



Using Sybase Failover in a High Availability System

Adaptive Server[®] Enterprise

15.7

DOCUMENT ID: DC00356-01-1570-02

LAST REVISED: February 2012

Copyright © 2012 by Sybase, Inc. All rights reserved.

This publication pertains to Sybase software and to any subsequent release until otherwise indicated in new editions or technical notes. Information in this document is subject to change without notice. The software described herein is furnished under a license agreement, and it may be used or copied only in accordance with the terms of that agreement.

Upgrades are provided only at regularly scheduled software release dates. No part of this publication may be reproduced, transmitted, or translated in any form or by any means, electronic, mechanical, manual, optical, or otherwise, without the prior written permission of Sybase, Inc.

Sybase trademarks can be viewed at the Sybase trademarks page at <http://www.sybase.com/detail?id=1011207>. Sybase and the marks listed are trademarks of Sybase, Inc. ® indicates registration in the United States of America.

SAP and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP AG in Germany and in several other countries all over the world.

Java and all Java-based marks are trademarks or registered trademarks of Oracle and/or its affiliates in the U.S. and other countries.

Unicode and the Unicode Logo are registered trademarks of Unicode, Inc.

All other company and product names mentioned may be trademarks of the respective companies with which they are associated.

Use, duplication, or disclosure by the government is subject to the restrictions set forth in subparagraph (c)(1)(ii) of DFARS 52.227-7013 for the DOD and as set forth in FAR 52.227-19(a)-(d) for civilian agencies.

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

Contents

CHAPTER 1	Failover and Failback.....	1
	What is failover?	1
	Client connections during fail over	3
	User logins in failover	3
	What is failback?	4
	Performing failback.....	5
	Cluster locks in a high availability node.....	6
CHAPTER 2	What is High Availability?	9
	Differences between active-active and active-passive.....	11
	Requirements for failover	12
	Resource requirements	13
	How does Sybase Failover work with high availability?	14
	Single-system presentation	16
	Special considerations for Sybase Failover	16
	Installing the monitoring table scripts	16
	Using disk mirroring.....	17
	Running the installhasvss script	17
	Creating a SYB_HACMP server entry.....	18
	Defining user-defined datatypes.....	18
	Adaptive Server and two-phase commit transactions	18
CHAPTER 3	Asymmetric and Symmetric Configurations	19
	Asymmetric and symmetric configuration.....	19
	Configuring the asymmetric companion	19
	Configuring the symmetric companion	21
	Auditing in a high availability system.....	23
	Setting auditing options	23
CHAPTER 4	Modes of Failover.....	25
	What are modes?	25
	Different modes of a companion server	26
	Determining the companion's mode.....	28

	Domains	29
CHAPTER 5	Proxy Databases, User Databases, and Proxy System Tables .	31
	Proxy databases	31
	Creating proxy databases	32
	Size of the proxy databases	34
	Commands and system procedures in proxy databases	34
	Manually updating the proxy databases	36
	Proxy system tables in master	36
CHAPTER 6	Running do_advisory	37
	What is the do_advisory option?	37
	Running the do_advisory option	41
	Quorum attributes	42
CHAPTER 7	Configuring Adaptive Server for Failover on HP	45
	Hardware and operating system requirements	45
	Preparing Adaptive Server for high availability	46
	Installing Adaptive Servers	46
	Adding entries for both Adaptive Servers to the interfaces file	46
	Setting the value of \$SYBASE	47
	Configuring sybha executable	48
	Creating a new default device other than master	49
	Adding the local server to syssservers	49
	Adding the secondary companion to syssservers	49
	Running installhasvss	50
	Assigning ha_role to system administrator	50
	Verifying configuration parameters	51
	Configuring HP for failover	51
	Creating the package configuration	52
	Editing the ASE_HA.sh script	53
	Verifying and distributing the configuration	61
	Configuring companion servers for failover	64
	Running sp_companion with do_advisory option	64
	Creating an asymmetric companion configuration	64
	Creating the symmetric configuration	66
	Administering Sybase Failover	66
	Failing back to the primary companion and resuming normal companion mode	67
	Suspending companion mode	67
	Dropping companion mode	68
	Troubleshooting Sybase Failover on HP	69

Error message 18750.....	69
Recovering from a failed prepare_failback.....	70
Location of error logs.....	71
Upgrading Adaptive Server	72

CHAPTER 8

Configuring Adaptive Server for Failover on IBM AIX HACMP. 75

Hardware and operating system requirements	75
Requirements for running Failover on IBM AIX HACMP	76
Preparing Adaptive Server to work with high availability.....	77
Installing Adaptive Servers.....	77
Adding entries for both Adaptive Servers to the interfaces file	78
Setting the value of \$SYBASE	79
sybha executable	79
Verifying configuration parameters.....	80
Adding thresholds to the master log.....	81
Creating a new default device other than master.....	81
Adding a local server to syssservers.....	82
Adding a secondary companion to syssservers.....	82
Running installhasvss script	82
Assigning ha_role to System Administrator.....	83
Configuring the IBM AIX subsystem for Sybase Failover	83
Modifying the ASE_HA.sh script	84
Configuring resource groups in HACMP	88
Configuring companion servers for failover.....	91
Running sp_companion with do_advisory option	92
Creating an asymmetric companion configuration	92
Creating the symmetric configuration.....	93
Starting the primary companion as a monitored resource.....	94
Administering Sybase Failover.....	95
Failing back to the primary node	95
Suspending companion mode.....	97
Resuming normal companion mode.....	98
Dropping companion mode	99
Troubleshooting fail over for HACMP for AIX.....	99
Error message 18750.....	100
Recovering from a failed prepare_failback.....	100
Location of error logs.....	101
Upgrading Adaptive Server	102

CHAPTER 9

Active-Active Configuration for Sun Cluster 3.0 and 3.1 105

Hardware and operating system requirements	105
Active-active setup in Sun Cluster.....	106
Preparing Adaptive Server for active-active setup	108

- Installing Adaptive Servers 108
- Adding entries for both Adaptive Servers to the interfaces file 108
- The value of \$SYBASE is the same for both companions 109
- Executing sybha 110
- Creating new default devices 111
- Adding the local server to syssservers..... 111
- Adding secondary companion to syssservers..... 112
- Assigning the ha_role to System Administrator..... 112
- Running installhasvss script 112
- Verifying configuration parameters..... 113
- Adding thresholds to the master log..... 113
- Adding user and login for fault monitor..... 114
- Configuring the Sun Cluster subsystem..... 114
 - Using the syscadm script 116
 - Adaptive Server resource extension properties 120
 - Configuring Adaptive Server resource groups 122
 - Using SUNW.HASStoragePlus..... 124
- Configuring companion servers for failover..... 126
 - High availability services library within Adaptive Server..... 126
 - Running sp_companion with do_advisory 127
 - Creating an asymmetric companion configuration 128
 - Setting up a symmetric configuration 130
- Administering Sybase Failover..... 131
 - Failing back to the primary companion..... 131
 - Suspending normal companion mode..... 132
 - Resuming normal companion mode..... 133
 - Dropping companion mode 133
- Verifying high availability on Sun Cluster 133
- Configuring the resource groups manually 134
 - Primary companion resource group 135
 - Secondary companion resource group 137
- Upgrading Adaptive Server 140
- Troubleshooting 143
 - Recovering from a failed prepare_failback..... 143
 - Recovering from a secondary failover on the secondary companion 144
 - Preventing the failover of secondary companion 144
 - Changing resource and resource group state 145
 - Location of the error logs..... 145

- CHAPTER 10 Active-Passive Configuration for Sun Cluster 3.0 and 3.1 147**
 - Hardware and operating system requirements 148
 - Active-passive setup in Sun Cluster..... 148
 - Failing back in an active-passive configuration 152

Clients in an active-passive configuration	152
Preparing Adaptive Server for active-passive setup	152
Installing Adaptive Server.....	153
Passing environment to Adaptive Server	154
Running the SySam license manager in the cluster.....	154
Adding an entry for Adaptive Server to the interfaces file	155
Verifying configuration parameters.....	157
Adding thresholds to the master log.....	157
Adding user and login for fault monitor.....	158
Configuring the Sun Cluster subsystem.....	159
Using the syscadm script	160
Configuring the Adaptive Server resource group	164
Using SUNW.HAStoragePlus.....	166
Verifying the active-passive configuration	167
Configuring the resource group manually	169
Upgrading Adaptive Server	172
Location of the error logs	173

CHAPTER 11

**Configuring Adaptive Server for Failover on Veritas 5.0 and later
175**

Hardware and operating system requirements	176
Preparing Adaptive Server to work with high availability.....	178
Installing Adaptive Servers.....	179
Adding entries for both Adaptive Servers to the interfaces file	179
sybha executable	180
Creating a new default device	181
Adding the local server to syssservers.....	181
Assigning ha_role.....	182
Installing high availability stored procedures.....	182
Verifying configuration parameters.....	183
Adding thresholds to the master log.....	183
Configuring the Veritas subsystem for Sybase Failover	184
Installing the HAase agent	184
Creating an Adaptive Server login file	185
Importing the HAase resource type.....	185
Starting the HAase agent	186
Adding the HAase resource	186
Configuring an instance of the HAase resource for each service group	188
Configuring companion servers for failover.....	189
Adding user and login for high availability monitor	189
Running sp_companion with do_advisory option	189
Verifying the high availability agent	190
Creating an asymmetric companion configuration	190

	Configuring for symmetric configuration.....	192
	Administering Sybase Failover.....	193
	During failover	193
	Failing back to the primary companion.....	193
	Suspending normal companion mode.....	194
	Resuming normal companion mode.....	194
	Dropping companion mode	195
	Upgrading from an agent of resource type Sybase.....	195
	Upgrading Adaptive Server	196
	Troubleshooting failover for Veritas Cluster	199
	Recovering from a failed prepare_failback.....	200
	Location of the logs	201
CHAPTER 12	Configuring Adaptive Server for Failover on Windows	203
	Hardware and operating system requirements	203
	Installing Adaptive Servers.....	204
	Changing the domain administration account	205
	Adding entries for both Adaptive Servers to sql.ini.....	205
	Creating a new default device other than master.....	206
	Adding the primary companion as a local server	207
	Adding secondary companion to syssservers.....	207
	Assigning ha_role to the System Administrator.....	207
	Running insthasv to install high availability stored procedures	207
	Verifying configuration parameters	208
	Running sp_companion with do_advisory option	208
	Configuring Windows	209
	Asymmetric setup from the command line	209
	Symmetric configuration from the command line	211
	Configuring Sybase companion servers using the Windows	
	management tools	212
	Configuring and securing Microsoft Cluster Server.....	214
	Securing the MSCS cluster	215
	Troubleshooting	216
	Error message 18750.....	216
	Recovering from a failed prepare_failback.....	217
	Upgrading Adaptive Server	218
APPENDIX A	Open Client Functionality in a Failover Configuration.....	221
	CTLIB application changes	221
APPENDIX B	Troubleshooting Secondary Points of Failure.....	223
	Troubleshooting with dbcc ha_admin.....	223

	Reinstalling installmaster	224
	Rerunning installhasvss	224
	Using dbcc ha_admin for rolling back failover commands.....	225
	Using @@hacmpservername	225
	Error messages	226
APPENDIX C	Changes to Commands, System Procedures, and Databases	229
	Changes to commands.....	229
	Changes to system procedures.....	231
	System procedures hold table lock.....	231
	System procedures that synchronize changes	232
	Other changes to system procedures.....	233
	dbcc options for high availability systems.....	235
	Glossary	237
	Index	239



This chapter describes the characteristics of failover and failback.

Topic	Page
What is failover?	1
What is failback?	4
Cluster locks in a high availability node	6

What is failover?

When a computer system fails, the databases, metadata, and user connections are moved to a secondary server so that users can still access data. This is known as failover.

With Adaptive Server[®], you set up a high availability cluster that is configured for failover. 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 fails over to the secondary companion.

Steps 2 and 3 are described below. See your high availability system documentation for a description of step 1.

During fail over, a secondary server detects the primary failure through the operating system's high availability system and initiates the failover mechanism, which:

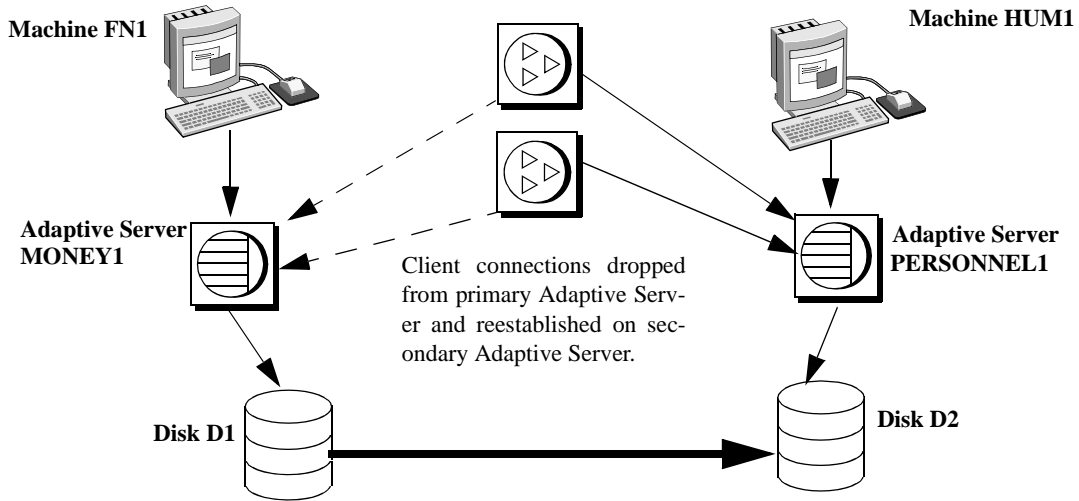
- 1 Performs a disk reinit to remap the master device path name to its local drive. disk reinit does not disturb the contents of the master device.
- 2 Mounts the master database, recovers it, and brings it online.

- 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`, `sybssystemprocs` becomes `sybssystemprocs_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 and clients using the `CS_FAILOVER` property are retained and reestablished on the secondary companion. Uncommitted transactions must be resubmitted.

Figure 1-1: Failover process



Master databases and system databases are migrated to secondary Adaptive Server and appended with suffix `_companion`. Proxy databases are renamed, shut down, and replaced with user databases.

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 fail over are rolled back. After fail over is complete, clients or users must resubmit rolled-back transactions.

Client connections during fail over

Clients with the failover property reconnect automatically during fail over. 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.

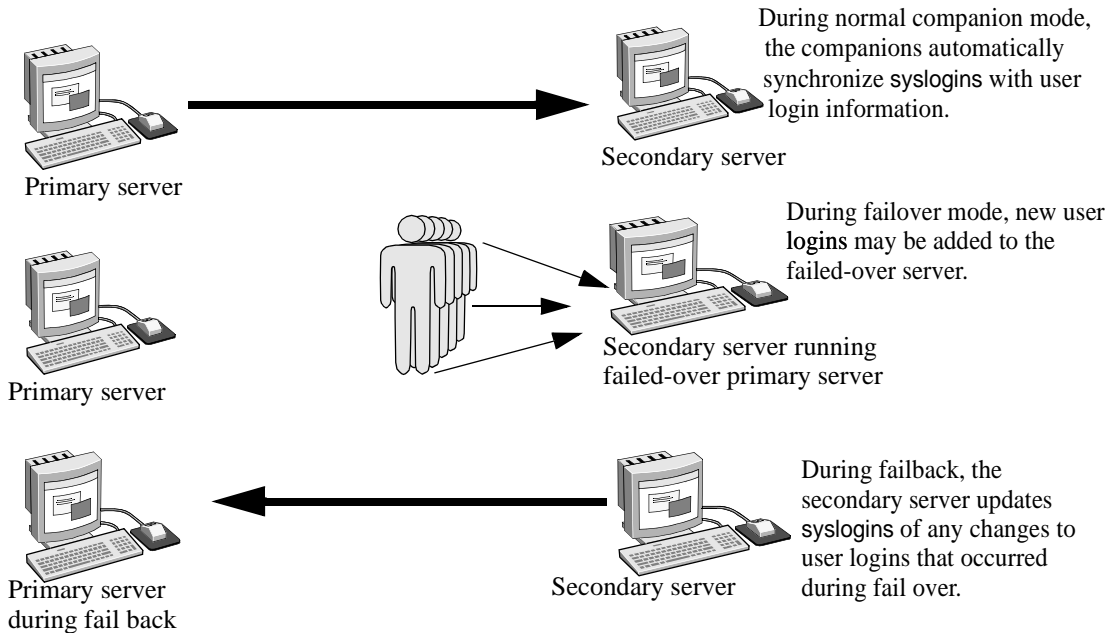
For more information about adding this information to the *interfaces* file or *sql.ini*, see the appropriate platform-specific configuration chapter.

Client applications must resend any queries that were interrupted by fail over. See Appendix A, “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 reestablished once the companion has successfully failed-over.

Figure 1-2: Synchronizing syslogins between primary and secondary servers



All user roles and privileges are maintained after fail over.

What is failback?

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

Issue `prepare_failback` from the secondary companion to release database devices and databases.

```
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 a 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)
```

Move the devices back to the primary node according to individual platform subsystem.

- 2 The high availability system restarts the primary companion automatically.
- 3 Run `sp_companion` with the `do_advisory` option to make sure there are no attribute settings to prevent a failback from succeeding. See Chapter 6, “Running `do_advisory`.”

- 4 After fail back is complete, issue `sp_companion` from the primary companion (the companion that originally failed) to return to normal companion mode. See the appropriate platform-specific chapter for more information about `sp_companion resume`

Note You cannot connect clients with the failover property 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 companion servers 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 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 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 before it is finished), the client that failed leaves behind a lock that blocks the next client from attempting to obtain a cluster lock.

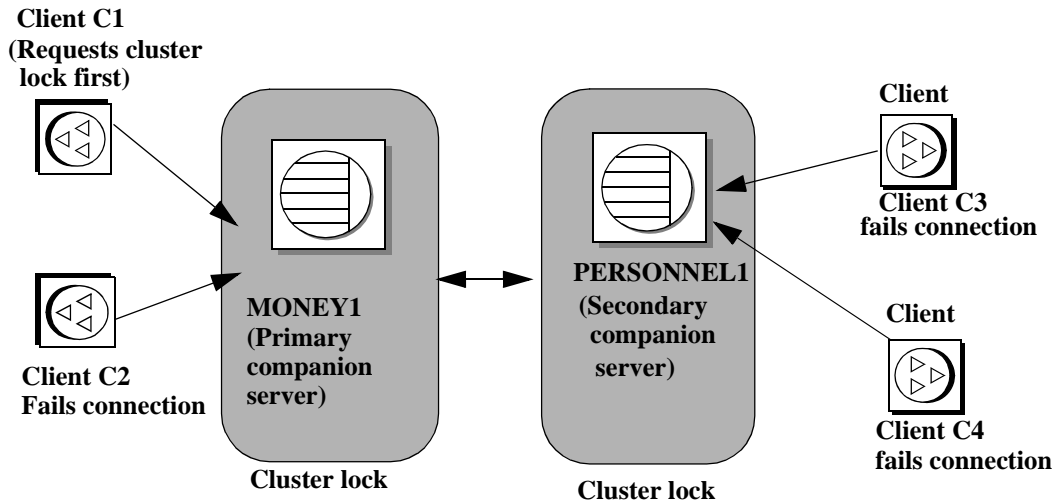
Issue `dbcc ha_admin` to acquire or release cluster locks:

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

For more information about `dbcc ha_admin`, see “`dbcc` options for high availability systems” on page 235.

Figure 1-3 describes two companion servers to which four clients are connecting. All of them are attempting to perform cluster operations:

Figure 1-3: 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 obtain 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 finished with its cluster operation, it releases the cluster-wide lock.
- 7 Client connections C2, C3, and C4 inform the System Administrator that they cannot 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.

What is High Availability?

Topic	Page
Differences between active-active and active-passive	11
Requirements for failover	12
Resource requirements	13
How does Sybase Failover work with high availability?	14
Single-system presentation	16
Special considerations for Sybase Failover	16

As more businesses depend on computer systems they also expect these systems to be available at all times. High availability means that a system is setup so that if a computer system or network has a hardware or software failure, the system fails over into a backup system. Business can then go on as usual. Once the problem is resolved, the system fails back to the primary system.

Sybase[®] High Availability Failover enables Adaptive Server Enterprise to work in a cluster of servers in a network in a specific configuration such as active-active or active-passive. Such a system allows for failover and then the failback on servers. This manual includes information about how to set up and run the Adaptive Server high availability system.

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.

When the **primary companion** or machine fails, the databases, metadata, and user connections are moved to a secondary server so that users can still access data. This is known as **failover**.

When the primary companion or machine is prepared to resume operation, the user with the `ha_role` performs a failback, which returns the servers to normal companion mode.

Active-active configuration

An **active-active** setup is a two-node configuration where both nodes in the **cluster** include Adaptive Servers managing independent workloads, 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 fail over. After the primary companion is ready to resume its workload, it is moved back to its original node. This movement is called a failback.

When a system fails over, clients that are connected to the primary companion and use 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 fail over. See your operating system documentation for information about configuring 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 appropriate platform-specific configuration chapter of this manual.

Active-passive configuration

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

When the Adaptive Server cannot run on the primary node, it fails over, 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 fail over or fail back, clients connected to the Adaptive Server reestablish 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 or. Contact your provider for other cluster platforms. See Chapter 10, “Active-Passive Configuration for Sun Cluster 3.0 and 3.1,” 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.

Adaptive Server supports following cluster platforms for Sybase-HA configuration:

- HPIA – MCSG 11.19
- IBM AIX – HACMP 6.2
- Sun Solaris - VCS 5.0, SunCluster 3.2
- Linux:
 - RHEL 5.0 / VCS 5.0
 - RHEL 6.0 / VCS 5.1 SP1
 - SuSE Enterprise 11 / VCS 5.1
- Win2000 - Cluster Manger 5.0
- Win2003 - Cluster Manger 5.2

Note Adaptive Server version 15.7 and later does not support high availability on Windows x86 32-bit. If you are currently running high availability on Windows x86 32-bit, contact your customer support representative.

Differences between active-active and active-passive

Table 2-1 summarizes the differences between an active-active and an active-passive configuration.

Table 2-1: Difference between active-active and active-passive

Active-active	Active-passive
<i>Setup:</i> Two Adaptive Servers are configured as companion servers, each with independent workloads. These companions run on the primary and secondary nodes, respectively, as individual servers until one fails over.	<i>Setup:</i> A single Adaptive Server runs either on the primary node or on the secondary node. The Adaptive Server runs on the primary node before a fail over and the secondary node after fail over.
<i>Failover:</i> When fail over 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.	<i>Failover:</i> When a system fails over, the Adaptive Server and its associated resources are relocated to, and restarted on, the secondary node.
<i>Failback:</i> 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.	<i>Failback:</i> 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.
<i>Client Connection failover:</i> 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.	<i>Client Connection failover:</i> 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 your platform 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 15.0 or higher
- Be running the latest version of Open Client
- Be at the same release level
- Have a compatible configuration
- Be running Component Integration Services (CIS)
- Be running a high availability system (for example Sun Cluster)

- Be configured for either parallel or nonparallel processing.

Resource requirements

Adaptive Servers configured as companions in a high availability system have different resource requirements than Adaptive Servers that function individually. These differences exist because the **secondary companion** must process all the work during fail over. This is true even if the companions are set up asymmetrically. Consequently, an Adaptive Server in a high availability system has higher resource requirements than if it is a single server. For more information, see “Single-system presentation” on page 16.

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 Adaptive Server 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.
- `sp_configure` option number of open databases – on an Adaptive Server configured for failover, the number of open databases is reduced by two to ensure successful failover. That is, if you currently have the number of open databases as ten, you can open only eight databases.
- `sp_configure` option number of user connections – on an Adaptive Server configured for failover, the number of user connections is reduced by two to ensure successful failover. That is, if the number of user connections is 50, you can use only 48 user connections.

Client applications that connect to companion servers must relink their libraries with the libraries included with failover software. See “CTLIB application changes” on page 221 for more information about using Open Client with failover.

How does Sybase Failover work with high availability?

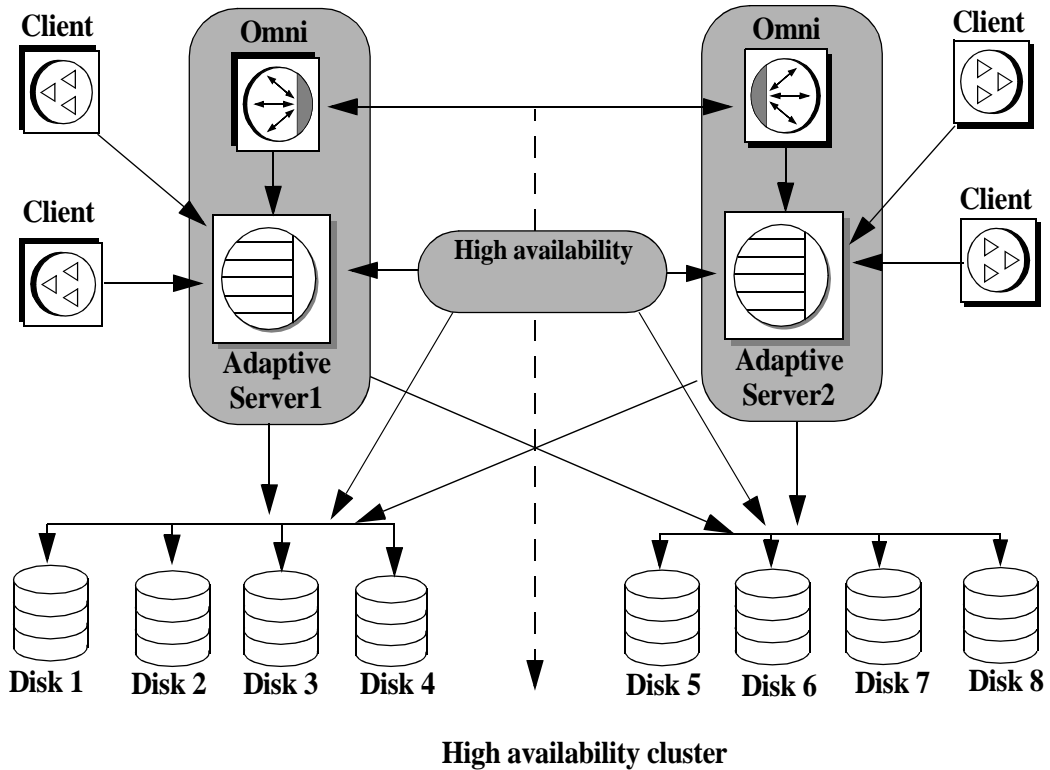
A high availability system includes both hardware and software. Sybase Failover is software that allows a companion server to withstand a single point of failure in the cluster.

