Sun Cluster Data Services Planning and Administration Guide for Solaris OS

Sun Microsystems, Inc.
4150 Network Circle
Santa Clara, CA 95054
U.S.A.

Part No: 817–6564–10
September 2004, Revision A

DOCUMENTATION IS PROVIDED "AS IS" AND ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT, ARE DISCLAIMED, EXCEPT TO THE EXTENT THAT SUCH DISCLAIMERS ARE HELD TO BE LEGALLY INVALID.

CETTE PUBLICATION EST FOURNIE "EN L'ETAT" ET AUCUNE GARANTIE, EXPRESSE OU IMPLICITE, N'EST ACCORDEE, Y COMPRIS DES GARANTIES CONCERNANT LA VALEUR MARCHANDE, L'APTITUDE DE LA PUBLICATION A REPONDRE A UNE UTILISATION PARTICULIERE, OU LE FAIT QU'ELLE NE SOIT PAS CONTREFAISANTE DE PRODUIT DE TIERS. CE DENI DE GARANTIE NE S'APPLIQUERAIT PAS, DANS LA MESURE OU IL SERAIT TENU JURIDIQUEMENT NUL ET NON AVENU.

Adobe PostScript™

040812@9495

Contents

Preface

Sun Cluster Data Services Planning and Administration Guide for Solaris OS explains how to install and configure Sun™ Cluster data services on both SPARC® and x86 based systems.

Note – In this document, the term “x86” refers to the Intel 32-bit family of microprocessor chips and compatible microprocessor chips made by AMD.

This document is intended for system administrators with extensive knowledge of Sun software and hardware. Do not use this document as a planning or presales guide. Before reading this document, you should have already determined your system requirements and purchased the appropriate equipment and software.

The instructions in this document assume knowledge of the Solaris™ Operating System and expertise with the volume manager software that is used with Sun Cluster.

Note – Sun Cluster software runs on two platforms, SPARC and x86. The information in this document pertains to both platforms unless otherwise specified in a special chapter, section, note, bulleted item, figure, table, or example.

UNIX Commands

This document contains information about commands that are specific to installing and configuring Sun Cluster data services. The document does *not* contain comprehensive information about basic UNIX® commands and procedures, such as shutting down the system, booting the system, and configuring devices. Information about basic UNIX commands and procedures is available from the following sources:

- Online documentation for the Solaris Operating System
- Solaris Operating System man pages
- Other software documentation that you received with your system

Typographic Conventions

The following table describes the typographic changes that are used in this book.

TABLE P–1 Typographic Conventions

Typeface or Symbol	Meaning	Example
`AaBbCc123`	The names of commands, files, and directories, and onscreen computer output	Edit your `.login` file. Use `ls -a` to list all files. `machine_name% you have mail.`

`AaBbCc123`	What you type, contrasted with onscreen computer output	`machine_name%` **`su`**
		`Password:`

AaBbCc123	Command-line placeholder: replace with a real name or value	The command to remove a file is `rm` *filename*.
AaBbCc123	Book titles, new terms, and terms to be emphasized	Read Chapter 6 in the *User's Guide*. These are called *class* options. Do *not* save the file. (Emphasis sometimes appears in bold online.)

Shell Prompts in Command Examples

The following table shows the default system prompt and superuser prompt for the C shell, Bourne shell, and Korn shell.

TABLE P–2 Shell Prompts

Shell	Prompt
C shell prompt	`machine_name%`
C shell superuser prompt	`machine_name#`
Bourne shell and Korn shell prompt	`$`
Bourne shell and Korn shell superuser prompt	`#`

Related Documentation

Information about related Sun Cluster topics is available in the documentation that is listed in the following table. All Sun Cluster documentation is available at `http://docs.sun.com`.

Topic	Documentation
Data service administration	*Sun Cluster Data Services Planning and Administration Guide for Solaris OS* Individual data service guides
Concepts	*Sun Cluster Concepts Guide for Solaris OS*
Overview	*Sun Cluster Overview for Solaris OS*
Software installation	*Sun Cluster Software Installation Guide for Solaris OS*
System administration	*Sun Cluster System Administration Guide for Solaris OS*
Hardware administration	*Sun Cluster 3.x Hardware Administration Manual for Solaris OS* Individual hardware administration guides
Data service development	*Sun Cluster Data Services Developer's Guide for Solaris OS*

Topic	Documentation
Error messages	*Sun Cluster Error Messages Guide for Solaris OS*
Command and function reference	*Sun Cluster Reference Manual for Solaris OS*

For a complete list of Sun Cluster documentation, see the release notes for your release of Sun Cluster at `http://docs.sun.com`.

