/config/* on local system
- Makes a new directory, */opt/VRTSvcs/bin/HAase*, if it does not already exist.
- Copies the following agent binary and scripts to */opt/VRTSvcs/bin/HAase/* on the local system:
  - *HAaseAgent*
  - *online*
  - *offline*
  - *clean*
  - *sybhautil.pm*
  - *attr_changed*

Creating an Adaptive Server login file

Create a file that contains the Adaptive Server login information for system administrator and for the user you added for the fault monitor. A sample file containing a template for this information is located in: *$SYBASE/$SYBASE_ASE/install/veritas/HAase/ase_login_file*.

This file consists of two lines. The first line is the login and password for system administrator; the second line is the monitor user login and password.

```
login-type<tab>login string
login-type<tab>login string
```

The *login-type* and the *login string* must be separated by a `tab` character.

Note If you use another file at a different location, specify the full path for the resource extension property *Dataserver_login_file* when configuring the *HAase* resource.

The default value for *login-type* is `normal`. Values for *login string* are in the form *login-name/password*. For example:

```
normal     sa/sa-password
normal     probe-user/probe-password
```

For security reasons, the *ase_login_file* should be securely protected so that `read` and `write` access permissions are restricted to root. To maintain this security, perform the following:

```
chmod 400 ase_login_file
chown root ase_login_file
chgrp  sys  ase_login_file
```

Note Sybase strongly recommends that you use a non-empty password. If you use an empty password, the agent scripts generate a warning message.

Importing the HAase resource type

There are two methods to import the HAase resource type:

- Use the cluster GUI tool to import the new resource type, *HAase*. See your VCS User Guide for more information.

- Use cluster commands hatype and haattr to manually import the new resource type from the command line. See your VCS User Guide for more information.

Starting the HAase agent

You can start the *HAase* agent by either:

- Restarting the Veritas Cluster, or
- Using the cluster commands to manually start the *HAase* agent

The second method is more desirable since it causes no disruption. The steps to manually start the *HAase* agent are:

1 Check the status of *HAase* agent with the *haagent* utility:

```
#haagent -display HAase
#Agent   Attribute  Value
HAase     AgentFile
HAase     Faults     0
HAase     Running    No
HAase      Started      No
```

2 Start the *HAase* agent on the host *myhost* with the *haagent* utility:

```
# haagent -start HAase -sys myhost
VCS:10001:Please look for messages in the log file
```

3 Check the status of *HAase* agent using the *haagent* utility:

```
# haagent -display HAase
#Agent    Attribute   Value
HAase     AgentFile
HAase     Faults      0
HAase     Running     Yes
HAase      Started       Yes
```

Adding the *HAase* resource

Each service group should contain an *HAase* resource. The following table shows the attributes of an *HAase* resource.

Property	Datatype, dimension, and default	Description
Sybase_home	string, scalar, null	The home directory of the Adaptive Server installation, and the same as the value for the environment variable SYBASE in an Adaptive Server installation.
Dataserver_name	string, scalar, null	Name of the Adaptive Server that is supplied at the time of configuration.
Backup_server_name	string, scalar, null	Name of the Backup Server that is supplied at the time of configuration.
Monserver_name	string, scalar, null	Name of the Monitor Server that is supplied at the time of configuration.
Textserver_name	string, scalar, null	Name of the full-text search server that is supplied at the time of configuration.
Secondary_companion_name	string, scalar, null	Name of secondary companion server which is set when you run the 'sp_companion configure' command.
Dataserver_login_file	string, scalar, null	Absolute path to a file containing login information for current data server. The file consists of two lines; the first line is the login and password for system administrator, the second line is the user login and password used for thorough probe used by the HA agent monitor.
RUN_server_file	string, scalar, null	Absolute path to an alternative *RUN_server* file, which overwrites the default *$SYBASE/$SYBASE_ASE/install/RUN_SERVER*.
Thorough_probe_cycle	int, scalar, 3	The number of "shallow" probes before a thorough probe is performed.

Property	Datatype, dimension, and default	Description
Thorough_probe_script	string, scalar, null	Absolute path to an alternative file containing SQL scripts for the fault monitoring program to perform a thorough probe. If it is set to null, the agent uses the default SQL commands. For security reasons, this file should only restrict write access to owner of *$SYBASE* directory). **Note** This value is ignored by the *HAase* resource.
Debug	Boolean, scalar, 0	If set to 1 (true), the monitor logs debugging messages to *$VCS_LOG/log/HAase_A.log*; other scripts log debugging messages to *$VCS_LOG/log/engine_A.log*. The message number range is 2,000,001 and greater.
Log_max_size	int, scalar, 5000000	Maximum size for the *$VCS_LOG/log/HAase_A.log* file.
Failback_strategy	string, scalar, null	Reserved for future use.
HA_config	boolean, scalar, 0	Reserved for future use.
Cmpstate	boolean, scalar, 0	Reserved for future use.

Note The default value for *$VCS_LOG* is */var/VRTSvcs*.

The following table shows a sample configuration of an *HAase* instance:

Attribute	Value
Sybase_home	*/release/rel125*
Dataserver_name	Money1
Backup_server_name	
Monserver_name	
Textserver_name	
Secondary_companion_name	
Dataserver_login_file	*/release/rel125/ASE-12_5/install/MONEY1_login*

Attribute	Value
RUN_server_file	*/release/rel125/ASE-12_5/install/RUN_MONEY1*
Thorough_probe_cycle	3
Thorough_probe_script	
Debug	
Log_max_size	5000000
Failback_strategy	
HA_config	0
Cmpstate	0

Configuring an instance of HAase resource for each service group

You can configure an instance of the *HAase* resource by either:

- Using the cluster GUI tool to configure an instance of *HAase*. (see your VCS User Guide for more information), or,
- Using cluster commands to manually add a new resource and configure its attributes. This is described below.

  The following are the cluster commands used to add resource *syb_ase125* that uses the configuration described in the table above. (The configuration of service group *SybASE* is shown in Figure 14-1 on page 249):

  - Add the *HAase* resource:

```
#hares -add syb_ase125 HAase SybASE
VCS:10245:Resource added
NameRule and Enabled attributes must be set before agent monitors
# hares -modify syb_ase125 Dataserver_name MONEY1
# hares -modify syb_ase125 RUN_server_file  /release/rel125/ASE-
12_5/install/RUN_MONEY1
# hares -modify syb_ase125 Log_max_size 5000000
# hares -modify syb_ase125 Dataserver_login_file /release/rel125/ASE-
12_5/install/MONEY1_login
# hares -modify syb_ase125 Sybase_home  /release/rel125
# hares -modify syb_ase125 Thorough_probe_cycle 3
```

  - Configure the agent to monitor the status of resource *syb_ase125*:

```
# hares -modify syb_ase125 Enabled 1
```

Note After you add the new resource to service group, you must establish the resource dependency between the *HAase* resource and other storage and network access resources access.

Use the following cluster commands to establish a resource dependency between *syb_ase125* and resources of types *Mount*, *Volume*, and *IP* (refer to Figure 14-1 for more details):

```
# hares -link syb_ase125 ha1_mnt_ase
# hares -link syb_ase125 ha1_mnt_fsdb
# hares -link syb_ase125 vrtsdg1_vol_master
# hares -link syb_ase125 vrtsdg1_vol_procs
#hares -link syb_ase125 ha1_ip
```

Configuring companion servers for Failover

Perform the tasks in this section to configure the Adaptive Servers as primary and secondary companions in a high availability system.

Adding user and login for HA monitor

Create a special user and login for the monitor for each data server associated with the HAase resource. Use isql to connect to the data servers and issue:

```
sp_addlogin probe_ase, password

sp_adduser probe_ase
```

For example:

```
sp_addlogin joe, joe_password

sp_adduser joe
```

Note During Adaptive Server configuration, the System Administrator should take into account that the user and login used for probe actually reduces by one the total number of connections available for other purposes.

For more information about storing the monitor login information, see “Creating an Adaptive Server login file” on page 256.

Running *sp_companion* with *do_advisory*

You must configure the secondary companion with sufficient resources to perform the work of both servers during failover. The secondary companion may have attributes that will prevent a successful cluster operation. For example, if both the primary and secondary companions are configured for 250 user logins, during failover, the secondary companion only has the resources for half the number of potential user logins necessary. Instead, both MONEY1 and PERSONNEL1 should be configured for 500 user logins.

The sp_companion do_advisory option checks the configuration options on both the primary and the secondary companion to make sure a cluster operation will be successful. sp_companion do_advisory advises you of any configuration options that should be changed.

See Chapter 6, “Running do_advisory” for a complete description of the sp_companion do_advisory option.

Verifying the HA agent

Because machines running the Solaris operating system can support different cluster software, sp_companion includes the show_cluster option to query the HA agent currently running and the set_cluster option to set the HA agent.

If you are running the Veritas Cluster Server subsystem, you must specify this with sp_companion. Adaptive Server assumes it is running the Sun Cluster software unless you specify otherwise.

The syntax is:

sp_companion *companion_server_name*, [show_cluster]
sp_companion *companion_server_name*, [set_cluster ["SC-2.2" | "SC-3.0" | "VCS-Sybase" | "VCS-HAase"]]

In the following example, Adaptive Server is using the default HA agent for Sun Cluster 2.2:

```
sp_companion MONEY1, show_cluster

The default cluster is: SC-2.2.
The current cluster is set to default.
```

```
Supported cluster systems for SunOS are:
SC-2.2
VCS-Sybase
SC-3.0
VCS-HAase
```

To change the Adaptive Server to use the *HAase* agent for the Veritas Cluster:

```
sp_companion MONEY1, set_cluster, "VCS-HAase"

The current cluster is set to VCS-HAase
```

The Adaptive Server now uses the *HAase* agent for the VCS subsystem.

Note You must not change to another HA agent type when Adaptive Server is configured for normal companion mode on your VCS system

Configuring for asymmetric configuration

Two Adaptive Servers are configured for asymmetric configuration. From the secondary Adaptive Server, issue:

```
sp_companion "primary_server_name", configure, with_proxydb, login_name,
password
```

Where:

- *primary_server_name* is the name of the primary Adaptive Server as defined in the *interfaces file* entry and in sysservers.
- *with_proxydb* indicates that proxy databases are created on the secondary companion for all databases other than system databases. Any subsequent databases that are added also create proxy databases.
- *login_name* is the name of the user performing this cluster operation (they must have the ha_role).
- *password* is the password of the person performing this cluster operation.

This example configures an Adaptive Server named MONEY1 as a primary companion (issue the command from the secondary companion, PERSONNEL1):

```
sp_companion "MONEY1", configure, null, "Think2Odd", "password"
Server 'PERSONNEL1' is alive and cluster configured.
Step: Access verified from Server:'PERSONNEL1' to Server:'MONEY1'.
Server 'MONEY1' is alive and cluster configured.
```

```
Step: Access verified from Server:'MONEY1' to Server:'PERSONNEL1'.
(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)
...
(1 row affected)
(1 row affected)
(1 row affected)
Step: Companion server's configuration check succeeded.
Step: Server handshake succeeded.
Step: Master device accessible from companion.
Step: Added the servers 'PERSONNEL1' and 'MONEY1' for cluster configuration.
Step: Server configuration initialization succeeded.
Step: Synchronizing Application Specific information from companion server
Step: Synchronizing Roles from companion server
Step: Synchronizing Login Roles from companion server
Step: Synchronizing Remote Logins from companion server
Step: Synchronizing Groups in sysusers from companion server
Step: Synchronizing Sysattributes from companion server
Step: Synchronizing server logins from companion server
Step: Synchronizing server-wide privs from companion server
Step: User information synchronization succeeded.
Step: Server configured in normal companion mode.
```

If user databases already exist while you are using sp_companion, you see messages similar to these:

```
Step: Created proxy database 'pubs2'
 Step: Proxy status for database has been set. Please Checkpoint the database
'pubs2'
 Step: Server configured in normal companion mode"
 Starting companion watch thread
```

See Chapter 3, "Asymmetric and Symmetric Setup" for more information about asymmetric configuration.

Note The *login_name* and *password* in the above sp_companion configure command cannot be null. After you successfully execute sp_companion configure, the operating system creates a new file, */etc/VRTSvcs/conf/config/ha_companion.remote_server_name*. Make sure this file has read and write access only for the user who runs the server, otherwise security may be compromised.

Configuring for symmetric configuration

After you configure your companions for asymmetric failover, you can configure them for symmetric configuration. In a symmetric configuration, both servers act as primary and secondary companions. See Figure 3-2 on page 22 for a description of symmetric configuration.

Issue sp_companion from the primary companion to set it up for symmetric configuration. Use the same syntax as for asymmetric configuration. See "Configuring for asymmetric configuration" on page 263, above, for a description of the syntax for sp_companion.

The following example adds an Adaptive Server named MONEY1 as the secondary companion to the Adaptive Server named PERSONNEL1 described in "Configuring for asymmetric configuration" on page 263 (issue this command from primary companion MONEY1):

```
sp_companion 'PERSONNEL1', configure, with_proxydb, null, sa, Think2Odd
Server 'MONEY1' is alive and cluster configured.
Step: Access verified from Server:'MONEY1' to Server:'PERSONNEL1'
Server 'PERSONNEL1' is alive and cluster configured.
Step: Access verified from Server:'PERSONNEL1' to Server:'MONEY1'
(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)
.....
Step: Companion servers configuration check succeeded
Step: Server handshake succeeded
Step: Master device accessible from companion
Step: Added the servers 'MONEY1' and 'PERSONNEL1' for cluster config
Step: Server configuration initialization succeeded
Step: Synchronizing server logins from companion server
Step: Synchronizing remoteserver from companion server
Step: Synchronizing roles from companion server
Step: Synchronizing server-wide privs from companion server
Step: User information syncup succeeded
```

```
Step: Server configured in normal companion mode
```

Note The *login_name* and *password* in the above sp_companion configure command cannot be null. After you successfully execute sp_companion configure, the operating system creates a new file, */etc/VRTSvcs/conf/config/ha_companion.remote_server_name*. Make sure this file has read and write access only for the user who runs the server, otherwise security may be compromised.

Administering Sybase Failover

This section includes information about using Sybase Failover.

During failover

When the primary node fails over to the secondary node, the service group that is online on the primary node is switched to the secondary node. At this point, all the resources except the Adaptive Server binary are online on the secondary node. The Adaptive Server on the secondary node takes over these resources.

Note When one service group fails over from the primary host to the secondary host, the Adaptive Server on the secondary host takes over all its resources, but the Adaptive Server on the failed-over group is not started.

Failing back to the primary companion

Failback switches the service group that originally belonged to the primary node from the secondary node back to the primary node and brings it online.

To initiate failback:

- After your primary node is ready to take back the service group, issue the following from the secondary companion:

  ```
  sp_companion primary_companion_name,
  prepare_failback
  ```

where *primary_companion_name* is the name of primary companion. This command switches the primary node's service group from secondary node back to primary node. For example, to fail back the primary companion MONEY1, issue this command from the secondary companion PERSONNEL1:

```
sp_companion "MONEY1", prepare_failback
Step: Primary databases are shutdown in secondary.
Step: Primary databases dropped from current secondary.
Step: Primary devices released from current secondary.
Step: Prepare failback for primary server completed successfully.
```

- Make sure the primary nodes service group is successfully switched to primary node by issuing this command from the command line:

  ```
  hastatus -group service_group_name
  ```

  This command displays the status of the primary nodes service group.

- To resume normal companion mode, issue the following from the primary companion:

  ```
  sp_companion secondary_companion_name, resume
  ```

  where *secondary_companion_name* is the name of the secondary companion server. For example, to resume normal companion mode for primary companion MONEY1:

  ```
  sp_companion "PERSONNEL1", resume
  ```

Note You cannot connect clients with the failover property (for example isql -Q) to Adaptive Server until you issue sp_companion resume. If you do try to reconnect them after issuing sp_companion prepare_failback, the client hangs until you issue sp_companion resume.