A system that uses Sybase Failover includes 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 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. When the second server assumes another server’s workload, it is called **fail over**. Moving the workload back to the original server once it is up and running again is called a **failback**.

Figure 2-1 describes a typical configuration consisting of two Adaptive Servers.

Included with the operating system is a high availability system (for example, Sun Cluster for Sun) that detects and broadcasts to the cluster that part of the system is failing or is being shut down for maintenance. When Adaptive Server goes down, the high availability system tells the second machine to take over the workload. Any clients connected to the Adaptive Server that is failing are automatically reconnected to the second Adaptive Server.

Figure 2-1: High availability system using Sybase Failover



The machines in Figure 2-1 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 Server1 is the primary companion and it fails, Adaptive Server2, as the secondary companion, reads its disks (1 – 4) and manages any databases on them until Adaptive Server1 can be brought back online. Any clients that are connected to Adaptive Server1 and are using the failover property are connected automatically to Adaptive Server2.

Single-system presentation

One of the hallmarks of a cluster system is that users are unaware that they are logged in to a system made up of two Adaptive Servers; it appears as if they are logging in to a single system with access to all the databases on the cluster. Applications also see only a single system. They log in to either companion 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 the same. Both individual Adaptive Servers, as well as the cluster, may require system maintenance.

Special considerations for Sybase Failover

The Adaptive Server functions discussed in this section require special consideration when you configure Sybase Failover.

Installing the monitoring table 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 primary and the secondary 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
```

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

Using disk mirroring

Sybase Failover and the high availability system enable users to access data while the server to which they were originally connected has failed. However, neither of these systems prevent disk failures. To make sure you do not lose any data because of a disk failure, use Sybase 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.

Running the *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, this script is *insthasv*, and is located in `%SYBASE%\%SYBASE_ASE%\scripts`.

Note You must rerun *installhasvss* if you rerun the *installmaster* script (see Reinstalling *installmaster* and “Rerunning *installhasvss*” on page 224).

Do not use an *installhasvss* script from a different version of Adaptive Server.

For more information, see the appropriate platform-specific configuration chapter.

Creating a **SYB_HACMP** server entry

The *installhasvss* script creates an entry in *syservers* 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 *syservers* entry allows the primary companion to communicate with the secondary companion using the entries in the *interfaces* file. Do not use the SYB_HACMP server entry to create any queries or stored procedures with the companion server.

Do not delete the SYB_HACMP server entry. If this entry is inadvertently deleted, you must re-run *installmaster* and *installhasvss*.

Defining user-defined datatypes

Updates to tables that include either Java or user-defined datatypes are not synchronized after Adaptive Servers in a 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. You must manually update any changes made to columns that store user-defined datatypes.

Additionally, for another example, if the *pubs2* database on the primary companion includes a table that uses user-defined datatypes, 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 Sybase two-phase commit (SYB2PC) transactions, which use the Sybase two-phase commit protocol.

Asymmetric and Symmetric Configurations

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

Topic	Page
Asymmetric and symmetric configuration	19
Auditing in a high availability system	23

Asymmetric and symmetric configuration

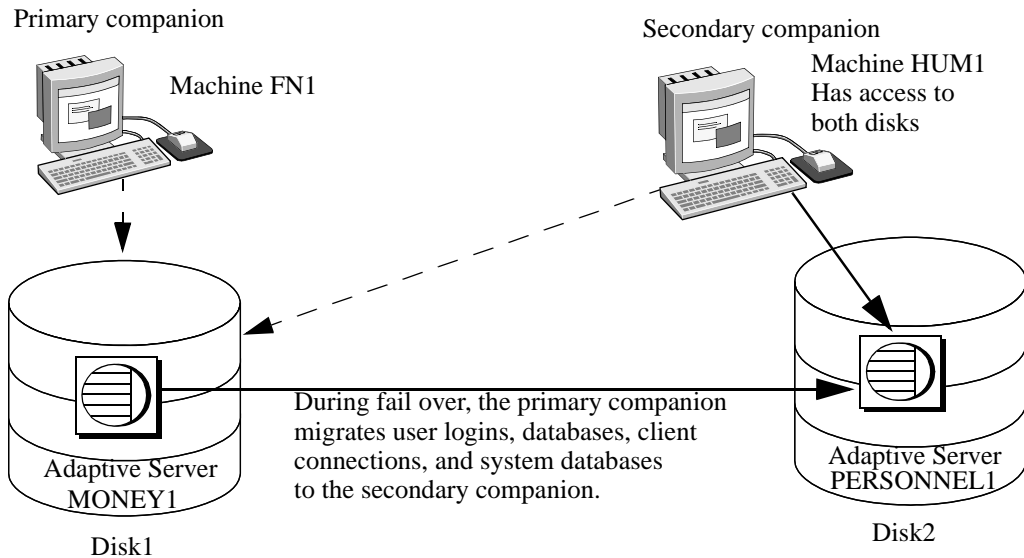
You can configure companion servers either asymmetrically or symmetrically. You must configure companions asymmetrically before you can configure them symmetrically.

Configuring the asymmetric companion

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 scheduled maintenance. The secondary companion is an independent Adaptive Server, and can have its own applications running. To configure for fail over, 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 fail over, 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 *Reference Manual: Procedures* for information about `sp_companion`.

Figure 3-1: Asymmetric configuration in 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 FN1 with a dual-ported SCSI. Because this is an asymmetric setup, PERSONNEL1 cannot fail over to MONEY1. Disk1 must be shared, and Disk2 can be a local disk.

See the appropriate platform-specific configuration chapter for detailed information about configuring Adaptive Server for an asymmetric setup.

Performance of Adaptive Server in an asymmetric configuration

During normal companion mode, performance of system procedures that update user information (`sp_addlogin`, `sp_addrole`, and so on) and of commands such as `create database` is slightly degraded because the primary companion must perform the command locally, then synchronize 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 fail over depends on the configuration of the secondary companion. If the secondary server is configured similarly to the primary server, performance should be similar before and after fail over. However, if the secondary server is not as robust (for example, has less memory or fewer CPUs) as the primary server, then performance after fail over is degraded. The performance of the secondary companion can also be degraded after fail over because it is running the primary companion as well as any applications.

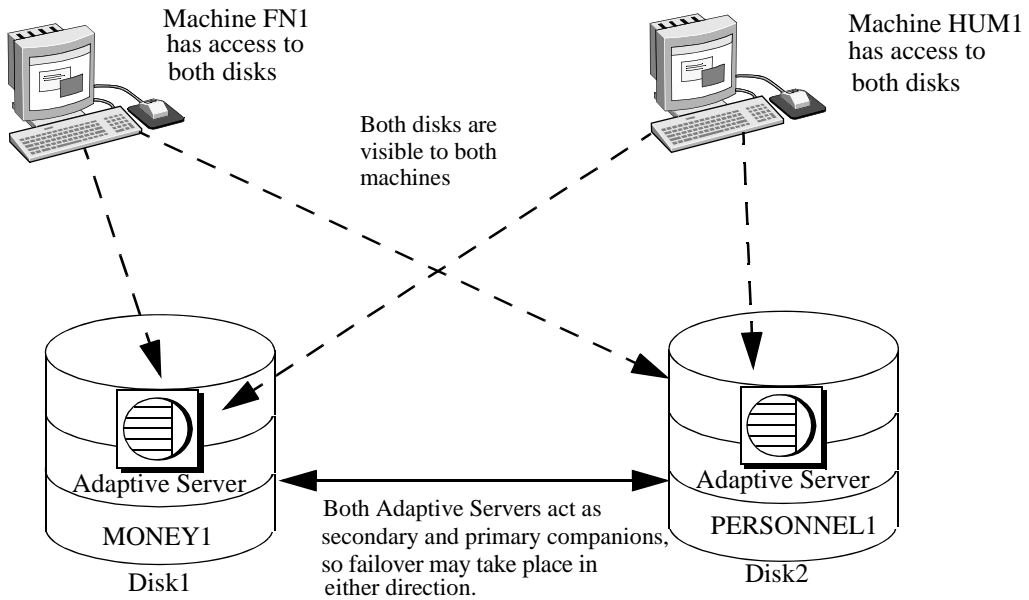
Configuring the symmetric companion

Like asymmetric configuration, symmetric configuration consists of two fully functional Adaptive Servers running on separate machines, each 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 as asymmetric companions.

Figure 3-2 describes a symmetric configuration between a financial department machine (FN1 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. Disk1 and Disk2 are both shared disks.

See the appropriate platform-specific configuration chapter for detailed information about configuring Adaptive Server 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, each machine can 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 with a reasonable level of performance. If both Adaptive Servers maximize their system resources, failover succeeds, but performance may be poor.

Auditing in a high availability system

Configure a companion for auditing the same way you configure a server that does not use failover. For more information, see “Setting auditing options” on page 23.

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

For the following auditing parameters, both companions must be configured identically. These parameters are checked as a quorum attribute, or when explicitly listed with `do_advisory`:

- allow procedure grouping
- unified login required
- secure default login
- systemwide password expiration
- use security services
- check password for digit
- minimum password length
- maximum failed logins
- auditing – turning this parameter on and off is not synchronized dynamically for the companions. If you change this parameter locally, you must manually update the remote companion.

Setting auditing options

You can configure auditing options (global, database-wide, and for each 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 failover.

- Users can still set database-wide options.
- The audit options of the local domain are used for both local and remote logins.

sybsecurity and Sybase Failover

The 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.

Audit trails and Sybase Failover

Audit trails are logged in the audit tables of the sybsecurity database. During fail over, 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 fail over, any new audit trails are placed in the audit table of the secondary companion. Any configuration or individual record changes related to auditing that are made on one companion are not automatically implemented on the other companion. For example, if you change an auditing configuration parameters on the primary companion, this change is not made on the secondary companion. 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.

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

Topic	Page
What are modes?	25
Domains	29

What are modes?

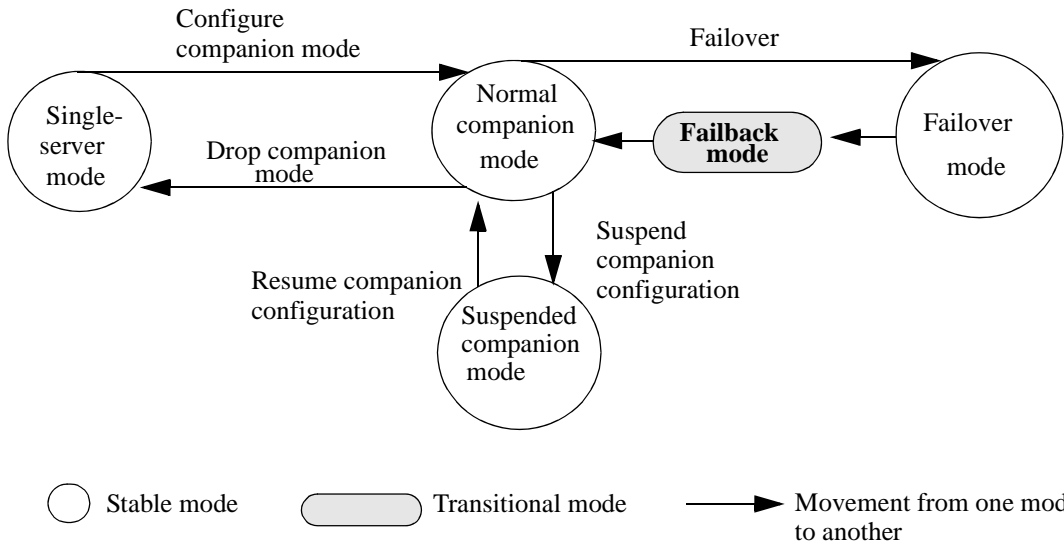
There are a series of modes that Adaptive Server runs through during high availability. There are two types of modes, stable and transitional. A **stable mode** is a system state in which Adaptive Server can exist for an extended period of time, such as the day-to-day operation of Adaptive Server.

Stable modes include:

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

Failback mode, which is **transitional**, occurs when Adaptive Server shifts from a failover mode to a normal companion mode, and is typically of very short duration. The movement that the primary companion makes while changing modes is 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

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 fail over, they operate in normal companion mode. This is the mode in which the day-to-day operations of Adaptive Server occur. In **asymmetrical** systems, the primary companion can fail over to the secondary companion. In **symmetric** systems, either companion can fail over to the remaining companion.

Suspended mode	<p>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.</p> <p>The companions cannot fail over, but the nodes upon which they are working can; you must perform some platform-specific steps to suspend node fail over. Also, before you shut down a companion in suspended mode, you must perform some platform-specific tasks. See the appropriate platform-specific chapter for more information.</p> <p>Many utilities and commands are severely restricted in suspended mode. See Appendix C, “Changes to Commands, System Procedures, and Databases,” for more information.</p> <hr/> <p>Note You should suspend a companion mode only from the secondary companion.</p> <hr/>
Failback mode	<p>Adaptive Server must enter a failback mode to move from failover mode on the secondary companion to normal companion mode on the primary companion.</p> <p>Failback 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 <code>sp_companion prepare_failback</code> to initiate fail back and migrate the failed-over Adaptive Server to its original node. See “Performing failback” on page 5.</p>
Resuming normal companion mode from suspended mode	<p>To resume normal companion mode, enter:</p> <pre>sp_companion "primary_server_name", resume</pre>
Dropping failover mode	<p>To permanently disable companion mode, enter:</p> <pre>sp_companion "server_name", 'drop'</pre> <p>When this command is complete, the two Adaptive Servers are no longer companion servers and are running in single-server mode.</p> <hr/> <p>Note <code>drop</code> 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.</p> <hr/>
	<p>If the companion you drop is in a symmetric setup, the cluster automatically assumes an asymmetric setup between the companions.</p>

Determining the companion's mode

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.
```

MONEY1 is configured for symmetric failover and is running in normal companion mode. You can also use the `@@cmpstate` global variable to determine the mode. At the `isql` prompt, enter:

```
select @@cmpstate
```

Note The `@@cmpstate` global variable is not used in a non-high availability environment, and reports a value of -2.

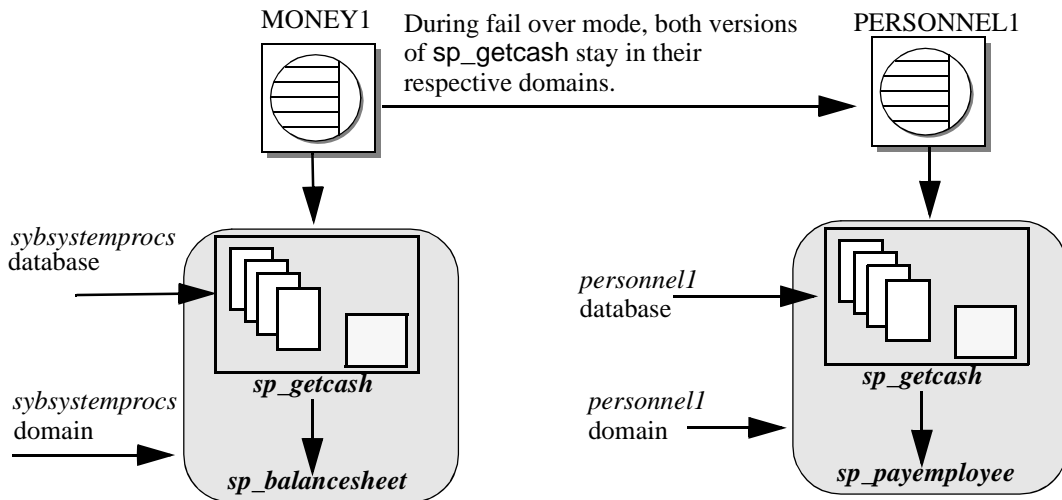
Table 4-1: @@cmpstate return values

@@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

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 to which database these objects belong. For example, 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 `sybssystemprocs` domain. In PERSONNEL1, `sp_getcash` (which issues a secondary stored procedure named `sp_payemployee`) is defined in the `personnell1` domain. During fail over, even though `sybssystemprocs` for MONEY1 migrates to PERSONNEL1 as `sybssystemprocs_companion`, its domain does not change, nor do the objects that are defined for this domain. Users that issue `sp_getcash` in `sybssystemprocs` for MONEY1 during fail over mode still issue the correct secondary stored procedure, `sp_balancesheet`.

The concept of domains is transparent to users.

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.

Proxy Databases, User Databases, and Proxy System Tables

Topic	Page
Proxy databases	31
Proxy system tables in master	36

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 **Adaptive Servers** as companions in an asymmetric setup. 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. **Proxy databases** are created dynamically as they are needed. See the *Reference Manual: Procedures*.

Do not use the `with_proxydb` on a symmetric setup.

Databases in companion servers are either primary or proxy databases. Primary databases are where 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
sybssystemdb
sybssystemprocs
tempdb
```

Note When the primary companion fails over, 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 you can configure them for Sybase Failover.

CIS does not import column defaults that exist in the primary database into the proxy table.

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. Proxy tables 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?

After the companions are configured with sp_companion...with_proxydb option:

- Proxy databases for all the primary companion's user databases are created when a companion configuration is created.
- Proxy databases are created for any new user databases that you 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.

Commands and system procedures in proxy databases

The behavior of some commands and system procedures changes when 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 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 a system procedure in a proxy database, any changes it makes do not appear in the primary database, and vice-versa.

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` proxy database, 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-defined 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 `sysystemprocs`, 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-defined 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 the high availability 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 *Reference Manual: Procedures*.

***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

You can use the `for proxy_update` option with `alter database` to manually synchronize your proxy databases with their primary databases.

. 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 `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 only the primary companion in a cluster environment. Also, a database needs not 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.

See the *Reference Manual: Procedures*.

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
```

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

Topic	Page
What is the <code>do_advisory</code> option?	37
Quorum attributes	42

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 fail over mode, or the companions may be configured for different languages.

To prevent these problems, `sp_companion` includes a `do_advisory` option that checks hundreds of attribute settings for each companion and issues warnings about any settings that may prevent a successful cluster operation. The attributes do not necessarily require the same values on both companions, but they must be compatible.

`sp_companion...do_advisory` does not change any attributes; it only advises about potential problems.

`sp_companion...do_advisory` is not triggered automatically; run it periodically to verify that there are no compatibility issues between your companions.

`do_advisory` allows you to specify the attributes to investigate. You can either include all the attributes, or you can specify subsets of attributes.

You can select subsets of *group*, *base*, or *quorum* attributes. A *group* subset includes a broad set of server settings (for example, all the login attributes or all the space attributes); a *base* subset includes specific settings within the *group* subset (for example, user logins or CIS settings). `do_advisory` reports only the attributes of the specified subset that will prevent a successful cluster operation.

Quorum attributes are configuration parameters that `sp_companion` checks every time, 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 42.

The following describes the server settings that make up each group:

- 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 – verifies that 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 The archives 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 companion, 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 companion are the same as those used by the secondary companion, if they exist.
- User types – verifies that all user-defined datatype definitions in the master database 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, you must drop and re-create 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 logins and permissions are consistent between the primary and secondary companions.
 - 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, and aliases, and user names in master are compatible across the companions. `do_advisory` automatically corrects any issues that it finds.
 - Default login incompatibilities of probe, qcollector, qrepository, and so on are fixed automatically.
- Roles group – verifies that all user-defined roles, login roles, and server-wide permissions are compatible between the primary and secondary companions.

- 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` in an asymmetric setup).

Running 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 to include 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 `sp_companion` to report.
- *base_attribute_name* is the name of the base attribute upon which you want `sp_companion do_advisory` reports.

`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 comparing 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 do not affect the cluster operation.
 - 1 – the attributes are not configured for the best configuration, but they will not prevent a cluster operation.
 - 2 – you cannot proceed with the cluster operation without altering the attributes.

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. The values in the Local Values column for these three attributes differs from the values in the Remote Values. The companions must have the same or compatible values.

Quorum attributes

If you issue `sp_companion` 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.

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. `do_advisory` checks the following configuration parameters as 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`
- `partition 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

do_advisory also checks the following database attributes:

- Charsets
- Java archives
- Languages
- Remote servers
- Sort order
- Time ranges
- User types
- Unique dbid
- Devnames
- Logins
- Roles

Configuring Adaptive Server for Failover on HP

This chapter contains the information for configuring Adaptive Server for Failover on HP.

Topic	Page
Hardware and operating system requirements	45
Preparing Adaptive Server for high availability	46
Configuring HP for failover	51
Configuring companion servers for failover	64
Administering Sybase Failover	66
Troubleshooting Sybase Failover on HP	69
Upgrading Adaptive Server	72

Hardware and operating system requirements

High availability requires the following hardware and system components:

- Two homogenous, network systems with similar configurations in terms of resources such as CPU, memory, and so on
- High availability system 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
- Third-party mirroring for media failure protection

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

Preparing Adaptive Server for high availability

This section discusses how to prepare Adaptive Server for a high availability configuration.

Installing Adaptive Servers

Install the primary and secondary servers. They must be installed in the same location on each node. The primary companion can be either newly installed, or it can be upgraded from an earlier version, 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 configuration for fail over is complete, you can add user logins and databases to the secondary companion.

If you are installing on the local disk, ensure that 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 the primary and secondary companions must include entries for both companions. The *interfaces* file for both companions must include a host name (instead of an IP address) that matches the host name string in the output from:

```
/usr/sbin/cmviewcl -p package_name
```

The server entry in the *interfaces* file must use the network name that is specified in *sysservers*. For information see the installation documentation for your platform.

Adding entries to *interfaces* file for client connections

To enable clients to reconnect to a 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 the 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 FN1 4100
    query tcp ether FN1 4100
    hafailover PERSONNEL1
```

Use *dsedit* to add entries to the *interfaces* file. If the *interfaces* entries already exist, you modify them as necessary to work for fail over.

See the *Utility Guide* for information about *dsedit*.

Setting the value of **\$SYBASE**

If you installed **\$SYBASE** on a local file system, **\$SYBASE** must point to the same directory name on both companions:

- Make 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 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 */usr/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.

Configuring *sybha* executable

The *sybha* executable enables the Adaptive Server High Availability Basic Services Library to interact with each platform's high availability cluster subsystem. Before *sybha* can run, you must change its ownership and permissions. You must also edit a file named *sybhauser* in *\$\$SYBASE/\$SYBASE_ASE/install* 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 *sybhagr*. You can either add this group to the */etc/group* file or you can add it to your NIS maps. Add the *sybase* user (the user that owns the *\$\$SYBASE* directory) to this group. 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, you must add each of these users 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 *sybhagr*:

```
chgrp sybhagr sybha
```

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

```
chmod 4550 sybha
```

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

```
cd $$SYBASE/$SYBASE_ASE/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 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 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, using `disk init` to create a new device. Use `sp_diskdefault` to specify the new device as the default before you configure Adaptive Server as a companion for fail over. 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`.

Adding the local server to *syssservers*

Use `sp_addserver`, to add the local server as the local server in *syssservers*, 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 restart Adaptive Server for this change to take effect.

Adding the secondary companion to *syssservers*

Add the secondary companion as a remote server in *syssservers*:

```
sp_addserver server_name
```

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

Running *installhasvss*

Note You must perform the tasks described in “Adding entries for both Adaptive Servers to the interfaces file” on page 46, before executing *installhasvss*. If you run *installhasvss* before performing these tasks, re-run *installmaster* to reinstall all the system stored procedures.

Enable High Availability, then restart Adaptive Server:

```
sp_configure "enable HA", 1
```

The *installhasvss* script:

- Installs the stored procedures required for Failover (for example, *sp_companion*).
- Installs the SYB_HACMP server in *syssservers*.

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

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.

Assigning *ha_role* to system administrator

You must have *ha_role* permission 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.

Verifying configuration parameters

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

- `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 restart Adaptive Server for it 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 *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 system for Sybase Failover. This section assumes that you have:

- Familiarized yourself with HP MC/ServiceGuard.
- Configured a two-node cluster hardware for MC/ServiceGuard.
- Installed HP MC/ServiceGuard version 11.15 on both nodes running under HP-UX 11.11.
- Installed and configured the cluster system.
- Set up volume groups to contain all the database devices in the cluster on the shared disk devices.
- Made all the shared volume groups part of the cluster configuration.