Related Third-Party Web Site References

Third-party URLs that are referenced in this document provide additional related information.

Note – Sun is not responsible for the availability of third-party web sites mentioned in this document. Sun does not endorse and is not responsible or liable for any content, advertising, products, or other materials that are available on or through such sites or resources. Sun will not be responsible or liable for any actual or alleged damage or loss caused or alleged to be caused by or in connection with use of or reliance on any such content, goods, or services that are available on or through such sites or resources.

Accessing Sun Documentation Online

The docs.sun.com^{SM} Web site enables you to access Sun technical documentation online. You can browse the docs.sun.com archive or search for a specific book title or subject. The URL is `http://docs.sun.com`.

Ordering Sun Documentation

Sun Microsystems offers select product documentation in print. For a list of documents and how to order them, see “Buy printed documentation” at `http://docs.sun.com`.

Help

If you have problems installing or using Sun Cluster, contact your service provider and provide the following information:

- Your name and email address (if available)
- Your company name, address, and phone number
- The model and serial numbers of your systems
- The release number of the Solaris Operating System (for example, Solaris 8)
- The release number of Sun Cluster (for example, Sun Cluster 3.0)

Use the following commands to gather information about each node on your system for your service provider.

Command	Function
`prtconf -v`	Displays the size of the system memory and reports information about peripheral devices
`psrinfo -v`	Displays information about processors
`showrev –p`	Reports which patches are installed
SPARC: `prtdiag -v`	Displays system diagnostic information
`scinstall -pv`	Displays Sun Cluster release and package version information

Also have available the contents of the `/var/adm/messages` file.

CHAPTER 1

Planning for Sun Cluster Data Services

This chapter provides planning information and guidelines to install and configure Sun Cluster data services. This chapter contains the following sections.

See the *Sun Cluster Concepts Guide for Solaris OS* document for conceptual information about data services, resource types, resources, and resource groups.

Sun Cluster software can provide service only for those data services that are either supplied with the Sun Cluster product or are created with the Sun Cluster data services application programming interfaces (APIs)

If your applications are not currently offered as Sun Cluster data services, see the *Sun Cluster Data Services Developer's Guide for Solaris OS* for information on how to develop highly available data services for applications.

Note – Sun Cluster software does not have a data service for the `sendmail`(1M) subsystem. The `sendmail` subsystem can run on the individual cluster nodes, but the `sendmail` functionality will not be highly available, including the functionality of mail delivery and mail routing, queuing, or retry.

Configuration Guidelines for Sun Cluster Data Services

This section provides configuration guidelines for Sun Cluster data services.

Identifying Data Service Special Requirements

Identify requirements for all of the data services **before** you begin Solaris and Sun Cluster installation. Failure to do so might result in installation errors that require that you completely reinstall the Solaris and Sun Cluster software.

For example, the Oracle Parallel Fail Safe/Real Application Clusters Guard option of Sun Cluster Support for Oracle Parallel Server/Real Application Clusters has special requirements for the hostnames that you use in the cluster. Sun Cluster HA for SAP also has special requirements. You must accommodate these requirements before you install Sun Cluster software because you cannot change hostnames after you install Sun Cluster software.Also note that both Sun Cluster Support for Oracle Parallel Server/Real Application Clusters and Sun Cluster HA for SAP are not supported for use in x86 based clusters.

Determining the Location of the Application Binaries

You can install the application software and application configuration files on one of the following locations.

- **The local disks of each cluster node –** Placing the software and configuration files on the individual cluster nodes provides the following advantage. You can upgrade application software later without shutting down the service.

  The disadvantage is that you then have several copies of the software and configuration files to maintain and administer.

- **The cluster file system –** If you put the application binaries on the cluster file system, you have only one copy to maintain and manage. However, you must shut down the data service in the entire cluster to upgrade the application software. If you can spare a small amount of downtime for upgrades, place a single copy of the application and configuration files on the cluster file system.