Suspending normal companion mode

Suspended mode temporarily disables the ability of the primary companion to fail over to the secondary companion. To switch from normal companion mode to suspended mode:

1 As root, use hares to change the attribute *Critical* for the *Sybase* resource on primary node to 0. The syntax is:

```
hares -modify name_of_Sybase_resource Critical 0
```

For example, to modify the attribute *Critical* of the *Sybase* resource, *syb_ase125* for primary companion, MONEY1:

```
hares -modify syb_ase125 Critical 0
```

(See the *Veritas Cluster Server User's Guide* for more information about the hares command.)

2 Suspend normal companion mode. From the secondary companion, issue:

```
sp_companion companion_name, suspend
```

For example, to suspend primary companion MONEY1 for maintenance, connect to secondary companion PERSONNEL1 and issue:

```
sp_companion MONEY1, suspend
```

Resuming normal companion mode

To move from suspended mode to normal companion mode:

1 Make sure both companions are running. As root, issue:

```
hastatus
```

2 Change the *Critical* attribute of the *Sybase* resource on the primary node to 1. As root, issue:

```
hares -modify name_of_Sybase_resource Critical 1
```

For example, to modify the *Critical* attribute of the *Sybase syb_ase125* resource for primary companion MONEY1:

```
hares -modify syb_ase125 Critical 1
```

3 Resume normal companion mode. From the secondary companion, issue:

```
sp_companion primary_companion_name, resume
```

For example, to resume normal companion mode for primary companion MONEY1:

```
sp_companion MONEY1, resume
```

Note You cannot connect clients with the failover property (for example isql -Q) until you issue sp_companion resume. If you do try to reconnect them after issuing sp_companion prepare_failback, the client hangs until you issue sp_companion resume.

Dropping companion mode

To drop companion mode, issue:

```
sp_companion companion_name, "drop"
```

Dropping companion mode is irreversible; you must reconfigure the companion servers before they will fail over in a high availability system and retain all the functionality that Sybase Failover provides. However, the companion server is still monitored by the HA agent. Before dropping companion mode, you must first disable the agent to monitor Adaptive Server. Issue the following command:

```
hares -modify Sybase_resource_name Enabled 0
```

To drop the companion mode, issue sp_companion ... drop.

For example, to drop the companionship with primary companion MONEY1, connect to secondary companion, PERSONNEL1 and issue:

```
sp_companion "MONEY1", "drop"
```

Troubleshooting Failover for Veritas Cluster

This section includes troubleshooting information about common errors.

- Turn on the debugging for Adaptive Server. Use the trace flag 2205 to get high availability-related debugging information. The following isql session turns on the debugging and redirects its messages to the console:

  ```
  dbcc traceon(2205)
  dbcc traceon(3604)
  ```

- When your system reports errors, first check the error log. In the VCS system log, */var/VRTSvcs/log/engine_A.log*, any error message with a message ID greater than 2,000,000 is an error message from *HAase* agent.
- The VCS error logs are located in:

  /var/VRTSvcs/log/log_name.log

  Among them, the *engine_A.log* is an important source of information.

  The system error log is located in */var/log/syslog*.
- Sybase recommends that you use the following monitoring tools to find information about your system:
  - *hagui* – a GUI tool
  - *hastatus* – a command line tool.
  - The following trigger scripts, which alert you of events on the VCS system: *injeopardy*, *preonline*, *postonline*, *postoffline*, *resnotoff*, *resfault*, *sysoffline*, *violation*.
- When one service group fails over from the primary host to the secondary host, the Adaptive Server on the secondary host takes over all its resources, but the Adaptive Server on the failed-over group is not started, and VCS may indicate that the HAase resource is "faulted" on the secondary host. Use the following command on the secondary host to clear the state after failover:

  ```
  hares -clear sybase_res_name -sys
  secondary_host_name
  ```

Recovering from a failed *prepare_failback*

During a failback, if prepare_failback was executed successfully on the secondary companion but the primary companion does not boot:

1 Check the primary companion's errorlog and the cluster error log to identify why the server did not boot, and correct the problems.

2 To clear the FAULTED state of the *HAase* resource, issue:

```
hasybase clear HAase_res_name
```

3 As root, issue the following to move the primary logical host back to the secondary node:

```
hagrp -switch primary_service_group -to
```

```
secondary_host_name
```

4 Log in to the secondary companion and issue:

```
dbcc ha_admin ("", "rollback_failback")
```

Your companion servers should both be back in failover mode. For more information about dbcc ha_admin, see “dbcc Options for High Availability Systems” on page 354.

5 Reissue sp_companion...prepare_failback on the secondary companion.

Location of the logs

Use the information in these logs to debug your high availability subsystem:

- Adaptive Server error log (defined in the RUNSERVER file).
- Veritas cluster log, located in */var/VRTSvcs/log/engine_A.log*.
- Operating system messages are in */var/log/syslog*.
- *HAase* agent log, located in */var/VRTSvcs/log/HAase_A.log.*

Upgrading from an agent of resource type *Sybase*

If you are using an earlier version of the resource type *Sybase* on VCS 3.5, you can continue to use the HA agent from this version on an upgraded VCS version that is certified by Sybase. If you are using an HA agent from an earlier release of VCS for resource type *Sybase* and you want to use the new agent for the resource type *HAase*, perform the following steps to switch from the old to the new agent:

1 Install the new agent for resource type *HAase*. See “Installing the HAase agent” on page 255.

2 Import the new resource type, *HAase*. See “Importing the HAase resource type” on page 256.

3 Start the new agent for resource type, HAase. See “Starting the HAase agent” on page 257.

4 Disable the Sybase resource monitoring:

```
haconf -makerw
```

```
hares -modify Sybase_resource_name Enabled 0
haconf -dump -makero
```

5 Drop the existing resource instances of *Sybase* from the service group.

```
haconf -makerw
hares -delete sybase_resource_name
haconf -dump -makero
```

6 Configure a new resource instance of resource type *HAase*. See “Adding the HAase resource” on page 257“.

7 Enable the new *HAase* resource with the following cluster command:

```
hares -modify HAase_resurce_name Enabled 1
```

CHAPTER 15

Configuring Adaptive Server for Failover on SGI IRIX

Perform the tasks in this chapter to configure Adaptive Server for Failover on SGI IRIX.

Configuring hardware and operating system for high availability

This document outlines the tasks you must perform to configure Adaptive Server to work in an IRIX high availability environment. For information about configuring SGI IRIX FailSafe, see the FailSafe documentation at *http://techpubs.sgi.com*.

Sybase high availability requires the following hardware and system components:

- Two homogenous, network systems. The hardware configurations need not be similar. However in a symmetric environment, each machine should have sufficient resources to accommodate the load of the other node during a failover.
- A minimum of one public network

- A minimum of one private network
- Shared disks for high availability
- Logical volumes created on the shared disks
- Optionally, file systems on the logical volumes
- Sybase Adaptive Server 12.5.0.2 or later
- Sybase high availability license
- SGI IRIX 6.5.16 or above
- SGI FailSafe 2.1.3 or above

For more information about commands used to run IRIX FailSafe, see the *IRIX FailSafe Administrator's Guide*

Preparing Adaptive Server to work with the high availability subsystem

Perform the tasks in this section to prepare Adaptive Server to work in a high availability configuration.

Planning the installation

Ensuring a smooth implementation requires proper planning. Consider the following items before starting the implementation

Starting and stopping Adaptive Server from the UNIX account

You must include the Sybase environment variables in your shell startup script (for example, the */bin/tcsh* startup script finds its environment variables in the.*tcshrc* file). Make sure that you can start and stop Adaptive Server from the shell before adding IRIX FailSafe to the Adaptive Server startup procedures.

Location of the Adaptive Server software

The Adaptive Server installations may reside on local or shared file systems. If they reside on a shared file system, they cannot have the same absolute path. For example, Adaptive Server may be installed in */node1_Sybase* on the primary node and on */node2_Sybase* on the secondary node.

Sybase recommends that each node have its own copy of Adaptive Server. If synchronization of information is required, use the command rsync to perform the synchronization. For more information about rsync, see your SGI documentation.

Logical volumes

You should use logical volumes even when they comprise one disk because this allows you to add more drives if there is disk contention without affecting the Adaptive Server administration.

Note Never allocate a disk drive to more than one Adaptive Server.

Location of the master device

The master device for each installation must reside on shared logical volumes or shared file systems.

Location of Adaptive Server's devices

Adaptive Server's devices must reside on the shared disks. Determine which raw logical volumes or file system files will be used for Adaptive Server's devices. You must determine the type of device Adaptive Server uses (file system files or raw logical volumes).

Raw logical volumes give better performance than file systems.

Using symbolic links

Use symbolic links to map raw device names to local file names when you issue the `disk init ...physname` parameter. If the logical volume name changes, stop Adaptive Server, change the volumes, recreate the symbolic link, and restart Adaptive Server.

Adaptive Server names

Determine the names for both the primary and secondary companions.

See “Selecting the server name” on page 277, below, for additional information.

Determining Adaptive Server configuration parameters

The primary and secondary Adaptive Server companions must have enough resources configured to allow for successful failover. Resources such as memory, user logins, and so on need to be considered during planning.

Make the value of *$SYBASE* the same for both companions

If *$SYBASE* is installed on the local disk, then *$SYBASE* on both companions must point to the same directory path name. This is not necessary if *$SYBASE* is installed on the shared disk. You can accomplish this by either:

- Making sure that the *$SYBASE* release directory on each companion is created in the same directory.
- If the companions have the *$SYBASE* release directory in different locations, create a directory with the same path on both companions that acts as a symbolic link to the actual *$SYBASE* release directory.

  For example, even though primary companion MONEY1 has a release directory of */usr/u/sybase1* and PERSONNEL1 has uses */usr/u/sybase2* as its release directory, their *$SYBASE* must point to the same path.

  Both MONEY1 and PERSONNEL1 have */SYBASE*, which they establish as a symbolic link to their respective *$SYBASE* release directories. On MONEY1, */SYBASE* is a link to */usr/u/sybase1*, and on PERSONNEL1, */SYBASE* is a link to */usr/u/sybase2*.

Install Adaptive Servers

Install the primary and the secondary servers. They must be installed in the same location on each node. The primary companion can be either a newly installed Adaptive Server or can be upgraded from a previous version of Adaptive Server with existing databases, users, and so on. The secondary companion must be a newly installed Adaptive Server and cannot have any user logins or user databases. This is to make sure that all user logins and database names are unique within the cluster.After configuration for failover is complete, you can add user logins and databases to the secondary companion.

If you are installing on the local disk, make sure any databases are created on the multihost disk.

See the installation documentation for your platform for information about installing and configuring Adaptive Server.

Selecting the server name

A server name is assigned when Adaptive Server is built. However, you must follow these naming requirements or failover will not work:

- 32 bit Adaptive Server requires that the server name be unique within the first four characters
- 64 bit Adaptive Server requires that the server name be unique within the first eight characters

Errorlog location

The errorlog must be located in the *$SYBASE/$SYBASE_ASE/install* directory.

HA login and password requirements

The high availability login and password must be the same value on both nodes. The login requires both the sa_role and ha_role, so you should login as system administrator.

Install Sybase licenses

Sybase high availability licenses must be installed on both servers. Check the errorlog to confirm that Adaptive Server recognizes the high availability license.

Setting up the license server to start at boot-up

Configure the Sybase License Server to start at boot-up.

Build user databases on the primary companion

Create the user databases and logins on the primary companion before installing any high availability specific software. Doing so simplifies installing and configuring Adaptive Server for high availability.

Add Entries for Both Adaptive Servers to the *Interfaces File*

The *interfaces file* for both primary and secondary companion must include entries for both companions. For example, the *interfaces file* for the servers used in the setups described in this manual would have entries for both MONEY1 and PERSONNEL1. The server entry in the *interfaces file* must use the same network name that is specified in sysservers. For information about adding entries to the *interfaces file*, see the installation documentation for your platform.

Add entries to *interfaces* file for client connections during failover

To enable clients to reconnect to the failed over companion, you must add an additional line to the *interfaces file*. By default, clients connect to the port listed in the query line of the server entry. If that port is not available (because that server has failed over), the client connects to the server listed in the *hafailover* line of the server entry. Here is a sample *interfaces file* for a primary companion named MONEY1 and a secondary companion named PERSONNEL1:

```
MONEY1
       master tcp ether MONEY1 56000
       query tcp ether MONEY1 56000
       hafailover PERSONNEL1
```

Use dsedit to add entries to the *interfaces file*. If the interfaces entries already exist, you must modify them to work for Failover.

See the the Utility Programs manual for your platform for information about dsedit.

Create new default device other than master

By default, master is the default device in a newly installed Adaptive Server. This means that, if you create any databases (including the proxy databases used by failover), they are automatically created on the master device. However, adding user databases to master makes it more difficult to restore the master device from a system failure. To make sure that the master device contains as few extraneous user databases as possible, create a new device using disk init. Use sp_diskdefault to specify the new device as the default before you configure Adaptive Server as a companion for failover.

For example, to add a new default device named money_default_1 to the MONEY1 Adaptive Server, enter:

```
sp_diskdefault money1_default1, defaulton
```

The master device continues to also be a default device until you specifically issue the following to suspend it as the default device:

```
sp_diskdefault master, defaultoff
```

See the *Adaptive Server Reference Manual* for more information about disk init and sp_diskdefault.

Add the local server to *sysservers*

Using sp_addserver, add the local server as the local server in sysservers using the network name specified in the *interfaces file*. For example, if the companion MONEY1 uses the network name of MONEY1 in the *interfaces file*:

```
sp_addserver MONEY1, local, MONEY1
```

You must reboot Adaptive Server for this change to take effect.

Add secondary companion to *sysservers*

Add the secondary companion as a remote server in *sysservers*:

```
sp_addserver server_name
```

By default, Adaptive Server adds the server with an srvid of 1000. You do not need to reboot Adaptive Server for the change to take effect.

Assign *ha_role* to system administrator

You must have the ha_role on both Adaptive Servers to run sp_companion. To assign the ha_role, issue the following from isql:

```
sp_role "grant", ha_role, sa
```

You must log out and then log back in to the Adaptive Server for the change to take effect.

Run *installhasvss* to install HA stored procedures

Note You must perform the tasks described in "Add Entries for Both Adaptive Servers to the Interfaces File" on page 278, above, before running *installhasvss*. If you run *installhasvss* before performing these tasks you will have to re-run *installmaster* to re-install all the system stored procedures.

The *installhasvss* script performs the following tasks to configure Adaptive Server for failover:

- Installs the stored procedures required for failover (for example, sp_companion).
- Installs the SYB_HACMP server in sysservers.

You must have System Administrator privileges to run the *installhasvss* script.

installhasvss in located in the *$SYBASE/ASE-12_5/scripts* directory. To execute the *installhasvss* script, enter:

```
$SYBASE/OCS-12_5/bin/isql -Usa -Ppassword -Sservername
< ../scripts/installhasvss
```

installhasvss prints messages as it creates stored procedures and creates the SYB_HACMP server.

Verify configuration parameters

You must enable the following configuration parameters before you configure Adaptive Server for failover:

- enable CIS – Enables Component Integration Services (CIS). This configuration parameter is enabled by default.
- enable xact coordination – Enables Distributed Transaction Management (DTM). This configuration parameter is enabled by default.
- enable HA – Enables Adaptive Server to function as a companion in a high availability system. enable HA is off by default. This configuration is static, so you must reboot Adaptive Server for it to take effect. This parameter causes a message to be written to your errorlog stating that you have started the Adaptive Server in a high availability system.