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

Creating 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 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, you might 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 uses 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 accessible only by “root”:

```
chmod 700 /etc/cmcluster/MONEY1
```
- 3 Create the same subdirectory on the secondary node. For example, to create this directory on machine FN1 for primary companion MONEY1:

```
rsh FN1 "mkdir /etc/cmcluster/MONEY1"
```
- 4 Change the permissions for this directory so it is only accessible by “root”:

```
rsh FN1 chmod 700 /etc/cmcluster/MONEY1
```
- 5 Generate a package configuration template for the primary companion using the *cmmakepkg* command:

```
/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,
- *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 changes will differ, depending on your machine names and parameters):

```
PACKAGE_NAME           MONEY1
FAILOVER_POLICY        CONFIGURED_NODE
FAILBACK_POLICY        MANUAL
NODE_NAME              FN1
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 the configuration 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
```

Editing the *ASE_HA.sh* script

The *ASE_HA.sh* template script configures the high availability system 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 in the previous section, 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*.

- 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. This indicates whether the *\$\$SYBASE* directory structure is *\$\$SYBASE/\$SYBASE_ASE/bin* (starting with 12.0) or *\$\$SYBASE/bin* (before 12.0)

Set this to:

- Yes if both servers are using Sybase Adaptive Server version 12.0 or later.
- 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 system 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 Failover (that is, they are running in single-server mode), the script attempts to start 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 *PACKAGE_NAME* must be the same as the companion name.

- *MONITOR_INTERVAL* – the amount of time, in seconds, this script 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 suspended companion server that prevents the halt script from completing. The value of *SHUTDOWN_TIMEOUT* must be less than the *timeout* variable set in the package configuration file.
- *RECOVERY_TIMEOUT* – the maximum amount of time the high availability system waits, in seconds, before determining that the companion failed to start. Set this number high enough to allow a loaded companion to restart. *RECOVERY_TIMEOUT* is also used as the maximum amount of time the subsystem waits for the failover and failback to complete.
- *SYBASE* – the location where 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* – the installation directory of Sybase Adaptive Server products.

- *SYBASE_OCS* – the installation directory for Sybase Open Client products.
- *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 HALOGIN. This must be the same on both the primary and secondary companion.

Note 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 each node.
- *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 *ASE_HA.sh* settings in *MONEY1.sh* for the primary companion MONEY1 running on host FN1, 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 not running or not in companion mode.

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

Variable	Setting
ASE_12_0	yes
ASE_HAFAILOVER	yes
BASIC_FAILOVER	yes
PACKAGE_NAME	MONEY1
MONITOR_INTERVAL	5
SHUTDOWN_TIMEOUT	60
RECOVERY_TIMEOUT	300
SYBASE_ASE	ASE-15_0
SYBASE_OCS	OCS-15_0
HALOGIN	“SA”
HAPASSWD	“Odd2Think”
PRIM_ASE_HOME	Defaults to directory <i>\$\$SYBASE/\$\$SYBASE_ASE</i>
PRIM_SYBASE	<i>/opt/sybase</i>
PRIM_SERVER	MONEY1
PRIM_HOSTNAME	FN1
PRIM_CONSOLE_LOG	<i>\$\$PRIM_SYBASE/\$\$SYBASE_ASE/install/MONEY1.cs_log</i>
PRIM_RUNSCRIPT	Name of RUNSERVER file – default to <i>\$\$SYBASE/\$\$SYBASE_ASE/install/RUN_<servername></i>
SYBASE	PRIM_SYBASE for primary and SEC_SYBASE if on the secondary host. Automatically set.
SEC_ASE_HOME	Default <i>\$\$SYBASE/\$\$SYBASE_ASE</i>
SEC_SYBASE	<i>/opt/sybase</i>
SEC_SERVER	PERSONNEL1
SEC_HOSTNAME	HUM1

Variable	Setting
ISQL	Default <code>\$\$SYBASE/\$SYBASE_OCS/bin/isql</code>
SEC_CONSOLE_LOG	<code>\$\$PRIM_SYBASE/\$SYBASE_ASE/install/PERSONNEL1.cs_log</code>
	<p>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 <i>MONEY1.sh</i>, enter:</p> <pre>chmod 700 MONEY1.sh</pre> <p>6 Distribute the script to the secondary node. For example, to distribute the file to the secondary node HUM1:</p> <pre>rcp /etc/cmcluster/MONEY1/MONEY1.sh HUM1:/etc/cmcluster/MONEY1/MONEY1.sh</pre> <p>7 Repeat the above steps for the secondary companion.</p> <p>The secondary companion package script uses values for PRIM_SERVER, PRIM_HOSTNAME, PRIM_SYBASE, SEC_SERVER, SEC_HOSTNAME, and SEC_SYBASE that are the opposite of the primary companion package script. Table 7-2 shows values for <i>PERSONNEL1.sh</i>.</p>

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

Variable	Setting
ASE_12_0	yes
ASE_HAFAILOVER	yes
BASIC_FAILOVER	yes
PACKAGE_NAME	PERSONNEL1
MONITOR_INTERVAL	5
SHUTDOWN_TIMEOUT	60
RECOVERY_TIMEOUT	300
SYBASE_ASE	ASE-15_0
SYBASE_OCS	OCS-15_0
HALOGIN	“SA”
HAPASSWD	“Odd2Think
PRIM_SYBASE	<code>/opt/sybase</code>
PRIM_SERVER	PERSONNEL1
PRIM_HOSTNAME	HUM1
PRIM_CONSOLE_LOG	<code>\$\$PRIM_SYBASE/\$SYBASE_ASE/install/MONEY1.cs_log</code>
PRIM_RUNSCRIPT	Name of RUNSERVER file – default to <code>\$\$SYBASE/\$SYBASE_ASE/install/RUN_<servername></code>

Variable	Setting
SEC_SYBASE	/opt/sybase
SEC_SERVER	MONEY1
EC_HOSTNAME	FN1
SEC_CONSOLE_LOG	\$PRIM_SYBASE/\$SYBASE_ASE/install/PERSONNEL1.cs_log

Creating 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 this syntax:

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

Where

- *package_name* – the name of the directory you created
- *companion_name* – the name of the companion you are configuring

- 2 Edit the package control script to reflect your cluster environment:

- a Define the volume groups that are used by this companion server package:

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

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

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

- b 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 the primary companion 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]=""
```

- c 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 “Editing the *ASE_HA.sh* script” on page 53). 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
}
```

- d 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 “Editing the *ASE_HA.sh* script” on page 53). 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, enter:

```

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
}

```

- e 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, to configure monitoring for primary companion MONEY1, enter:

```

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

```

- f 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

```

- g Repeat these steps for the secondary companion.

Verifying and distributing the configuration

- 1 Use the cmquerycl utility to create the cluster configuration information file:

```
cmquerycl -n primary_node_name -n secondary_node_name -v -C  
/etc/cmcluster/cmclconfig.ascii
```

primary_node_name – the host name for the primary node.

secondary_node_name – the host name for the secondary node.

In the *cmclconfig.ascii* file, change the *max_configured_packages* to 2.

- 2 Use the *cmcheckconf* utility to verify that the package configuration file is correct. *cmcheckconf* uses this syntax:

```
cmcheckconf -C /etc/cmcluster/cmclconfig.ascii -P  
/etc/cmcluster/package_name/primary_companion_name.ascii -P  
/etc/cmcluster/secondary_package_name/secondary_companion_n  
ame.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
```

- 3 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:

```
/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, *secondary_package_name* is the name of the secondary directory you created, and *secondary_companion_name* is the name of the secondary companion. For example, to generate a binary configuration file for MONEY1, enter:

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

c To deactivate the cluster lock volume group:

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

Note The cluster lock volume group can 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. Deactivate the cluster lock volume group on the configuration node after `cmapplyconf` is executed.

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

Starting the primary and secondary companions

Start the companion cluster, including its packages, using this syntax as “root” on one companion node:

```
cmruncl -v
```

To view the information on the companion cluster, enter:

```
cmviewcl -v
```

To add packages to an individual node, enter:

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

For example:

```
/usr/cmrunpkg -n FN1 MONEY1
```

Configuring companion servers for failover

This section discusses how to configure the Adaptive Servers as primary and secondary companions in a high availability system.

Running *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 prevents 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 the resources for only half the number of potential user logins necessary. Instead, configure both MONEY1 and PERSONNEL1 for 500 user logins.

sp_companion do_advisory 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.

Creating an asymmetric companion configuration

To configure the primary companion asymmetrically, issue this command from the secondary companion:

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

- *primary_server_name* is the name of the primary Adaptive Server as defined in the *interfaces* file entry and in *sys.servers*.
- *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, NULL, 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
```

Creating the symmetric configuration

After you configure your companion 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. See “Creating an asymmetric companion 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 “Creating an asymmetric companion configuration” on page 64. Issue the following command from the `MONEY1` server:

```
sp_companion 'PERSONNEL1', configure, 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 includes information about using Sybase 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. A failback is a planned event. To fail back to the primary companion:

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

Note The high availability system automatically restarts the primary companion.

- 2 Issue the following from the secondary companion, where *primary_companion_name* is the name of primary companion server:
`sp_companion primary_companion_name, prepare_failback`
- 3 From the primary companion, issue, where *secondary_companion_name* is the name of the secondary companion server:
`sp_companion secondary_companion_name, 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 until you issue `sp_companion resume`. If you do try to reconnect them after issuing `sp_companion prepare_failback`, the client stops responding until you issue `sp_companion resume`.

Suspending companion mode

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

- 1 As “root”, issue `cmhaltserv` to disable the monitoring process so it does not trigger a fail over when you shut down the companion server, where *primary_package_name* is the name of the primary package, which is also the same as the name of the primary companion:

```
cmhaltserv -v primary_package_name
```

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

```
sp_companion primary_server_name, suspend
```

You can now shut down the primary companion as necessary and it does not fail over 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 the primary companion, where *primary_package_name* is the name of the primary package, which is the same as the name of the primary companion server:

```
cmhaltpkg primary_package_name
```

- 2 As “root,” issue `cmmodpkg` and `cmrunpkg` from the primary companion to run the package which restarts the primary companion, where *primary_package_name* is the name of the primary package, which is the same as the name of the primary companion server:

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

Dropping companion mode

To drop companion mode, issue:

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

Dropping companion mode is irreversible; to reestablish failover, you must reconfigure the Adaptive Server companion servers. However, the nodes upon which the Adaptive Servers are running are still monitored by the high availability system.

If you drop companion mode while the monitor script is running, the script continues to monitor the server. If you plan to shut down the server and do not want the node to fail over, kill the monitor process by issuing:

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

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 stopped responding, 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 must recover manually. 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 error 18750:

- 1 Shut down both the primary and the secondary companions by halting both their packages.
- 2 Restart 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
```

Make sure that this command executes successfully.

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

```
sp_companion secondary_companion, resume
```

Recovering from a failed *prepare_failback*

During a fail back, if *prepare_failback* executed successfully on the secondary companion but the primary companion does not start, perform the following to roll back 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 start, and correct the problems.
- 2 If the package for the primary companion is running on the primary node, halt the package.
- 3 Log in 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 fail back to normal companion mode, issue *sp_companion...prepare_failback*.

Location of error logs

Sybase Failover and HP MC/ServiceGuard includes the following error logs:

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

For output from the companion start, stop, and monitor scripts, search for “SYBASE HA”.

For MC/ServiceGuard package failure output, search for the string “ERROR”.

- *\$PRIM_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 for the primary.
- *SEC_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 for the secondary.

Upgrading Adaptive Server

To upgrade an Adaptive Server in a high availability configuration, you must temporarily break the companionship between the primary and secondary companion, and disable monitoring of the Adaptive Server packages. You can shut down or restart either Adaptive Server companion independently during the upgrade process without triggering unexpected failovers by the MC/ServiceGuard cluster.

Note Unless you specify a different node with the `-n node_name` parameter, the MC/ServiceGuard commands for starting packages assume you are issuing the command for the node on which the command is performed. Before you issue these commands, use the the MC/ServiceGuard documentation to verify the packages are started on the correct node.

You cannot add, delete, or modify any databases, objects, users, or logins during the upgrade process. Making these changes after the companionship is dropped and before it is reestablished may cause the upgrade to fail or destabilize the cluster by causing inconsistencies between servers.

❖ Stopping the monitoring service and dropping companionship

- 1 Drop the companionship. From the secondary companion, issue:

```
sp_companion primary_server_name, "drop"
```

- 2 From the primary companion, issue:

```
sp_companion secondary_server_name, "drop"
```

- 3 Ensure that both nodes are in single-server mode by issuing, on each node:

```
sp_companion
```

If the companions are in single-server mode, they return:

```
Server 'server_name' is not cluster configured.  
Server 'server_name' is currently in 'Single server'  
mode.
```

- 4 On Adaptive Server packages on all nodes in the cluster stop the monitoring service. As root, issue:

```
cmhaltserv -v primary_package_name
```

❖ Upgrading Adaptive Server

- 1 On each node, disable high availability:

```
sp_configure 'enable HA', 0
```

Restart Adaptive Server for this change to take effect.

- 2 Follow the instructions in the installation guide to upgrade each server.
- 3 On all nodes, reenable high availability:

```
sp_configure 'enable HA', 1
```

Restart Adaptive Server for this change to take effect.

- 4 On the upgraded servers, reinstall the *installmaster* and *installhasvss* scripts. See “Reinstalling installmaster” and “Rerunning installhasvss” on page 224. When you reinstall *installmaster*, you must reinstall *installhasvss*.
- 5 Ensure that permissions are set correctly for the *sybha* binary and *sybhausers* file.

As root, issue these commands from `$$SYBASE/$$SYBASE_ASE/bin`:

```
chown root sybha
chmod 4550 sybha
```

As root, perform these tasks from `$$SYBASE/$$SYBASE_ASE/install`:

- 1 Ensure that the *sybase* user is included in the *sybhauser* file.
- 2 Issue:

```
chown root sybhauser
chmod 600 sybhauser
```

- 6 Changes are properly reflected in package properties or any files related to high availability in the new installation (for example, *PRIM_SYBASE*, *PRIM_RUNSCRIPT*, *PRIM_CONSOLE_LOG*, and so on) in the `/etc/cmcluster/package_name/package_name.sh` script.

❖ **Reestablishing companionship and resuming monitoring**

- 1 On each node, manually restart Adaptive Server.
- 2 As root, from the primary node, restart the monitoring service:

```
cmmodpkg -e primary_package_name
```

- 3 Verify you have performed the prerequisite steps for establishing companionship described in “Configuring companion servers for failover” on page 64.
- 4 Reestablish companionship between the servers.

For asymmetric configurations, issue these commands on the secondary server; for symmetric configurations, issue these commands on both companions:

```
dbcc traceon (2209)
sp_companion primary_server_name, configure, NULL, user_name, password
```

If user databases exist on the secondary server, you may see one or more warning messages, which you can safely ignore:

```
Msg 18739, Level 16, State 1:
Server 'svr2', Procedure 'sp_hacmpcfgvrfy', Line 102:
Database 'svr2_db1': a user database exists. Drop this
database and retry the configuration again.
```

- 5 As root, take the packages offline:

```
dbcc traceoff(2209)
cmhaltpkg "primary_package_name"
cmhaltpkg "secondary_package_name"
```

- 6 Restart the packages on their appropriate nodes. As root on the primary node, issue:

```
cmrunpkg -v "primary_package_name"
```

As root on the secondary node, enter:

```
cmrunpkg -v "secondary_package_name"
```

- 7 Run `sp_companion` to verify that the system is properly configured for failover. To verify that failover and failback works for the companion servers, relocate the primary package to the secondary node.

Configuring Adaptive Server for Failover on IBM AIX HACMP

This chapter contains the information for configuring Adaptive Server for Failover on IBM AIX HACMP.

Topic	Page
Hardware and operating system requirements	75
Preparing Adaptive Server to work with high availability	77
Configuring the IBM AIX subsystem for Sybase Failover	83
Configuring companion servers for failover	91
Administering Sybase Failover	95
Troubleshooting fail over for HACMP for AIX	99
Upgrading Adaptive Server	102

Hardware and operating system requirements

High availability requires the following hardware and system components:

- Two homogenous, network systems with similar configurations in terms of resources such as CPU, memory, and so on
- High availability system package and the associated hardware
- Devices that are accessible to both nodes
- A logical volume manager (LVM) to maintain unique device path names across cluster nodes
- Third-party mirroring, for media failure protection

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

Requirements for running Failover on IBM AIX HACMP

Configuring for high availability on IBM High Availability Cluster Multiprocessing (HACMP) requires:

- Two hardware-compatible nodes running HACMP for AIX, Version 5.2, that are configured in the same cluster.
- Each node must have three IP addresses, one for service, one for start, 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. Both nodes must 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 system.

Sybase also recommends that you identify the following resources in advance:

- 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 5.2, the entire node fails over, not only 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 timeout. Because of this, when you use Sybase 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 high availability

This section discusses how to prepare Adaptive Server for a high availability configuration.

Installing Adaptive Servers

Before you install Adaptive Server, start the HACMP services on the same node on which you are installing Adaptive Server. The HACMP node must be running on its service IP address, not the start 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 systems cannot be the same for each companion. This is to prevent the file systems from overwriting each other during a device fail over. 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 systems, the name of the file systems must be the same. For example, both the primary and the secondary companion can 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,” which means make the volume group active and accessible on this node.

If you are creating an asymmetric configuration, you can use any device (shared or local) for the database device of the secondary companion. 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,” which means make the volume group active and accessible on this node.

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 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.

Adding 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. The server entry in the *interfaces* file must use the same network name that is specified in *syssservers*. 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

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 the 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 FN1 4100
    query tcp ether FN1 4100
    hafailover PERSONNEL1
```

Use *dsedit* to add entries to the *interfaces* file. If the interfaces entries already exist, modify them to work for fail over.

See the *Utility Guide* for information about `dsedit`.

Setting 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, creating 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` use `/usr/u/sybase2` as its release directory, `$SYBASE` must point to the same path.

Both `MONEY1` and `PERSONNEL1` uses `/sybase`, which is established 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`.

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

`sybha` executable

The Adaptive Server High Availability Basic Services library calls `sybha`, which 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` 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”

- 1 Add a new group named *sybhagr*. You can either add this group to the */etc/group* file, or you can add it to your NIS maps. Add the *sybase* user (the user that owns the *\$SYBASE* directory) to this group. When the server is started, the *sybase* user runs the data server. If you have multiple servers running, and different users own the *\$SYBASE* directory for each of them, each user must be added to the group.
- 2 Add “*sybase*” user to the group “*hacmp*”. This group is created as part of HACMP Cluster Software installation.
- 3 Change to the *\$SYBASE/\$SYBASE_ASE/bin* directory.
- 4 Change the ownership of *sybha* to “*root*”:

```
chown root sybha
```
- 5 Change the group for the *sybha* program to *sybhagr*:

```
chgrp sybhagr sybha
```
- 6 Modify the file permissions for *sybha* to 4550:

```
chmod 4550 sybha
```
- 7 Change to the *\$SYBASE/\$SYBASE_ASE/install* directory:

```
cd $SYBASE/$SYBASE_ASE/install
```
- 8 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
```
- 9 Change the ownership of *sybhauser* to “*root*”:

```
chown root sybhauser
```
- 10 Modify the file permissions for *sybhauser*:

```
chmod 600 sybhauser
```

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. This configuration is static, so you must restart 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 Chapter 5, “Setting Configuration Parameters,” in the *System Administration Guide: Volume 1*.

Adding thresholds to the master log

If you have not already done so, 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 *Reference Manual: Procedures* 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 Perform both steps on the primary and secondary servers, then restart both servers to allow the static configuration parameters to take effect.

Creating a 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 proxy databases used by failover), they are automatically created on the master device. However, adding user databases to the master device 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, use `disk_init` to create a new device. Use `sp_diskdefault` to specify the new device as the default.

For example, to add a new default device named `money1_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 *Reference Manual: Procedures* for more information about `disk init` and `sp_diskdefault`.

Adding a local server to `syssservers`

Use `sp_addserver`, to add the local server as the local server to `syssservers` 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 restart Adaptive Server for this change to take effect.

Adding a secondary companion to `syssservers`

Add the secondary companion as a remote server in `syssservers`:

```
sp_addserver server_name
```

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

Running `installhasvss` script

Note You must perform the tasks described in “Adding entries for both Adaptive Servers to the `interfaces` file” on page 78, before running `installhasvss`. If you run `installhasvss` before performing these tasks, you must then re-run `installmaster` to reinstall all the system stored procedures.

The `installhasvss` script:

- Installs the stored procedures required for fail over (for example, `sp_companion`)
- Installs the `SYB_HACMP` server in `syssservers`

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

installhasvss is located in the `$$SYBASE/$SYBASE_ASE/scripts` directory. To execute the 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.

Assigning *ha_role* to System Administrator

To run `sp_companion` you must have the `ha_role` on both Adaptive Servers. 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.

Configuring the IBM AIX subsystem for Sybase Failover

This section discusses how to configure IBM AIX for failover.

Modifying 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. 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 depend on whether the script is for the primary or secondary companion. Each node must have 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 must have read, write, and execute permissions for “root”. The easiest way to do this is to first modify both scripts on the same node, copy both the scripts to the other node, 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 enter the following, where *servername* is the Adaptive Server to be monitored.:

```
RUNHA_<servername>.sh
```

- 3 You must edit the *RUNHA_<servername>.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_<servername>.sh* waits between checks to see if the data server process is alive.
 - *RECOVERY_TIMEOUT* – the maximum amount of time, in seconds, the high availability system waits before determining that the companion did not start. Set this number high enough for a loaded companion to restart. *RECOVERY_TIMEOUT* is also used as the maximum amount of time the subsystem waits for the failover and failback to complete.

- *SHUTDOWN_TIMEOUT* – the maximum amount of time the high availability system 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. By default this is 360 seconds. If your server takes longer than this to start, 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 has stopped responding. By default, *RESPONSE_TIMEOUT* is set to 999999.
- *ASE_FAILOVER* – set to:
 - yes – monitors the companion server for stopped or dead processes and stops HACMP services on this node so the devices fail over to the secondary node. You must also run sp_companion configure on the server.
 - no – do not bring down the HACMP subsystem on this node even if the primary companion fails over. This setting is useful if you must bring down a companion for maintenance or reconfigure.

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

Warning! Set *ASE_FAILOVER* to “yes” only if both servers are running Adaptive Server Version 12.0 or later. For version earlier than 12.0 set *ASE_FAILOVER* to “no”.

- *BASIC_FAILOVER* – set to:

- *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 Failover (that is, they have `enable ha` set to 1), if 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 the primary companion on the secondary node.
- *no* – do not revert to mode 0 failover even if Sybase Failover criteria is not met. That is, if `BASIC_FAILOVER` is set to *no*, failover neither at the node nor the companion level.
- *retry* – the number of times the HACMP subsystem attempts to restart 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 restart itself if it ever goes down. The default is 0, which means that the companion does not restart on the same node if it goes down.
- `SYBASE_ASE` – the installation directory of Sybase Adaptive Server products.
- `SYBASE_OCS` – the installation directory of Sybase Open Client products.
- `PRIM_SERVER` – the name of the primary companion.
- `SEC_SERVER` – the name of the secondary companion.
- `PRIM_HOST` – the name of the primary host as returned by `hostname`.
- `SEC_HOST` – the name of the secondary host as returned by the `hostname`.

Warning! You must set `PRIM_HOST` and `SEC_HOST` to the string as returned by `hostname` command, otherwise, failover or failback may not happen properly.

- `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*.
 - *SEC_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/\$SYBASE_ASE/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/\$SYBASE_ASE/install*.
- 4 Edit the script for the primary companion.
 - 5 Edit the script for the secondary companion. These values differ depending on whether you are using an asymmetric or a symmetric configuration.

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.

Configuring resource groups in HACMP

Note You can perform the steps described in this section from the command line or through the SMIT configuration utility. See your IBM documentation for information about starting and using the SMIT utility. This document explains how to use SMIT to configure resource groups. The steps below assume you are using HACMP5 2. If you are using a later version, see your operating system documentation. See the IBM HACMP for AIX documentation for information about configuring resource groups from the command line.

Shut down the cluster services on both nodes in graceful mode, then log in to start IP addresses of the primary node as “root” and perform these tasks:

- 1 Start SMIT from the command line - smit hacmp.
- 2 From Extended Configuration menu:
 - Select Extended Resource Configuration.
 - Select HACMP Extended Resource Group Configuration.
 - 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 Select the appropriate start, fallover, and fallback policies when defining the resource groups:

For the Resource Group for the primary companion:

Field name	Enter
Resource Group Name	[<resgrp1>]
Participating Nodes (Default Node Priority)	[<primary_node> <secondary_node>]
Start Policy	“Online On Home Node Only”
Fallover Policy	“Fallover To Next Priority Node In The List”
Fallback Policy	“Fallback To Higher Priority Node In The List”

Click OK.

For the Resource Group for the secondary companion, Asymmetric Failover Configuration:

Field name	Enter
Resource Group Name	[<resgrp2>]
Participating Nodes (Default Node Priority)	[<secondary_node>]
Start Policy	“Online On Home Node Only”
Fallover Policy	“Fallover To Next Priority Node In The List”
Fallback Policy	“Fallback To Higher Priority Node In The List”

Click OK.

For the Resource Group for the secondary companion, as a Symmetric Failover Configuration:

Field name	Enter
Resource Group Name	[<resgrp2>]
Participating Nodes (Default Node Priority)	[<secondary_node> <primary_node>]
Start Policy	“Online On Home Node Only”
Fallover Policy	“Fallover To Next Priority Node In The List”
Fallback Policy	“Fallback To Higher Priority Node In The List”

Click OK.

Note The start and stop policies can also be “Bring Offline (On Error Node Only)” and “Never Fallback” for secondary companion in Asymmetric HA configuration.

- 4 Define the primary and secondary companions as application servers in HACMP. From Extended Configuration menu:
 - Select Extended Resource Configuration
 - Select HACMP Extended Resources Configuration
 - Select Configure HACMP Applications
 - Select Configure HACMP Application Servers
 - Select Add an Application Server for defining a new Application Server or Change/Show an Application Server if you are changing an existing Application Server, then enter these values:

Field name	Enter
Server Name	the corresponding companion server name for this Application Server being created.
Start Scrip	the full path name of the corresponding script you created in section “Modifying the ASE_HA.sh script” on page 84, for this companion server. Enter “monitor” as argument for the start script.
Stop Script	the full path name of the corresponding script you created in section “Modifying the ASE_HA.sh script” on page 84, for this companion server. Enter “failover” as argument for the stop script corresponding to primary and symmetric secondary companion servers. Enter “stop” as argument for asymmetric secondary companion server.
Application Monitor Name(s)	leave blank

Click OK when information is completed.

- As examples,

For the primary companion server MONEY1:

Field name	Enter
Server Name	[MONEY1].
Start Scrip	[/usr/sbin/cluster/events/RUN_MONEY1.sh monitor]
Stop Script	[/usr/sbin/cluster/events/RUN_MONEY1.sh failover]
Application Monitor Name(s)	[]

For secondary companion server PERSONNEL1 in Symmetric HA configuration:

Server Name	[PERSONNEL1]
Start Script	[/usr/sbin/cluster/events/RUN_PERSONNEL1.sh monitor]
Stop Script	[/usr/sbin/cluster/events/RUN_PERSONNEL1.sh failover]
Application Monitor Name(s)	[]

For secondary companion server PERSONNEL1 in Asymmetric HA configuration:

Server Name	[PERSONNEL1]
Start Script	[/usr/sbin/cluster/events/RUN_PERSONNEL1.sh monitor]
Stop Script	[/usr/sbin/cluster/events/RUN_PERSONNEL1.sh stop]
Application Monitor Name(s)	[]

Only the secondary node is listed in the Participating Nodes field for secondary resource group in Asymmetric HA configuration, whereas both nodes are listed in reverse order, when compared to primary resource group, in the Symmetric HA configuration.