  See the planning chapter of the *Sun Cluster Software Installation Guide for Solaris OS* for information on how to create cluster file systems.

- **Highly available local file system –** Using `HAStoragePlus`, you can integrate your local file system into the Sun Cluster environment making the local file system highly available. `HAStoragePlus` provides additional file system capabilities such as checks, mounts, and unmounts enabling Sun Cluster to fail over local file systems. In order to failover, the local file system must reside on global disk groups with affinity switchovers enabled.

See the individual data service guides or "Enabling Highly Available Local File Systems" on page 86 for information on how to use the `HAStoragePlus` resource type.

Verifying the `nsswitch.conf` File Contents

The `nsswitch.conf` file is the configuration file for name-service lookups. This file determines the following information.

- which databases within the Solaris environment to use for name-service lookups
- in what order to consult the databases

Some data services require that you direct "group" lookups to "files" first. For these data services, change the "group" line in the `nsswitch.conf` file so that the "files" entry is listed first. See the documentation for the data service that you plan to configure to determine whether you need to change the "group" line.

See the planning chapter in the *Sun Cluster Software Installation Guide for Solaris OS* for additional information on how to configure the `nsswitch.conf` file for the Sun Cluster environment.

Planning the Cluster File System Configuration

Depending on the data service, you might need to configure the cluster file system to meet Sun Cluster requirements. See the documentation for the data service that you plan to configure to determine whether any special considerations apply.

See the planning chapter of the *Sun Cluster Software Installation Guide for Solaris OS* for information on how to create cluster file systems.

The resource type `HAStoragePlus` enables you to use a highly available local file system in a Sun Cluster environment configured for failover. See "Enabling Highly Available Local File Systems" on page 86 for information on setting up the `HAStoragePlus` resource type.

Relationship Between Resource Groups and Disk Device Groups

Sun Cluster uses the concept of **node lists** for disk device groups and resource groups. Node lists are ordered lists of primary nodes, which are potential masters of the disk device group or resource group. Sun Cluster uses a **failback policy** to determine what happens when a node has been down and then rejoins the cluster, and the rejoining node appears earlier in the node list than the current primary node. If failback is set to `True`, the device group or resource group will be switched off of the current primary and switched onto the rejoining node, making the rejoining node the new primary.

To ensure high availability of a failover resource group, make the resource group's node list match the node list of associated disk device groups. For a scalable resource group, the resource group's node list cannot always match the device group's node list because, currently, a device group's node list must contain exactly two nodes. For a greater-than-two-node cluster, the node list for the scalable resource group can have more than two nodes.

For example, assume that you have a disk device group, `disk-group-1`, that has nodes `phys-schost-1` and `phys-schost-2` in its node list, with the failback policy set to `Enabled`. Assume that you also have a failover resource group, `resource-group-1`, which uses `disk-group-1` to hold its application data. When you set up `resource-group-1`, also specify `phys-schost-1` and `phys-schost-2` for the resource group's node list, and set the failback policy to `True`.

To ensure high availability of a scalable resource group, make the scalable resource group's node list a superset of the node list for the disk device group. Doing so ensures that the nodes that are directly connected to the disks are also nodes that can run the scalable resource group. The advantage is that, when at least one cluster node connected to the data is up, the scalable resource group runs on that same node, making the scalable services available also.

See the *Sun Cluster Software Installation Guide for Solaris OS* for information on how to set up disk device groups. See the *Sun Cluster Concepts Guide for Solaris OS* document for more details on the relationship between disk device groups and resource groups.

Understanding `HAStorage` and `HAStoragePlus`

The `HAStorage` and the `HAStoragePlus` resource types can be used to configure the following options.

- Coordinate the boot order of disk devices and resource groups by causing the START methods of the other resources in the same resource group that contains the HAStorage or HAStoragePlus resource to wait until the disk device resources become available.
- With AffinityOn set to True, enforce colocation of resource groups and disk device groups on the same node, thus enhancing the performance of disk-intensive data services.

In addition, HAStoragePlus is capable of mounting any global file system found to be in an unmounted state. See "Planning the Cluster File System Configuration" on page 17 for more information.