See the *System Administration Guide* for information about enabling configuration parameters.

Add thresholds to the master log

If you have not already done so, you must add a threshold to the master log.

1. Define and execute sp_thresholdaction on the master database's log to set a threshold on the number of pages left before a dump transaction occurs. Sybase does not supply sp_thresholdaction. See the Adaptive Server Reference Manual for information about creating this system procedure.
2. Place thresholds on the master log segment so it does not fill up:

```
sp_addthreshold "master", "logsegment", 250, sp_thresholdaction
```

3. You must reboot the primary companion for this static parameter to take effect.

Configure companion servers for failover

Perform the tasks in this section to configure the Adaptive Servers as primary and secondary companions in a high availability system.

Run *sp_companion* with *do_advisory* option

You must configure the secondary companion with sufficient resources to perform the work of both servers during failover. The secondary companion may have attributes that will prevent a successful cluster operation. For example, if both the primary and secondary companions are configured for 250 user logins, during failover, the secondary companion only has the resources for half the number of potential user logins necessary. Instead, both MONEY1 and PERSONNEL1 should be configured for 500 user logins.

The sp_companion do_advisory option checks the configuration options on the primary and the secondary companion to make sure a cluster operation (such as configuring an Adaptive Server as a secondary companion) will be successful. sp_companion do_advisory advises you of any configuration options that should be changed.

See Chapter 6, “Running do_advisory,” for a complete description of the sp_companion do_advisory option.

Configure for asymmetric configuration

Use sp_companion to configure the primary companion for asymmetric configuration:

```
sp_companion "primary_server_name", configure, with_proxydb, login_name,
password, cluster_login, cluster_password
```

Where:

- *primary_server_name* is the name of the primary Adaptive Server as defined in the *interfaces file* entry and in sysservers.
- *with_proxydb* indicates that proxy databases are created on the secondary companion for all databases other than system databases. Any subsequent databases that are added also create proxy databases.
- *login_name* is the name of the user performing this cluster operation (must have the ha_role).
- *password* is the password of the person performing this cluster operation
- *cluster_login* is the name of the user logging into this cluster.
- *password* is the password of the person logging into this cluster.

This example configures an Adaptive Server named PERSONNEL1 as a secondary companion (for more information about the syntax, see the *Adaptive Server Reference Manual* and your SGI documentation):

```
sp_companion "PERSONNEL1", configure, with_proxydb, null, sa, Odd2Think
Server 'MONEY1' is alive and cluster configured.
Step: Access verified from Server:'MONEY1' to Server:'PERSONNEL1'
Server 'PERSONNEL1' is alive and cluster configured.
Step: Access verified from Server:'PERSONNEL1' to Server:'MONEY1'
(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)
.....
Step: Companion servers configuration check succeeded
Step: Server handshake succeeded
Step: Master device accessible from companion
Step: Added the servers 'MONEY1' and 'PERSONNEL1' for cluster config
Step: Server configuration initialization succeeded
Step: Synchronizing server logins from companion server
Step: Synchronizing remoteserver from companion server
Step: Synchronizing roles from companion server
Step: Synchronizing server-wide privs from companion server
Step: User information syncup succeeded
Step: Server configured in normal companion mode
```

If user databases already exist while you are using sp_companion, you see messages similar to these:

```
Step: Created proxy database ‘pubs2’
 Step: Proxy status for database has been set. Please Checkpoint the database
'pubs2'
 Step: Server configured in normal companion mode”
 Starting companion watch thread
```

See Chapter 3, “Asymmetric and Symmetric Setup,” for more information about asymmetric configuration.

Configure for symmetric configuration

After you configure your companions for asymmetric failover, you can configure them for symmetric configuration. In a symmetric configuration, both servers act as primary and secondary companions. See Figure 3-2 on page 22 for a description of symmetric configuration.

Issue sp_companion from the secondary companion to configure it for symmetric configuration. Use the same syntax as for asymmetric configuration. See "Configure for asymmetric configuration" on page 282," above, for a description of the syntax for sp_companion.

The following example adds an Adaptive Server named MONEY1 as the secondary companion to the Adaptive Server named PERSONNEL1 described in "Configure for asymmetric configuration" on page 282:

```
sp_companion 'MONEY1', configure, with_proxydb, null, sa, Think2Odd
Server 'MONEY1' is alive and cluster configured.
Step: Access verified from Server:'MONEY1' to Server:'PERSONNEL1'
Server 'PERSONNEL1' is alive and cluster configured.
Step: Access verified from Server:'PERSONNEL1' to Server:'MONEY1'
(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)
.....
Step: Companion servers configuration check succeeded
Step: Server handshake succeeded
Step: Master device accessible from companion
Step: Added the servers 'MONEY1' and 'PERSONNEL1' for cluster config
Step: Server configuration initialization succeeded
Step: Synchronizing server logins from companion server
Step: Synchronizing remoteserver from companion server
Step: Synchronizing roles from companion server
Step: Synchronizing server-wide privs from companion server
Step: User information syncup succeeded
Step: Server configured in normal companion mode
```

Configure the IRIX failSafe with Sybase's failover

This section describes the process for configuring the IRIX FailSafe software to work with Adaptive Server. For performance reasons, SGI recommends that you use raw devices. If your environment uses raw devices, ignore the steps concerning file systems.

The examples in this section are based on a setup with a primary companion named spinnaker and a secondary companion named mizzen.

This section assumes that:

- The IRIX FailSafe environment is fully configured except for configuring the nodes. Configuring nodes starts the last phase in configuring IRIX FailSafe for production. See the *IRIX FailSafe Administrator's Guide* for more information.
- You have configured Adaptive Server for either asymmetric or symmetric mode.
- You have stopped Adaptive Server, and all devices that comprise its resource group (volumes or file systems) are dismounted.

Hardware configuration

The hardware configuration is a basic two-node IRIX FailSafe configuration:

- Two nodes
- A set of shared disks
- One public NIC on each node, connected to the Corporate LAN
- One private NIC on each node, connected directly, using a cross-over cable, to the NIC of the other node
- One serial line connected directly to the other node's system controller

The following is a sample configuration for a primary companion named spinnaker and a secondary companion named mizzen. For information about this configuration, consult your SGI documentation.

Configuration	Primary Companion	Secondary Companion
Logical Name	spinnaker	mizzen
Host Name	spinnaker	mizzen
$DSQUERY	SPINNAKER	MIZZEN
Node ID	1	2
Logical Volume	xlv1	xlv2
Mount Point	*/vol1*	*/vol2*
Public IP address	10.22.110.133	10.22.110.132
Private IP address	130.214.121.79	130.214.121.114
Public Network Accepts	Heartbeat and Control	Heartbeat and Control
Private Network Accepts	Heartbeat and Control	Heartbeat and Control
Port Type	MSC	MSC
Port Password	""	""
Logical Name of Reset Owner	mizzen	spinnaker
TTY Device	*/dev/ttyd2*	*/dev/ttyd2*
set reset parameter	Checked	Checked

Note You can configure IRIX FailSafe using either a GUI (with the fstask utility) or the command line (using the cluster_mgr utility). Examples for both methods are included for each step.

For more information about the command-line arguments, see your SGI documentation.

SGI provides unsupported, template command line files. Although these files are not supported by either SGI or Sybase, they have been used and tested within our Engineering organization. You can download these files from:

www.sybase.com/detail?id=1022631

Note In the examples below, when the command line argument is called for, it is assumed that the corresponding script has been previously been edited to reflect your environment.

The following list provides a high-level overview of the steps required to setup IRIX FailSafe. The details of each action are listed in the subsequent sections:

1 Install the FailSafe scripts

2 Start FailSafe Configuration Tool

3 Define the primary and secondary nodes

4 Define the failover policies for the primary and secondary nodes

5 Define the cluster

6 Define the logical volume resources for the primary and secondary nodes

7 Define the failsafe resources for the primary and secondary nodes

8 Define the Adaptive Server resources for the primary and secondary nodes

9 Define the resource groups for the primary and secondary nodes

10 Start the high availability Services

11 Bring the resource groups online

Install the FailSafe scripts

As super user (su), run the following to install the Adaptive Server components in the corresponding FailSafe directories and create the following scripts in the */var/cluster/ha/resource_type/SYBASE_DB* directory. First run the following to become the Sybase super user:

```
su - sybase
```

Then run:

```
su root –c $SYBASE/$SYBASE_ASE/install/inst_ha_script
```

Scripts	Description
start	Starts Adaptive Server and issues a resume if the state mandates and if so configured – see *SWITCHES* below
stop	Stops Adaptive Server and issues a prepare_failback if so configured – see *SWITCHES* below
monitor	Using isql, ensures that Adaptive Server is up and running
restart	Similar to *start*
probe	Attempts to access the resource definitions from the FailSafe database. Returns either `success` or `failure`

Scripts	Description
exclusive	Determines whether Adaptive Server is running or not
resourceAttributes	Besides retrieving and setting global environment variables for the scripts, the area where common functions live.
cmgr-create-resource-SYBASE_DB	Sample script to create the *SYBASE_DB* resource group
create_resource_type	Defines the *SYBASE_DB* resource type in the FailSafe database
SWITCHES	FailSafe and Adaptive Server switches used to alter the behavior of the scripts (that is automatic failback versus manual, logging information, and so on). Any changes to this file need to be sent to the other node(s) through rcp or rsync.
README	Detailed DBA and system administrator notes.

The software is installed locally on both the primary and secondary nodes.

Start the FailSafe configuration tool

From the GUI

Log in as root to a machine that has the FailSafe software installed. The FailSafe binary is located in the /usr/sbin directory and is named fstask. Usually one of the nodes already has the software installed.

Note Make sure the *$DISPLAY* environment variable points to your local machine.

To start the FailSafe GUI, located in */usr/sbin* directory, enter fstask at the command line.

After the FailSafe splash screen, a login window appears. Fill in the following fields:

Field	Sample value	Description
Server	mizzen	The hostname(1) of either node.

Field	Sample value	Description
Login	root	A UNIX account that has root access on the node.
Password	None	The password for the UNIX account.

Select OK to login.

From the command line

Make sure the program and */usr/cluster/bin* is part of your *PATH* environment variable.

Login as root on either node.

Enter `cluster_mgr` to begin the command line interface. Enter `quit` to end this session.

Define the primary node

From the GUI

1 Select Tasks from fstask.

2 Select Nodes from the pull-down menu. and fill in the following fields

Field	Sample values	Description
Hostname	spinnaker	The value returned by hostname(1).
Logical Name	spinnaker	The same value as Hostname.
Networks for Incoming Cluster Messages	Sample Values	Description
Network	130.215.121.79	The IP address or the hostname of the private network.
Messages to Accept	Heartbeat and Control	Set to *enabled* after serial reset cables are connected to this node.

Field	Sample values	Description
Network	130.215.121.79	The IP address or the hostname of the private network.
Messages to Accept	Heartbeat and Control	Set to *enabled* after serial reset cables are connected to this node.

3 Select Add.

Field	Sample values	Description
Network	10.22.110.133	Network address
Message to Accept	Heartbeat and Control	Accept hearbeat and control messages on the public or network address.

4 Select Add

Field	Sample value	Description
Node ID	1	(Optional) An integer in the range 1–32767 that is unique among the nodes in the pool. If you do not specify a number, FailSafe calculates an ID for you.
Partition ID	Unset	(Optional) An integer between 1–32767 that is unique among the nodes in the pool. If you do not specify a number, FailSafe calculates an ID for you.
Set Reset Parameters	Check	A special hardware port on a node that provides a way to power-cycle or reset the node remotely.

5 Select Next.

6 Select the Set Reset Parameters box. The following fields are enabled:

Field	Sample value	Description
This Node	NA	Information line
Port Type	MSC	Depending on the system controller, the values be L1, L2, MSC, or MMSC.
Port Password	Unset	The system controller password for privileged commands, not the node's root password.
Temporarily Disable Port	Unchecked	If this box is checked, FailSave cannot reset the node.

Field	Sample value	Description
Owner (node that sends reset command)	N/A	Information line
Logical Name	mizzen	Name of the node that sends the remote reset command.
TTY Device	*/dev/ttyd2*	Name of the terminal port (TTY) on the owner node to which the system controller is connected.

From the command line

From the command line, enter the following:

```
cluster_mgr –f pri_node_def
```

Define the secondary node

From the GUI

Use the same steps used to define the primary nodes, but supply information specific to the secondary node.

1 Select Tasks from fstask

2 Select Nodes from the pull-down menu. and fill in the following fields:

Field	Sample Values	Description
Hostname	mizzen	The value returned by *hostname(1)*.
Logical Name	mizzen	The same value as Hostname.
Network	130.215.121.114	The IP address or the hostname of the private network.
Messages to Accept	Heartbeat and Control	Set to *enabled* after serial reset cables are connected to this node.

3 Select Add.

Field	Sample values	Description
Network	10.22.110.133	Network address
Message to Accept	Heartbeat and Control	Set to *enabled* after serial reset cables are connected to this node.

4 Select Add

Field	Sample value	Description
Node ID	2	(Optional) An integer in the range 1– 32767 that is unique among the nodes in the pool. If you do not specify a number, FailSafe calculates an ID for you.
Partition ID	Unset	(Optional) An integer between 1–32767 that is unique among the nodes in the pool. If you do not specify a number, FailSafe calculates an ID for you.
Set Reset Parameters	Check	A special hardware port on a node that provides a way to power-cycle or reset the node remote.

5 Select Next.

6 Select the Set Reset Parameters box. The following fields are enabled:

Field	Sample value	Description
This Node	N/A	Information line
Port Type	MSC	Depending on the system controller, the values be L1, L2, MSC, or MMSC.
Port Password	Unset	The system controller password for privileged commands, not the node's root password.
Temporarily Disable Port	Unchecked	If checked, FailSafe cannot reset the node.
Owner (name of node that sends reset command)	N/A	Information line
Logical Name	spinnaker	Name of the node that sends the remote reset command.
TTY Device	*/dev/ttyd2*	Name of the terminal port (TTY) on the owner node to which the system controller is connected.

From the command line

From the command line, enter the following:

```
cluster_mgr –f sec_node_def
```

Define the failover policies for the primary node

From the GUI

1 Select Tasks from fstask

2 Select Failover Policies

3 Select Define a Failover Policy and fill in the following fields:

Field	Sample value	Description
Failover Policy	*pri_staying_afloat*	The name of the failover policy
Script	ordered	The name of an existing failover script. Two scripts are provided by FailSafe: *ordered* and *round-robin*.
Failback	Controlled_Failback	Value passed to the failover script. Either *Auto_Failback* or *Controlled_Failback* with optional attributes of *Auto_Recovery* or *InPlace_Recovery* as well as *Critical_RG* and *Node_Failures_Only*.
Recovery	In Place	The recovery attribute: Let FailSafe Choose, Automatic, or In Place.
Critical Resource Group	Unchecked	Allows monitor failure recovery to succeed even when there are resource group release failures.
Node Failures Only	Unchecked	Controls failover on resource monitoring failures.
Other Attributes	None	Determined by the user-defined failover scripts.

Field	Sample value	Description
Ordered Nodes in Failover Domains	N/A	The ordered list of nodes on which a given resource group can be allocated.
#1	spinnaker	Name of node in cluster
#2	mizzen	Name of node in cluster

From the command line

From the command line, enter the following:

```
cluster_mgr –f pri_failover_policy_def
```

Define the failover policy for the secondary node

From the GUI

1 Select Tasks.

2 Select Failover Policies.

3 Select Define a Failover Policy and fill in the following the fields:

Field	Sample value	Description
Failover Policy	*sec_staying_afloat*	The name of the failover policy.
Script	ordered	The name of an existing failover script. Two scripts are provided by FailSafe: *ordered* and *round-robin*.
Failback	Controlled_Failback	Value passed to the failover script. Either *Auto_Failback* or *Controlled_Failback* with optional attributes of *Auto_Recovery* or *InPlace_Recovery* as well as *Critical_RG*.
Recovery	In Place	The recovery attribute: Let FailSafe Choose, Automatic, or In Place.
Critical Resource Group	Unchecked	Allows monitor failure recovery to succeed even when there are resource group release failures.
Node Failures Only	Unchecked	Controls failover on resource monitoring failures.