- 5 Configure each of the resource groups you defined for the Application Servers. From Extended Configuration menu:
 - Select Extended Resource Configuration.
 - Select HACMP Extended Resource Group Configuration.
 - Select Changes/Show Resources and Attributes for a Resource Group.
 - On the Single Select List, select the resource group you want to configure.
 - In the Application Server field, enter corresponding application server name defined in step (4) for the resource group (such as, primary companion name for primary resource group and secondary companion server name for the secondary resource group).

Enter information in all the required fields, such as IP Label, Volume Groups, and File systems. Repeat this step for each of the companions.
- 6 Synchronize the cluster resources. Using `smit` on the node where you performed steps 1 through 5, go to Extended configuration Screen and select “Extended Verification and Synchronization”. 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 restart both nodes before performing the synchronization. Verify that the synchronization does not produce any errors.

Configuring companion servers for failover

This section contains instructions for configuring the Adaptive Servers as primary and secondary companions in a high availability system.

Running `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 has the resources for only half the number of potential user logins necessary. Instead, configure both MONEY1 and PERSONNEL1 for 500 user logins.

`sp_companion do_advisory` 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.

Creating an asymmetric companion configuration

To configure the primary companion asymmetrically, issue this command from the secondary companion:

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

- `primary_server_name` is the name of the primary Adaptive Server as defined in the `interfaces` file entry and in `sys.servers`.
- `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, NULL, 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

```

Creating the symmetric configuration

After you configure your companion 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. See “Creating an asymmetric companion 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 “Creating an asymmetric companion configuration” on page 92. Issue the following command from the MONEY1 server:

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

Starting the 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 system.

Note Before monitoring begins on the primary companion, make sure that there is no need to shut down 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 restart the server using the steps described here.

To start the primary companion as a resource monitored for fail over:

- 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_<servername>.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 start the secondary companion.

Administering Sybase Failover

This section includes information about using Sybase Failover. The steps below assume you are using HACMP5 2. If you are using a later version, see your operating system documentation.

Failing back to the primary node

Failback occurs automatically on HACMP. When you start HACMP on the primary node, the `stop_server` event on the secondary node triggers the monitoring script to execute `sp_companion 'prepare_failback'`.

To failback to the primary node, make sure that the secondary companion is in secondary failover mode, and start 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 the secondary node is running and the secondary companion is running in secondary failover mode. If the secondary companion or secondary node is not up and running, do not start 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.

Failing back manually

Note If the automatic failback fails, 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 Stop the HACMP subsystem with the takeover mode on the primary node (See your IBM documentation for information). This shuts down the primary companion and fails over its resources to the secondary companion.
- 2 Shut down and restart the secondary companion.
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 **HA_LOGIN**, as specified in *RUNHA_<servername>.sh*, to the secondary companion through **isql** and verify 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 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 until you issue `sp_companion resume`. If you do try to reconnect them after issuing `sp_companion prepare_failback`, the client stops responding 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 companions does not occur, and the primary companion cannot fail over to the secondary companion. However, suspended mode is useful for performing such maintenance tasks as changing configuration parameters.

- 1 To move to suspended mode, issue:

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

- 2 Kill the monitoring process so it does not trigger a fail over when the companion server goes down. As “root”, enter:

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

- 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 does not fail over.

Restarting companion during suspended mode

Use the start-up script in `$SYBASE/$SYBASE_ASE/install` to restart the primary companion without monitoring it:

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

If you use this script to start a companion server, it does not fail over when the server goes down, even if it is configured to do so. Use this method only if you are performing maintenance and you do not want the server databases to be accessible while 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 from failover mode

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

- 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 irreversible; you must reconfigure the Adaptive Servers companion servers to retain the failover functionality. However, the nodes upon which the Adaptive Servers are running are still monitored by the high availability system.

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 stopped instances. If you plan to shut down the server and do not want the node to fail over, 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 attempts to restart the companion from either the primary or secondary node, depending on your settings for *RETRY* and *BASIC_FAILOVER*.

Troubleshooting fail over for 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 the failure of an *sp_companion* 'prepare_failback' command on the secondary companion. You can determine whether this happened by examining the HACMP log (located in */tmp/hacmp.out*) on the secondary node. To recover:

- 1 Shut down both the primary and the secondary companions.
- 2 Restart 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 sysdatabases 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
```

Make sure that this command executes successfully.

- 7 Restart the HACMP services on the primary node.

Recovering from a failed *prepare_failback*

During failback, if *prepare_failback* executes successfully on the secondary companion but the primary companion fails to start, perform the following to roll back, 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 start, and correct the problems.
- 2 Stop the HACMP services on the primary node with takeover.
- 3 Log in 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 failover mode.
- 4 Restart HACMP on the primary node.

Location of error logs

Sybase Failover includes the following logs, which logs may be helpful for investigating and diagnosing errors encountered during failover:

- */tmp/hacmp.out* – contains output of the HACMP activities, as well as the output from the *RUNHA_<servername>.sh* monitoring script. For general HACMP failure, search for the string “ERROR”. For output of the *RUNHA_<servername>.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 perform a Recover From Script Failure before continuing.

If a node does not include a sufficient amount of space in a particular file system, HACMP stops responding in the middle of fail over or fail back 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_<servername>.sh* monitoring script. This error log includes the Adaptive Server information from the most recent execution of the *RUNHA_<servername>.sh* script.

Upgrading Adaptive Server

To upgrade an Adaptive Server in a high availability configuration, you must temporarily break the companionship between the primary and secondary companion, and disable monitoring of the Adaptive Server resource groups. You can shutdown or restart either Adaptive Server independently during the upgrade process without triggering unexpected failovers by the HACMP cluster.

Note You cannot add, delete, or modify any databases, objects, users, or logins during the upgrade process. Making these changes after the companionship is dropped and before it is reestablished may cause the upgrade to fail or destabilize the cluster by causing inconsistencies between servers.

❖ Stopping the monitoring service and dropping companionship

- 1 As root, issue these commands to take the resource group offline:

```
dbcc traceoff(2209)
clRGmove -g secondary_resource_group -d -s false
clRGmove -g secondary_resource_group -d -s true
clRGmove -g group_name -d -s false
clRGmove -g group_name -d -s true
```

You may also use SMIT (see your SMIT user documentation).

- 2 On all nodes in the cluster, stop the monitoring service. As root, issue:

```
ps -ef | grep "RUNHA_server_name.sh monitor"
kill -9 pid
```

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

- 3 From the secondary companion, issue:

```
sp_companion primary_server_name, "drop"
```

- 4 (For symmetric configuration) Drop the secondary's companionship from the primary companion:

```
sp_companion secondary_server_name, "drop"
```

- 5 Ensure that both nodes are in single-server mode by issuing, on each node:

```
sp_companion
```

If the companions are in single-server mode, they return:

```
Server 'server_name' is not cluster configured.
```


Server '*server_name*' is currently in 'Single server' mode.

❖ Upgrading Adaptive Server

- 1 On each node, disable high availability:

```
sp_configure 'enable HA', 0
```

Restart Adaptive Server for this change to take effect.

- 2 Follow the instructions in the installation guide to upgrade each server.

- 3 On all nodes, reenables high availability:

```
sp_configure 'enable HA', 1
```

Restart Adaptive Server for the change to take effect.

- 4 On the upgraded servers, reinstall the scripts (*installmaster*, *installhasvss*, *installsecurity*, and so on). See “Reinstalling *installmaster*” and “Rerunning *installhasvss*” on page 224. When you reinstall *installmaster*, you must reinstall *installhasvss*.
- 5 Ensure that permissions are set correctly for the *sybha* binary and *sybhausers* file.

As root, issue these commands from *\$\$SYBASE/\$SYBASE_ASE/bin*:

```
chown root sybha
chgrp sybhagrp sybha
chmod 4550 sybha
```

As root, perform these tasks from *\$\$SYBASE/\$SYBASE_ASE/install*:

- 1 Ensure that the *sybase* user is included in the *sybhauser* file.
- 2 Issue:

```
chown root sybhauser
chmod 600 sybhauser
```

- 6 Verify:

- Changes are properly reflected in resources, Resource Group properties, or any files related to high availability in the new installation (for example, *PRIM_SYBASE_HOME*, *PRIM_RUNSCRIPT*, *PRIM_CONSOLE_LOG*, and so on) in the */usr/sbin/cluster/event/RUNHA_server_name.sh* script.

- You have performed all actions required for establishing companionship described “Preparing Adaptive Server to work with high availability” on page 77 and “Configuring the IBM AIX subsystem for Sybase Failover” on page 83 and the system maintains these changes after the upgrade is complete.

❖ **Reestablishing companionship and resuming package monitoring**

- 1 On each node, manually restart Adaptive Server.
- 2 As root, restore the monitoring service for the cluster by issuing this command, which automatically executes the `RUNHA_server_name.sh` monitoring script:

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

- 3 Verify you have performed the prerequisite steps for establishing companionship described in “Configuring companion servers for failover” on page 91.
- 4 Reestablish companionship between the servers. On the secondary server, issue:

```
dbcc traceon (2209)
sp_companion primary_server_name,configure
```

Note For symmetric configurations, issue this command on both companions.

If the secondary server includes user databases, you may see one or more warning messages, which you can safely ignore:

```
Msg 18739, Level 16, State 1:
Server 'server_name', Procedure 'sp_hacmpcfgvrfy', Line 102:
Database 'database_name': a user database exists. Drop this
database and retry the configuration again.
```

- 5 Restart the resource groups on their appropriate nodes. As root, on the primary node, enter:

```
clRGmove -g group_name -u -s false
```

As root, on the secondary node, enter:

```
clRGmove -g group_name -u -s true
```

- 6 Run `sp_companion` to verify that the system is properly configured for failover. Verify failover and failback.

Active-Active Configuration for Sun Cluster 3.0 and 3.1

This chapter contains the information for configuring Adaptive Server Enterprise on Sun Cluster 3.0 and 3.1 in an active-active setup.

Topic	Page
Hardware and operating system requirements	105
Preparing Adaptive Server for active-active setup	108
Configuring the Sun Cluster subsystem	114
Configuring companion servers for failover	126
Administering Sybase Failover	131
Verifying high availability on Sun Cluster	133
Configuring the resource groups manually	134
Upgrading Adaptive Server	140
Troubleshooting	143

Adaptive Server Enterprise version 15.0 does not support Sun Cluster version 2.2. If you currently have these clusters configured, you must upgrade the respective cluster versions to configure Adaptive Server 15.0 for High Availability on Sun Solaris.

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 host name that 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.

See the documentation for Sun Cluster for more information about requirements, and for installing platform-specific high availability software.

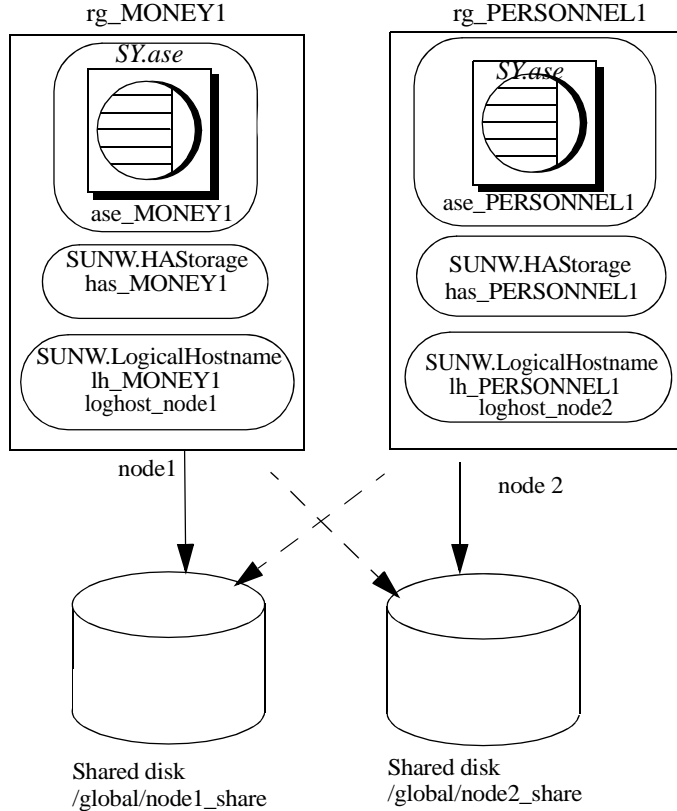
Active-active setup in Sun Cluster

Figure 9-1 on page 107 depicts an active-active configuration in Sun Cluster.

In Sun Cluster, Adaptive Server runs as a data service and is managed by the Sun Clusters Resource Group Manager (RGM). Adaptive Server is associated with a resource group that contains the Adaptive Server resource and all other resources it requires, such as the *SUNW.HAStorage*, *SUNW.HAStoragePlus*, and *SUNW.LogicalHostname*.

SY.ase is the Adaptive Server resource and defines various extension properties for the resources of type *SY.ase*. See “Adaptive Server resource extension properties” on page 120 for more information. See the Sun Cluster documentation for more information on standard resource properties.

Figure 9-1: 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 a 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 *loghost_node1*. The Adaptive Server resource *ase_MONEY1* uses 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 *loghost_node2*. The Adaptive Server resource *ase_PERSONNEL1* uses on *has_PERSONNEL1* and *lh_PERSONNEL1*.

Preparing Adaptive Server for active-active setup

This section discusses how to set up Adaptive Server for active-active high availability.

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. 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 for the primary and secondary companions must include entries for both companions. The server entry in the *interfaces* file must use the network name that is specified in *syssservers*. 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` (lowercase L) parameter of the `scrgadm` command used to add a `SUNW.LogicalHostname` resource when you configure Adaptive Server to work with the Sun Cluster 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
    hfailover PERSONNEL1

PERSONNEL1

    query tcp ether loghost_node2 9866
    master tcp ether loghost_node2 9866
    hfailover 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, modify them to work for fail over.

See the *Utility Guide* for information about `dsedit`.

The value of `$SYBASE` is the same for both companions

`$SYBASE` on both companions must point to the same directory path name. You can accomplish this by:

- 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 uses 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 use `/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`.

Executing *sybha*

The Adaptive Server High Availability Basic 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 you can run *sybha*, 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”:

- 1 Add a new group called `sybhagr` either in the `/etc/group` file or to your NIS maps.
- 2 Add the Sybase user to `sybhagr`. This is the user who owns the `$$SYBASE` directory, and when the server is started, this user runs the data server. If you have multiple servers running, with different users owning the `$$SYBASE` directory, you must add all of these users to `sybhagr`.
- 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 `sybhagr`:

```
chgrp sybhagr 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 proxy databases used by fail over) 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, use `disk init` to create a new device. Use `sp_diskdefault` to specify the new device as the default.

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 *Reference Manual: Commands* for information about `disk init` and see the *Reference Manual: Procedures* for information about `sp_diskdefault`.

Adding the local server to `sys.servers`

Use `sp_addserver` to add the local server in `sys.servers` 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 restart Adaptive Server for this change to take effect.

Adding secondary companion to *syservers*

Add the secondary companion as a remote server in *syservers*:

```
sp_addserver server_name
```

By default, Adaptive Server adds the server with *svid* of 1000. You need not restart Adaptive Server for the change to take effect.

Assigning the *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.

Running *installhasvss* script

Note You must perform the tasks described in “Adding entries for both Adaptive Servers to the interfaces file” on page 108 before you run *installhasvss*. If you run *installhasvss* before performing these tasks, you must re-run *installmaster* to reinstall all of the system stored procedures.

The *installhasvss* script:

- Installs the stored procedures required for fail over (for example, *sp_companion*)
- Installs the *SYB_HACMP* server in *syservers*

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

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 fail over:

- 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. Restart 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 Chapter 5, “Setting Configuration Parameters in the *System Administration Guide: Volume 1*.”

Adding thresholds to the master log

If you have not already done so, 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 *Reference Manual: Procedures* 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 the primary companion for this static parameter to take effect.

Adding user and login for fault monitor

When the high availability agent fault monitor, *ase_monitor*, runs thorough probe, it:

- 1 Connects to the Adaptive Server.
- 2 Creates a temporary 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>
```

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 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 high availability system.
- Installed Adaptive Server and created the required database device files on the shared disk.
- Configured Adaptive Server following the instructions in “Preparing Adaptive Server for active-active setup” on page 108.
- Created `$$SYBASE/SYBASE.sh` and edited the file with the required environment for Adaptive Server.

As the file is executed in the high availability agent scripts, protect the file from unauthorized access and make sure only “root” has read and executable permissions.

- Created `$(SYBASE)/$(SYBASE_ASE)/install/RUN_<Datasever_name>` file. You must specify the Adaptive Server error log with the `-e` option in this file.

If you specify the `-s` option, it must be the same as the Adaptive Server resource property `Datasever_name`.

- Installed `$(SYBASE)/$(SYBASE_ASE)/SC-3_0` properly. This directory must contain all the required files for the Adaptive Server high availability 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` initially contains no files, but eventually contains *Callback_log* and *Monitor_log* files once you create the Adaptive Server resource.

Using the *syscadm* script

Use the *syscadm* script to configure and administer Adaptive Server resource groups and their associated resources in Sun Cluster. You can use *syscadm* to create, remove, or no longer control the Adaptive Server resource group and its resources. 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, specified resources and adds them to the resource group
- Establishes resource dependencies for the 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*.

syscadm works with an input file called *sysc_input_file*, which you edit to provide the correct input values 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. Sybase suggests that you 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 companion.
- Names ending with 2 correspond to the secondary companion.

See “Sample sysc_input_file” on page 117 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 data server name, the Nodelist, and so on.
- Section 2 – enter right-side values for the required entries. For example, if you are using only the 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 need not provide right-side values except to override the defaults.

For example, to edit the file for the Adaptive Server resource name, change this line:

```
ASE_RNAME="ase_${Datasever_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"
```

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 9-1 on page 107:

```
#####
##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-15_0/SC-3_0/etc/ase_login_file"

Dataserver_name2="PERSONNEL1"
Dataserver_login_file2="/sybase/ASE-15_0/SC-3_0/etc/ase_login_file"

#####
## Section2: Must specify right hand side values, if required        ##
#####

# if using Logical Hostname
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="$SYBASE/ASE-15_0/SC-3_0/etc/$RT_NAME"

# Resource Group names
RG_NAME1="rg_$Dataserer_name1"
RG_NAME2="rg_$Dataserer_name2"

# ASE Resource names and space separated extended properties
ASE_RNAME1="ase_$Dataserer_name1"
ASE_RNAME2="ase_$Dataserer_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_$Dataserer_name1"
LOGHOST_RNAME2="lh_$Dataserer_name2"

# HA Storage Resource names
HASTORAGE_RNAME1="has_$Dataserer_name1"
HASTORAGE_RNAME2="has_$Dataserer_name2"

# HA Storage Plus Resource names
HASTORAGE_PLUS_RNAME1="hasp_$Dataserer_name1"
HASTORAGE_PLUS_RNAME2="hasp_$Dataserer_name2"
```

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 unmanages 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`.

Adaptive Server resource extension properties

Table 9-1 summarizes all extension properties for the Adaptive Server resource. Refer to the respective Sun Cluster documentation for more information about resources.

Table 9-1: Extension properties for the SY.ase resource

Property	Default	Description
<i>Sybase_home</i>	None	The home directory of the Adaptive Server installation, and the same as the value for the <code>\$SYBASE</code> environment variable in an Adaptive Server installation. This property is required to create the Adaptive Server resource.
<i>Environment_file</i>	<i>Sybase_home/SYBASE.sh</i>	Absolute path to the environment file where you specify the environment to pass to the Adaptive Server. This file must be available for proper functioning of the high availability agent.
<i>Dataserver_name</i>	None	Name of the Adaptive Server data server. This property is required to create the Adaptive Server resource.
<i>Backup_server_name</i>	None	Name of the Backup Server.
<i>Text_server_name</i>	None	Name of the full-text search server.
<i>Secondary_companion_name</i>	None	Name of the secondary companion server, which is automatically set or un-set by <code>sp_companion</code> commands <code>configure</code> or <code>drop</code> . Reserved for active-active setup. Do not set this property manually.
<i>Dataserver_login_file</i>	<i>Sybase_home/\$SYBASE-ASE/SC-3_0/etc/ase_login_file</i>	Absolute path to a file containing login information for the data server. The file consists of two lines; the first line is the login and password for the System Administrator, the second line is the user login and password for the thorough probe used by the fault monitor program <i>ase_monitor</i> .

Property	Default	Description
<i>Action_file</i>	<i>Sybase_home/\$SYBASE_ASE/SC-3_0/etc/ase_monitor_action</i>	Absolute path to a file that associates error codes with actions to be taken by the fault monitor program <i>ase_monitor</i> .
<i>RUN_server_file</i>	<i>Sybase_home/\$SYBASE_ASE/install/RUN_<Datasever_name></i>	Absolute path to the <i>RUN_SERVER</i> file for the Adaptive Server specified by the property <i>Datasever_name</i> . Do not include environment variables in this file.
<i>Thorough_probe_script</i>	Ignored. Reserved for future use.	Absolute path to a file containing SQL scripts for the fault monitoring program to perform thorough probe.
<i>Monitor_log</i>	<i>Sybase_home/\$SYBASE_ASE/SC-3_0/log/ase_monitor_<Datasever_name>.log</i>	Absolute path to the log file for the fault monitor program, <i>ase_monitor</i> .
<i>Callback_log</i>	<i>Sybase_home/\$SYBASE_ASE/SC-3_0/log/ase_callback_<Datasever_name>.log</i>	Absolute path to the log file used by Adaptive Server high availability agent callback scripts in <i>\$SYBASE/\$SYBASE_ASE/SC-3_0/bin</i> .
<i>Callback_log_max_size</i>	5000000	Maximum size for the callback log file. If the log size exceeds this limit, the callback log is renamed using the current date and time as its extension. Any new log messages are written to the <i>Callback_log</i> .
<i>Monitor_log_max_size</i>	Ignored. Reserved for future use.	
<i>Probe_timeout</i>	30	Time, in seconds, after which the fault monitoring probe times out and registers an error.
<i>Restart_delay</i>	30	Time, in seconds, to delay the next probe after a restart.
<i>Debug_monitor</i>	FALSE	If TRUE, the fault monitor program <i>ase_monitor</i> logs debugging message to the file specified by property <i>Monitor_log</i> .
<i>Debug_callback</i>	FALSE	If TRUE, the Adaptive Server high availability agent scripts log debugging messages to the file specified by property <i>Callback_log</i> .
<i>Connect_cycle_count</i>	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.
<i>Failback_strategy</i>	Ignored. Reserved for future use.	

Configuring Adaptive Server resource groups

To configure Adaptive Server resource groups on Sun Cluster:

- 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 high availability agent. Change the value to point to the installation location of *\$\$SYBASE/\$\$SYBASE_ASE/SC-3_0/bin*.

For example:

```
RT_BASEDIR=/sybase/ASE-15_0/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 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. 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*.

The file consists of two lines; the first line is the login and password of the System Administrator, and the second line is the login and password of *monitor_user*. The fault monitoring program, *ase_monitor*, performs the thorough probe as user *monitor_user*.

```
login_type <tab> login_string
login_type <tab> login_string
```

Valid values for login type are “encrypted” and “normal”. If you set *login_type* to “normal”, the value of the *login_string* is in the form “*login_name/password*”. If you set *login_type* to “encrypted”, the value of *login_string* is the encrypted string you get from the *haisql* utility (located in *\$\$SYBASE/\$\$SYBASE_ASE/bin*). Sybase recommends usage of “encrypted” *login_type* so the sensitive information in the file is well protected. To use *haisql* to generate the encrypted login string:

- a Run *haisql* with no arguments to generate the encrypted string for a given *login_name* and *password*:

```
/$$SYBASE/ASE-12_5/bin/haisql
Enter Username: sa
Enter Password:
```

```
TWAS8n1jSF2gBsvayUlW97861.cyTKaS1YhavBRQ2qKcJwtX.TmFBarGS2K1553WDR7
g8m5vrf86t@K4CU62HEccm4zkeexsP9E=FeuvX
```

- b Copy and then paste the encrypted string to the *ase_login_file* file.

The following is an example of the *ase_login_file* using the “encrypted” login type:

```
encrypted
TWAS8n1jSF2gBsvayUlW97861.cyTKaS1YhavBRQ2qKcJwtX.TmFBarGS2K1553WDR7g8m5
vrf86t@K4CU62HEccm4zkeexsP9E=FeuvX
encrypted
rX2S8n1jSF2gBuD0q=AXEXKCZvzGcK5K3kWnp_P+e4avf=67kYVSzy7+h640@97FSP_dlkH
_oV2Zima5+7tUyHnsm4zmSIHIUnKSTPoTD
```

The following is an example of the *ase_login_file* file using “normal” login type:

```
normal    sa/sa_password
normal    monitor_user/monitor_user_password
```

Note The two lines of the *ase_login_file* may use different login types.

You should protect the *ase_login_file* file with proper access permissions, particularly if you are not using the encrypted login strings. Perform the following to make the file readable only to the root user after editing the file with proper *login_type* and *login_string* values:

```
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 *syscadmin* command, which registers the resource type, creates the resource group, adds resources to the resource group, and establishes resource dependencies.

For example, to run the *syscadmin* script with an input file named *sysc_input_file*, enter:

```
syscadmin -c both -f sysc_input_file
```

For more information on the *syscadmin* script, see “Using the *syscadmin* script” on page 116.

You can also perform these steps manually. See “Configuring the resource groups manually” on page 134 for more information.

- 4 For the primary Adaptive Server resource group, run the `scswitch` command to complete the following tasks:
 - Move the resource group to the “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 and use the same steps as in number 4.

## 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, set the *SUNW.HAStoragePlus* resource properties *GlobalDevicePaths* and *FilesystemMountPoints* as required. If you are using *syscadm*, you can specify values for corresponding entries in the *sysc\_input\_file*. To enable connection, you must set the *SUNW.HAStoragePlus* resource property *AffinityOn* to `TRUE`.

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 you are using *SUNW.HAStoragePlus* resources, you can create Adaptive Server database devices either on a 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, specify all the storage resource names as a comma-separated list.

```
scrgadm -c -j ase_resource_name
-y Resource_dependencies=hasp_resource_name,hasorage_name
```

For example:

```
scrgadm -c -j ase_MONEY1
-y Resource_dependencies=hasp_MONEY1,has_MONEY1
```

See the Sun Cluster documentation for more information about *SUNW.HAStoragePlus* resource type.