Note – If the device group is switched to another node while the HAStorage or HAStoragePlus resource is online, AffinityOn has no effect and the resource group does **not** migrate along with the device group. On the other hand, if the resource group is switched to another node, AffinityOn being set to True causes the device group to follow the resource group to the new node.

See "Synchronizing the Startups Between Resource Groups and Disk Device Groups" on page 80 for information about the relationship between disk device groups and resource groups. The SUNW.HAStorage(5) and SUNW.HAStoragePlus(5) man pages provide additional details.

See "Enabling Highly Available Local File Systems" on page 86 for procedures for mounting of file systems such as VxFS in a local mode. The SUNW.HAStoragePlus(5) man page provides additional details.

Determining Whether Your Data Service Requires HAStorage or HAStoragePlus

- In cases where a data service resource group has a node list in which some of the nodes are not directly connected to the storage, you must configure HAStorage or HAStoragePlus resources in the resource group and set the dependency of the other data service resources to the HAStorage or HAStoragePlus resource. This requirement coordinates the boot order between the storage and the data services.
- If your data service is disk intensive, such as Sun Cluster HA for Oracle and Sun Cluster HA for NFS, ensure that you perform the following tasks.
  - Add a HAStorage or HAStoragePlus resource to your data service resource group.
  - Switch the HAStorage or HAStoragePlus resource online.
  - Set the dependency of your data service resources to the HAStorage or HAStoragePlus resource.

  - Set AffinityOn to True.

    When you perform these tasks, the resource groups and disk device groups are colocated on the same node.

  - The failback settings must be identical for both the resource group and device group(s).

- If your data service is **not** disk intensive—such as one that reads all of its files at startup (for example, Sun Cluster HA for DNS)—configuring the HAStorage or HAStoragePlus resource type is optional.

Choosing Between HAStorage and HAStoragePlus

To determine whether to create HAStorage or HAStoragePlus resources within a data service resource group, consider the following criteria.

- Use HAStoragePlus if you are using Sun Cluster 3.0 5/02 or Sun Cluster 3.1. (If you want to integrate any file system locally into a Sun Cluster configured for failover, you must upgrade to Sun Cluster 3.0 5/02 or Sun Cluster 3.1 and use the HAStoragePlus resource type. See "Planning the Cluster File System Configuration" on page 17 for more information.)
- Use HAStorage if you are using Sun Cluster 3.0 12/01 or earlier.

Considerations

Use the information in this section to plan the installation and configuration of any data service. The information in this section encourages you to think about the impact your decisions have on the installation and configuration of any data service. For specific considerations for a data service, see the documentation for the data service.

- When using data services that are I/O intensive and that have a large number of disks configured in the cluster, the application may experience delays due to retries within the I/O subsystem during disk failures. An I/O subsystem may take several minutes to retry and recover from a disk failure. This delay can result in Sun Cluster failing over the application to another node, even though the disk may have eventually recovered on its own. To avoid failover during these instances, consider increasing the default probe timeout of the data service. If you need more information or help with increasing data service timeouts, contact your local support engineer.
- For better performance, Install and configure your data service on the cluster nodes with direct connection to the storage.

- Client applications that run on cluster nodes should not map to logical IP addresses of an HA data service. After a failover, these logical IP addresses might no longer exist, leaving the client without a connection.

Node List Properties

You can specify the following three node lists when configuring data services.

1. `installed_nodes` – A property of the resource type for the data service. This property is a list of the cluster node names on which the resource type is installed and enabled to run.
2. `nodelist` – A property of a resource group that specifies a list of cluster node names where the group can be brought online, in order of preference. These nodes are known as the potential primaries or masters of the resource group. For failover services, configure only one resource group node list. For scalable services, configure two resource groups and thus two node lists. One resource group and its node list identifies the nodes on which the shared addresses are hosted. This list is a failover resource group on which the scalable resources depend. The other resource group and its list identifies nodes on which the application resources are hosted. The application resources depend on the shared addresses. Therefore, the node list for the resource group that contains the shared addresses must be a superset of the node list for the application resources.
3. `auxnodelist` – A property of a shared address resource. This property is a list of physical node IDs that identify cluster nodes that can host the shared address but never serve as primary in the case of failover. These nodes are mutually exclusive with the nodes identified in the node list of the resource group. This list pertains to scalable services only. See the `scrgadm`(1M) man page for details.