Field	Sample value	Description
Other Attributes	None	Determined by the user-defined failover scripts.
Ordered Nodes in Failover Domains		The ordered list of nodes on which a given resource group can be allocated.
#1	mizzen	Name of node in cluster
#2	spinnaker	Name of node in cluster

From the command line

From the command line, enter the following:

```
cluster_mgr –f sec_failover_policy_def
```

Define the cluster

From the GUI

Note This task may take a few minutes to complete.

1 Select Tasks from fstask.

2 Select Cluster.

3 Select Define a Cluster and fill in the following fields.

Field	Sample values	Description
Cluster Name	afloat	The logical name of the cluster
Cluster Mode	Normal	Set to *Experimental* when debugging as it turns off resetting of nodes.
Notify Administrator	Never	Send a notification message to the administrator on cluster and node status changes; By e-mail, By other command or Never.

Note Do not add or remove nodes until the cluster icon appears in the tree view.

To add nodes to a cluster:

a Select View: Nodes in Cluster.

b Select *afloat*.

c Select Add or Remove Nodes in Cluster from Applicable Tasks.

d To add a node, select its logical name from the Available Nodes menu then select Add. The node name appears in the list titled Nodes to Go into Cluster. In the example above, the nodes are spinnaker and mizzen.

From the command line

From the command line, enter the following:

```
cluster_mgr –f cluster_def
```

Define the logical volume resource for the primary node

From the GUI

1 Select Tasks from fstask

2 Select Resources.

3 Select Define a Resource.

4 Select Next. and fill in the following fields:

Field	Sample value	Description
Resource Type	volume	The type of resource to define. The FailSafe system includes pre-defined resource types; you can define your own resource types as well
Resource	*x1v1*	The name of the resource to define. The name is the Volume as defined in *xlv(1).* See "*xlv_mgr –c 'show all_objects*" in your SGI documentation.

5 Select Next.

6 In the following screen, enter the following:

Field	Sample value	Description
Device Group	sys	The group name of the XLV device file
Device Owner	root	The user name (login name) of the owner of the XLV device file.
Device Mode	660	The device file permissions, specified in octal notation.

From the command line

From the command line, enter the following:

```
cluster_mgr –f pri_vol_resource_def
```

Define the logical volume resource for the secondary node

From the GUI

1 Select Tasks from fstask.

2 Select Resources.

3 Select Define a Resource.

4 Select Next and fill in the following fields.

Field	Sample value	Description
Resource Type	volume	The type of resource to define. The FailSafe system includes pre-defined resource types; you can define your own resource types as well.
Resource	*x1v2*	The name of the resource to define. The name is the Volume as defined in xlv(2) xlv_mgr –c 'show all_objects'.

5 Select Next and fill in the following fields.

Field	Sample value	Description
Device Group	sys	The group name of the XLV device file.
Device Owner	root	The user name (login name) of the owner of the XLV device file.
Device Mode	660	The device file permissions, specified in octal notation.

From the command line

From the command line, enter the following:

```
cluster_mgr –f sec_vol_resource_def
```

Define the file system resource for the primary

From the GUI

1 Select Resources from fstask.

2 Select Define a Resource and fill in the following fields.

Field	Sample value	Description
Resource Type	file system	The type of resource to define. The FailSafe system includes pre-defined resource types; you can define your own resource types as well.
Resource	*/vol1*	The name of the resource to define. For file system resource types, this is the mount point.

3 Select Next. Enter the type-specific attributes in the following fields:

Field	Sample value	Description
Volume Name	*x1v1*	The name of the XLV volume associated with the file system, as defined previously.
Mount Options	*rw*	The mount options to be used for mounting the file system, which are the mount options that have to be passed to the –o option of the mount (1M) command. The list of available options is provide in fstab(4).
Monitoring Level	2	The monitoring level to be used for the file system. A monitoring level of 1 specifies to check whether the file system exists in */etc/mtab*. Monitoring level 2 is a more intrusive check that is more reliable.

4 Select Tasks.

5 Select Resources.

6 Select Add/Remove Dependencies for a Resource Definition. Enter the following type-specific attributes:

Field	Sample value	Description
Resource Type	file system	Select a value from the pull-down list.
Resource	*/vol1*	The list of values is displayed for the Resource Type.
List of Dependencies:	N/A	Information line
Dependency Type	volume	Similar to Resource Type.

Field	Sample value	Description
Dependency Name	*x1v1*	Similar to Resource – the list of values is limited based on the Dependency Type.

From the command line

From the command line, enter the following:

```
cluster_mgr –f pri_fs_resource_def
```

Define the file system resource for the secondary node

From the GUI

1 Select Resources from fstask.

2 Select Define a Resource and fill in the following fields.

Field	Sample value	Description
Resource Type	file system	The type of resource to define. The FailSafe system includes pre-defined resource types; you can define your own resource types as well.
Resource	*/vol2*	The name of the resource to define. For file system resource types, this is the mount point.

3 Select Next. Enter the type-specific attributes in the following fields:

Field	Sample value	Description
Volume Name	*x1v2*	The name of the XLV volume associated with the file system, as defined previously.
Mount Options	*rw*	The mount options to be used for mounting the file system, which are the mount options that have to be passed to the –o option of the mount (1M) command. The list of available options is provide in fstab(4).
Monitoring Level	2	The monitoring level to be used for the file system. A monitoring level of 1 specifies to check whether the file system exists in */etc/mtab*. Monitoring level 2 is a more intrusive check that is more reliable.

4 Select Tasks.

5 Select Resources.

6 Select Add/Remove Dependencies for a Resource Definition. Enter the following type-specific attributes:

Field	Sample value	Description
Resource Type	file system	Select a value from the pull-down list.
Resource	*/vol2*	The list of values is displayed for the Resource Type.
List of Dependencies:	N/A	Information line
Dependency Type	volume	Similar to Resource Type.
Dependency Name	*x1v2*	Similar to Resource – the list of values is limited based on the Dependency Type.

From the command line

Enter the following:

```
cluster_mgr –f sec_fs_resource_def
```

Define the Adaptive Server resources on the primary

From the GUI

1 Select Tasks from fstask.

2 Select Resources.

3 Select Define a Resource

Field	Sample value	Description
Resource Type	*SYBASE_DB*	The type of resource to define. The FailSafe system includes pre-defined resource types; you can define your own resource types as well.
Resource	pri_ase	Name of the resource to define.

4 Select Next. Complete the following fields with the appropriate values.

Field	Sample value	UOM	Description
HA_PWD	' '	string	To enter a NULL password, use single quotes.
SYBASE1	*/usr/sybase*	string	The local Adaptive Server's *$SYBASE* value.

Field	Sample value	UOM	Description
SYBASE2	*/usr/sybase*	String	The remote Adaptive Server's *$SYBASE* value.
SYBUSER	sybase	String	The IRIX user name who owns *$SYBASE*. The value is used by IRIX FailSafe to login and start Adaptive Server.
MONITOR_INTERVAL	5	Seconds	How often you ping Adaptive Server using isql. Each ping creates a new isql session, then terminates it. Use this data in determining the setting.
SYBASE_ASE	ASE-12_5	String	The subdirectory within *$SYBASE* where the Adaptive Server exists.
SYBASE_OCS	OCS-12_5	String	The subdirectory within *$SYBASE* where the Open Client Server binaries exist; specifically, isql.
HOST1	spinnaker	String	The local *hostname(1)* value.
HOST2	mizzen	String	The remote *hostname(1)* value.
RECOVERY_TIMEOUT	90	Seconds	The number of seconds that you are willing to wait for Adaptive Server to run through recovery on all its databases. This can be a while, depending on the size of the transaction logs and how many databases are being rolled forward. If you exceed the limit specified, IRIX FailSafe issues an error.
SHUTDOWN_TIMEOUT	30	Seconds	The number of seconds you are willing to wait before erroring out.

Field	Sample value	UOM	Description
HA_LOGIN	sa	syslogins	A user with ha_role assigned who has the ability to monitor Adaptive Server and shut it down.
SYBSERVER1	SPINNAKER	String	The local Adaptive Server's *DSQUERY* value.
SYBSERVER2	MIZZEN	String	The remote Adaptive Server's DSQUERY value.

5 Select Tasks

6 Select Resources

7 Select Add/Remove Dependencies for a Resource Definition. Enter values for the following fields

Field	Sample value	Description
Resource Type	SYBASE_DB	Select a value from the pull-down list.
Resource	pri_ase	The list of values is displayed for the Resource Type.
List of Dependencies	N/A	Information line
Dependency Type	file system	Similar to a Resource Type.
Dependency Name	*/vol1*	Similar to a Resource. The list of values is limited based on the Dependency Type.

From the command line

From the command line, enter the following:

```
cluster_mgr –f pri_ase_resource_def
```

Define the Adaptive Server resources for the secondary node

From the GUI

1 Select Tasks from fstask.

2 Select Resources.

3 Select Define a Resource and fill in the following fields.

Field	Sample value	Description
Resource Type	*SYBASE_DB*	The type of resource to define. The FailSafe system includes pre-defined resource types; you can define your own resource types as well.
Resource	sec_ase	Name of the resource to define.

4 Select Next. Complete the following fields with the appropriate values.

Field	Sample value	UOM	Description
HA_PWD	‘ ‘	string	To enter a NULL password, use single quotes.
SYBASE1	*/usr/sybase*	string	The local Adaptive Server’s *$SYBASE* value.
SYBASE2	*/usr/sybase*	String	The remote Adaptive Server’s *$SYBASE* value.
SYBUSER	sybase	String	The IRIX user name who owns *$SYBASE*. The value is used by IRIX FailSafe to login and start Adaptive Server.
MONITOR_INTERVAL	5	Seconds	How often we ping Adaptive Server using isql. Each ping creates a new isql session, then terminates it at the end. Use this data in determining the setting.
SYBASE_ASE	ASE-12_5	String	The subdirectory within *$SYBASE* where the Adaptive Server binaries exist.
SYBASE_OCS	OCS-12_5	String	The subdirectory within *$SYBASE* where the Open Client Server binaries exist; specifically, isql.
HOST1	mizzen	String	The local *hostname(1)* value.
HOST2	spinnaker	String	The remote *hostname(1)* value.

Field	Sample value	UOM	Description
RECOVERY_TIMEOUT	90	Seconds	The number of seconds that you are willing to wait for Adaptive Server to run through recovery on all its databases. This can be a while depending on the size of the transaction logs and how many databases are being rolled forward. If you exceed the limit specified, IRIX FailSafe issues an error.
SHUTDOWN_TIMEOUT	30	Seconds	The number of seconds you are willing to wait before we error out.
HA_LOGIN	sa	syslogins	A user with ha_role assigned who has the capability to monitor Adaptive Server and shut it down.
SYBSERVER1	MIZZEN	String	The local Adaptive Server's *DSQUERY* value.
SYBSERVER2	SPINNAKER	String	The remote Adaptive Server's *DSQUERY* value.

5 Select Tasks.

6 Select Resources.

7 Select Add/Remove Dependencies for a Resource Definition. Enter values for the following fields:

Field	Sample value	Description
Resource Type	*SYBASE_DB*	Select a value from the pull-down list.
Resource	*sec_ase*	The list of values is displayed for the Resource Type.
List of Dependencies	N/A	Information line
Dependency Type	file system	Similar to a Resource Type.
Dependency Name	*/vol2*	Similar to a Resource; the list of values is limited based on the Dependency Type.

From the command line

From the command line, enter the following:

```
cluster_mgr –f sec_ase_resource_def
```

Define the resource groups on the primary node

From the GUI

1 Select Tasks from fstask.

2 Select Resource Groups from the pull-down menu.

3 Select Define a Resource Group.

4 Fill in the following fields:

Field	Sample values	Description
Failover Policy	*pri_staying_afloat*	The name of a previously defined failover policy.
Resource Group Name	*pri_rg*	The name of a previously defined resource group name.

5 Select OK.

6 Select Tasks.

7 Select Resource Groups from the pull-down menu.

8 Select Add/Remove Resources in Resource Group.

Field	Sample value	Description
Resource Group	*pri_rg*	Name of the resource group to be modified.
Resources		List of resources
Resource Type	volume	Type of resource to added or removed
Resource Name	*xlv1*	Name of resource to added or removed.

9 Select Add. Fill in the following fields:

Field	Sample value	Description
Resource Type	file system	Type of resource to added or removed
Resource Name	*/vol1*	Name of resource to added or removed

10 Select Add, and fill in the following fields.

Field	Sample value	Description
Resource Type	*SYBASE_DB*	Type of resource to added or removed
Resource Name	*pri_ase*	Name of resource to added or removed

11 Select Add.

From the command line

From the command line, enter the following:

```
cluster_mgr –f pri_rg_def
```

Define the resource group for the secondary node

From the GUI

1 Select Tasks from fstask.

2 Select Resource Groups from the pull-down menu.

3 Select Define a Resource Group.

4 Fill in the following fields:

Field	Sample values	Description
Failover Policy	*sec_staying_afloat*	The name of a previously defined failover policy.
Resource Group Name	*sec_rg*	The name of a previously defined resource group name.

5 Select OK.

6 Select Tasks.

7 Select Resource Groups from the pull-down menu.

8 Select Add/Remove Resources in Resource Group.

Field	Sample value	Description
Resource Group	sec_rg	Name of the resource group to be modified
Resources		List of resources
Resource Type	volume	Type of resource to added or removed
Resource Name	*xlv2*	Name of resource to added or removed

9 Select Add. Fill in the following fields:

Field	Sample value	Description
Resource Type	file system	Type of resource to added or removed
Resource Name	*/vol2*	Name of resource to added or removed

10 Select Add, and fill in the following fields:

Field	Sample value	Description
Resource Type	*SYBASE_DB*	Type of resource to added or removed
Resource Name	sec_ase	Name of resource to added or removed

11 Select Add.

From the command line

From the command line, enter the following:

```
cluster_mgr –f sec_rg_def
```

Start the high availability services

From the GUI

This task may take a few minutes to complete.

1 Select Tasks from fstask.

2 Select FailSafe HA Services on the pull-down menu.

3 Select Start FailSafe HA Services and fill in the following fields.

Field	Sample value	description
Cluster Name	afloat	Defaults to the defined cluster name.
One Node Only	Blank	If blank, starts services on all nodes.

From the command line

From the command line, enter:

```
cluster_mgr –f start_services
```

Bring the primary resource group online

From the GUI

Note If the high availability services are not yet running, the groups go into an "online state" until the services are running.

1 Select Tasks from fstask.

2 Select Resource Groups from the pull-down menu.

3 Select Bring a Resource Group Online. Enter the following data into the fields provided:

Field	Sample value	Description
Group to Bring Online	pri_rg	A pull-down of previously defined resource groups.

From the command line

Note If the high availability services are not yet running, the groups go into an "online state" until the services are running.

From the command line, enter the following:

```
cluster_mgr –f pri_online_rg
```

Bring the secondary resource group online

From the GUI

1 Select Tasks from fstask.

2 Select Resource Groups from the pull-down menu.

3 Select Bring a Resource Group Online and fill in the following:

Field	Sample value	Description
Group to Bring Online	sec_rg	A pull-down of previously defined resource groups.

From the command line

From the command line, enter the following:

```
cluster_mgr –f sec_online_rg
```

Administering the High-Availability Environment

This section includes information about administrating the High-Availability Environment.