## Configuring companion servers for failover

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

### High availability services library within Adaptive Server

You must load the high availability services library for Sun Cluster.

First, verify that the high availability services library is available. Use `isql` to connect to any Adaptive Server:

```
sp_companion "MONEY1", show_cluster
```

You see:

```
The default cluster is: SC.
The current cluster is set to default.
Supported cluster systems for SunOS are:
SC
VCS
```

Set the high availability services library for SC3.0. For example, from `PERSONNEL1`, enter:

```
sp_companion "MONEY1", set_cluster, "SC"
The current cluster is set to SC.
```

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 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, there should be no user login and password used for thorough probe on secondary companion server. If they do exist, both `sp_companion configure` and `sp_companion do_advisory` fail during the user information synchronization process,

To drop the user and login of user probe in the secondary companion server, use `sp_dropuser` and `sp_droplogin`.

## Running *sp\_companion* with *do\_advisory*

### Before initiating *sp\_companion*

Before executing `sp_companion do_advisory` and `sp_companion configure`:

- 1 Disable the monitoring of the secondary Adaptive Server:  

```
scswitch -n -M -j secondary-resource
```
- 2 Drop the user and login for monitor for the secondary Adaptive Server, where *secondary\_probe\_ase* is the login and user created in “Adding user and login for fault monitor” on page 114:

```
sp_dropuser secondary_probe_ase
sp_droplogin secondary_probe_ase
```

After successfully executing `sp_companion do_advisory` and `sp_companion configure` for an asymmetric configuration (see the two sections below for detail), perform the following steps:

- 1 Add the user and login for monitor for secondary Adaptive 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 “Adding user and login for fault monitor” on page 114.
- 2 Enable monitoring of secondary Adaptive 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 the resources for only half the number of potential user logins necessary. Instead, configure both MONEY1 and PERSONNEL1 for 500 user logins.

`sp_companion do_advisory` 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.

## Creating an asymmetric companion configuration

Before you configure for an asymmetric setup, you must first use `scswitch` 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
```

- *primary\_server\_name* is the name of the primary Adaptive Server as defined in the *interfaces* file entry and in *sys.servers*.
- *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, NULL, 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

```

Use `scswitch` to enable the monitoring of the primary resource:

```
scswitch -e -M -j primary_resource
```

To prevent the failover of the secondary companion server in an asymmetric configuration, you must disable the monitoring of the secondary resource after failover.

See “Configuring the asymmetric companion” on page 19 for more information about asymmetric configuration.

## Setting up a symmetric configuration

After you configure the companions for asymmetric fail over, you can set them up 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 asymmetric 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 primary companion to configure it for symmetric configuration. Use a syntax similar to the one for asymmetric configuration, but replace `with_proxydb` by `NULL`. See “Creating an asymmetric companion configuration” on page 128 for a description of the syntax for `sp_companion`.

In the following example, `PERSONNEL1` is the secondary server of `MONEY1`. This is an asymmetric configuration, and will be changed to a symmetric one. Connect to `MONEY1`.

```
sp_companion 'PERSONNEL1', configure, NULL, 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 *scswitch* 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.

---

**Note** Alternatively, you can use this command to fail back the resource group:

```
scswitch -z -h primary_host -g failed_over_group
```

For example, perform a failback to the primary companion MONEY1 on *node1*, issue the following command 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:

```
scswitch -n -M -j primary_resource
```

- 4 Issue the following from the primary companion:

```
sp_companion secondary_companion_name, resume
```

- 5 Enable the monitoring of the primary resource with:

```
scswitch -e -M -j primary_resource
```

- 6 If you are in symmetric mode, use *scswitch* to enable monitoring of the secondary resource.

---

**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 attempt 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 system 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 To stop the high availability system from monitoring the companions. Issue:

```
scswitch -n -M -j primary-resource-name
scswitch -n -M -j secondary-resource-name
```

- 2 To drop companion mode, issue:

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

Dropping companion mode is irreversible; you must reconfigure the Adaptive Server companion servers before they again fail over in a high availability system.

## Verifying high availability on Sun Cluster

To ensure that you have properly configured high availability on the Sun Cluster, perform the verification tests in this section.

The following steps assume that you have configured two Adaptive Server resource groups in asymmetric mode.

- 1 Log in to the primary node for the Adaptive Server resource group.
- 2 Set the Adaptive Server environment variables: SYBASE, SYBASE\_ASE, SYBASE\_OCS, and so on. Use the *Environment\_file* extension property to specify environment variables.
- 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
```
- 5 Switch the primary resource group to the secondary node. This is a simulated fail over.

```
scswitch -z -g primary-resource-group -h secondary-host
```
- 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
```
- 7 Follow the instructions in “Failing back to the primary companion” on page 131 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
```

## 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, you can perform these steps manually, for example to configure, reconfigure, or troubleshoot the Adaptive Server resource groups. Make sure you have properly modified *SY.ase* and *ase\_login\_file* as explained in steps 1 and 2 in “Configuring Adaptive Server resource groups” on page 122.

You must be logged on as “root” to run these Sun Cluster 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-15_0/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 fail over:

```
scrgadm -a -j hstorage_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 host name that can be relocated to the secondary node in case of fail over.

```
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 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-15_0/SC-3_0/etc/ase_login_file \
-x RUN_server_file=/sybase/ASE-15_0/install/RUN_MONEY1
```

Specify any standard resource property values and extension property values.

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 high availability agent fault monitor: *Cheap\_probe\_interval*, *Thorough\_probe\_interval*, *Retry\_count*, and *Retry\_interval*.

For more information about the standard resource properties, see the Sun Cluster documentation. Table 9-1 on page 120 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=hasstorage_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.LogicalHostname* resource if one is added to the resource group.

---

- 8 For the primary Adaptive Server resource group, execute *scswitch* to:

- 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 124 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-15_0/SC-3_0/etc/ase_login_file \
-x RUN_server_file=/sybase/ASE-15_0/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=hasstorage_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
```

## Upgrading Adaptive Server

To upgrade an Adaptive Server in a high availability configuration, you must temporarily break the companionship between the primary and secondary companion, and disable monitoring of the Resource Groups. You can shutdown or restart either Adaptive Server independently during the upgrade process without triggering unexpected failovers by the SunCluster subsystem.

---

**Note** You cannot add, delete, or modify any databases, objects, users, or logins during the upgrade process. Making these changes after the companionship is dropped and before it is reestablished may cause the upgrade to fail or destabilize the cluster by causing inconsistencies between servers.

---

### ❖ Stopping the monitoring service and dropping companionship

- 1 Halt the monitoring service and stop management for the Adaptive Server resource groups on all nodes in the cluster. As root, issue:

```
scswitch -F -g primary_resourcegroup_name
scswitch -u -g secondary_resourcegroup_name
```

- 2 From the secondary companion, issue:

```
sp_companion primary_server_name, "drop"
```

- 3 (For symmetric configuration) Drop the secondary's companionship. Log in to the primary companion and issue:

```
sp_companion secondary_server_name, "drop"
```

- 4 Ensure that both nodes are in single-server mode by issuing on each node:

```
sp_companion
```

If the companions are in single-server mode, they return:

```
Server 'server_name' is not cluster configured.
Server 'server_name' is currently in 'Single server'
mode.
```

The servers are now running on their installation node and may be stopped and started independently without the cluster attempting to failover the resources between nodes.

### ❖ Upgrading Adaptive Server

- 1 On each node, disable high availability:

```
sp_configure 'enable HA', 0
```

Restart Adaptive Server for this change to take effect.

---

**Note** Alternatively, if the companions are shut down, you can edit their server configuration files (*server\_name.cfg*), changing the value of enable HA to zero

---

2 Follow the instructions in the installation guide to upgrade each server.

3 On each node, reenable high availability:

```
sp_configure 'enable HA', 1
```

Restart Adaptive Server for the change to take effect. See the *Configuration Guide*.

4 On the upgraded servers, reinstall the scripts (*installmaster*, *installhasvss*, *installsecurity*, and so on). See “Reinstalling *installmaster*” and “Rerunning *installhasvss*” on page 224. When you reinstall *installmaster*, you must reinstall *installhasvss*.

5 Ensure that permissions are set correctly for the *sybha* binary and *sybhausers* file.

As root, issue these commands from *\$\$SYBASE/\$SYBASE\_ASE/bin*:

```
chown root sybha
chgrp sybhagrp sybha
chmod 4550 sybha
```

As root, perform these tasks from *\$\$SYBASE/\$SYBASE\_ASE/install*:

1 Ensure that the *sybase* user is included in the *sybhauser* file.

2 Issue:

```
chown root sybhauser
chmod 600 sybhauser
```

6 Verify:

- Changes are properly reflected in the resource group and resource properties (for example, *Sybase\_Home*, *runserver* files, *Dataserver\_login\_file*, and so on) in the *\$\$SYBASE* installation location, or any related files related to high availability in the new installation

- You have performed all actions required for establishing companionship described “Preparing Adaptive Server for active-active setup” on page 108 and “Configuring the Sun Cluster subsystem” on page 114 and the system maintains these changes after the upgrade is complete.
- The following files exist and have correct information:

```
$$SYBASE/$$SYBASE_ASE/SC-3_0/etc/hacompanion.server_name
$$SYBASE/$$SYBASE_ASE/SC-3_0/etc/ase_login_file
```

❖ **Reestablishing companionship and resuming resource monitoring**

- 1 On each node, manually restart Adaptive Server.
- 2 As root, issue this command to restore the monitoring service:

```
scswitch -z -g primary_resourcegroup_name -h primary_node
scswitch -z -g secondary_resourcegroup_name -h secondary_node
```

- 3 Verify both the resource groups and the Adaptive Server resources are online on their nodes with the `scstat -g` command (see your Sun documentation).
- 4 Log in to the primary and secondary companions with `isql` to verify both are running.
- 5 Select *SunCluster3.x* as the high availability library (see “High availability services library within Adaptive Server” on page 126).
- 6 Reestablish companionship between the servers (see “Creating an asymmetric companion configuration” on page 128 or “Setting up a symmetric configuration” on page 130):

```
dbcc traceon (2209)
sp_companion primary_server_name,configure
dbcc traceoff(2209)
```

---

**Note** For symmetric configurations, issue this command on both companions.

---

If the secondary server includes user databases, you may see one or more warning messages, which you can safely ignore:

```
Msg 18739, Level 16, State 1:
Server 'server_name', Procedure 'sp_hacmpcfgvrfy', Line 102:
Database 'database_name': a user database exists. Drop this
database and retry the configuration again.
```



- 7 Run `sp_companion` to verify that the system is properly configured for symmetric or asymmetric configuration.

❖ **Verify failover and failback**

- 1 Failover the primary companion by relocating the associated resource group to secondary node. As root:

```
scswitch -z -g primary_resourcegroup_name -h secondary_node
```

Log in to the secondary companion and issue `sp_companion` to verify the failover successfully completed.

- 2 Failback the Adaptive Server by following instructions in section “Failing back to the primary companion” on page 131. Log in to the primary and secondary companions and issue `sp_companion` to verify the failover successfully completed.

## Troubleshooting

This section includes troubleshooting information about common errors.

### Recovering from a failed *prepare\_failback*

During fail back, if `prepare_failback` executes successfully on the secondary companion but the primary companion also fails, 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 failover mode, the cluster is in an inconsistent state, and you must recover manually. The inconsistent state may be caused by `sp_companion 'prepare_failback'` failing on the secondary companion. To recover:

- 1 Issue `sp_helpdb` on the 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")
```

## Preventing the failover of secondary companion

You must disable monitoring after fail over.

## Changing resource and resource group state

When you are performing maintenance on a cluster, bring all resources in the Adaptive Server resource group offline and move Adaptive Server resource group to an unmanaged state:

```
scswitch -F -g primary-resource-group
scswitch -F -g secondary-resource-group
scswitch -u -g primary-resource-group
scswitch -u -g secondary-resource-group
```

## Location of the error logs

Use the information in these logs to debug the high availability system:

- Adaptive Server error log – the location is defined in the RUNSERVER file. For example:

```
/sybase/ASE-15_0/install/MONEY1.log
```

- Adaptive Server high availability 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
```



## Active-Passive Configuration for Sun Cluster 3.0 and 3.1

This chapter discusses configuring Adaptive Server on Sun Cluster in an active-passive setup.

| Topic                                              | Page |
|----------------------------------------------------|------|
| Hardware and operating system requirements         | 148  |
| Active-passive setup in Sun Cluster                | 148  |
| Preparing Adaptive Server for active-passive setup | 152  |
| Configuring the Sun Cluster subsystem              | 159  |
| Configuring the resource group manually            | 169  |
| Upgrading Adaptive Server                          | 172  |
| Location of the error logs                         | 173  |

Adaptive Server Enterprise version 15.0 does not support Sun Cluster version 2.2. If you currently have these clusters configured, you have to upgrade the respective cluster versions to configure Adaptive Server 15.0 for High Availability on Sun Solaris.

An active-passive configuration includes two nodes and a single Adaptive Server. The node that hosts the Adaptive Server under normal conditions is the primary node; the node that can potentially host the Adaptive Server is called the secondary node.

When the Adaptive Server or any of the resources it depends on, such as a disk or the node itself, crashes, the Adaptive Server, along with all required resources, is relocated and restarted on the 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 a planned failback, or until the Adaptive Server on the new primary node fails, causing another fail over.

After failover, all existing client connections are lost. The clients must reestablish 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 performed automatically by using high availability connections and self-referencing the `hafailover` entry in the `interfaces` file. See “Configuring the interfaces file on the client side” on page 156 for 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 host name or floating IP address that can be bound to any of the primary and secondary nodes

For more information about requirements for running Sun Cluster, see the Sun Cluster documentation.

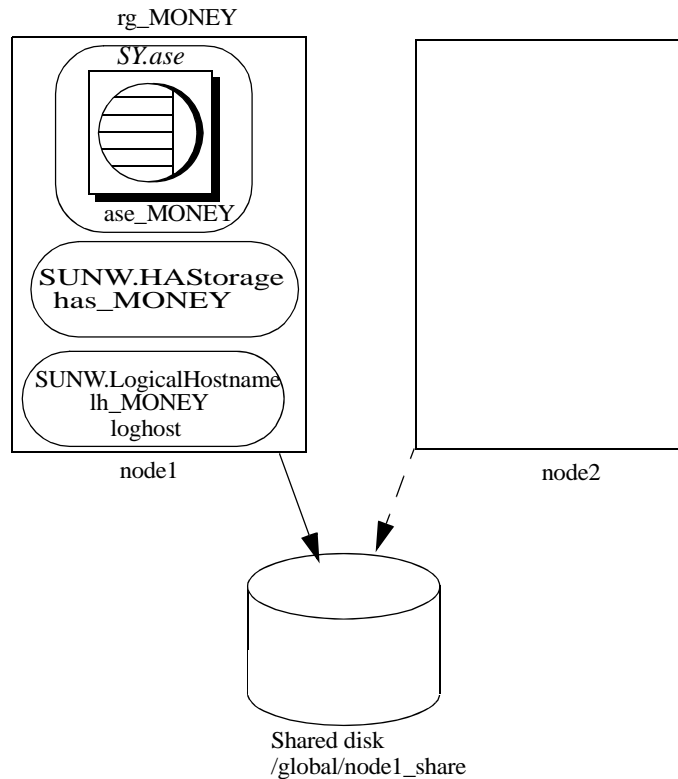
See your hardware and operating system documentation for information about installing platform-specific high availability software.

## Active-passive setup in Sun Cluster

Two-node active-passive configuration is described in Figure 10-1.

In Sun Cluster, Adaptive Server runs as a data service and is managed by the Sun Cluster Resource Group Manager (RGM). Adaptive Server is associated with a resource group that contains the Adaptive Server resource and all other resources it requires, such as the *SUNW.HAStorage*, *SUNW.HAStoragePlus*, and *SUNW.LogicalHostname* resources.

*SY.ase* is the Adaptive Server resource and it defines various extension properties. See “Adaptive Server resource extension properties” on page 120 for more information. See the Sun Cluster documentation for more information on standard resource properties.

**Figure 10-1: Active-passive setup in Sun Cluster before failover**

In Figure 10-1, 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`. `has_MONEY` manages the global file system `/global/node1_share` on the shared disk. The logical host resource is associated with the logical host name or floating IP address `loghost`. `ase_MONEY` uses `has_MONEY` and `lh_MONEY`.

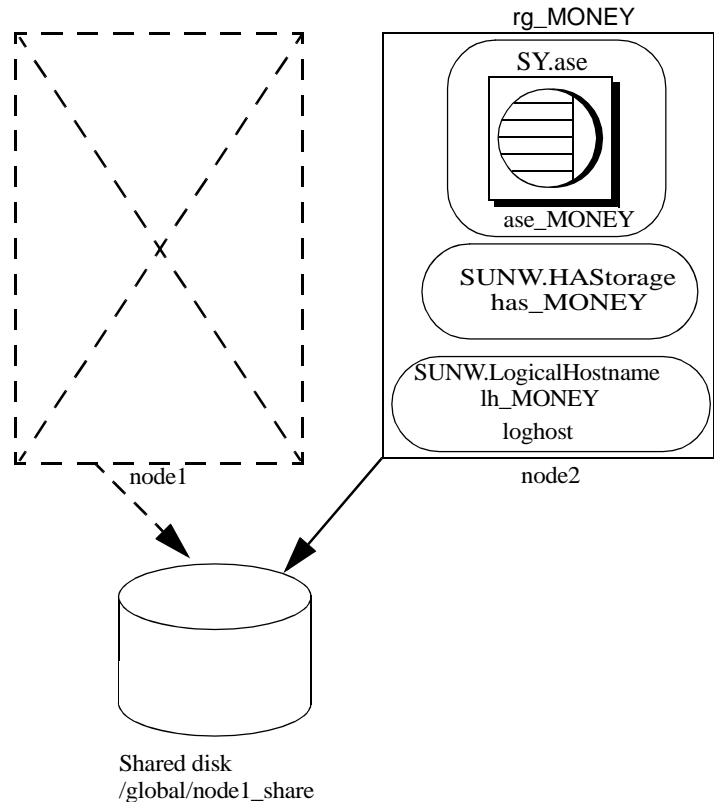
Initially, the Adaptive Server resource group, `rg_MONEY`, is hosted by the primary node, node1, and Adaptive Server MONEY serves its clients through the logical host name `loghost` associated with `lh_MONEY`.

When node1 crashes, the resource group `rg_MONEY` and all of its resources is relocated and restarted on the secondary node as shown in Figure 10-2.



After failover, the Adaptive Server runs on node2 and continues to serve its clients using the same *loghost*.

**Figure 10-2: Active-passive setup on Sun Cluster after failover**



The resource group properties `Pingpong_interval` and `Global_resources_used` may affect fail over. For example, in update 1 of Sun Cluster version 3.0 documentation, 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 stop fail over 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
```

## 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. This is called failback. A 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, then relocating and starting them on the primary node. A failback is not required, but can be performed solely for administration purposes. If fail back is not done, the recovered primary node acts as secondary node until another fail over.

## Clients in an active-passive configuration

When a failover or a failback occurs, all existing client connections are lost. Clients do not see any difference between the two events. However, the client connection failover happens differently, depending on the type of connection the client has established with Adaptive Server. Client connections are either high availability connections or non-high-availability connections.

High availability connections must have the `CS_HAFAILOVER` property set in the connection handle, and the `hafailover` entry in the *interfaces* file. For clients that use the high availability connection, fail over is transparent; the broken connections are automatically reestablished when the Adaptive Server restarts on the secondary node. However, the client must resubmit any uncommitted transactions.

non-high-availability connections do not reconnect automatically; clients must first reestablish their connections to Adaptive Server, then resubmit uncommitted transactions.

For more information see “Configuring the interfaces file on the client side” on page 156.

## Preparing Adaptive Server for active-passive setup

This section discusses how to set up Adaptive Server for active-passive high availability.

## Installing Adaptive Server

You can install Adaptive Server on a global file system or on the local file systems of 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 one 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 fail over.

---

**Note** Do not install *\$\$SYBASE* on a failover file system managed by a *SUNW.HAStoragePlus* resource.

---

### Installing on local file system

If you install Adaptive Server on a local file system:

- The installation directory *\$\$SYBASE* must use the same directory path on the primary and secondary nodes. If different nodes use the *\$\$SYBASE* directory in different locations, create a directory with the same path on the primary and secondary nodes that acts as a symbolic link to the respective actual *\$\$SYBASE* release directory paths.

For example, if the directory on node1 is */usr/sybase1* and on node2 is */usr/sybase2*, 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 the *etc* and *bin* directories, must be consistent.
  - You must apply upgrades and patches consistently.

- Various log files are created on both 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 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

Use the *SYBASE.sh* file to specify the environment to pass to the Adaptive Server. Protect *SYBASE.sh* from unauthorized access; make sure only “root” has read and execute permissions.

The high availability agent looks for the file in *\$\$SYBASE* or as specified in the Adaptive Server resource property *Environment\_file*. The high availability agent may not behave as expected if *SYBASE.sh* is not available.

---

**Note** *SYBASE.csh* file is not supported.

---

## Running the SySam license manager in the cluster

You must run the *SySam* license manager on the primary and secondary nodes in the cluster. 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 both nodes using the same *license.dat* file.

- Create the same alias in the */etc/hosts* file for the respective physical host names of the primary and secondary nodes.

For example, if *node1* and *node2* are the host names 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 host name in the `SERVER` line to the alias host name defined in the `/etc/hosts` file. Following the above example, the `SERVER` line changes from:

```
SERVER node1 any 1700
SERVER license_host any 1700
```

See the *Sybase Software Asset Management (SySAM) Users Guide*.

## Adding an 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 host name `loghost`:

```
MONEY
 master tcp ether loghost 4010
 query tcp ether loghost 4010
```

Make sure the logical host name is accessible on all primary and secondary nodes by properly updating the `/etc/hosts` or NIS hosts map and `/etc/nsswitch.conf` files.

---

**Note** Sybase recommends that you use the local `/etc/hosts` rather than 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 10-1 on page 150 looks like this:

```
#
#internet host table
#
10.22.98.43 node1
10.22.98.44 node2
10.22.98.165 loghost
```

The 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 high availability connections or non-high-availability connections. In either case, client connections require:

- Adequate 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 and recovery time.
- The logical host name be accessible from the client machine.

### non-high-availability connections

non-high-availability connections do not include neither the *hafailover* entry in the *interfaces* file nor the CS\_HAFAILOVER property set in the client connection. When non-high-availability connections are lost, clients must reconnect to the Adaptive Server after failure. To reestablish the connections, clients must retry enough times, or wait long enough between retries, until fail over 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
```

### High-availability connections

High availability 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 fail over.

In an active-passive configuration, clients must self-reference the *hafailover* entry because they reconnect to the same Adaptive Server after fail over. That is, they must set the same server name as the *hafailover* server in the *interfaces* file because the same Adaptive Server is restarted 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 A, “Open Client Functionality in a Failover Configuration.”

## Verifying configuration parameters

To set up Adaptive Server for an 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, enter:

```
sp_configure "enable HA", 2
```

You must restart Adaptive Server for this parameter to take effect.

See Chapter 5, “Setting Configuration Parameters,” in the *System Administration Guide: Volume 1*.

## Adding thresholds to the master log

If you have not already done so, 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 Chapter 16, “Managing Free Space with Thresholds,” in the *System Administration Guide: Volume 2* and the *Reference Manual: Procedures* 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.

## Adding user and login for fault monitor

When the high availability agent fault monitor, *ase\_monitor*, runs the *thorough\_probe* in Sun Cluster and higher, it thoroughly checks the performance of the Adaptive Server. *thorough\_probe*:

- 1 Connects to the Adaptive Server.
- 2 Creates a temporary 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 *thorough\_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 subsystem

See the *Sun Cluster Installation Guide* for information about installing the high availability system.

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 high availability system.
- 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 152.
- Created `$$SYBASE/SYBASE.sh` and edited the file with the required environment for Adaptive Server. As the file is executed in the high availability agent scripts, protect the file from unauthorized access and make sure only the “root” user has read and execute permissions.
- Created the `$$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 high availability agent.

The default `$$SYBASE/$SYBASE_ASE/SC-3_0/` contains these directories:

- `bin`
- `etc`

`$$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* initially contains no files, but will eventually contain *Callback\_log* and *Monitor\_log* files once the Adaptive Server resource is created.

If a *log* directory does not exist, you must create one to store the *callback\_log* and *monitor\_log* files.

## Using the *syscadm* script

Use the *syscadm* script to configure and administer Adaptive Server resource groups, and their associated resources in Sun Cluster. You can use *syscadm* to create, remove, or disable the Adaptive Server resource group and its resources. 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 dependencies for the 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

- Brings the resource group to the unmanaged state

---

**Note** You must be logged in as “root” to run the *sycadm*.

---

*sycadm* works with an input file called *syc\_input\_file*, which you edit to provide the correct input values for your configuration. The *syc\_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 *syc\_input\_file*. If erroneous values are included in this file, they may affect your installation. Sybase suggests that you change the permissions on this file so only System Administrators can edit it.

---

When editing the *syc\_input\_file*:

- Do not include any spaces around “=” in the “<name>=<value>” entries.
- Start comments with #.
- Use names that end with 1 to correspond to the primary companion, and 2 to the secondary companion.

### Sample *syc\_input\_file*

The following is the *syc\_input\_file* used to create and configure the Adaptive Server resource group *rg\_MONEY* and its resources as shown in Figure 10-1 on page 150:

```
#####
##NOTE:
1. This file will be executed by ksh to set environment of sycadm
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-15_0/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="$SYBASE/ASE-15_0/SC-3_0/etc/$RT_NAME"
```

```
Resource Group names
RG_NAME1="rg_$(Datserver_name1)"
RG_NAME2="rg_$(Datserver_name2)"

ASE Resource names and space separated extended properties
ASE_RNAME1="ase_$(Datserver_name1)"
ASE_RNAME2="ase_$(Datserver_name2)"

OTHER_PROPERTIES1="RUN_server_file=/sybase/ASE-15_0/install/RUN_MONEY"
OTHER_PROPERTIES2="RUN_server_file= Callback_log= Monitor_log="

Logical Host Resource names
LOGHOST_RNAME1="lh_$(Datserver_name1)"
LOGHOST_RNAME2="lh_$(Datserver_name2)"

HA Storage Resource names
HASTORAGE_RNAME1="has_$(Datserver_name1)"
HASTORAGE_RNAME2="has_$(Datserver_name2)"

HA Storage Plus Resource names
HASTORAGE_PLUS_RNAME1="hasp_$(Datserver_name1)"
HASTORAGE_PLUS_RNAME2="hasp_$(Datserver_name2)"
```

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 data server name, the node list, and so on.
- Section 2 – enter right-side values for the required entries. For example, if you are using only the SUNW.HAStoragePlus resource, 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 need not provide the right-side values except to override the defaults.