Overview of the Installation and Configuration Process

Use the following procedures to install and configure a data service.

- Install the data service packages from the installation medium on which the packages are supplied.
  - Sun Java™ Enterprise System CD
  - Sun Java Enterprise System Accessory CD Volume 3
- Install and configure the application to run in the cluster environment.

- Configure the resources and resource groups that the data service uses. When you configure a data service, specify the resource types, resources, and resource groups that the Resource Group Manager (RGM) will manage. The documentation for the individual data services describe these procedures.

Before you install and configure data services, see the *Sun Cluster Software Installation Guide for Solaris OS*, which includes procedures on how to install the data service software packages and how to configure Internet Protocol Network Multipathing (IP Networking Multipathing) groups that the network resources use.

Note – You can use SunPlex™ Manager to install and configure the following data services: Sun Cluster HA for Oracle, Sun Cluster HA for Sun Java System Web Server, Sun Cluster HA for Sun Java System Directory Server, Sun Cluster HA for Apache, Sun Cluster HA for DNS, and Sun Cluster HA for NFS. Note that Sun Cluster HA for Oracle and Sun Cluster HA for Apache are supported for use in only SPARC based clusters. See the SunPlex Manager online help for more information.

Installation and Configuration Task Flow

The following table shows a task map of the procedures to install and configure a Sun Cluster failover data service.

TABLE 1–1 Task Map: Sun Cluster Data Service Installation and Configuration

Task	For Instructions, Go to
Install the Solaris and Sun Cluster software	*Sun Cluster Software Installation Guide for Solaris OS*
Set up IP Networking Multipathing groups	*Sun Cluster Software Installation Guide for Solaris OS*
Set up multihost disks	*Sun Cluster Software Installation Guide for Solaris OS*
Plan resources and resource groups	Appendix C
Decide the location for application binaries, and configure the `nsswitch.conf` file	"Determining the Location of the Application Binaries" on page 16 "Verifying the `nsswitch.conf` File Contents" on page 17
Install and configure the application software	The appropriate Sun Cluster data services book.
Install the data service software packages	*Sun Cluster Software Installation Guide for Solaris OS* or the appropriate Sun Cluster data services book.

TABLE 1–1 Task Map: Sun Cluster Data Service Installation and Configuration *(Continued)*

Task	For Instructions, Go to
Register and configure the data service	The appropriate Sun Cluster data services book.

SPARC: Example

The example in this section shows how you might set up the resource types, resources, and resource groups for an Oracle application that has been instrumented to be a highly available failover data service.

The main difference between this example and an example of a scalable data service is that, in addition to the failover resource group that contains the network resources, a scalable data service requires a separate resource group (called a scalable resource group) for the application resources.

The Oracle application has two components, a server and a listener. Sun supplies the Sun Cluster HA for Oracle data service, and therefore these components have already been mapped into Sun Cluster resource types. Both of these resource types are associated with resources and resource groups.

Because this example is a failover data service, the example uses logical hostname network resources, which are the IP addresses that fail over from a primary node to a secondary node. Place the logical hostname resources into a failover resource group, and then place the Oracle server resources and listener resources into the same resource group. This ordering enables all of the resources to fail over as a group.

For Sun Cluster HA for Oracle run to on the cluster, you must define the following objects.

- `LogicalHostname` resource type – This resource type is built in, and therefore you do not need to explicitly register the resource type.
- Oracle resource types – Sun Cluster HA for Oracle defines two Oracle resource types—a database server and a listener.
- Logical hostname resources – These resources host the IP addresses that fail over in a node failure.
- Oracle resources – You must specify two resource instances for Sun Cluster HA for Oracle—a server and a listener.
- Failover resource group – This container is composed of the Oracle server and listener and logical hostname resources that will fail over as a group.

Tools for Data Service Resource Administration

This section describes the tools that you can use to perform installation and configuration tasks.

The SunPlex Manager Graphical User Interface (GUI)

SunPlex Manager is a web-based tool that enables you to perform the following tasks.