Failing over

RIX FailSafe monitors the health of the nodes. In the event of a node failure, it migrates the resource groups to the surviving node. Sybase recommends that you read the IRIX FailSafe Administrator's Guide for more information about configuring and monitoring FailSafe. See at *http://techpubs.sgi.com* for information about the frequency of the health checks. The more you check the health of the nodes, the narrower the window between failure and restart. This must be balanced between false alerts and resource consumption.

Failing back to the primary companion

Failback moves the primary node's resources from the secondary node back to the primary node and starts the primary Adaptive Server. IRIX FailSafe and Sybase's Failover software offer two different methods to fail back:

- Automatic – Failback controlled by IRIX FailSafe.
- Manual – Failback handled by IRIX FailSafe in conjunction with the DBA issued `sp_companion` commands.

The following table describes the transition from an initial state of failover, to failback, and then to steady-state. This example uses two resources groups, *pri_rg* (the primary node) and *sec_rg* (the secondary node). The *pri_rg* is on the primary node and the *sec_rg* is on the secondary node.

Table 15-1 describes the automatic transition states.

Table 15-1: Automatic transition states

Action	Primary node	Secondary node	Primary server	Secondary server
Initial state	The node is down	The node is up and the *pri_rg* and *sec_rg* are online	Not running	Adaptive Server is running the Primary and Secondary databases.

Action	Primary node	Secondary node	Primary server	Secondary server
FailSafe Services started on the Primary Node	The node is up with no resource groups assigned	No change	No change	No change.
The DBA issues a prepare_failback on the secondary server	No change	No changes	No change	The primary server's databases and devices are released by the secondary server.
The primary resource group *pri_rg* is taken offline	No change	The primary server's resources are dismounted	No change	No change.
The primary resource group *pri_rg* is brought online	The node is up and the *pri_rg* is online	The node is up and the *sec_rg* is online	Adaptive Server is started	No change.
The DBA issues a resume on the primary companion.	No change	No changes	The user databases are brought back online.	No change.
Final State	The node is up and the *pri_rg* is online	The node is up and the *sec_rg* is online	Adaptive Server is running the primary databases	Adaptive Server is running the secondary databases.

Preform the following steps to move from the failed-over state to a normal state.

Re-Start FailSafe Services on the primary node

From the GUI

1 Select Tasks from fstask.

2 Select FailSafe HA Services from the pull-down menu.

3 Select Start FailSafe HA Services and fill in the following:

Field	Sample value	Description
Cluster Name	afloat	The name of the cluster is defaulted

Field	Sample value	Description
One Node Only	spinnaker	If you want HA services to be started on one node only, choose its name. If you leave this field blank, HA services will be started on every node in the cluster

From the command line

Enter:

```
cluster_mgr –f pri_start_services
```

Issue *prepare_failback* on the secondary companion

From an isql session connected to the secondary companion, issue:

```
sp_companion 'primary_companion_name',
'prepare_failback
```

Take the primary companion's resource group offline

From the GUI

Taking the primary resource group offline releases the secondary companion's resources.

1 Select Tasks from fstask.

2 Select Resource Groups from the pull-down menu.

3 Select Take a Resource Group Offline and fill-in the following fields:

Field	Sample value	Description
Detach Only	Unchecked	Check this box to stop monitoring the resource group. The resource group is not stopped, but FailSafe will not have any control over the group.
Detach Force	Unchecked	Same as Detach Only. In addition, FailSafe clears all errors.
Force Offline	Unchecked	Stops all resources in the group and clears all errors.
Group to Take Offline	*pri_rg*	Select the name of the resource group you want to take offline. The menu displays only resource groups that are currently online.

From the command line

Enter:

```
cluster_mgr –f pri_offline_rg
```

Bring the primary resource group online

From the GUI

This resumes the resources on the primary node.

1 Select Tasks from fstask.

2 Select Resource Groups from the pull-down menu.

3 Select Take a Resource Group Online, and fill-in the following fields:

Field	Sample value	Description
Group to Bring Online	*pri_rg*	Use the pull-down list to select the name of the resource group you want to bring online. The menu displays only resource groups that are not currently online.

From the command line

Enter:

```
cluster_mgr –f pri_online_rg
```

The table below describes the manual transition states.

Action	Primary node	Secondary node	Primary server	Secondary server
Initial state	The node is down	The node is up and the *pri_rg* and *sec_rg* are online	Not running	Adaptive Server is running the Primary and Secondary databases.
FailSafe Services started on the Primary Node	The node is up with no resource groups assigned	No change	No change	No change.
The DBA issues a prepare_failback on the secondary server	No change	No changes	No change	The primary server's databases and devices are released by the secondary server.
The primary resource group *pri_rg* is taken offline	No change	The primary server's resources are dismounted	No change	No change.
The primary resource group *pri_rg* is brought online	The node is up and the *pri_rg* is online	The node is up and the *sec_rg* is online	Adaptive Server is started	No change.

Action	Primary node	Secondary node	Primary server	Secondary server
The DBA issues a resume on the primary companion.	No change	No changes	The user databases are brought back online.	No change.
Final State	The node is up and the *pri_rg* is online	The node is up and the *sec_rg* is online	Adaptive Server is running the primary databases	Adaptive Server is running the secondary databases.

Issues *resume* on the primary Adaptive Server

Issue the following command from the isql prompt on the primary companion:

```
sp_companion 'primary_companion_name', 'resume'
```

For example:

```
sp_companion 'MIZZEN', 'resume'
```

Suspending normal companion

Suspended mode temporarily disables the ability of the primary companion to fail over to the secondary companion. To switch from normal companion mode to suspended mode requires that you no longer monitor the node and instruct Adaptive Server to suspend high availability services. The following steps describe how to switch from normal companion mode to suspended mode.

Suspend FailSafe monitoring

From the GUI

1 Select Tasks from fstask.

2 Select Resource Groups from the pull-down menu.

3 Select Suspend Monitoring a Resource Group and fill in the following field:

Field	Sample value	Description
Group to Stop Monitoring	*pri_rg*	Select the name of the group you want to stop monitoring. Only those resource groups that are currently online and monitored are displayed in the menu.

From the command line

Enter the following from the command line:

```
cluster_mgr –f pri_rg_stop_monitoring
```

Suspend secondary from primary companion

Log into the primary companion through isql and enter the following:

```
sp_companion ‘seconary_companion’, ‘suspend’
```

For example:

```
sp_companion ‘MIZZEN’, ‘suspend’
```

Resuming normal companion mode

Before you can move from suspended mode to normal mode:

- Adaptive Server must be running on both nodes.
- FailSafe monitoring needs to be resumed.
- Adaptive Server needs to resume normal companion mode.

This section assumes that Adaptive Server is already running on both nodes.

From the GUI

1 Select Tasks from fstask.

2 Select Resume Groups from the pull-down menu.

3 Select Resume Monitoring a Resource Group and fill in the following field:

Field	Sample value	Description
Group to Start Monitoring	*pri_rg*	Select the name of the group you want to start monitoring. Only those resource groups that are currently offline and not monitored are displayed in the menu.

From the command line

Enter:

```
cluster_mgr –f pri_rg_start_monitoring
```

Resume from primary companion

Enter the following from the isql prompt:

```
sp_companion 'secondary_companion', 'resume'
```

For example:

```
sp_companion 'MIZZEN', 'resume'
```

Dropping Companion Mode

Before dropping companion mode, you must either suspend IRIX FailSafe monitoring or delete the resource group, the resources, and the node from IRIX FailSafe. Sybase recommends that you suspend monitoring in case you plan on restarting companion mode at some future date. For more information, see, Suspend FailSafe Monitoring "Suspend FailSafe monitoring" on page 313. To drop companion mode, enter the following at the isql prompt from the primary companion:

```
sp_companion 'secondary_companion', 'drop'
```

For example:

```
sp_companion 'MIZZEN', 'drop'
```

Tailoring IRIX FailSafe with Sybase's Failover

As noted in "Install the FailSafe scripts" on page 287, there are several scripts that get installed under IRIX FailSafe control in order to merge Sybase HA. These scripts are korn shell scripts that you can tailor. SGI recommends that you read the README file before you start your development efforts.

The *SWITCHES* File

The *SWITCHES* file contains settings for various high-level behavior settings. Review the comments in this script for more information.

When this file is changed, it must be copied to the other node. It's not necessary to stop the resource group if it's running.

The following is a list of the switches:

Variable	Default Value	Description
AUTO_FAILBACK	1	Values: 0 or non-zero. Sets failback to either Automatic or Manual Failback.

Variable	Default Value	Description
NOWAIT	1	Possible values: zero or non-zero. When a resource group is brought offline, the Adaptive Server resource issues a shutdown. If Adaptive Server does not stop within the prescribed time and NOWAIT is non-zero, Adaptive Server issues shutdown with nowait.
DEBUG_SCRIPT_NAME	0	Possible values: zero or non-zero. FailSafe executes its scripts in a specific order. During debugging, it is helpful to see when the scripts are invoked. Setting this variable to non-zero causes the scripts to log the date and time and their name to */usr/tmp/SYBASE_DB.log*. For more information, you can issue `tail -f` on this file.
DEBUG_TRACE	0	Possible values: zero or non-zero. Setting this parameter to non-zero causes each script to dump trace information into */usr/tmp/script_name.log*. This file is always appended and can grow quite large.

Calls to Replication Server and other applications

Currently, there are no Replication Server-specific FailSafe scripts. However, the IRIX FailSafe scripts contain documented 'hook sections' where you can make the respective calls to Replication Server (or any other application).

When adding Replication, the following scripts need to be modified:

- *monitor*
- *start* and *stop* – see resourceAttributes -> start_restart_ASE() function.
- *stop*

Example hook in *start* script

The *resourceAttributes* script contains common functions. The *start* and *restart* scripts share the common start_restart_ASE() function. This function contains the following section:

#

```
# Start:  hook section -------------
#
# If you need to start additional Sybase products (such as
# replication) that are dependent on the same resources
# as ASE, add them here.
#
# End:  hook section -------------
#
```

If the same UNIX user starts Replication Server as Adaptive Server, the new code looks similar to the following:

```
#
# Start:  hook section -------------
#
# If you need to start additional Sybase products (such as
# replication) that are dependent on the same resources
# as ASE, add them here.
#
su - ${SYBUSER} –c “$SYBASE/$SYBASE_ASE/install/RUN_REP_SEVER >& /dev/null” &
#
# End:  hook section -------------
```

Troubleshooting IRIX FailSafe with Sybase Failover

This section describes some typical troubleshooting situations for Adaptive Server running on SGI IRIX in a high availability configuration.

Simulating a node failure

There are several ways to simulate a node failure to test the environment:

- Unplug the node’s power cord – This is not recommended.
- As root, issue `uadmin 1 1` – This works very quickly, however rebooting the machine takes time.
- Stop the FailSafe services on the node. This is described below.

Stopping the FailSafe service on a node

To stop FailSafe services on all nodes or a specific node:

1 Select Start from fstask

2 Select FailSafe HA Services from the pull-down menu.

3 Select Stop FailSafe HA Services and enter the following fields:

Field	Sample value	Description
Force	Uncheck	Select the check box to forcibly stop the services even if there are errors that would normally prevent them from being stopped.
Cluster Name	afloat	The name of the cluster is pre-selected.
One Node Only	spinnaker	Select the node name or leave blank for all nodes.

Stopping the FailSafe service on a node from the command line

From the command line, enter:

```
cluster_mgr –f pri_stop_services
```

Recovering from a failed manual *prepare_failback*

If you inadvertently take the failed resource group offline before issuing prepare_failback, the primary companion's devices will be removed by FailSafe. This causes an Adaptive Server error 840 on all the primary databases that have been moved to the secondary companion. The databases are marked as not recovered, offline.

This error can be corrected, but requires advanced system administration. If you're not comfortable performing these steps, contact Sybase Technical Support.

Perform the following to recover from a failed manual prepare_failback:

1 Bring the Primary Resource Group back Online. See "Bring the primary resource group online" on page 308 for more information.

2 Take the Secondary Resource Group Offline, then Online. See "Take the primary companion's resource group offline" on page 311 and "Bring the primary resource group online" on page 308 but use sec_rg instead of pri_rg for the resource group name.

3 Allow updates to the system catalogs. Log into the secondary companion using isql and enter:

```
sp_configure “allow updates”, 1
```

4 Unset the `not recovered flag`. Log into the secondary companion using isql and enter:

```
update sysdatabases set status = status & ~256
where status & 256 = 256
```

5 Disallow Updates to the System Catalogs. Log into the secondary companion using isql and enter:

```
sp_configure “allow updates”, 0
```

6 (This is a repeat of step 2) Take the secondary resource group offline, then online. See “Take the primary companion’s resource group offline” on page 311 and “Bring the primary resource group online” on page 308, but use sec_rg instead of pri_rg for the resource group name.

7 Confirm that the databases are recovered. Log into the secondary companion using isql, and enter:

```
sp_helpdb
```

Inadvertently shutting down the secondary companion

Currently, if the secondary companion is brought down when its in failback status (that is, @cmpstate = 14) and Failback is set to Automatic, the resource group cannot be brought back online by FailSafe.

To recover from this situation, temporarily set IRIX FailSafe to Manual Failback, bring the resource group online, then switch back to Automatic Failback, then you can continue with the Automatic Failback from whichever step you last left off. To recover:

1 Set IRIX FailSafe to manual failback on the secondary node. As root, enter the following at the command line:

```
cd /var/cluster/ha/resource_types/SYBASE_DB
```

2 As root, edit the *SWITCHES* file so that AUTO_FAILBACK is set to 0.

Note FailSafe does not need to be restarted for the new setting to take effect.

3 Bring the resource group online.This is the same as described in “Bring the primary resource group online” on page 308, but use sec_rg instead of pri_rg for the Resource Group name.

4 Set IRIX FailSafe to automatic failback on the secondary node. As root, move to the following directory:

```
cd /var/cluster/ha/resource_types/SYBASE_DB
```

5 As edit the *SWITCHES* file to set AUTO_FAILBACK to 1.

Location of the IRIX FailSafe logs

FailSafe maintains system logs for each of the FailSafe daemons. You can customize the system logs according to the level of logging you wish to maintain. Consult the IRIX FailSafe Administrator's Guide.

The following are the default log file names located in the */var/cluster/ha/log* directory:

File name	Description
cmsd_nodename	Log file for the FailSafe membership services daemon in node *nodename.*
gcd_nodename	Log file for group communication daemon in node *nodename.*
srmd_nodename	Log file for system resource manager daemon in node *nodename.*
failsafe_nodename	Log file for the Failsafe demon, a policy implemented for resource groups, in node *nodename.*
agent_nodename	Log file for monitoring agent named *agent* in node *nodename*. For example, *ifd_nodename* is the log file for the interface daemon monitoring agent that monitors interfaces and IP addresses and performs local failover of IP addresses.
crsd_nodename	Log file for reset daemon in node *nodename.*
script_nodename	Log file for scripts in node *nodename.*
cli_nodename	Log file for internal administrative commands in node *nodename* invoked by the GUI and `cmgr.`

Errors from resource groups running on two nodes

The following errors occur when FailSafe determines that part of the resource group is running on at least two different nodes in the cluster. This may be caused by a failed start, then a forced offline:

- `SPLIT RESOURCE` error.
- Resource groups starting on wrong nodes.
- `srmd` executable error.

A resource group that starts on the wrong node may also occur when the FailSafe Services have recently started, and they haven't quiesced. Try waiting a minute or two after starting the FailSafe Services on a node before moving a resource group.