For example, to edit the file for the Adaptive Server resource name, change this line:

```
ASE_RNAME="ase_$(Datserver_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 the *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 unmanages the resource groups.
- -f specifies the input file.
- -v is verbose (displays the Sun Cluster commands as they are being executed).
- -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 the -c option to create the Adaptive Server resource group.

---

## Configuring the Adaptive Server resource group

- 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 high availability agent. Change the value to point to the installation location of *\$\$SYBASE/\$\$SYBASE\_ASE/SC-3\_0/bin*.

For example:

```
RT_BASEDIR=/sybase/ASE-15_0/SC-3_0/bin/
```

- 2 Create or edit a file that contains Adaptive Server login information for the 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 in 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`. The value for login string is in the form `login-name/password`. This is an example of `$$SYBASE/$SYBASE_ASE/SC-3_0/etc/ase_login_file`:

```
normal <tab> sa/sa-password
normal <tab> ase_monitor_user/ase_monitor_user_password
```

---

**Note** After editing the file with proper values, make the file only readable to the “root” user:

```
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 `syscadm`, 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 “Using the `syscadm` script” on page 116.

You can also run the steps performed by the `syscadm` command manually. See “Configuring the resource group manually” on page 169 for more information.

---

**Note** For a list of the extension properties see Table 9-1 on page 120.

---

- 4 Run `scswitch` to:
  - 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
```

## 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 instead 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, set the *SUNW.HAStoragePlus* resource properties *GlobalDevicePaths* and *FilesystemMountPoints* as required. If you are using *syscadm*, you can specify values for corresponding entries in the *sysc\_input\_file*. To enable connection, the *SUNW.HAStoragePlus* resource property *AffinityOn* must be set to TRUE.

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 you are using *SUNW.HAStoragePlus* resources, you can create Adaptive Server database devices either on the global file system or on the 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.

- 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 isql to establish the high availability connection (modify the *interfaces* file to self-reference the hafailover entry, if necessary).

```
isql -Usa -Ppassword -SMONEY -Q
1> select @@servername
2> go
```

```

MONEY

(1 row affected)
```

- 3 Simulate fail over, 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 the connection failover by issuing the following in the isql session started in step 2:

```
1> select @@servername
2> go
CT-LIBRARY error:
 ct_results(): user api layer: internal
Client Library error:
HAFILOVER:Trying to connect to MONEY server.
1> select @@servername
2> go

MONEY

(1 row affected)
```

- 5 Simulate a failback by relocating the resource group back to the primary node.

```
scswitch -z -h node1 -g rg_MONEY
```

- 6 Check connection failover by issuing this command in the isql session started in step 2:

```

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)

```

## 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, you can perform these steps 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 the Adaptive Server resource group” on page 164.

You must be logged on as “root” to run these Sun Cluster 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-15_0/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 fail over:

```
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 host name or floating IP address that can be relocated to the secondary node in case of fail over.

```
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 that are used by the high availability 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-15_0/SC-
3_0/etc/ase_login_file\
-x RUN_server_file=/sybase/ASE-15_0/install/RUN_MONEY
```

For more information about the standard resource properties, see the Sun Cluster documentation. Table 9-1 on page 120 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=hasstorage_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.LogicalHostname* resource if one is added to the resource group.

---

- 8 Run `scswitch` to:
  - 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 124 to create and add the *SUNW.HAStoragePlus* resource to the Adaptive Server resource group.

---

## Upgrading Adaptive Server

To upgrade an Adaptive Server in a high availability configuration, you must temporarily break the companionship between the primary and secondary companion, and disable monitoring of the Adaptive Server Resource Groups. You can shutdown or restart either Adaptive Server independently during the upgrade process without triggering unexpected failovers by the SunCluster subsystem.

---

**Note** You cannot add, delete, or modify any databases, objects, users, or logins during the upgrade process. Making these changes after the companionship is dropped and before it is reestablished may cause the upgrade to fail or destabilize the cluster by causing inconsistencies between servers.

---

### ❖ Stop the monitoring service and drop companionship

Halt the monitoring service and stop management for the Adaptive Server resource groups on both nodes in the cluster.

- As root, issue these commands to bring the SunCluster resource group associated with Adaptive Server offline, and shut down Adaptive Server:

```
scswitch -F -g primary_resourcegroup_name
scswitch -u -g secondary_resourcegroup_name
```

### ❖ Upgrading Adaptive Server

- 1 On both node, disable high availability:

```
sp_configure 'enable HA', 0
```

Restart Adaptive Server for this change to take effect.

---

**Note** Alternatively, if the companions are shut down, edit their server configuration files (*server\_name.cfg*), changing the value of enable HA to 0

---

- 2 Follow the instructions in the *Installation Guide* to upgrade each server.
- 3 Reenable high availability on both nodes:

```
sp_configure 'enable HA', 2
```

You must restart Adaptive Server for the change to take effect. See the *Configuration Guide* for your platform.

- 4 Verify:

- Changes are properly reflected in the resource group and resource properties (for example, *Sybase\_Home*, *runserver* files, *Dataserver\_login\_file*, and so on) in the *\$SYBASE* installation location, or any related files related to high availability in the new installation
- You have performed all actions required for establishing companionship described “Preparing Adaptive Server for active-passive setup” on page 152 and “Configuring the Sun Cluster subsystem” on page 159 and the system maintains these changes after the upgrade is complete.

#### ❖ Reestablishing companionship and resuming resource monitoring

Start the resource group (which starts Adaptive Server, if it is not already started) and restore the monitoring service

- As root, issue:

```
scswitch -z -g ase_resourcegroup_name -h
primary_node
```

#### ❖ Verify failover and failback

- 1 Failover the primary companion by relocating the associated resource group to secondary node. As root, issue:

```
scswitch -z -g ase_resourcegroup_name -h secondary_node
```

Verify Adaptive Server is properly running on the secondary node by checking for the Adaptive Server operating system process and by logging into Adaptive Server with *isql*.

- 2 Failback Adaptive Server by relocating its associated resource group to primary node. As root, issue:

```
scswitch -z -g ase_resourcegroup_name -h primary_node
```

Verify Adaptive Server is properly running on the primary node by checking for the Adaptive Server operating system process and by logging into Adaptive Server with *isql*.

## Location of the error logs

Use this information to debug your high availability system:

- Adaptive Server error log – the location is specified in the RUNSERVER file. For example:

`/sybase/ASE-15_0/install/MONEY.log`

- Adaptive Server high availability 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`



# Configuring Adaptive Server for Failover on Veritas 5.0 and later

This chapter discusses how to configure Adaptive Server for failover on Veritas Cluster Server (VCS), version 5.0 and later.

| Topic                                                    | Page |
|----------------------------------------------------------|------|
| Hardware and operating system requirements               | 176  |
| Preparing Adaptive Server to work with high availability | 178  |
| Configuring the Veritas subsystem for Sybase Failover    | 184  |
| Configuring companion servers for failover               | 189  |
| Administering Sybase Failover                            | 193  |
| Upgrading from an agent of resource type Sybase          | 195  |
| Upgrading Adaptive Server                                | 196  |
| Troubleshooting failover for Veritas Cluster             | 199  |

---

**Note** Adaptive Server Enterprise version 15.0 does not support Veritas Cluster versions 2.2 or 3.0. If you currently have these clusters configured, you have to upgrade the respective cluster versions to configure Adaptive Server for High Availability on Veritas.

---

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 Veritas, first review “Upgrading from an agent of resource type Sybase” on page 195 before performing the tasks in this chapter.

---

## Hardware and operating system requirements

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. See Chapter 2, “What is High Availability?,” for the installation configuration.

You should also install the VCS graphical user interface to facilitate configuration and administration.

- The resource type *HAase*, imported into the VCS.
- Access to shared multihost disks, which store the databases for the Adaptive Server configured for high availability.
- Veritas Volume Manager 3.1 or later to manage disks and create resources like DiskGroup and Volume.
- Third-party vendor mirroring for media failure protection.
- 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 Adaptive Server. A sample service group and the resource dependency graph is shown in Figure 11-1. 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.

---

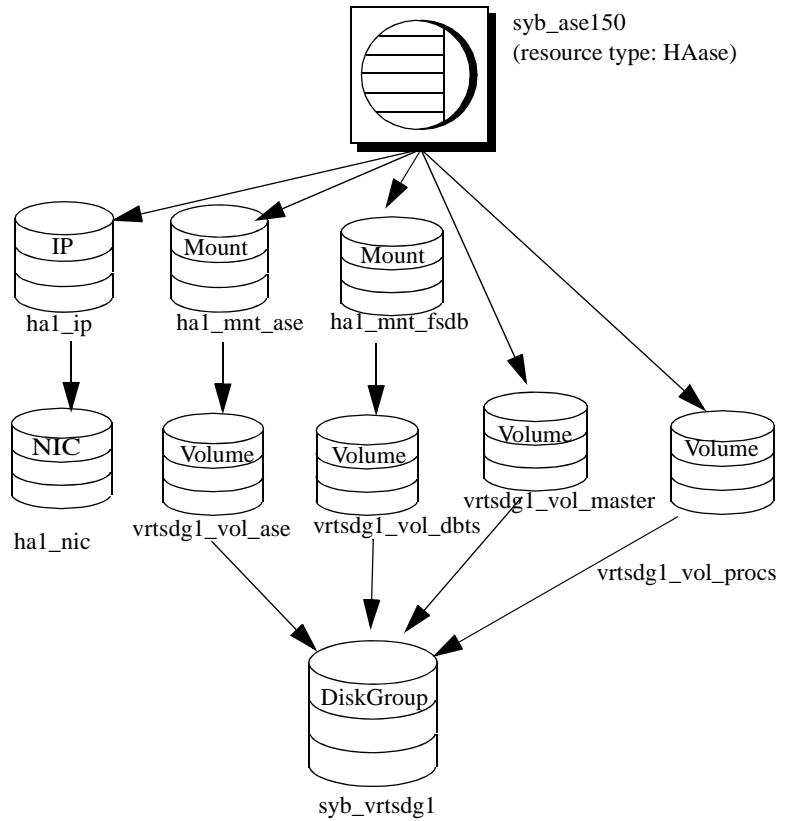
- Both public and private networks on both the nodes.

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

In Figure 11-1 on page 178, the configuration of the service group has one DiskGroup, *syb\_vrtsdgl*, 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\_ase150* of type *HAase* is the Adaptive Server installation, which sits on top of the mount resources. *syb\_ase150* also requires resource *IP*, which also requires resource *NIC* for public network access.

Not shown in Figure 11-1 on page 178, the service group *SybASE* running on the primary node and service group, *SybASE2* running on the secondary node, with a similar configuration:

**Figure 11-1: Sample service group running on Veritas Cluster Server**



## Preparing Adaptive Server to work with high availability

This section contains the instructions necessary to prepare Adaptive Server for a high availability configuration.

## Installing Adaptive Servers

Install both the primary and the secondary servers. They can be installed 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, which 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.

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. 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 fail over

To enable clients to reconnect to the failed-over companion, 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 the 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, modify them to work for fail over.

See the *Utility Guide* for information about `dsedit`.

## **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 `$$SYBASE/$SYBASE_ASE/bin`. Before you can *sybha*, you must change its ownership and permissions. You must also edit a file named *sybhauser* in `$$SYBASE/$SYBASE_ASE/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":

- 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/$SYBASE_ASE/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

master is the default device in a newly installed Adaptive Server. This means that any databases you create (including the proxy databases used by failover) 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 fail over.

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 issue the following to suspend it as the default device:

```
sp_diskdefault master, defaultoff
```

See the *Reference Manual: Commands* for information about disk init. See the *Reference Manual: Procedures* for information about `sp_diskdefault`.

## Adding the local server to `syssservers`

Use `sp_addserver` to add the local server in `syssservers` 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
```

Restart Adaptive Server for this change to take effect.

## Adding secondary companion to syssservers

Add the secondary companion as a remote server in syssservers:

```
sp_addserver server_name
```

By default, Adaptive Server adds the server with a svid of 1000. You need not restart Adaptive Server for this change to take effect.

## Assigning ha\_role

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

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

You can use the `sa_role` to turn the `ha_role` on or off for this session.

You must log out and then log in for these changes to take effect.

## Installing high availability 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 must reinstall all the system stored procedures.

---

The *installhasvss* script:

- Installs the stored procedures required for fail over (for example, `sp_companion`)
- Installs the *SYB\_HACMP* server in syssservers

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

*installhasvss* is located in `$$SYBASE/$SYBASE_ASE/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

Enable the following configuration parameters before you configure Adaptive Server for fail over:

- 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. Restart 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 must purchase the ASE\_HA license option separately. See the installation guide for your platform for information about enabling the ASE\_HA license.

See Chapter 5, “Setting Configuration Parameters,” in the *System Administration Guide: Volume 1*.

## 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 Chapter 16, “Managing Free Space with Thresholds,” in the *System Administration Guide: Volume 2* and the *Reference Manual: Procedures* 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 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 system. See your *VCS Installation Guide* and *VCS User's Guide* for information about installing and using the Veritas Cluster Server high availability system.

### Installing the HAase agent

To install the *HAase* agent on each node of the cluster (you must have “root” permission to run these 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 the 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 the 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 you configure 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, protect the *ase\_login\_file* so that read and write access permissions are restricted to “root”.

```
chmod 400 ase_login_file
chown root ase_login_file
chgrp sys ase_login_file
```

---

**Note** Sybase strongly recommends that you use a password. If you use an empty password, the agent scripts generate a warning message.

---

## Importing the HAase resource type

There are two ways you can import the *HAase* resource type:

- Use the cluster GUI tool to import the new resource type, *HAase*. See your *VCS User's 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 causes no disruption. To manually start the *HAase* agent:

- 1 Check the status of the *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 *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 must contain an *HAase* resource. Table 11-1 shows the attributes of an *HAase* resource.

**Table 11-1: HAase resource**

| Property                        | Datatype, dimension, and default | Description                                                                                                                                                                                                                                                                                                                                                                            |
|---------------------------------|----------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>Sybase_home</i>              | 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.                                                                                                                                                                                                                              |
| <i>Dataserver_name</i>          | string, scalar, null             | Name of the Adaptive Server that is supplied at the time of configuration.                                                                                                                                                                                                                                                                                                             |
| <i>Backup_server_name</i>       | string, scalar, null             | Name of the Backup Server that is supplied at the time of configuration.                                                                                                                                                                                                                                                                                                               |
| <i>Textserver_name</i>          | string, scalar, null             | Name of the full-text search server that is supplied at the time of configuration.                                                                                                                                                                                                                                                                                                     |
| <i>Secondary_companion_name</i> | string, scalar, null             | Name of secondary companion server that is set when you run the 'sp_companion configure' command.                                                                                                                                                                                                                                                                                      |
| <i>Dataserver_login_file</i>    | 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 high availability agent monitor.                                                                                   |
| <i>RUN_server_file</i>          | string, scalar, null             | Absolute path to an alternative <i>RUN_server</i> file, which overwrites the default <code>\$\$SYBASE/\$\$SYBASE_ASE/install/RUN_SERVER</code> .                                                                                                                                                                                                                                       |
| <i>Thorough_probe_cycle</i>     | int, scalar, 3                   | The number of shallow probes before a thorough probe is performed.                                                                                                                                                                                                                                                                                                                     |
| <i>Thorough_probe_script</i>    | 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.<br><br>For security reasons, this file should restrict write access to the owner of <code>\$\$SYBASE</code> directory.<br><br><b>Note</b> This value is ignored by the <i>HAase</i> resource. |
| <i>Debug</i>                    | Boolean, scalar, 0               | If set to 1 (true), the monitor logs debugging messages to <code>\$VCS_LOG/log/HAase_A.log</code> ; other scripts log debugging messages to <code>\$VCS_LOG/log/engine_A.log</code> . The message number range is 2,000,001 and greater.                                                                                                                                               |
| <i>Log_max_size</i>             | int, scalar, 5000000             | Maximum size for the <code>\$VCS_LOG/log/HAase_A.log</code> file.                                                                                                                                                                                                                                                                                                                      |
| <i>Failback_strategy</i>        | string, scalar, null             | Reserved for future use.                                                                                                                                                                                                                                                                                                                                                               |
| <i>HA_config</i>                | Boolean, scalar, 0               | Reserved for future use.                                                                                                                                                                                                                                                                                                                                                               |
| <i>Cmpstate</i>                 | Boolean, scalar, 0               | Reserved for future use.                                                                                                                                                                                                                                                                                                                                                               |

---

**Note** The default value for `$VCS_LOG` is `/var/VRTSvcs`.

---

## Configuring an instance of the *HAase* resource for each service group

Configure an instance of the *HAase* resource by either:

- Using the cluster GUI tool to configure an instance of *HAase* (see your *VCS User's Guide* for more information), or,
- Using cluster commands to manually add a new resource and configure its attributes, as described below. The configuration of service group *SybASE* is shown in Figure 11-1 on page 178:
  - Add the *HAase* resource:

```
#hares -add syb_ase150 HAase SybASE
VCS:10245:Resource added
NameRule and Enabled attributes must be set before agent monitors
hares -modify syb_ase150 Dataserver_name MONEY1
hares -modify syb_ase150 RUN_server_file /release/rel150/ASE-
15_0/install/RUN_MONEY1
hares -modify syb_ase150 Log_max_size 5000000
hares -modify syb_ase150 Dataserver_login_file /release/rel150/ASE-
15_0/install/MONEY1_login
hares -modify syb_ase150 Sybase_home /release/rel150
hares -modify syb_ase150 Thorough_probe_cycle 3
```

- Configure the agent to monitor the status of resource *syb\_ase150*:

```
hares -modify syb_ase150 Enabled 1
```

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\_ase150* and resources of types *Mount*, *Volume*, and *IP* (refer to Figure 11-1 for more details):

```
hares -link syb_ase150 ha1_mnt_ase
hares -link syb_ase150 ha1_mnt_fsdb
hares -link syb_ase150 vrtsdg1_vol_master
hares -link syb_ase150 vrtsdg1_vol_procs
#hares -link syb_ase150 ha1_ip
```

## Configuring companion servers for failover

This section discusses how to configure the Adaptive Servers as primary and secondary companions in a high availability system.

### Adding user and login for high availability 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
```

---

**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 185.

### Running *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 has the resources for only half the number of potential user logins necessary. Instead, configure both `MONEY1` and `PERSONNEL1` for 500 user logins.

`sp_companion do_advisory` 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.

## Verifying the high availability agent

Because machines running the Solaris operating system can support different cluster software, `sp_companion` includes the `show_cluster` option to query the high availability agent currently running and the `set_cluster` option to set the high availability agent.

If you are running the Veritas Cluster Server subsystem, you must specify the cluster with `sp_companion`. Adaptive Server assumes it is running the cluster software for your operating system unless you specify otherwise.

The syntax is:

```
sp_companion companion_server_name, [show_cluster]
sp_companion companion_server_name, [set_cluster, "SC"|"VCS"]
```

To change the Adaptive Server to use the *HAase* agent for the Veritas Cluster:

```
sp_companion MONEY1, set_cluster, "VCS"
```

```
The current cluster is set to VCS
```

---

**Note** Do not change to another high availability agent type when Adaptive Server is configured for normal companion mode on your VCS system

---

## Creating an asymmetric companion configuration

To configure the primary companion asymmetrically, issue this command from the secondary companion:

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

Where:

- *primary\_server\_name* – the name of the primary Adaptive Server as defined in the *interfaces* file entry and in *syssservers*.
- *login\_name* – the name of the user performing this cluster operation (this person must have the *ha\_role*).



- *password* – 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, NULL, 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
```

## 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. You use the same syntax as the asymmetric setup, except you cannot use the `with_proxydb` option. See [Creating an asymmetric companion configuration](#), 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` (issue this command from primary companion `MONEY1`):

```
sp_companion 'PERSONNEL1', configure, 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`. This file should have 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 switches 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 primary resources, but the Adaptive Server on the failed-over group is not started.

---

### Failing back to the primary companion

The 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 a 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.

- 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.

---

**Note** You cannot connect clients with the failover property to Adaptive Server until you issue `sp_companion resume`. If you do try to reconnect them after issuing `sp_companion prepare_failback`, the client stops responding 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
```

- 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. 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
```

- 3 Resume normal companion mode. From the secondary companion, issue:

```
sp_companion primary_companion_name, resume
```

---

**Note** You cannot connect clients with the failover property 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

Dropping companion mode is irreversible; you must reconfigure the companion servers before they 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 high availability 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 companion mode, issue:

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

## Upgrading from an agent of resource type Sybase

If you are using a high availability agent from an earlier release of VCS for resource type *Sybase* and you want to use the new agent for the resource type *HAase*, switch from the old to the new agent:

- 1 Install the new agent for resource type *HAase*. See “Installing the HAase agent” on page 184.
- 2 Import the new resource type, *HAase*. See “Importing the HAase resource type” on page 185.

- 3 Start the new agent for resource type, *HAase*. See “Starting the HAase agent” on page 186.
- 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 186.
- 7 Enable the new *HAase* resource:

```
haconf -makerw
hares -modify Sybase_resource_name Enabled 1
haconf -dump -makero
```

## Upgrading Adaptive Server

To upgrade an Adaptive Server in a high availability configuration, you must temporarily break the companionship between the primary and secondary companion, and disable monitoring of the Adaptive Server service groups in VCS. You can shutdown or restart either Adaptive Server independently during the upgrade process without triggering unexpected failovers by VCS.

---

**Note** You cannot add, delete, or modify any databases, objects, users, or logins during the upgrade process. Making these changes after the companionship is dropped and before it is reestablished may cause the upgrade to fail or destabilize the cluster by causing inconsistencies between servers.

---

### ❖ Stop the monitoring service and drop companionship

- 1 On all nodes in the cluster halt the monitoring service. As root, issue:

```
hares -modify primary_resource Enabled 0
hares -modify primary_resource Critical 1
```

If you configured the system for symmetric failover, disable monitoring for the secondary resource:

```
hares -modify secondary_resource Enabled 0
hares -modify secondary_resource Critical 0
haconf -dump -makero
```

- 2 As root, issue:

```
hares -offline primary_resource -sys primary_system_name
hares -offline secondary_resource -sys secondary_system_name
```

- 3 From the secondary companion, issue:

```
sp_companion primary_server_name, "drop"
```

- 4 (For symmetric configuration) Drop the secondary's companionship log in the primary companion and issue:

```
sp_companion secondary_server_name, "drop"
```

- 5 Ensure that both nodes are in single-server mode:

```
sp_companion
```

If the companions are in single-server mode, they return:

```
Server 'server_name' is not cluster configured.
Server 'server_name' is currently in 'Single server'
mode.
```

The servers are now running on their installation node and may be stopped and started independently without the VCS attempting to failover the resources between nodes.

#### ❖ **Upgrading Adaptive Server**

- 1 On each node, disable high availability:

```
sp_configure 'enable HA', 0
```

Restart Adaptive Server for this change to take effect.

- 2 Follow the instructions in the installation guide to upgrade each server.

- 3 On all nodes, reenable high availability:

```
sp_configure 'enable HA', 1
```

Restart Adaptive Server for this change to take effect.

- 4 Ensure that permissions are set correctly for the sybha binary and *sybhausers* file.

As root, issue these commands from `$$SYBASE/$SYBASE_ASE/bin`:

```
chown root sybha
chgrp sybhagrp sybha
chmod 4550 sybha
```

As root, perform these tasks from `$$SYBASE/$$SYBASE_ASE/install`:

- 1 Ensure that the sybase user is included in the `sybhauser` file
- 2 Issue:

```
chown root sybhauser
chmod 600 sybhauser
```

- 5 Verify:

- Changes are properly reflected in the service group and resource properties (for example, `Sybase_Home`, runserver files, `Dataserver_login_file`, and so on) in the `$$SYBASE` installation location, or any related files related to high availability in the new installation
- You have performed all actions required for establishing companionship described “Preparing Adaptive Server to work with high availability” on page 178 and “Configuring the Veritas subsystem for Sybase Failover” on page 184 and the system maintains these changes after the upgrade is complete.

❖ **Reestablishing companionship and resuming resource monitoring**

- 1 On each node, manually restart Adaptive Server.
- 2 As root from the primary node, restore the monitoring service:

```
haconf -makerw
hares -modify primary_resource Enabled 1
hares -modify primary_resource Critical 1
```

If you configured the system for symmetric failover, enable monitoring for the secondary resource:

```
hares -modify secondary_resource Enabled 1
hares -modify secondary_resource Critical 1
haconf -dump -makero
```

- 3 Verify you have performed the prerequisite steps for establishing companionship described in “Configuring companion servers for failover” on page 189.
- 4 Reestablish companionship between the servers (see “Creating an asymmetric companion configuration” on page 190 or “Configuring for symmetric configuration” on page 192):



```
dbcc traceon (2209)
sp_companion primary_server_name,configure
dbcc traceoff(2209)
```

---

**Note** For symmetric configurations, issue this command on both companions.

---

If the secondary server includes user databases, you may see one or more warning messages, which you can safely ignore:

```
Msg 18739, Level 16, State 1:
Server 'server_name', Procedure 'sp_hacmpcfgvrfy', Line 102:
Database 'database_name': a user database exists. Drop this
database and retry the configuration again.
```

- Restart the Adaptive Server resources on their appropriate nodes. As root on the primary node, enter:

```
hares -online primary_resource -sys primary_system_name
```

As root on the secondary node, enter

```
hares -online secondary_resource -sys secondary_system_name
```

- Run `sp_companion` to verify that the system is properly configured for symmetric or asymmetric failover.

## Troubleshooting failover for Veritas Cluster

This section includes troubleshooting information about common errors.