- Install a cluster.
- Administer a cluster.
- Create and configure resources and resource groups.
- Configure data services with the Sun Cluster software.

See the *Sun Cluster Software Installation Guide for Solaris OS* for instructions on how to use SunPlex Manager to install cluster software. SunPlex Manager provides online help for most administrative tasks.

SPARC: The Sun Cluster Module for the Sun Management Center GUI

The Sun Cluster module enables you to monitor clusters and to perform some operations on resources and resource groups from the Sun Management Center GUI. See the *Sun Cluster Software Installation Guide for Solaris OS* for information about installation requirements and procedures for the Sun Cluster module. Go to `http://docs.sun.com` to access the Sun Management Center software documentation set, which provides additional information about Sun Management Center.

The `scsetup` Utility

The `scsetup`(1M) utility is a menu-driven interface that you can use for general Sun Cluster administration. You can also use this utility to configure data service resources and resource groups. Select option 2 from the `scsetup` main menu to launch the Resource Group Manager submenu.

The scrgadm Command

You can use the scrgadm command to register and configure data service resources. See the procedure on how to register and configure your data service in the book for the data service. If, for example, you use Sun Cluster HA for Oracle, see the procedure in *Sun Cluster Data Service for Oracle Guide for Solaris OS*. Chapter 2 also contains information on how to use the scrgadm command to administer data service resources. Finally, see the scrgadm(1M) man page for additional information.

Data Service Resource Administration Tasks

The following table lists which tool you can use in addition to the command line for different data service resource administration tasks. See Chapter 2 for more information about these tasks and for details on how to use the command line to complete related procedures.

TABLE 1–2 Tools You Can Use for Data Service Resource Administration Tasks

Task	SunPlex Manager	SPARC: Sun Management Center	The scsetup Utility
Register a resource type	Yes	No	Yes
Create a resource group	Yes	No	Yes
Add a resource to a resource group	Yes	No	Yes
Bring a resource group online	Yes	Yes	No
Remove a resource group	Yes	Yes	No
Remove a resource	Yes	Yes	No
Switch the current primary of a resource group	Yes	No	No
Disable a resource	Yes	Yes	No
Move the resource group of a disabled resource into the unmanaged state	Yes	No	No
Display resource type, resource group, and resource configuration information	Yes	Yes	No
Change resource properties	Yes	No	No
Clear the STOP_FAILED error flag on resources	Yes	No	No
Add a node to a resource group	Yes	No	No

CHAPTER **2**

Administering Data Service Resources

This chapter describes how to use the scrgadm(1M) command to manage resources, resource groups, and resource types within the cluster. See "Tools for Data Service Resource Administration" on page 24 to determine if you can use other tools to complete a procedure.

This chapter contains the following procedures.

- "How to Set Up HAStorage Resource Type for New Resources" on page 80
- "How to Set Up HAStorage Resource Type for Existing Resources" on page 82
- "How to Set Up HAStoragePlus Resource Type" on page 86
- "How to Add File Systems to an Online HAStoragePlus Resource" on page 89
- "How to Remove File Systems From an Online HAStoragePlus Resource" on page 91
- "How to Recover From a Fault After Modifying an HAStoragePlus Resource" on page 94
- "How to Set Up an RGOffload Resource" on page 105
- "How to Replicate Configuration Data on a Cluster Without Configured Resource Groups, Resource Types, and Resources" on page 109
- "How to Upgrade Configuration Data on a Cluster With Configured Resource Groups, Resource Types, and Resources" on page 110

See Chapter 1 and the *Sun Cluster Concepts Guide for Solaris OS* document for overview information about resource types, resource groups, and resources.

Administering Data Service Resources

Table 2–1 lists the sections that describe the administration tasks for data service resources.