Perform the following to clear the error:

1 Force the Resource Group Offline:

a Select Tasks.

b Select Resource Groups from the pull-down menu.

c Select Bring a Resource Group Offline and fill in the following fields:

Detach Only	**Sample value**	**Description**
Detach Only	Unchecked	Stops monitoring the resource group. The resource group will not be stopped, but FailSafe will not have any control over the group.
Detach Force	Unchecked	Same as Detach Only. In addition, FailSafe clears all errors.
Force Offline	Check	Stops all resources in the group and clear all errors.
Group to Take Offline	Unchecked	Select the name of the resource group you want to take offline. The menu displays only resource groups that are currently online.

If you are using the command line, enter the following:

```
cluster_mgr –f pri_offline_rg_force_hard
cluster_mgr –f sec_offline_rg_force_hard
```

2 Verify that no resources are still online and running on any node. Adaptive Server should not be running, and any logical volumes should be dismounted – check with the df(1) command.

3 Verify that Adaptive Server is not running on either node. If Adaptive Server is still running, determine its process id number and kill(1) it. If you have configured multiple engines, terminate them as well.

4 Make sure the volumes are still mounted on each node. Use the umount(1M) command to dismount any volumes that need to be dismounted.

5 Verify that the volumes are disassembled on each node. Perform the following:

a Make sure that volumes listed in the resource are not in the kernel's memory. Enter the following at the command line:

```
xlv_mgr –c 'show kernel'
```

b If volumes are listed which belong to the offline resource group, disassemble them. The xlv_mgr command lists the volume names which can be fed to the xlv_shutdown command. For example, xlv_mgr displays something similar to the following:

```
VOL xlv1              flags=0x1, [complete]          (node=NULL)
DATA      flags=0x0()         open_flag=0x0() device=(192, 5)
```

The volume name is *x1v1*. To shut it down, enter:

```
xlv_shutdown –n xlv1
```

c Check that the volumes have the ownership set to *none*. For example, the following shows the volumes before their ownership is set to *none*:

```
#xlv_mgr -c 'show all_objects'
#Volume:         xlv2 (complete)
#Volume:         xlv1 (complete; node=none)
#
#Vol: 2; Standalone Plex: 0; Standalone Ve: 0
```

and then after their ownership is set to *none*:

```
#xlv_mgr -c 'show all_objects'
#Volume:         xlv2 (complete; node=none)
#Volume:         xlv1 (complete; node=none)
#
# Vol: 2; Standalone Plex: 0; Standalone Ve: 0
#
```

d Run the following from the command line:

```
xlv_mgr –c ‘show all_objects’
```

e Set *xlv2*’s node name to be *none*:

```
xlv_mgr –c ‘change nodename none xlv2’
```

f Verify that all works correctly:

```
xlv_mgr –c ‘show all_objects’
```

Manually mounting the file systems

To manually start Adaptive Server on both nodes, run maintenance, correct an error, and so on, the volumes or file systems must be made available. Perform the following steps on the primary node:

1 Follow the steps outlined in “Errors from resource groups running on two nodes” on page 321.

2 Set the XLV node name. To change the node name, enter the following at the command line:

```
xlv_mgr –c ‘change nodename spinnaker xlv1’
```

Verify that the name is correct:

```
xlv_mgr –c ‘show all_objects’
```

3 Assemble the logical volumes:

```
xlv_assemble
```

4 Optionally, you can mount the file systems:

```
mount /dev/xlv/xlv1 /vol1
```

5 Clean up. After you’ve completed your work, it’s important to return the FailSafe environment to a pristine state – see “Errors from resource groups running on two nodes” on page 321. Alternatively, if all the resources that comprise the resource group have been manually brought online. You can bring the resource group online and FailSafe will pickup from there. See “Bring the primary resource group online” on page 308 as an example of starting the primary resource group; you’ll need to bring online all affected resource groups.

CHAPTER 16

Configuring Adaptive Server for Failover on Windows NT

This chapter lists the steps necessary to configure Adaptive Server for Failover on Windows NT.

It includes the following sections:

Configuring hardware and operating system for High Availability

Sybase high availability requires the following hardware and system components:

- Windows NT Enterprise Edition (with Service Pack 6 and 6a) and Microsoft Cluster Server installed on both nodes, residing on local disk storage with the same path on both nodes (for example, *C:\WINNT* and *C:\WINNT\Cluster* on both nodes).
- A Microsoft certified cluster. See your Microsoft documentation for a description of what constitutes a certified cluster.
- Adaptive Server software installed on both cluster nodes, with the Sybase release directory (*%SYBASE%*) residing on local disk storage on the nodes (rather than shared disk storage).
- Sybase data devices on shared disk drives.

- Both Adaptive Servers have an independent shared disk (or set of shared disks) for their data device storage. This area of shared disk stores all the companion database device files. The other companion cannot use this area of shared disk for any of its data devices.

Prepare Adaptive Server for HA configuration

Perform the tasks in this section to prepare Adaptive Server for a high availability configuration.

Install Adaptive Servers

Install both the primary and secondary Adaptive Servers according to the instructions in the Windows NT installation guide. Do not use the machine name as the Adaptive Server name.

The primary companion can be either a newly installed Adaptive Server, or it can be upgraded from a previous version of Adaptive Server with existing databases, users, and so on.

The secondary companion must be a newly installed Adaptive Server without any user logins or user databases. This ensures that all user logins and database names are unique within the cluster. After configuration for failover is complete, you can add user logins and databases to the secondary companion.

Place all data and log devices (including the master and sybsystemprocs devices) on dedicated shared disks, and the corresponding cluster resources must be in a dedicated MSCS group.

If you are installing on the local disk, make sure any databases are created on the shared disk.

See *Installing Adaptive Server and OmniConnect for Windows NT* for more information.

changing the domain administration account

After you install the Adaptive Servers, they run under an operating system account known as “LocalSystem”. For a regular installation of Adaptive Server, this works fine, however, for a clustered operation, the Adaptive Server must be able to communicate over the network to the other cluster node using Windows NT operating system services. Because the Local System account is not allowed to access any NT operating system services related to the network, it cannot communicate with the other node. You must reconfigure both Adaptive Servers to run under a domain administration account.

To configure Adaptive Server to run as a domain administrator:

1 Start the Services application from the Windows NT Control Panel.

2 Select the service corresponding to the Adaptive Server. Its service name uses this syntax:

Sybase SQLServer _ *server_name*

For example, Sybase SQLServer_MONEY1

3 Click Startup to display the service's startup properties dialog box.

4 Select the This Account radio button from the Log On As group.

5 Enter a valid domain administration account name (for example, *MYDOMAIN\AdminUser1*). Enter and then confirm this account’s password.

6 Click OK to save these changes.

7 Restart the Adaptive Server to use these changes.

Add entries for both Adaptive Servers to *sql.ini*

The *sql.ini* file must include entries for both companions. For example, the *sql.ini* file for the cluster described in this manual would have entries for both MONEY1 and PERSONNEL1. The server entry in the *sql.ini* file must use the same network name that is specified in *sysservers*. For information about adding entries to *sql.ini*, see *Installing Adaptive Server and OmniConnect for Windows NT*.

Add entries to sql.ini for client connections during failover

By default, clients connect to the port listed in the query line of the server entry. If that port is not available (because that server has failed over), the client connects to the server listed in the hafailover line of the server entry in sql.ini. Here is a sample *sql.ini* file for a primary companion named MONEY1 and a secondary companion named PERSONNEL1:

```
[MONEY1]
    query=TCP,FN1,9835
    master=TCP,FN1,9835
    hafailover=PERSONNEL1

[PERSONNEL1]
    query=TCP,HUM1,7586
    master=TCP,HUM1,7586
    hafailover=MONEY1
```

Use dsedit to add entries to the *sql.ini* file. If *sql.ini* entries already exist, you must modify them to work for Failover.

See *Installing Adaptive Server and OmniConnect for Windows NT* for information about dsedit.

Create new default device other than master

The master device is the default device in a newly installed Adaptive Server. This means that, if you create any databases (including the proxy databases used by Failover), they are automatically created on the master device. However, adding user databases to master makes it more difficult to restore the master device from a system failure. To make sure that the master device contains as few extraneous user databases as possible, create a new device using disk init (make sure this device is on a dedicated shared disk). Use sp_diskdefault to specify the new device as the default before you configure Adaptive Server as a companion for failover. For example, to add a new default device named *money_default1* to the MONEY1 Adaptive Server, enter:

```
sp_diskdefault money1_default1, defaulton
```

The master device continues to also be a default device until you suspend it as the default device:

```
sp_diskdefault master, defaultoff
```

See the *Adaptive Server Reference Manual* for more information about disk init and sp_diskdefault.

Add primary companion as a local server

Using sp_addserver, list the local server as the local server in *sysservers* using the network name specified in the *sql.ini* file. For example, if the companion MONEY1 uses the network name of MONEY1 in the *sql.ini* file:

```
sp_addserver MONEY1, local, MONEY1
```

You must reboot Adaptive Server for this change to take effect.

Add secondary companion to sysservers

Add the secondary companion as a remote server in sysservers:

```
sp_addserver server_name
```

By default, Adaptive Server adds the server with a srvid of 1000. You do not need to reboot Adaptive Server for the change to take effect.

Run *insthasv* to install HA stored procedures

Run the *insthasv* script on both Adaptive Servers. The *insthasv* script:

- Installs the stored procedures required for Failover (for example, sp_companion).
- Installs the SYB_HACMP server in sysservers.

  You must have System Administrator privileges to run the *insthasv* script.

  insthasv is located in the *%SYBASE%\ASE-12_5\scripts* directory. To execute *insthasv*, enter:

```
%SYBASE%\OCS-12_5\bin\isql -Usa -Ppassword -Sservername
< %SYBASE\ASE-12_5\scripts\insthasv
```

  insthasv prints messages as it creates stored procedures and creates the SYB_HACMP server.

Assign *ha_role* to system administrator

You must have the ha_role on both Adaptive Servers to run sp_companion. To assign the ha_role, issue the following from isql:

```
sp_role "grant", ha_role, user_name
```

You must log out and then log back in to the Adaptive Server for the change to take effect.

Verify configuration parameters

You must enable the following configuration parameters before you configure Adaptive Server for failover:

- enable CIS – enables Component Integration Services (CIS). This configuration parameter is enabled by default.
- enable xact coordination – enables Distributed Transaction Management (DTM). This configuration parameter is enabled by default.
- enable HA – enables Adaptive Server to function as a companion in a high availability system. enable HA is off by default. This configuration is static, so you must reboot Adaptive Server for it to take effect. This parameter causes a message to be written to your error log stating that you have started the Adaptive Server in a high availability system.

  See the *System Administration Guide* for information about enabling configuration parameters.

Run *sp_companion* with *do_advisory* option

You must configure the secondary companion with sufficient resources to perform the work of both servers during failover. For example, if both the primary and secondary companions are configured for 250 user logins, during failover, the secondary companion has only half the number of potential user logins necessary. MONEY1 and PERSONNEL1 should both be configured for 500 user logins.

The sp_companion do_advisory option checks the configuration options on both the primary and the secondary companion to make sure a cluster operation (such as configuring an Adaptive Server as a secondary companion) will be successful. sp_companion do_advisory advises you of any configuration options you should change.

Configuring Windows NT for failover

You can configure Failover on Windows NT either from the command line or using the Cluster Administrator. Configuring from the command line is described below; configuring with the Cluster Administrator is described in "Configure Windows NT for failover using cluster administrator" on page 334.

If you are configuring for a symmetric setup, you must first configure the cluster for an asymmetric setup.

Configure for asymmetric configuration from the command Line

Configure the primary companion for asymmetric configuration. From the secondary companion, enter:

```
sp_companion "primary_server_name", configure, with_proxydb, login_name,
password, cluster_login, cluster_login_password
```

Where:

- *primary_server_name* is the name of the primary Adaptive Server as defined in the *sql.ini* file entry and in sysservers.
- The *with_proxydb* indicates that proxy databases are created on the secondary companion for all databases other than system databases. Any subsequent databases that are added also create proxy databases.
- *login_name* is the name of the user performing this cluster operation (they must have both the ha_role and sa_role).
- *password* is the password of the person performing this cluster operation.
- *cluster_login* The login that the high availability subsystem uses to log in to the companion to control it. This login must exist in the primary before running sp_companion...configure and must have sa_role and ha_role.
- *cluster_login_password* is the user's password for logging in to the cluster.

Note You must execute the above command only from the secondary companion.

This example configures an Adaptive Server named MONEY1 as a primary companion. Issue the following command from the secondary server PERSONNEL1:

```
1> sp_companion "MONEY1", configure, with_proxydb, sa, MyPassword,
sa_cluster_login, MyClusterPassword
2> go
Server 'PERSONNEL1' is alive and cluster configured.
Step: Access verified from Server:'PERSONNEL1' to Server:'MONEY1'
Server 'MONEY1' is alive and cluster configured.
Step: Access verified from Server:'MONEY1' to Server:'PERSONNEL1'
(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)
.....
Step: Companion servers configuration check succeeded
Step: Server handshake succeeded
Step: Master device accessible from companion
Step: Added the servers 'PERSONNEL1' and 'MONEY1' for cluster config
Step: Server configuration initialization succeeded
Step: Synchronizing server logins from companion server
Step: Synchronizing remoteserver from companion server
Step: Synchronizing roles from companion server
Step: Synchronizing server-wide privs from companion server
Step: User information syncup succeeded
Step: Server configured in normal companion mode
```

If user databases already exist when run sp_companion, you see these messages:

```
Step: Created proxy database 'pubs2'
Step: Proxy status for database has been set. Please Checkpoint the database
'pubs2'
Step: Server configured in normal companion mode"
Starting companion watch thread
```

Before you configure the companions for symmetric configuration, you must first configure them for asymmetric configuration.

See "Asymmetric companion configuration" on page 19 for more information about asymmetric configuration.

Configure for symmetric setup from the command line

You must configure your companions for an asymmetric setup before you can configure them for a symmetric setup. In a symmetric configuration, both servers act as primary and secondary companions. See “Symmetric companion configuration” on page 21 for a description of symmetric configuration.

Issue sp_companion from the secondary companion to configure it for symmetric configuration. Use the same syntax as for asymmetric configuration.

The following example adds an Adaptive Server named MONEY1 as the secondary companion to the Adaptive Server named PERSONNEL1 described in “Configure for asymmetric configuration from the command Line” on page 331. Issue the following command from the server MONEY1:

```
1> sp_companion 'PERSONNEL1', configure, with_proxydb, sa, MyPassword,
sa_cluster_login, MyClusterPassword
2> go
Server 'MONEY1' is alive and cluster configured.
Step: Access verified from Server:'MONEY1' to Server:'PERSONNEL1'
Server 'PERSONNEL1' is alive and cluster configured.
Step: Access verified from Server:'PERSONNEL1' to Server:'MONEY1'
(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)
(1 row affected)
.....
Step: Companion servers configuration check succeeded
Step: Server handshake succeeded
Step: Master device accessible from companion
Step: Added the servers 'MONEY1' and 'PERSONNEL1' for cluster config
Step: Server configuration initialization succeeded
Step: Synchronizing server logins from companion server
Step: Synchronizing remoteserver from companion server
Step: Synchronizing roles from companion server
Step: Synchronizing server-wide privs from companion server
Step: User information syncup succeeded
Step: Server configured in normal companion mode
```

Configure Windows NT for failover using cluster administrator

The Cluster Administrator utility is a graphical user interface that walks you through the configuration process. The section assumes that the Microsoft Cluster Server is installed on your system. Install the new resource type "Sybase Companion Server" and Cluster Administrator extensions with the following command on each node of the cluster:

```
%SYBASE%\ASE-12_5\bin\sybcpnin.exe -s
```

(*%SYBASE%* is the release directory for ASE executable.)