- Turn on debugging for Adaptive Server. Use trace flag 2205 to get high-availability-related debugging information. The following `isql` session turns on debugging and redirects messages to the console:

```
dbcc traceon(2205)
dbcc traceon(3604)
```

- When your system reports errors, first check the error log. Any error message with a message ID greater than 2,000,000 is an error message from the *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 fail over:

```
hares -clear sybase_res_name -sys
secondary_host_name
```

## Recovering from a failed *prepare\_failback*

During fail back, if *prepare\_failback* was executed successfully on the secondary companion but the primary companion does not start:

- 1 Check the primary companion’s error log and the cluster error log to identify why the server did not start, and correct the problems.
- 2 To clear the FAULTED state of the *HAase* resource, issue:

```
hares -clear primaryHAase_resname -sys primary_hostname
```

- 3 As “root”, 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 235.

- 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 system:

- 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*.



# Configuring Adaptive Server for Failover on Windows

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

| Topic                                             | Page |
|---------------------------------------------------|------|
| Hardware and operating system requirements        | 203  |
| Installing Adaptive Servers                       | 204  |
| Verifying configuration parameters                | 208  |
| Configuring Windows                               | 209  |
| Configuring and securing Microsoft Cluster Server | 214  |
| Troubleshooting                                   | 216  |
| Upgrading Adaptive Server                         | 218  |

## Hardware and operating system requirements

High availability requires the following hardware and system components:

- A Microsoft certified cluster. See your Microsoft documentation for a description of what constitutes a certified cluster
- Windows 2008 R2
- The operating system and cluster server software are installed on both nodes, residing on local disk storage with the same path on both nodes (for example, *C:\WINNT* and *C:\WINNT\Cluster*). See your Microsoft documentation for a description of what constitutes a certified cluster.
- Adaptive Server 64-bit 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 must 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. Each companion can use only its own area of shared disk for its data devices.
- You must install Adaptive Server on the same hard drive letter and path on all cluster nodes. For example, if you use *C:\Sybase* as the installation directory on the first node, you must also use *C:\Sybase* on the second node.

This section discusses how to prepare Adaptive Server for a high availability configuration.

## Installing Adaptive Servers

---

**Note** Start all command-line utilities, Xterms, and executables related to configuring and executing high availability on Windows as the administrator. To run these as administrator, right-click on the command-tool icon and select “Run as administrator.”

---

Install both the primary and secondary Adaptive Servers. 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 an earlier 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 subsystemprocs devices) on dedicated shared disks.

## Changing the domain administration account

After you install the Adaptive Servers, the servers run under an operating system account known as “LocalSystem”. For a clustered operation, the Adaptive Server must be able to communicate over the network to the other cluster node using Windows operating system services. Because the LocalSystem account is not allowed to access any Windows 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 an Adaptive Server to run as a domain administrator:

- 1 Start the Services application from the Windows Control Panel | Administrative Tools.
- 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 Double Click on `Sybase SQLServer_server_name`, for the Properties dialog box.
- 4 Select Log on tab.
- 5 Select This Account from Log on tab.
- 6 Enter a valid domain administration account name (for example, `MYDOMAIN\AdminUser1`). Enter and confirm this account’s password.
- 7 Click OK.
- 8 Restart the Adaptive Server.

## Adding entries for both Adaptive Servers to *sql.ini*

The *sql.ini* file must include entries for both companions. The server entry in the *sql.ini* file must use the same network name that is specified in *syssservers*. For information about adding entries to *sql.ini*.

## Adding entries to *sql.ini* for client connections

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, modify them to work for fail over.

## Creating a 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, use `disk init` to create a new device (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 fail over. For example, to add a new default device named *money1\_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 *Reference Manual: Commands* for information about `disk init`. See the *Reference Manual: Procedures* for information about `sp_diskdefault`.



## Adding the primary companion as a local server

Use `sp_addserver` to list the local server as the local server in `syssservers` 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, enter:

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

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

## Adding secondary companion to `syssservers`

Add the secondary companion as a remote server in `syssservers`:

```
sp_addserver server_name
```

By default, Adaptive Server adds the server with a `svid` of 1000. You do not need to restart Adaptive Server for this change to take effect.

## Assigning `ha_role` to the 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
```

Log out and then log back in to the Adaptive Servers for the change to take effect.

## Running `insthasv` to install high availability stored procedures

Run the `insthasv` script on both Adaptive Servers. `insthasv`:

- Installs the stored procedures required for fail over (for example, `sp_companion`).
- Installs the SYB\_HACMP server in `syssservers`.

You must

- Create the sybsecurity database before running `insthasv`. See the *Security Administration Guide*.
- Run `instsecu` (located in `%SYBASE%\ASE-15_0\scripts`)

- Have System Administrator privileges to run the *insthasv* script.

*insthasv* is located in the `%SYBASE%\%SYBASE_ASE%\scripts` directory. To execute *insthasv*, enter:

```
isql -Usa -Ppassword -Sservername -i %SYBASE%\%SYBASE_ASE%\scripts\insthasv
```

*insthasv* prints messages as it creates stored procedures and creates the SYB\_HACMP server.

## Verifying configuration parameters

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 restart 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 Chapter 5, “Setting Configuration Parameters,” in the *System Administration Guide: Volume 1*.

## Running *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 has the resources for only half the number of potential user logins necessary. Instead, configure both MONEY1 and PERSONNEL1 for 500 user logins.

`sp_companion do_advisory` 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.

## Configuring Windows

You can configure failover on Windows either from the command line or using the Cluster Administrator. Using the command line is described below; using the Cluster Administrator is described in “Configuring Sybase companion servers using the Windows management tools” on page 212.

If you are configuring for a symmetric setup, you must first configure the cluster for an asymmetric setup.

## Asymmetric setup from the command line

---

**Note** Start all command-line utilities, Xterms, and executables related to configuring and executing high availability on Windows as the administrator. To run these as administrator, right-click on the command-tool icon and select “Run as administrator.”

---

To configure the primary companion asymmetrically, issue this command from the secondary companion:

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

Where:

- *primary\_server\_name* – the name of the primary Adaptive Server as defined in the *interfaces* file entry and in *syservers*.
- *login\_name* – the name of the user performing this cluster operation (this person must have the *ha\_role*).

- *password* – 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, NULL, 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
```

## Symmetric configuration 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.

Issue `sp_companion` from the secondary companion to configure it for symmetric setup. Use the same syntax as for the asymmetric setup, except you cannot use the `with_proxydb` option.

The following example adds an Adaptive Server named `MONEY1` as the secondary companion to the Adaptive Server named `PERSONNEL1`. Issue the following command from the server `MONEY1`:

```
1> sp_companion 'PERSONNEL1', configure, null, 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
```

After `sp_companion` completes:

- 1 Check the completion status of the Wizard. If the configuration failed, check the setup log for errors, if it succeeded, open the Windows Failover Cluster Manager (as a shortcut, press and release the Windows key, type “failover” to bring up the Failover Cluster Manager icon, then press Enter).

- 2 Locate the new Sybase Companion server in the Services and Applications tree. The companion server is named *primary\_server\_name\_GRP*.

Click on companion server name to view the Summary of *primary\_server\_name\_GRP* display.

- 3 In the right-hand side of this display, click “Add Storage,” and add all shared storage devices the companion servers will use.
- 4 In the central part of the display, right-click the Sybase Companion server under “Other Resources,” select Properties, and highlight the Dependencies tab. Add a dependency for every shared storage device.

## Configuring Sybase companion servers using the Windows management tools

This section assumes that the Microsoft Cluster Server is installed on your system, and that the Microsoft .NET framework installer (*InstallUtil.exe*), a Microsoft distributed tool, is available.

Verify you have the *InstallUtil.exe* tool

- 1 Find *InstallUtil.exe*:

- You may already have it: Check this location on your system:

`%SystemRoot%\Microsoft.NET\Framework64\v2.0.50727\installutil.exe`

- If your system does not include the *InstallUtil.exe* tool, download and install the Microsoft.NET framework Version 2 utility from Microsoft.
- 2 Copy the *InstallUtil.exe* tool to the `%SYBASE%\%SYBASE_ASE%\install` directory

---

**Note** If *installUtil.exe* does not exist in the `%SYBASE%\%SYBASE_ASE%\install` directory, you cannot configure Sybase Failover using the Microsoft Management Console (MMC). However, you can configure Sybase Failover using the steps outlined in “Asymmetric setup from the command line” on page 209 and “Symmetric configuration from the command line” on page 211.

---

After you copy *installUtil.exe*, use this command on each node of the cluster to install the Sybase Companion Server resource type and Cluster Administrator extensions:

```
%SYBASE%\%SYBASE_ASE%\bin\sybcpnin.exe -s
```

where *%SYBASE%* is the release directory for the Adaptive Server executable.

To configure Sybase Failover on Microsoft Cluster Server:

---

**Note** These steps are unnecessary if you configured the companion server according to the steps described in “Asymmetric setup from the command line” on page 209.

---

- 1 Use Windows Explorer to move to *%SYBASE%\%SYBASE\_ASE%\bin* and double-click on *SybCpnSetup.msc* (as a shortcut, hold the Windows key and press “R”, type *SybCpnSetup.msc* in the dialog box, and press Enter).

The Microsoft Management Console appears.

- 2 Click on Sybase Companion Server Setup.
- 3 Click on “Create Sybase companion server” on the right-hand panel of the Microsoft Management Console. Click Yes to confirm.

Microsoft Management Console starts the Sybase companion installer Wizard.

- 4 On the ASE Server Information window, enter:
  - The name of the Adaptive Server you are configuring as the primary companion
  - The System 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

- 5 Select Next.

- 6 In the Companion Server Information field, enter the name of the Adaptive Server that is to be the secondary companion.

To configure the companions in a symmetric setup, select Symmetric, then click Next.

- 7 On the Cluster Parameters window, select Use System Generated Cluster Login. This provides a system-generated setup login that is used when the cluster logs into the Adaptive Server. Click 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.)

- 8 (Optional) On the Setup Options window, enter the path to the error log that records the steps made during this configuration. This log may be helpful if you need to call Technical Support. Select Finish.
- 9 The next window lists the configuration that you have selected for this cluster configuration. Select Back and reenter 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.

- 10 Check the completion status of the Wizard. If the configuration failed, check the setup log for errors, if it succeeded, open the Windows Failover Cluster Manager (as a shortcut, press and release the Windows key, type “failover” to bring up the Failover Cluster Manager icon, then press Enter).
- 11 Locate the new Sybase Companion server in the Services and Applications tree. The companion server is named *primary\_server\_name\_GRP*.  
Click on companion server name to view the Summary of *primary\_server\_name\_GRP* display.
- 12 In the right-hand side of this display, click “Add Storage,” and add all shared storage devices the companion servers will use.
- 13 In the central part of the display, right-click the Sybase Companion server under “Other Resources,” select Properties, and highlight the Dependencies tab. Add a dependency for every shared storage device.

## Configuring and securing Microsoft Cluster Server

This section describes how to set the pending timeout and the 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 online or offline, 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 timeout period must be long enough to start 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 timeout 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 these properties:

- 1 Start the Windows Failover Cluster Manager (as a shortcut, press and release the Windows key, type “failover” to bring up the Failover Cluster Manager icon, then press ENTER).
- 2 Locate the new Sybase Companion server in the Services and Applications tree. The companion server is named *primary\_server\_name\_GRP*.  
Click on companion server name to view the Summary of *primary\_server\_name\_GRP* display.
- 3 In the center panel under Other Resources, right-click the Sybase Companion server and select Properties.
- 4 Highlight the Policies tab of the Properties panel.
- 5 Change the Pending Timeout property to a value that is about two minutes larger than the longest time the server takes to recover.
- 6 Click OK.

## 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 in to Adaptive Server to control it for cluster operations.

The login and its password are stored as part of the Windows Registry Cluster Database (under *HKLM\Cluster*). Even though this information is encrypted to prevent users from obtaining privileged login information, 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.

To encrypt the cluster login and password:

- 1 Execute *REGEDIT.EXE*.
- 2 From the window titled *HKEY\_LOCAL\_MACHINE* on Local Machine, double-click the Cluster folder, then select the Resources key.
- 3 Select Permissions from the Security menu. A dialog called Registry Key Permissions displays.
- 4 Select Remove from the Registry Key Permissions dialog box to remove all entries 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.
- 5 Click OK.

Repeat this process on both cluster nodes.

## Troubleshooting

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, from which you must 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 from this:

- 1 Restart 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
```

Make sure that this command executes successfully.

- 6 Make sure the primary companion is up and running, then resume normal companion mode. From the primary companion, issue:

```
sp_companion secondary_companion, 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 use the Cluster Administrator to bring the resource online.

## Recovering from a failed *prepare\_failback*

During fail back, if `prepare_failback` was executed successfully on the secondary companion but the primary companion does not start, perform the following to roll back and then reissue the `prepare_failback` command:

- 1 Check the primary companion’s system event log to find the reason the server does not start, 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, it does use Move Group to move it.
- 3 Log in 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 Check that the MSCS resource for the primary server is online. If it is not, manually bring the resource online using the Cluster Administrator.

## Upgrading Adaptive Server

To upgrade an Adaptive Server in a high availability configuration, you must temporarily break the companionship between the primary and secondary companion, and disable monitoring of the Adaptive Server resource groups. You can shutdown or restart either Adaptive Server independently during the upgrade process without triggering unexpected failovers by the MSCS cluster.

---

**Note** You cannot add, delete, or modify any databases, objects, users, or logins during the upgrade process. Making these changes after the companionship is dropped and before it is reestablished may cause the upgrade to fail or destabilize the cluster by causing inconsistencies between servers.

---

### ❖ Halt the monitoring service and drop companionship

- 1 Drop the companionship. From the secondary companion, issue:

```
sp_companion primary_server_name, "drop"
```

- 2 (For symmetric configuration) From the primary companion, issue:

```
sp_companion secondary_server_name, "drop"
```

- 3 Ensure that both nodes are in single-server mode by issuing, on each node:

```
sp_companion
```

If the companions are in single-server mode, they return:

```
Server 'server_name' is not cluster configured.
Server 'server_name' is currently in 'Single server'
mode.
```

- 4 Use the Microsoft Cluster Manager to verify the Sybase Server resources corresponding to the primary companion (and to the secondary companion, if you are using a symmetric environment) have been removed. If not, manually delete them in the Cluster Manager.

**❖ Upgrade Adaptive Server**

- 1 On each node, disable high availability:

```
sp_configure 'enable HA', 0
```

Restart Adaptive Server for this change to take effect.

- 2 Follow the instructions in the installation guide to upgrade each server.
- 3 Reenable high availability on all nodes:

```
sp_configure 'enable HA', 1
```

You must restart Adaptive Server for the change to take effect.

- 4 Reinstall the *instmstr* and *insthasv* scripts on the upgraded servers. See “Reinstalling installmaster” and “Rerunning installhasvss” on page 224. When you reinstall *instmstr*, you must reinstall *insthasv*.

**❖ Reestablishing companionship and resuming package monitoring**

- 1 Verify you have performed the prerequisite steps for establishing companionship.
- 2 Reestablish companionship between the servers. On the secondary server, issue (For symmetric configurations, issue this command on both companions):

```
sp_companion primary_server_name,configure
```

If user databases exist on the secondary server, you may see one or more warning messages, which you can safely ignore:

```
Msg 18739, Level 16, State 1:
Server 'svr2', Procedure 'sp_hacmpcfgvrfy', Line 102:
Database 'svr2_db1': a user database exists. Drop this
database and retry the configuration again.
```

- 3 Run *sp\_companion* to verify that the system is properly configured for failover.
- 4 Verify the primary resource can successfully failover and failback by relocating it to the secondary node and then back to primary node. Repeat these steps for the secondary companion in symmetric configurations.



# Open Client Functionality in a Failover Configuration

This appendix discusses the changes required for Open Client to work with Sybase 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, you must also configure the secondary companion for parallel processing so it can run the application during fail over.

---

You must modify all applications that are written with CTLIB API calls before they can work with the Sybase Failover. Use:

- 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. 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 to reconnect to the secondary companion when the primary companion crashes, or when you issue a shutdown with `nowait`, triggering a failover.

See the “Adding entries for both Adaptive Servers to the *interfaces* file” section in the chapter for your platform for information about adding this line.

- 3 Write application failover messages according to these parameters:
  - As soon as the companion begins to go down, clients receive an informational message that fail over 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 a failover connection, and clients reconnect to the secondary companion appropriately.

However, if the failover property is set but the *interfaces* file does not have an entry for the *hafailover* server (or vice-versa), then it is not a failover connection. Instead, it is a normal connection with the failover property turned off. Inform the user to check the failover property to determine whether or not the connection is a failover connection.

- 4 Add return codes.

When a successful failover occurs, the client issues a return value named `CS_RET_HAFILOVER`, which is specific to the following CTLIB API calls:

```
ret = ct_results(cmd, result_type)
ret = ct_send(cmd)
```

`CS_RET_HAFILOVER` is returned from the API call during a synchronous connection. In an asynchronous connection, these APIs issue `CS_PENDING`, and the callback function returns `CS_RET_HAFILOVER`. Depending on the return code, the customer can perform the required processing, such as sending the next command to be executed.

- a Rebuild your applications, linking them with the libraries included with the failover software.

---

**Note** You cannot connect clients with the failover property until you issue `sp_companion resume`. If you do try to reconnect them after issuing `sp_companion prepare_failback`, the client stops responding until you issue `sp_companion resume`.

---



# Troubleshooting Secondary Points of Failure

This appendix discusses common problems that result from secondary points of failure with the high availability system.

## Troubleshooting with *dbcc ha\_admin*

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. Sybase Failover includes *dbcc ha\_admin*, which addresses second points of failure.

See “dbcc options for high availability systems” on page 235 for information about *dbcc ha\_admin* syntax and a complete list of options.

After you run *installhasvss* on a companion server, you should re-run this script only if the stored procedures it creates are corrupted, or to install a newer version of *installhasvss*. *dbcc ha\_admin* (' ', *state\_machine*) temporarily moves the companion to single-server mode so *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, run this command only when there is no concurrent activity.

---

## Reinstalling *installmaster*

After you install *installmaster* on a companion server, you should only rerun 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 *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.

---

Perform the following to re-install *installmaster*:

- Run `dbcc ha_admin` to the local companion server to single-server mode:

```
dbcc ha_admin (' ', 'state_machine', 'halt')
```

Where ' ' is used as an empty placeholder.

- Rerun *installmaster*.
- Run `dbcc ha_admin` to return the companion server to its original mode:

```
dbcc ha_admin (' ', 'state_machine', 'restart')
```

- You must reinstall *installhasvss* after you reinstall *installmaster*. See below.

## Rerunning *installhasvss*

- 1 Make a note of the `srvnetname` for the `SYB_HACMP` entry in `sys.servers`. When it is configured for Sybase Failover, `SYB_HACMP` points to the companion server's `srvnetname` (for example, the `srvnetname` for the `SYB_HACMP` entry on companion server `MONEY1` is `PERSONNEL1`).

- 2 Run `dbcc ha_admin` to move the companion to single-server mode:

```
dbcc ha_admin (' ', 'state_machine', 'halt')
```

Where ' ' is used as an empty 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 `syssservers`. If this occurs, add the name of the remote server to `syssservers` by issuing:

```
sp_addserver SYB_HACMP, null, 'remote_server_srvnetname'
```

Run `dbcc ha_admin` to return the companion server to its original mode:

```
dbcc ha_admin (' ', 'state_machine', 'restart')
```

## Using `dbcc ha_admin` for rolling back failover commands

`dbcc ha_admin` includes the `rollback_failover` and `rollback_failback` options. Use these `dbcc` options as a last resort; only System Administrators who are knowledgeable about the high availability system should issue them.

These options allow you to roll back the steps performed by:

- A fail over that did not complete because of either a problem with the high availability system (for example, all the disks were not available during fail over, so the companion mark all databases as suspect) or because the secondary companion crashed during fail over.
- A `sp_companion...prepare_failback` command that did not complete because of either a problem with the high availability system or because the primary companion did not restart during failback.

There are platform-specific steps you must perform before you issue either `dbcc ha_admin rollback_failover` or `rollback_failback`. See the configuration chapter for your platform in this manual for information.

## Using `@@hacmpservername`

Use the `@@hacmpservername` global variable to determine the name of the companion server:

```
select @@hacmpservername
```

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

```
select @@hacmpservername

PERSONE1
(1 row affected)
```

## Error messages

The following are common error messages you might receive.

- Error message 18805:

Warning: Server '%1!' is configured for ASE HA services. The network name 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.

This error occurs when the SYB\_HACMP network name is set to another server's network name. Use `sp_addserver` to set the `srvnetname` of SYB\_HACMP to the local server's network name. During normal companion mode, the `srvnetname` 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 release the lock when they are done. This error occurs when you perform a cluster operation before the previous cluster operation releases the cluster-wide lock. For information about releasing a cluster-wide lock, see “Cluster locks in a high availability node” on page 6.

- 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 attributes to confirm the compatibility between the companion servers. One of your companion servers has attribute settings that are not compatible with the other companion server. Run do\_advisory for a list of the problem attributes. See Chapter 6, “Running do\_advisory”.



# Changes to Commands, System Procedures, and Databases

This appendix discusses changes to commands, system procedures, and system databases when Adaptive Server is configured for failover.

## Changes to commands

*Table C-1: Changes to commands*

| Command                                            | Asymmetric setup                                                                                                                                                                                                                                                                                                                                                                                                                                                                   | Symmetric setup                                                                                                   |
|----------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------|
| create role<br>add role<br>drop role<br>alter role | <p>During normal companion mode, any changes made to the primary companion with these commands are synchronized with the secondary companion server.</p> <p>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 the failback mode.</p> <p>You cannot run drop role during failover mode.</p> <p>You cannot run these commands during suspended mode.</p> | <p>These commands have the same behavior in symmetric configuration as they have in asymmetric configuration.</p> |
| create database                                    | <p>During normal companion mode, create database creates a proxy database on the secondary companion.</p> <p>During failover mode, create database cannot run because the primary companion's model database is not in failover mode.</p> <p>During the failback mode, create database is allowed only under special circumstances.</p> <p>You cannot run create database during suspended mode.</p>                                                                               | <p>create database has the same behavior in symmetric setup as it has in asymmetric setup.</p>                    |
| alter database                                     | <p>During normal companion mode, alter database adds 2MB of space to the database.</p>                                                                                                                                                                                                                                                                                                                                                                                             | <p>alter database has the same behavior in symmetric setup as it has in asymmetric setup.</p>                     |

| Command                                                      | Asymmetric setup                                                                                                                                                                                                                                                                                                    | Symmetric setup                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|--------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| disk init                                                    | <p>During normal companion mode, disk init has the same behavior as in symmetric configuration.</p> <p>During failover mode, the secondary server can add devices to its local set by ensuring the unique device name space.</p> <p>During suspended mode, disk init cannot run.</p>                                | <p>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.</p> <p>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 such as log expansion.</p> <p>During suspended mode, disk init cannot run.</p> |
| <p>disk mirror</p> <p>disk remirror</p> <p>disk unmirror</p> | <p>Sybase mirroring is not supported for high availability.</p>                                                                                                                                                                                                                                                     |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| disk resize                                                  | <p>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.</p>                                           |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| drop database                                                | <p>During normal companion mode, drop database informs the companion server to free the database name space and may request to drop the proxy database.</p> <p>During failover mode, there are no restrictions on the drop database command.</p> <p>During suspended mode, you cannot run drop database.</p>        | <p>This command has the same behavior in symmetric setup as is has in asymmetric setup.</p>                                                                                                                                                                                                                                                                                                                                                                                                    |
| <p>grant</p> <p>revoke</p>                                   | <p>During normal companion mode, changes to permissions from these commands are synchronized across the companion servers.</p> <p>During failover mode, there are no restrictions for grant. You cannot run revoke during failover mode.</p> <p>During suspended mode, you cannot issue either grant or revoke.</p> | <p>This command has the same behavior in symmetric setup as is has in asymmetric setup.</p>                                                                                                                                                                                                                                                                                                                                                                                                    |
| <p>shutdown</p> <p>shutdown with nowait</p>                  |                                                                                                                                                                                                                                                                                                                     |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |



## Changes to system procedures

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 fail over. Because database IDs may change, system procedures are automatically recompiled after fail over 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 performed only in failover mode.

## System procedures hold table lock

System procedures cannot acquire explicit table locks on system tables. However, in a system using Sybase Failover, system procedures on both companions may attempt to modify the system tables at the same time.

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 `syslogins` are released.

Any other system procedures that must 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 use the `sp_configure` “`dtm lock timeout period`” command to set the amount of time system procedures wait in the queue for the locked proxy system table. See Chapter 5, “Setting Configuration Parameters,” in the *System Administration Guide: Volume 1*.

## System procedures that synchronize changes

The following 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.

|                                    |                                       |
|------------------------------------|---------------------------------------|
| <code>sp_addexternlogin</code>     | <code>sp_dropremotelogin</code>       |
| <code>sp_addlanguage</code>        | <code>sp_drop_resource_limit</code>   |
| <code>sp_addlogin</code>           | <code>sp_dropserver</code>            |
| <code>sp_addremotelogin</code>     | <code>sp_drop_time_range</code>       |
| <code>sp_add_resource_limit</code> | <code>sp_locklogin</code>             |
| <code>sp_addserver</code>          | <code>sp_modifylogin</code>           |
| <code>sp_add_time_range</code>     | <code>sp_modify_resource_limit</code> |
| <code>sp_defaultdb</code>          | <code>sp_modify_time_range</code>     |
| <code>sp_defaultlanguage</code>    | <code>sp_password</code>              |
| <code>sp_dropexternlogin</code>    | <code>sp_remoteoption</code>          |
| <code>sp_droplanguage</code>       | <code>sp_serveroption</code>          |
| <code>sp_droplogin</code>          | <code>sp_setlangalias</code>          |

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`

## Other changes to system procedures

This section describes system procedures that exhibit behavior changes when Adaptive Server is configured for failover. After Adaptive Server is configured as a companion server:

- System procedures exhibit no changes to their default functionality when they are run in single-server mode.
- You cannot run any of the system procedures listed in Table C-2 or Table C-3 during the failback mode.
- The “Normal companion mode,” column of Table C-2 and Table C-3 describes the behavioral changes for system procedures issued from an asymmetric primary, asymmetric secondary, or symmetric companion.
- The “Failover mode,” column of Table C-2 and Table C-3 describes the behavioral changes for system procedures issued during either asymmetric secondary failover or symmetric failover.