The example shows the console output from running the above command:

```
Installed Resource Type 'Sybase Companion Server' using resource DLL
'sybcpnrs.dll' on the cluster

Installed Admin Extension DLL 'sybcpnae.dll' with MSCS and COM

Successful installation complete
SI_SUCCESS
```

Perform the following steps to configure ASE HA on Microsoft Cluster Server:

1. Create a cluster group. See you Microsoft Cluster Server documentation for information.
2. Move the dedicated shared disks for the companion you are configuring into the cluster group you created in step 1. See your Microsoft Cluster Server documentation for information.
3. Select | Administrative Tools | Cluster Administrator.
4. Select File | Resource | New Resource.
5. On the New Resource screen enter:
   - Name – the name of the package you are configuring.
   - Description – a brief description of the package. This field is not required.
   - Resource type – select Sybase Companion Server
   - Group – group in which you want this cluster included. This field is not required

Click OK to create this group.

Note You may not find the resource type "Sybase Companion Server" in the list box if the command at the beginning of this section is not run:

```
%SYBASE%\ASE-12_5\bin\sybcpnin.exe -s
```

6 Select Change Group then select the name of the group to move the physical disk resources (data and log devices) of the primary companion to this new group. Click OK. You see the Possible Owners screen.

7 The Possible Owners screen specifies the nodes on which this resource can be brought online. Both nodes must be listed as possible owners in the right-hand window of this screen. If the list is not correct, use Add or Remove to correct it. Select Next.

8 The Dependencies screen lists the services that must be brought online first before starting this resource. Make sure the shared disk device is listed as a dependency. Select Next.

Note You may only find the Finish button at this step and can not perform the following step 9 through 13 if the command at the beginning of this section is not run:

```
%SYBASE%\ASE-12_5\bin\sybcpnin.exe -s
```

9 On the ASE Server Information screen enter:

- The name of the Adaptive Server you are configuring as the primary companion.
- Thesystem administrator login for this companion (this must be sa
- The system administrator password for this login.
- A password check to make sure the password you entered is correct.

  Select Next.

10 Enter the name of the Adaptive Server that is to be the secondary companion in the Companion Server Information field.

   To configure the companions in a symmetric setup, select symmetric.

   Select Next.

11 On the Cluster Parameters screen, select Use System Generated Cluster Login. This provides a system-generated setup log that is used when the cluster logs into the Adaptive Server. Select Next. (Alternatively, you can create the login on the primary companion, assign it both the sa_role and ha_role before you perform this step.)

12 (Optional) On the Setup Options screen, enter the path to the error log that records the steps made during this configuration (this log is very helpful if you need to call Technical Support). Select Finish.

13 The next screen lists the configuration that you have selected for this cluster configuration.Select Back and re-enter the appropriate data to change any information. When the configuration is correct, select Next to configure this cluster resource.

You see a series of messages as the two Adaptive Servers are configured. If any error messages appear, address the issues and select Next. You do not have to start over again.

When the configuration is complete, the companions are in normal companion mode in either an asymmetric or symmetric setup, depending on what you specified in the Companion Server Information screen.

Configuring and securing Microsoft Cluster Server

This section describes the steps for setting the pending time-out and failback properties for the primary companion's cluster resource. If you are configuring a symmetric setup, you must set the properties for both companions.

- When the Microsoft Cluster Server (MSCS) takes the cluster resource for the primary companion on or off line, it allows for a certain amount of time to perform its processing before assuming that the operation will not complete. By default, this amount of time is 180 seconds (3 minutes). This value is known as the "pending timeout," and can be set for each resource in the MSCS cluster.

  For the Sybase Companion Server resource, the pending time-out period must be long enough to boot the Adaptive Server, run recovery on its databases, and possibly execute sp_companion resume. For companions that have large databases, it is likely that this processing will take more than 180 seconds, and you should set the pending time-out property to a higher number.

- If you are repairing or restarting the primary node after a failover, MSCS automatically fails back to the primary node as soon as the primary node comes back up unless the MSCS group containing the Sybase Companion Server resource is set to not automatically fail back.

To configure both of these properties:

1 Select Start | Administrative Tools | Cluster Administrator.

2 From the Cluster Administrator window, select Configure an Existing Resource.

3 Select Advanced from the Properties window.

4 Change the Pending Timeout property to a value that is comfortably larger than the longest time the server takes to recover, plus about 2 minutes.

5 Select Failback and make sure the Prevent Failback radio button is selected.

6 Click OK.

Check the MSCS configuration

Use the Cluster Administrator, to verify the configuration of MSCS is correct:

- There should be a new cluster resource of type “Sybase Companion Server” for each companion that can fail over. In an asymmetric setup, there is one of these resources, for a symmetric setup, there are two of these resources.

  The names of these resources are same as the names of the primary and secondary companions they are managing. For example,.if you created an asymmetric setup where PERSONNEL1 is the secondary companion for MONEY1, there should be a new cluster resource called MONEY1.

- The new cluster resources described above should all be in their own group, which is named *companion_name_GRP*, and where *companion_name* is the name of the companion server resources they contain.

- The cluster group described above should contain one for each physical cluster disk upon which the companions data devices reside.

Securing the MSCS cluster

The Sybase integration software that interfaces MSCS to Adaptive Server requires a login (with ha_role and sa_role) and password for the Adaptive Server you are configuring as a companion server. This allows the integration software to log into Adaptive Server when it needs to control it for cluster operations.

The login and its password are stored as part of the Windows NT registry Cluster Database (under *HKLM\Cluster*). This information is encrypted to prevent users from obtaining privileged login information by browsing the registry using tools like *REGEDIT.EXE* and *REGEDT32.EXE*. However, as with any reversible encryption, there is a possibility that a user could break the encryption. To address this possibility, Sybase recommends that you protect the appropriate area of the registry using a Discretionary Access Control List (DACL) that allows only administrators access to the information.

Perform the following to encrypt the cluster login and password

1 Run *REGEDT32.EXE*.

2 From the window titled HKEY_LOCAL_MACHINE on Local Machine, Double click on the Cluster folder. A subtree opens containing registry keys.

3 Select the Resources registry key.

4 Select Permissions from the Security menu. A dialog called Registry Key Permissions is displayed.

5 Select Remove from the Registry Key Permissions dialog box to remove all entries displayed except CREATOR OWNER and *machine_name\Administrators*, where *machine_name* is the local machine name. This prevents anyone except administrative users from reading this part of the registry

6 Click OK to commit the changes

Repeat this process on both cluster nodes.

Troubleshooting Sybase failover on Windows NT

This section includes troubleshooting information about common errors.

Error message 18750

If a companion server issues error message 18750, check the @@cmpstate of your servers. If your primary companion is in normal companion mode, but the secondary companion is in secondary failover mode, your cluster is in an inconsistent state, and you need to manually recover from this. This inconsistent state may be caused by an sp_companion 'prepare_failback' command failing on the secondary companion. You can determine whether this happened by examining the log on the secondary node. To recover from this, perform the following steps manually:

1 Reboot the secondary companion.

2 Repair all databases marked "suspect." To determine which databases are suspect, issue:

```
select name, status from sysdatabases
```

Databases marked suspect have a status value of 320.

3 Allow updates to system tables:

```
sp_configure "allow updates", 1
```

4 For each suspect, failed-over database, perform the following:

```
1> update sysdatabases set status=status-256 where name='database_name'
2> go
1> dbcc traceon(3604)
2> go
1> dbcc dbrecover(database_name)
2>go
```

5 From the secondary companion, issue:

```
sp_companion primary_companion_name, prepare_failback
```

For example, from secondary companion PERSONNEL1:

```
sp_companion MONEY1, prepare_failback
```

Make sure that this command executes successfully.

6 Make sure the primary companion is up and running, a then resume normal companion mode. From the primary companion, issue:

```
sp_companion secondary_companion, resume
```

For example, from the primary companion MONEY1:

```
sp_companion PERSONNEL1, resume
```

7 Make sure the Sybase Companion Server resource for the companion relationship is located on the primary node (use Move Group to move it if not) and is Offline. Then, bring the resource online using the Cluster Administrator.

Recovering from a failed *prepare_failback*

During a failback, if prepare_failback was executed successfully on the secondary companion but the primary companion fails to boot, perform the following to rollback and then reissue the prepare_failback command:

1 Check the primary companion's system event log to find the reason the server failed to boot, and correct the problems.

2 Check that the MSCS group that contains the resource for the primary server is located on the secondary node. If not, use Move Group to move it there

3 Login to the secondary companion and issue:

```
dbcc ha_admin ("", "rollback_failback")
dbcc ha_admin ("", "rollback_failover")
```

4 Verify secondary companion is in normal companion mode

5 Check that the MSCS resource for the primary server is online. If not, manually bring the resource online using the Cluster Administrator.

6 As root, start up the package for the primary companion to run on secondary node.

```
/usr/sbin/cmrunpkg -n <secondary_node> primary_companion_package_name
```

Your secondary companion is now in failover mode. Once you verify that everything is ready for the primary companion to failback to normal companion mode, you can either issue sp_companion...prepare_failback or Move Group.

APPENDIX A

Troubleshooting Second Point of Failures

This chapter discusses common problems that result from secondary point of failures with the high availability subsystem.

Troubleshooting with *dbcc ha_admin*

Sybase's Failover includes dbcc ha_admin, which addresses second point of failures. A second point of failure for a high availability system occurs when the primary companion is already in failover mode, and another point in the system fails.

See "dbcc Options for High Availability Systems" on page 354 for information about dbcc ha_admin syntax and a complete list of options.

Re-Installing *installmaster* and *installhasvss*

Perform the steps in the following sections to re-install either *installmaster* or *installhasvss*.

Re-Installing *installmaster*

After you install *installmaster* on a companion server, you should only re-run this script if the stored procedures it creates are corrupted, or if you need to install a newer version of *installmaster*. dbcc ha_admin (' ', state_machine) temporarily moves the companion to single-server mode so the *installmaster* can safely reinstall or update the stored procedures. Do not run *installmaster* without running dbcc ha_admin.

Note Because dbcc ha_admin moves the companion to single-server mode, you should only run this command when there is no concurrent activity.

Preform the following to re-install *installmaster*:

1 Run dbcc ha_admin to move the local companion server to single-server mode:

```
dbcc ha_admin (' ', 'state_machine', 'halt')
```

Where ' ' is used for a placeholder.

2 Re-run *installmaster*.

3 Run dbcc ha_admin to return the companion server to its original mode:

```
dbcc ha_admin (' ', 'state_machine', 'restart')
```

4 You *must* re-install *installhasvss* after you re-install *installmaster*. See "Re-Installing installhasvss" on page 342, for more information.

Re-Installing installhasvss

After you install *installhasvss* on a companion server, you should only re-run this script if the stored procedures it creates are corrupted, or if you need to install a newer version of *installhasvss*. dbcc ha_admin (' ', state_machine) temporarily moves the companion to single-server mode so the *installhasvss* can safely reinstall or update the stored procedures. If you attempt to run *installhasvss* without running dbcc ha_admin, the companion issues the following error message:

```
Server is not in single-server mode.
Please run dbcc ha_admin (' ', 'state_machine', 'halt') and try again
```

Note Because dbcc ha_admin moves the companion to single-server mode, you should only run this command when there is no concurrent activity.

Preform the following to re-install *installhasvss*:

1 Make a note of the srvnetname for the SYB_HACMP entry in sysservers. When it is configured for Sybase Failover, SYB_HACMP points to the companion server's svrnetname (for example, the srvnetname for the SYB_HACMP entry on companion server MONEY1 is PERSONNEL1). If the local node crashes while you are re-running *installhasvss*, this name is removed from sysservers, and you will have to replace it manually.

2 Run dbcc ha_admin to move the companion to single-server mode:

```
dbcc ha_admin (' ', 'state_machine', 'halt')
```

Where ' ' is used for a placeholder.

3 Re-run *installhasvss*. After *installhasvss* finishes, the companion server reverts to its original mode.

If the node crashes after you perform step 2, above, the srvnetname of the remote server is removed from sysservers. If this occurs, add the name of the remote server to sysservers by issuing:

```
sp_addserver SYB_HACMP, null, 'remote_server_svrnetname'
```

Run dbcc ha_admin to return the companion server to its original mode:

```
dbcc ha_admin (' ', 'state_machine', 'restart')
```

Using *dbcc ha_admin* to Address Second Point of Failures for Failover and *prepare_failback*

dbcc ha_admin includes the rollback_failover and rollback_failback options. These dbcc options should be only as a last resort, and only by System Administrators who are knowledgeable about the high availability subsystem.

These options allow you to rollback the steps performed by:

- A failover that did not complete because of either a problem with the high availability subsystem (for example, all the disks were not available during the failover, so the companion marks all the databases as suspect) or the secondary companion crashed during the failover.
- A sp_companion...prepare_failback the did not complete because of either a problem with the high availability subsystem or the primary companion failed to reboot during the failback steps.

You must preform platform specific steps before you issue either dbcc ha_admin rollback_failover or rollback_failback. See the configuration chapter for your platform for information.

Error Messages 18805, 18769, 18836

The following are common error messages you may receive:

- Error message 18805 –

```
Warning: Server '%1!' is configured for ASE HA services. The networkname
in its SYB_HACMP entry does not point to the local server. If this is due
to an earlier failed cluster command, refer to the System Administration
Guide
```

  If the local node is running in single-server mode and the srvnetname entry for SYB_HACMP is set correctly, its network name is the same as the local servers network name. This error occurs when the SYB_HACMP network name is set to another server's network name. If this occurs because of an earlier failed cluster operation, use sp_addserver to set the srvnetname of SYB_HACMP to the local servers network name. Note that during normal companion mode, the svrnetname for SYB_HACMP *always* points to the remote companion's network name, and should *never* be changed.

- Error message 18769 –

```
The HA cluster is currently in use for other cluster operations. Retry
the command later. If the problem persists, it may be due to an earlier
failed cluster command; check the System Administration Guide (Error
%1!).
```

  All cluster operations receive a cluster-wide lock and then immediately release the lock when they are done. This error occurs when you perform a cluster operation but the previous cluster operation did not release the cluster-wide lock. For information about releasing a cluster-wide lock, see "Cluster locks in a high availability node" on page 16.

- Error message 18836 –

```
Configuration operation '%1!' can not proceed due to Quorum AdvisoryCheck
failure. Please run 'do_advisory' command to find the incompatible
attribute and fix it
```

sp_companion checks a series of quorum attributes to confirm the compatibility between the companion servers. One of your companion servers has attribute settings that are not compatible. Run do_advisory for a list of the problem attributes. See Chapter 6, “Running do_advisory” for information.

APPENDIX B

Changes to Commands, System Procedures, System Databases, and New dbcc Commands, and Functions

This chapter discusses the changes to commands, system procedures, and system databases when Adaptive Server is configured for failover.

Changes to commands in asymmetric and symmetric mode

Table B-1: Changes to commands in asymmetric and symmetric mode

Command	Asymmetric setup	Symmetric setup
create role add role drop role alter role	During normal companion mode, any changes made to the primary companion with these commands are synchronized with the secondary companion server. During failover mode, the secondary companion is updated with create role, create role and alter role changes. The primary companion is updated with this information during failback mode. You cannot run drop role during failover mode. You cannot run these commands during suspended mode.	These commands have the same behavior in symmetric mode as they have in asymmetric configuration.

Command	Asymmetric setup	Symmetric setup
create database	During normal companion mode, create database creates a proxy database on the secondary companion. During failover mode, create database is not allowed to run because the primary companion's model database is not in failover mode. During failback mode, create database is allowed only under special circumstances. You cannot run create database during suspended mode.	create database has the same behavior in symmetric setup as it has in asymmetric setup.
alter database	During normal companion mode, alter database adds 2MB of space to the database.	alter database has the same behavior in symmetric setup as it has in asymmetric setup.
disk init	During normal companion mode, disk init has the same behavior as in symmetric configuration. During failover mode, the secondary server can add devices to its local set by ensuring the unique device name space. During suspended mode, disk init cannot run.	During normal companion mode, disk init ensures that the secondary companion does not already have a disk with same physical and logical name, and that the secondary companion server can access the device. disk init is not allowed to run during failover mode because it cannot verify access to the disk on the primary companion. However, disk init is allowed to perform some special duties like log expansion. During suspended mode, disk init cannot run.
disk mirror disk remirror disk unmirror	Sybase mirroring is not supported for high availability	Sybase mirroring is not supported for high availability
disk resize	disk resize does not alter the behavior of Adaptive Server running in a high availability environment. Adaptive Server assumes that the disk space allocated by the file system comes from a shared physical disk and not from a disk local to the primary server.	

Command	Asymmetric setup	Symmetric setup
drop database	During normal companion mode, drop database informs the companion server to free the database name space and may request to drop the proxy database. During failover mode, there are no restrictions on the drop database command. During suspended mode, you cannot run drop database.	This command has the same behavior in symmetric setup as is has in asymmetric setup.
grant revoke	During normal companion mode, changes to permissions from these commands are synchronized across the companion servers. During failover mode, there are no restrictions for grant. You cannot run revoke during failover mode. During suspended mode, you cannot issue either grant or revoke.	This command has the same behavior in symmetric setup as is has in asymmetric setup.
shutdown shutdown with nowait		

Changes to System Procedures in Adaptive Server Configured for Failover

Using proxy databases guarantees unique database names with the cluster, but it does not guarantee unique database IDs. The same database may have a different database ID before and after failover. Because the database IDs may change, system procedures are automatically recompiled after failover to make sure they do not use an incorrect or out-of-date database or object ID from sysprocedures.

During failover mode, Adaptive Server performs a domain check to make sure that, if there are system procedures with duplicate names in the two Adaptive Servers, the system procedure in the correct domain is run. This domain check is only performed in failover mode.

System Procedures Hold Table Lock When Modifying System Tables

System procedures cannot acquire table locks on system tables explicitly. However, in a system using Sybase's Failover, system procedures on both companions could attempt to modify the system tables at the same time.

To prevent deadlocks, if you issue a system procedure to modify a system table, the system procedure acquires a table lock on the proxy table of the system table it is modifying. That is, if you issue a system procedure to alter the syslogins system table on primary companion MONEY1, the system procedure acquires a table lock on the syslogins proxy table on the secondary companion, PERSONNEL1.

The system procedure then modifies the syslogins proxy table on PERSONNEL1, and the syslogins proxy table updates the syslogins system table on MONEY1. After the changes are committed, the table locks on the proxy syslogins system table are released.

Any other system procedures that need to make changes to the same system table are in a queue for that table. After the lock is released, they acquire the table lock.

You can set the amount of time, in seconds, system procedures wait in the queue for the locked proxy system table with the sp_configure "dtm lock timeout period" command. For more information, See the dtm lock timeout period parameter description in "Setting Configuration Parameters" in the *System Administration Guide*.

System procedures that synchronize changes between primary and secondary

Table B-2 lists the system procedures that synchronize changes between the primary companion and the secondary companion. For example, if you use sp_droplanguage to drop the French language from the primary companion, sp_droplanguage also drops it from the secondary companion.

You can issue these system procedures from any database.

Table B-2: System procedures that synchronize changes to both companions

sp_addexternlogin	sp_dropremotelogin,
sp_addlanguage	sp_drop_resource_limit
sp_addlogin	sp_dropserver
sp_addremotelogin	sp_drop_time_range
sp_add_resource_limit	sp_locklogin
sp_addserver	sp_modifylogin
sp_add_time_range	sp_modify_resource_limit
sp_defaultdb	sp_modify_time_range
sp_defaultlanguage	sp_password
sp_dropexternlogin	sp_remoteoption
sp_droplanguage	sp_serveroption
sp_droplogin	sp_setlangalias

The following system procedures synchronize changes between the primary companion and the secondary companion when you issue them from the master database.

- sp_addalias
- sp_addgroup
- sp_addtype
- sp_adduser
- sp_changegroup
- sp_dropalias
- sp_dropgroup
- sp_droptype
- sp_dropuser

Changes to System Procedures in A Failover Configuration

This section describes the system procedures whose behavior changes when Adaptive Server is configured for Failover. After Adaptive Server is configured as a companion server:

- System procedures have no change to their default functionality when they are run in single-server mode.

- You cannot run any of the system procedures listed in Table B-3 or Table B-4 during failback mode.
- The first column of Table B-3 and Table B-4 "Normal Companion Mode," describes the behavioral changes for system procedures issued from an asymmetric primary, asymmetric secondary, or symmetric companion.
- The last column of Table B-3 Table B-4 "Failover Mode," describes the behavioral changes for system procedures issued during either asymmetric secondary failover or symmetric failover.

Table B-3 lists the system procedures that change *server-wide* attributes (for example, the default language or the resource limit):

- During normal companion mode, all the system procedures listed in Table B-3 must be run from master.
- These system procedures cannot be run during asymmetric secondary suspended mode or symmetric suspended mode.
- An X indicates that the system procedure does not run in the listed mode.

Table B-3: Changes in system procedures that alter server-wide attributes

System procedure	Normal Companion Mode	Asymmetric Primary Suspended Mode	Failover Mode
sp_add_resource_limit			
sp_add_time_range			
sp_addexternlogin			
sp_addlanguage			
sp_addlogin			
sp_addremotelogin			
sp_addserver			
sp_defaultdb			
sp_defaultlanguage			
sp_drop_resource _limit	You must manually run this system procedure on the remote server as well to synchronize the companions	X	X
sp_drop_time_range		X	X
sp_dropexternlogin		X	X
sp_droplanguage		X	X
sp_droplogin		X	X
sp_dropremotelogin		X	X
sp_dropserver		X	X

System procedure	Normal Companion Mode	Asymmetric Primary Suspended Mode	Failover Mode
sp_locklogin			
sp_modify_resource _limit	You must manually run this system procedure on the remote server as well to synchronize the companions		
sp_modify_time_range			
sp_password			
sp_remotelogin			
sp_remoteoption			
sp_serveroption			
sp_setlangalias			

Table B-4 lists the system procedures that change the attributes of the database in which they are run (such as adding a user, alias, or group to the current database). You cannot run these system procedures from master during either secondary suspended or symmetric suspended mode. An X indicates that you cannot run the system procedure in the listed mode.

Table B-4: System procedures that alter database-wide attributes when they are run in master

System procedure	Normal Companion Mode	Asymmetric Primary Suspended Mode	Failover Mode	Notes
sp_addalias				
sp_addgroup				
sp_addtype				
sp_adduser				
sp_changedbowner			X	See below for additional restrictions for this system procedure.
sp_changegroup	You must manually run this system procedure on the remote server as well to synchronize the companions			
sp_dropalias		X	X	
sp_dropgroup		X	X	
sp_droptype		X	X	
sp_dropuser		X	X	

System procedure	Normal Companion Mode	Asymmetric Primary Suspended Mode	Failover Mode	Notes
sp_renamedb			X	See below for additional restrictions for this system procedure.

sp_changedbowner and sp_renamedb cannot run during failover mode, and have the following additional behavior changes:

- sp_changedbowner – After you run this procedure on local companion, you must manually run it on the remote server as well to synchronize the companions if the following are true:
  - You are not running this command in master.
  - The companion is in suspended or normal companion mode
  - The companion was configured using the with_proxydb option.
- sp_renamedb – You must first run this system procedure in the primary database and then run it in the proxy database on the remote server, if the following are true:
  - You do not run this command in master
  - The companion is in suspended or normal companion mode
  - The companion is configured using the with_proxydb option

dbcc Options for High Availability Systems

Sybase Failover includes dbcc ha_admin, which addresses second point of failures. Second point of failures a situations in which a cluster operation fails because of problems with the high availability subsystem. For example, if you issue sp_companion 'prepare_failback' and the secondary companion crashes. dbcc ha_admin provides a method of backing out of the cluster operation. After dbcc ha_admin is complete, you can re-issue the cluster operation.

Note dbcc ha_admin should only by a System Administrator who is familiar with the high availability subsystem. Issuing this command at the wrong time may only further trouble an already problematic situation.

Table B-5 includes information about the dbcc ha_admin options.

Table B-5: dbcc ha_admin options

Option name	Function	Syntax and comments
rollback_failback	Rolls back the effect of sp_companion... prepare_failback and returns the companion to the failover mode. This command works irrespective of the results of prepare_failback command.	`dbcc ha_admin (" ", rollback_failback)` Where `" "` is a required empty placeholder • Can only be used in failback mode. • Any failback threads waiting for the resume command are killed with this command. • You may need to perform platform-specific steps to prepare you companions for rollback_failback option. See the configuration chapter for your platform for more information. • This command is only issued from the secondary companion.
rollback_failover	Rolls back the effects of failover from the primary companion, and returns it to normal companion mode. rollback_failover does not affect the secondary companion.	`dbcc ha_admin (" ", rollback_failover)` Where `" "` is a required empty placeholder • This command can only be used in failover mode. • You may need to perform platform-specific steps to prepare you companions for rollback_failover option. See the configuration chapter for your platform for more information. • rollback_failover has no effect on the companion server that failed. The companion server that takes over the failed companion's work load resumes normal companion mode. • This command is only issued from the secondary companion. • This command works even when failover marked the databases "suspect"
drop_failedoverdb	Only used in failover mode. drop_failoeverdb drops the failed-over databases that could not be dropped with the drop database command. This command also cleans up the master_companion of all the metadata relating to the dropped database	`dbcc ha_admin (" ", drop_failedoverdb, database_name)` Where `" "` is a required empty placeholder, and *database_name* is the name of the database you are dropping. • Use as a last resort, when you must drop a database to complete the load of another database.
clusterlock	Acquires or releases cluster-wide locks during a cluster operation.	`dbcc ha_admin (" ", clusterlock, [acquire | release])` For more information about cluster-wide locks and releasing them, see "Cluster locks in a high availability node" on page 16.

Option name	Function	Syntax and comments
state_machine	Moves the companion server to single-server mode.	`dbcc ha_admin (' ', 'state_machine', 'halt')` Where `" "` is a required empty placeholder. For information about using this option, see "Re-Installing installmaster and installhasvss" on page 341.
session	Invokes clients that are sleeping because of a failed sp_companion...resume. Clients that are invoked disconnect from the secondary companion and connect to the primary companion.	`dbcc ha_admin (SYB_HACMP, session, "drop")`

dbcc dbrepair Option for Sybase Failover

Sybase Failover adds the dropproxydb option to dbcc dbrepair.

Table B-6: dbcc dbrepair dropproxydb option

Option Name	Function	Syntax and Comments
dropppoxydb	Drops proxy databases.	`dbcc dbrepair(`*`database_name`*`, dropproxydb)` Where *database_name* is the name of the database whose proxy database you are dropping.

APPENDIX C

Open Client Functionality in a Failover Configuration

This chapter discusses the changes required for Open Client to work with Sybase's Failover.

CTLIB application changes

Note An application installed in a cluster must be able to run on both the primary and secondary companions. That is, if you install an application that requires a parallel configuration, the secondary companion must also be configured for parallel processing so it can run the application during failover.

You must modify all of your applications that are written with CTLIB API calls before they can work with Sybase's failover software. The following steps describe the modifications:

1 Set the CS_HAFAILOVER property using the ct_config and ct_con_props CTLIB API calls. You can set this property at either the context or the connection level. This property is set using the following syntax:

```
ct_config(context, action, CS_HAFAILOVER, buf, buflen, outlen)
ct_con_props(connection, action, CS_HAFAILOVER, buf, buflen, outlen)
```

2 Modify the *interfaces file* so clients fail over to the secondary companion.

The *interfaces file* includes a line labeled *hafailover* that enables clients for reconnect to the secondary companion when the primary companion crashes or you issue a shutdown with nowait, triggering failover.

See “Add entries for both Adaptive Servers to the interfaces file” on page 52 for information about adding this line to the *interfaces file*.

3 Write application failover messages according to the following parameters:

- As soon as the companion begins to go down, clients receive an informational message that failover is about to occur. Treat this as an informational message in the client error handlers.
- Once the failover property is set (from step 1) and the *interfaces file* has a valid entry for the *hafailover* server, the client connection is an failover connection, and clients reconnect to the secondary companion appropriately.

However, if the failover property is set but the *interfaces file* does not have a entry for the hafailover server (or vice-versa), then it is a not a failover connection. Instead, it is a normal non-high availability connection with the failover property turned off. The user must check the failover property to know whether or not the connection was a failover connection

4 Add return codes

When a successful failover occurs, the client issues a return value named CS_RET_HAFAILOVER, which is specific to the following CTLIB API calls:

```
ret = ct_results(cmd, result_type)
ret = ct_send(cmd)
```

CS_RET_HAFAILOVER is returned from the API call during a synchronous connection. In an asynchronous connection, these API `s issue CS_PENDING and the callback function returns CS_RET_HAFAILOVER. Depending on the return code, the customer can do the required processing, such as sending the next command to be executed.

5 Rebuild your applications, linking them with the libraries included with the failover software.

Note You cannot connect clients with the failover property (for example isql -Q) until you issue sp_companion resume. If you do try to reconnect them after issuing sp_companion prepare_failback, the client hangs until you issue sp_companion resume.

Glossary

This glossary describes terms used in this book. For a description of Adaptive Server and SQL terms, refer to the *Adaptive Server Glossary*.

Asymmetrical — A high availability system consisting of one primary companion and one secondary companion. In an asymmetric system, only the primary companion can failover. In this system, the secondary Adaptive Server is also known as a "hot stand-by."

Companion Server — Each Adaptive Server in a high availability system is a companion. One of the Adaptive Servers is a **companion** (see below for definition) and the other is the **secondary** companion (see below for definition).

Cluster — A collection of nodes in a high availability system. A cluster for the Adaptive Server high availability system consists of two nodes.

Failback — The planned event during which Adaptive Server is migrated back to, and restarted on, the original machine. This involves moving the failed-over databases, devices, and client connections from the secondary companion to the restarted primary companion.

Failover — During failover, Adaptive Server migrates to another machine which takes over the responsibility of managing the failed over Adaptive Server. Failover can occur because of either a scheduled maintenance or a failure of Adaptive Server or the machine running Adaptive Server.

Failover Mode — The mode of the primary companion after it has failed over and is running on the secondary companion.

High Availability — A system that is designed to reduce the amount of downtime a system suffers.

Node — A machine in a high availability system.

Normal Companion Mode — The mode during which two Adaptive Servers in a high availability system are functioning as independent servers and are configured to failover during a scheduled maintenance or system failure.

Primary companion — The Adaptive Server whose databases and connections are migrated to the secondary Adaptive Server during failover.

Proxy Databases — Place holder databases created on the secondary companion for every user database on the primary companion. Proxy databases reserve the database names so that during failover, all database names are unique on the system. For more information about proxy databases, see Chapter 5, “Proxy Databases, User Databases, and Proxy System Tables”

Secondary companion — The Adaptive Server configured to accept the failed over primary Adaptive Server during failover

Single-Server Mode — The mode of Adaptive Server when it is being configured for high availability. During this mode, Adaptive Server cannot failover.

Suspended Companion Mode — The mode of Adaptive Server after companion mode has been suspended. During this mode, Adaptive Server cannot failover; it is working independently of the other Adaptive Server.

Symmetrical — A high availability system in which two independent Adaptive Servers act as failover servers for each other. That is, each Adaptive Server acts as both a primary and a secondary companion.

Index

Symbols

A

B

C

D

E

F

G

H

I

L

M

N

O

P

Q

R

S

T

U

V

W