Table C-2 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 C-2 must be run from master.
- You cannot these system procedures during asymmetric secondary suspended mode or symmetric suspended mode.
- An X indicates that the system procedure does not run in the listed mode.

**Table C-2: Changes in system procedures that alter server-wide attributes**

| System procedure       | Normal companion mode                                                                                   | Asymmetric primary suspended mode | Failover mode |
|------------------------|---------------------------------------------------------------------------------------------------------|-----------------------------------|---------------|
| 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             |
| sp_locklogin           |                                                                                                         |                                   |               |

| System procedure         | Normal companion mode                                                                                   | Asymmetric primary suspended mode | Failover mode |
|--------------------------|---------------------------------------------------------------------------------------------------------|-----------------------------------|---------------|
| sp_modify_resource_limit | You must manually run this system procedure on the remote server as well to synchronize the companions. |                                   |               |

Table C-3 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 C-3: Changes that alter database-wide attributes when they are run in master**

| System procedure | Normal companion mode                                                                                   | Asymmetric primary suspended mode | Failover mode | Notes                                                            |
|------------------|---------------------------------------------------------------------------------------------------------|-----------------------------------|---------------|------------------------------------------------------------------|
| 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             |                                                                  |
| sp_renamedb      |                                                                                                         |                                   | X             | See below for additional restrictions on this system procedure.  |

sp\_changedbowner and sp\_renamedb run during failover mode; additionally, they exhibit these 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, 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**

Table C-4 includes information about the `dbcc ha_admin` options.

**Table C-4: *dbcc ha\_admin* options**

| <b>Option name</b>             | <b>Function</b>                                                                                                                                                                                                              | <b>Syntax and comments</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|--------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>rollback_failback</code> | Rolls back the effect of <code>sp_companion...</code> , <code>prepare_failback</code> and returns the companion to failover mode. This command works regardless of the results of the <code>prepare_failback</code> command. | <code>dbcc ha_admin (" ", rollback_failback)</code><br>Where " " is a required empty placeholder. <ul style="list-style-type: none"> <li>• Can be used only in failback mode.</li> <li>• Any failback threads waiting for the resume command are killed when this command is executed.</li> <li>• You may need to perform platform-specific steps to prepare your companions for the <code>rollback_failback</code> option. See the appropriate chapter in this manual for more information.</li> <li>• You can issue this command only from the secondary companion.</li> </ul>                                                                                                                                                                                           |
| <code>rollback_failover</code> | Rolls back the effects of fail over from the primary companion, and returns it to normal companion mode. <code>rollback_failover</code> does not affect the secondary companion.                                             | <code>dbcc ha_admin (" ", rollback_failover)</code><br>Where " " is a required empty placeholder. <ul style="list-style-type: none"> <li>• This command can be used only in failover mode.</li> <li>• You may need to perform platform-specific steps to prepare you companions for the <code>rollback_failover</code> option. See the appropriate chapter in this manual for more information.</li> <li>• <code>rollback_failover</code> has no effect on the companion server that failed. The companion server that takes over the failed companion's workload resumes normal companion mode.</li> <li>• You can issue this command only from the secondary companion.</li> <li>• This command works even when fail over has marked the databases "suspect."</li> </ul> |

| Option name     | Function                                                                                                                                                                                                                                     | Syntax and comments                                                                                                                                                                                                                                                                                                                            |
|-----------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| drop_failoverdb | Used only in failover mode. drop_failoverdb 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. | <pre>dbcc ha_admin (" ", drop_failedoverdb, database_name)</pre> <p>Where " " is a required empty placeholder, and <i>database_name</i> is the name of the database you are dropping.</p> <ul style="list-style-type: none"> <li>Use only as a last resort, when you must drop a database to complete the load of another database.</li> </ul> |
| clusterlock     | Acquires or releases cluster-wide locks during a cluster operation.                                                                                                                                                                          | <pre>dbcc ha_admin (" ", clusterlock, [acquire   release])</pre> <p>For more information about cluster-wide locks and releasing them, see “Cluster locks in a high availability node” on page 6.</p>                                                                                                                                           |
| state_machine   | Moves the companion server to single-server mode.                                                                                                                                                                                            | <pre>dbcc ha_admin (' ', 'state_machine', 'halt')</pre> <p>Where " " is a required empty placeholder. For information about using this option, see Appendix A.</p>                                                                                                                                                                             |
| 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.                                                          | <pre>dbcc ha_admin (SYB_HACMP, session, "drop")</pre>                                                                                                                                                                                                                                                                                          |

### dbcc dbrepair option

Sybase Failover adds the dropproxydb option to dbcc dbrepair.

**Table C-5: dbcc dbrepair dropproxydb option**

| Option name | Function              | Syntax and comments                                                                                                                                        |
|-------------|-----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------|
| dropproxydb | Drops proxy databases | <pre>dbcc dbrepair(database_name, dropproxydb)</pre> <p>where <i>database_name</i> is the name of the database of the proxy database you are dropping.</p> |

# Glossary

This glossary includes only the terms used in this book. For a description of Adaptive Server and SQL terms, see the *Adaptive Server Glossary*.

**active-active**

A system that is set up as 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.

**active-passive**

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 that can host the Adaptive Server and its resources, if necessary.

**asymmetrical**

A high-availability system consisting of one primary companion and one secondary companion. In an asymmetric system, only the primary companion can fail over. The secondary Adaptive Server is also known as a "hot standby."

**cluster**

A collection of nodes in a high availability system. A cluster for the Adaptive Server high availability system consists of at least two nodes.

**companion server**

Each Adaptive Server in a primary availability system is a companion. One Adaptive Server is called a companion server and the other is called the secondary companion server.

**connection failover**

A connection that has the failover property set fails over to the secondary companion

**failback or fail back**

The planned event during which Adaptive Server is migrated back to, and restarted on, a primary companion, after failover has taken place. This involves moving the failed-over databases, devices, and client connections from the secondary companion to the restarted primary companion.

**failover mode**

The mode of the primary companion after it has failed-over and is running on the secondary companion.

|                                 |                                                                                                                                                                                                                                                          |
|---------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>failover or fail over</b>    | The process by which Adaptive Server migrates to another that which takes over the responsibility of managing the failed server. Failover may occur due to scheduled maintenance or a failure of Adaptive Server or the machine running Adaptive Server. |
| <b>high availability</b>        | A system designed to reduce downtime.                                                                                                                                                                                                                    |
| <b>node</b>                     | A machine in a high availability system.                                                                                                                                                                                                                 |
| <b>normal companion mode</b>    | The mode during which two Adaptive Servers in a high availability system are functioning as independent servers and are configured to fail over during a scheduled maintenance or system failure.                                                        |
| <b>primary companion</b>        | The Adaptive Server whose databases and connections are migrated to the secondary Adaptive Server during failover.                                                                                                                                       |
| <b>proxy databases</b>          | Placeholder 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.                              |
| <b>secondary companion</b>      | The Adaptive Server configured to accept a failed-over primary Adaptive Server.                                                                                                                                                                          |
| <b>single-server mode</b>       | The mode of Adaptive Server while it is being configured for high availability. During this mode, Adaptive Server cannot fail over.                                                                                                                      |
| <b>stable mode</b>              | A system state where Adaptive Server can exist for an extended period of time, such as the day-to-day operation of Adaptive Server.                                                                                                                      |
| <b>suspended companion mode</b> | The mode of Adaptive Server after companion mode has been suspended. During this mode, Adaptive Server cannot fail over; it is working independently of the companion Adaptive Server.                                                                   |
| <b>symmetric</b>                | 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.                                                             |
| <b>thorough_probe</b>           | A utility executed by <i>ase_monitor</i> ; runs the <i>thorough_probe</i> to thoroughly check the performance of the Adaptive Server.                                                                                                                    |
| <b>transitional mode</b>        | Occurs when Adaptive Server shifts from failed-over mode to normal companion mode; is typically of very short duration.                                                                                                                                  |



# Index

## Numerics

18750 error message on IBM 100

## A

Active 105

active-active setups

configuration Sun Clusters 114

active-passive setups 152

client configuration 152

configuring Sun Clusters 151

interface entries, Sun Clusters 155

Sun Clusters 147–174

Adaptive Server

changing domain administration accounts for  
Windows 205

considerations for IBM 76

performance in asymmetric configuration 20

performance in symmetric configuration 22

two-phase commit transactions and 18

Adaptive Server adding entries in interfaces file

active-active during failover 109

for HP 46

for IBM 78

for Sun Clusters, active-passive 155

for Veritas 179

add login for probe 114, 158, 238

**add role** command 229

add user 114, 158, 238

adding entries in the sql.ini file

for Windows 205

**alter role** command 229

ASE\_HA.sh script editing

for HP 53

for IBM 84

asymmetric configurations 19–22

companion 64

companion configuration 92, 128, 190

described 19

for Windows 209

performance of Adaptive Server in 20

asymmetric mode

**add role** command 229

**alter role** command 229

**create database** command 229

**create role** command 229

defined 237

**disk init** command 230

**disk mirror** command 230

**disk remirror** command 230

**disk resize** command 230

**disk unmirror** command 230

**drop database** command 230

**drop role** command 229

**grant** command 230

**revoke** command 230

**shutdown** command 230

**shutdown with nowait** command 230

asymmetrical, defined 237

audit trails and failover 24

auditing

configuration parameters 23

setting options 23

## C

client connections

cluster locks, for 7

failover and 3

cluster

defined 237

described 14

cluster locks

client connections for 7

cluster-wide locks 6

High Availability node, in 6

releasing 6

## Index

- clusterlock option in **dbcc ha\_admin** option 236
  - @@cmpstate global variable 28
  - companion
    - asymmetric configuration 64, 92, 128, 190
    - disk mirroring 17
    - failover 1
  - companion mode
    - dropping 133
    - for HP 68
    - for IBM 99
    - for Veritas 195
    - suspending Sun active-active 132
  - companion mode, suspending
    - for IBM 97
    - for Sun active-active 133
    - for Veritas 3.5 194
  - companion servers
    - configuring for IBM 91
    - configuring for Sun active-active 126
    - configuring for Veritas 189
    - defined 237
    - determining mode of 28
    - failback mode 27
    - modes of 26
    - naming with @@hacmpservername 225
    - normal companion mode 26
    - resuming from normal companion to suspended mode 27
    - single-server mode 26
    - suspended mode 27
  - Component Integration Services (CIS), creating proxy databases with 32
  - configuration
    - active-passive client 152
    - symmetric 66, 93
  - configuration parameters, verifying
    - for IBM 80
    - for Sun Clusters active-active 113
    - for Sun Clusters active-passive 157
    - for Veritas 183
    - for Windows 208
  - configuration requirements
    - for HP 45
    - for IBM 76
  - configurations
    - asymmetric 19–22
    - for Windows asymmetric 209
    - Microsoft Cluster Server (MSCS) in Windows 214
    - symmetric 19–22, 130
  - configurations, symmetric
    - for Veritas 192
    - for Windows 211
  - configuring
    - active-active for Sun 105–145
    - active-passive for Sun Clusters 152
    - Adaptive Servers 12
    - companion servers for auditing 23
    - High Availability Sun Clusters active-passive 147–173
    - parameters for auditing 23
    - resource group manually 169
    - resource group manually, Sun Clusters 134
    - Sun Clusters active-active 114
  - configuring High Availability
    - for HP 45–71
    - for IBM 75–101
    - for Veritas 176–201
    - for Windows 203–217
  - connection failover 1
  - create
    - proxy databases 33
  - create database** command 229
    - degraded performances of 20
  - create role** command 229
  - CTLIB API calls, modifying for failover 221
- ## D
- database IDs and failover 231
  - databases
    - creating proxy 32
    - proxy 31–36
    - required number of open databases 13
  - dbcc dbrepair** option
    - dropproxydb** option 236
  - dbcc ha\_admin** option 6, 235
    - clusterlock 236
    - described 223
    - drop\_failoverdb** option 236
    - prepare\_failback** option 225
    - rollback\_failback** option 225, 235

- rollback\_failover** option 225
- second points of failure 225
- session 236
- state\_machine** option 236
- default devices, creating new 111
  - for HP 49
  - for IBM 81
  - for Veritas 181
  - for Windows 206
- devices, required number of 13
- disk failures and failover 17
- disk init** command 230
- disk mirror** command 230
- disk mirroring and companion clusters 17
- disk remirror** command 230
- disk resize** command 230
- disk unmirror** command 230
- do\_advisory** option
  - described 37
  - failback, running in 5
  - for Sun 41
  - group attributes 38
  - output of 41
- domain
  - administration accounts in Windows, changing 205
  - checks during failover 231
- domains 29
- drop database** command 230
- drop role** command 229
- drop\_failoverdb** option in **dbcc ha\_admin** option 236
- dropproxydb** option in
  - dbcc dbrepair** option 236
- dtm lock timeout period** command 231

## E

- error message 18750 in IBM 100

## F

- failback 27
  - defined 237

- described 4, 9
- do\_advisory** option, running 5
- in active-passive Sun Clusters 152
- manual method in IBM 96
- performing 5
- primary companion Sun active-active 131
- primary node IBM 95
- sp\_companion**, issuing 5
- sp\_companion**, syntax for 5
- failover
  - adding entries in Adaptive Server interface files
    - during failover Sun active-active 109
  - adding entries in sql.ini files during for Windows 206
  - adding entries interface files during failover for Veritas 3.5 179
  - administering for IBM 95
  - administering for Veritas 193
  - administering Sun active-active 131
  - applications running with 14
  - audit trails and 24
  - changes in system procedures caused by 231
  - client connections and 3
  - companion failover 1
  - configuration considerations 16
  - configuring datatypes for 18
  - configuring in HP 51
  - configuring in Windows 209
  - connection failover 1
  - database IDs and 231
  - dbcc ha\_admin** option 235
  - defined 237, 238
  - described 1, 9
  - disk failures and 17
  - disk mirroring and 17
  - domain checks during 231
  - domains used in 29
  - hafailover label in interfaces file 3
  - High Availability 14
  - illustrated 3
  - logs location for IBM 101
  - modes 25–27
  - modifying CTLIB API calls 221
  - requirements 12
  - sequential steps for 1, 9
  - stable mode 25

- sybsecurity and 24
  - sysdevices, mapping 2
  - system failover 1
  - system procedures with database-wide changes and 234
  - system procedures with server-wide changes and 233
  - table locks and 231
  - transitional mode 25
  - user logins in 3
- G**
- grant** command 230
- H**
- ha\_role**
    - sp\_companion**, in Windows 207
  - ha\_role sp\_companion** 112
    - for IBM 83
  - ha\_role sp\_companion**
    - for HP 50
    - for Veritas 182
  - HACMP
    - configuring resource groups in IBM 88
    - troubleshooting for AIX in IBM 99
  - @@hacmpservername global variable 225
  - hafailover label, adding in interfaces file 3
  - high availability 9, 238
    - configurations illustrated 15
    - connections, active-passive Sun Clusters 156
    - defined 238
    - failover and 14
    - installing stored procedures in HP 50
    - nodes, cluster locks in 6
    - subsystem 14
  - HP configuration 45–71
    - \$SYBASE 47
    - adding local server to sysservers 49
    - adding secondary companion to sysservers 49
    - creating new default device 49
    - dropping companion mode 68
    - editing ASE\_HA.sh script 53
    - failover 51
    - ha\_role sp\_companion** 50
    - installhasvss script 50
    - installing Adaptive Server 46
    - interfaces files, adding entries in 46
    - package configuration 52
    - package control script in 59
    - parameters, verifying 51
    - preparing Adaptive Server 46
    - requirements 45
    - sybha** executable 48
- I**
- IBM configuration 75–101
    - \$SYBASE 79
    - Adaptive Server, installing 77
    - adding local servers to sysservers 82
    - adding secondary companions to sysservers 82
    - adding thresholds to master log 81
    - companion servers for failover 91
    - creating new default devices 81
    - dropping companion mode 99
    - editing ASE\_HA.sh script 84
    - error message 18750 100
    - failing back manually 96
    - failover log location 101
    - ha\_role sp\_companion** 83
    - HACMP resource groups 88
    - installhasvss script 82
    - installing Adaptive Server 77
    - interfaces files, adding entries in 78
    - parameters, verifying 80
    - preparing Adaptive Server 77
    - primary companions as monitored resource 94
    - recovering from failed **prepare\_failback** 100
    - requirements 76
    - restarting shut down companion during suspended mode 97
    - resuming normal companion mode 98
    - troubleshooting failover on HACMP 99
  - IBMconfiguration
    - sybha** executable 79
  - installhasvss script
    - for IBM 82
    - for Veritas 182

- installing stored procedures 17
- re-installing 223
- installing
  - for HP 46
  - for IBM 77
  - for Veritas 179
  - for Windows 204
- installmaster script
  - running installhasvss before 17
  - stored procedures for failover and 17
- insthasv script
  - installing High Availability stored procedures in Windows 207
- interfaces files
  - adding entries during failover active-active 109
  - for HP 46
  - for IBM 78
  - for Sun active-active 108
  - for Sun Clusters, active-passive 155
  - for Veritas 179
  - for Veritas during failover 179
  - hafailover label, adding 3

## L

- local servers, adding with syservers
  - for HP 49
  - for IBM 82
  - for Sun active-active 111
  - for Veritas 181
  - for Windows 207
- logins
  - failover, in 3
  - requirements 13
- logs
  - adding thresholds in master log Sun Clusters active-active 113
  - adding thresholds in master log Sun Clusters active-passive 157
  - adding thresholds in master log Veritas 183
  - failover log location IBM 101
  - location SunClusters active-passive 173
  - location Veritas 201

## M

- manual failback
  - for IBM 96
- master log, adding thresholds
  - for IBM 81
  - for Sun Clusters active-active 113
  - for Sun Clusters active-passive 157
  - for Veritas 183
- Microsoft Cluster Server (MSCS) in Windows 214
- modes
  - @@cmpstate, determining with 28
  - companion servers 26
  - failback 27
  - normal companion 26
  - resuming from normal companion to suspended 27
  - suspended 27
- Monitoring tables, installing with high availability 16
- MSCS (Microsoft Cluster Server) in Windows 214

## N

- node 14
  - defined 238
- non-High Availability connections
  - Sun Clusters active-passive 156
- normal companion mode 26
  - defined 238
  - resuming IBM 98
  - resuming Veritas 194
- normal companion mode, resuming in Sun Clusters
  - active-active 133
- number of
  - devices, requirements 13
  - open databases, requirements 13
  - open databases, **sp\_configure** command 13
  - user connections, requirements 13
  - user connections, **sp\_configure** command 13

## O

- open databases, required number of 13

**P**

- package
  - configuration in HP 52
  - control script, creating in HP 59
- parameters verifying configurations
  - for HP 51
  - for IBM 80
  - for Sun Clusters active-active 113
  - for Sun Clusters active-passive 157
  - for Veritas 183
  - for Windows 208
- prepare\_failback**
  - dbcc ha\_admin** option 225
  - recovering from 143
  - sp\_companion**, issuing 5
  - sp\_companion**, syntax for 5
- prepare\_failback**, recovering from
  - for IBM 100
  - for Veritas 200
- preparing
  - for HP 46
  - for IBM 77
  - for Windows 204
- primary companions 14
  - as monitored resource IBM 94
  - defined 238
- properties for SY.ase file 120, 187
- proxy databases 31–36
  - commands not used in 34
  - configuring for failover 31
  - create when 33
  - creating 32
  - defined 238
  - size 34
  - sp\_dboption** command and 35
  - system procedures, issuing 35
  - updating manually 36

**Q**

- quorum attributes of **sp\_companion** 42

**R**

- recovering from failed **prepare\_failback** 143
    - for IBM 100
    - for Veritas v3.5 200
  - re-installing
    - installhasvss script 223
  - remote servers adding with syssservers in HP 49
  - requirements
    - failover 12
    - resources 13
  - resource groups
    - configuring in HACMP in IBM 88
  - resource requirements 13
  - resuming operations 4, 9
  - revoke** command 230
  - rollback\_failback** option
    - dbcc ha\_admin** 225
    - in **dbcc ha\_admin** option 235
- 
- S**
  - secondary companions 14
    - defined 238
  - servers
    - adding local with syssservers in HP 49
    - adding remote with syssservers in HP 49
    - adding secondary companion with syssservers for IBM 82
    - adding secondary companion with syssservers for Sun active-active 112
    - adding secondary companion with syssservers for Veritas 182
    - adding secondary companion with syssservers for Windows 207
    - failback mode 27
    - normal companion mode 26
    - resuming from normal companion to suspended mode 27
    - suspended mode 27
    - SYB\_HACMP 18
  - servers, companion
    - modes of 26
    - naming with @@hacmpservername 225
    - session option in **dbcc ha\_admin** option 236

- shutdown** command 230
  - restarting during suspended mode in IBM 97
- shutdown with nowait** command 230
- single system 16
- single-server mode 26
  - defined 238
- sp\_companion** 28
  - do\_advisory** option described 37
  - failback, issuing during 5
  - failback, syntax for issuing 5
  - quorum attributes 42
- sp\_companion do\_advisory** option 128
- sp\_companion ha\_role**
  - for HP 50
  - for IBM 83
  - for Sun active-active 112
  - for Veritas 182
  - for Windows 207
- sp\_companion prepare\_failback** command 6
- sp\_companion resume** command 6
- sp\_configure** command
  - number of open databases 13
  - number of user connections 13
- sp\_dboption** command and proxy databases 35
- sql.ini files, adding entries in Adaptive Server in Windows during failover 206
- srids requirements 13
- stable failover mode 25
- state\_machine** option in **dbcc ha\_admin** option 236
- stored procedures
  - installhasvss script 17
- Sun Clusters active-active
  - active-active hardware 105
  - active-active operating system 105
  - add login for probe 114
  - add user 114
  - adding local servers to syssservers active-active 111
  - adding secondary companions to syssservers active-active 112
  - adding thresholds to master log 113
  - companion servers for failover 126
  - configuring for High Availability 114
  - configuring resource group manually 134
  - creating new default devices for active-active 111
  - dropping companion mode in Sun Clusters 133
  - ha\_role** and **sp\_companion for active-active** 112
  - installhasvss script active-active 112
  - installing Adaptive Server High Availability active-active 108
  - interfaces files during failover, adding entries, active-active 109
  - interfaces files, adding entries active-active 108
  - parameters, verifying 113
  - preparing Adaptive Server with High Availability active-active 108
  - recovering from failed **prepare\_failback** 143
  - resuming normal companion mode 133
  - sp\_companion** and **do\_advisory** option 128
  - steps 122
  - sybha** executable in active-active 110
  - symmetric configuration 130
  - troubleshooting failover 143
- Sun Clusters active-passive 147–171
  - adding login for probe 158
  - adding thresholds to master log 157
  - adding user 158
  - client configuration 152
  - configuring 151, 152, 159
  - configuring resource group manually 169
  - configuring Sun Clusters active-passive 159
  - failback 152
  - high availability connections 156
  - interface entries for active-passive 155
  - interfaces files, adding entries in 155
  - log locations 173
  - non-High Availability connections 156
  - parameters, verifying 157
  - steps 164
- Sun configuration
  - adding login for probe 238
  - adding user 238
- suspended mode 27
  - defined 238
  - restarting shutdown companion in IBM 97
- SY.ase file properties 120, 187
- SYB\_HACMP 18
  - installhasvss script and 18
  - procedures if dropped accidentally 18
- \$\$SYBASE setting value
  - for HP 47

- for IBM 79
- sybha** executable
  - running in HP 48
  - running in IBM 79
  - running in Sun active-active 110
  - running in Veritas 180
- sybsecurity and failover 24
- symmetric
  - configuration 66, 93
- symmetric configuration 19–22, 130
  - companion 21
  - for Veritas 192
  - for Windows 211
  - performance of Adaptive Server in 22
- symmetric mode
  - add role** command 229
  - alter role** command 229
  - create database command** 229
  - create role** command 229
  - defined 238
  - disk init** command 230
  - disk mirror** command 230
  - disk remirror** command 230
  - disk resize** command 230
  - disk unmirror** command 230
  - drop database** command 230
  - drop role** command 229
  - grant** command 230
  - revoke** command 230
  - shutdown** command 230
  - shutdown with nowait** command 230
- sysdevices, mapping during failover 2
- syssservers adding secondary companions in HP 49
- syssservers, adding local servers
  - for HP 49
  - for IBM 82
  - for Sun active-active 111
  - for Veritas 181
  - for Windows 207
- syssservers, adding secondary companions
  - for IBM 82
  - for Sun active-active 112
  - for Veritas 182
  - for Windows 207
- system failover 1

- system procedures
  - changes due to failover 231
  - proxy databases, issuing in 35

## T

- table locks and failover 231
- thresholds, adding to master log
  - for IBM 81
  - for Sun Clusters active-active 113
  - for Sun Clusters active-passive 157
  - for Veritas 183
- transitional failover mode 25
- troubleshooting
  - dbcc ha\_admin** option 223
- troubleshooting failover for
  - HACMP for AIX in IBM 99
  - Sun Clusters active-active 143
  - Veritas clusters 199
- two-phase commit transactions
  - Adaptive Server and 18

## U

- updating proxy databases 36
- user
  - connections, required number of 13
  - logins in failover 3

## V

- Veritas configuration 176–201
  - adding local servers to syssservers 181
  - adding secondary companions to syssservers 182
  - adding thresholds to master log 183
  - administering failover 193
  - companion servers for failover 189
  - configuring Veritas clusters 184
  - creating new default devices 181
  - dropping companion mode 195
  - ha\_role sp\_companion** 182
  - installhasvss script 182
  - installing Adaptive Server 179



---

- interfaces files during failover, adding entries in 179
- interfaces files, adding entries in 179
- log locations 201
- parameters, verifying 183
- recovering from failed **prepare\_failback** 200
- resuming normal companion mode 194
- sybha** executable 180
- symmetric configuration 192
- troubleshooting failover for Veritas clusters 199

## W

- Windows configuration 203–217
  - adding local servers to `syssservers` 207
  - adding secondary companions to `syssservers` 207
  - asymmetric configuration 209
  - changing domain administration accounts after installing 205
  - creating new default devices 206
  - failover 209
  - ha\_role** and **sp\_companion** 207
  - installing Adaptive Server 204
  - `insthasv` script 207
  - Microsoft Cluster Server (MSCS) 214
  - parameters, verifying 208
  - preparing Adaptive Server 204
  - `sql.ini` files during failover, adding entries in 206
  - symmetric configuration 211