TABLE 2–1 Task Map: Data Service Administration

Task	For Instructions, Go To …
Register a resource type	"How to Register a Resource Type" on page 31
Upgrade a resource type	"How to Migrate Existing Resources to a New Version of the Resource Type" on page 34 "How to Install and Register an Upgrade of a Resource Type" on page 33
Create failover or scalable resource groups	"How to Create a Failover Resource Group" on page 39 "How to Create a Scalable Resource Group" on page 40

TABLE 2–1 Task Map: Data Service Administration *(Continued)*

Task	For Instructions, Go To …
Add logical hostnames or shared addresses and data service resources to resource groups	"How to Add a Logical Hostname Resource to a Resource Group" on page 42 "How to Add a Shared Address Resource to a Resource Group" on page 44 "How to Add a Failover Application Resource to a Resource Group" on page 46 "How to Add a Scalable Application Resource to a Resource Group" on page 48
Enable resources and resource monitors, manage the resource group, and bring the resource group and its associated resources online	"How to Bring Online Resource Groups" on page 51
Disable and enable resource monitors independent of the resource	"How to Disable a Resource Fault Monitor" on page 53 "How to Enable a Resource Fault Monitor" on page 53
Remove resource types from the cluster	"How to Remove a Resource Type" on page 54
Remove resource groups from the cluster	"How to Remove a Resource Group" on page 55
Remove resources from resource groups	"How to Remove a Resource" on page 57
Switch the primary for a resource group	"How to Switch the Current Primary of a Resource Group" on page 58
Disable resources and move their resource group into the `UNMANAGED` state	"How to Disable a Resource and Move Its Resource Group Into the `UNMANAGED` State" on page 60
Display resource type, resource group, and resource configuration information	"Displaying Resource Type, Resource Group, and Resource Configuration Information" on page 62
Change resource type, resource group, and resource properties	"How to Change Resource Type Properties" on page 63 "How to Change Resource Group Properties" on page 64 "How to Change Resource Properties" on page 65
Clear error flags for failed Resource Group Manager (RGM) processes	"How to Clear the `STOP_FAILED` Error Flag on Resources" on page 67

TABLE 2–1 Task Map: Data Service Administration *(Continued)*

Task	For Instructions, Go To …
Reregister the built-in resource types LogicalHostname and SharedAddress	"How to Reregister Preregistered Resource Types After Inadvertent Deletion" on page 71
Upgrade the built-in resource types LogicalHostname and SharedAddress	"Upgrading a Resource Type" on page 33 "Upgrading a Preregistered Resource Type" on page 69
Update the network interface ID list for the network resources, and update the node list for the resource group	"Adding a Node to a Resource Group" on page 72
Remove a node from a resource group	"Removing a Node From a Resource Group" on page 75
Set up HAStorage or HAStoragePlus for resource groups so as to synchronize the startups between those resource groups and disk device groups	"How to Set Up HAStorage Resource Type for New Resources" on page 80
Set up HAStoragePlus to enable highly available local file systems for failover data services with high I/O disk intensity	"How to Set Up HAStoragePlus Resource Type" on page 86
Modify online the resource for a highly available file system	"Modifying Online the Resource for a Highly Available File System" on page 88
Upgrade the HAStoragePlus resource type	"Upgrading a Resource Type" on page 33 "Upgrading the HAStoragePlus Resource Type" on page 95
Distribute online resource groups among cluster nodes	"Distributing Online Resource Groups Among Cluster Nodes" on page 97
Configure a resource type to automatically free up a node for a critical data service.	"How to Set Up an RGOffload Resource" on page 105
Replicate and upgrade configuration data for resource groups, resource types, and resources	"Replicating and Upgrading Configuration Data for Resource Groups, Resource Types, and Resources" on page 109
Tune fault monitors for Sun Cluster data services	"Tuning Fault Monitors for Sun Cluster Data Services" on page 111

Note – The procedures in this chapter describe how to use the scrgadm(1M) command to complete these tasks. Other tools also enable you to administer your resources. See "Tools for Data Service Resource Administration" on page 24 for details about these options.

Configuring and Administering Sun Cluster Data Services

Configuring a Sun Cluster data service is a single task composed of several procedures. These procedures enable you to perform the following tasks.

- Register a resource type.
- Upgrade a resource type.
- Create resource groups.
- Add resources into the resource groups.
- Bring the resources online.

Use the procedures in this chapter to update your data service configuration after the initial configuration. For example, to change resource type, resource group, and resource properties, go to "Changing Resource Type, Resource Group, and Resource Properties" on page 62.

Registering a Resource Type

A resource type provides specification of common properties and callback methods that apply to all of the resources of the given type. You must register a resource type before you create a resource of that type. See Chapter 1 for details about resource types.

▼ How to Register a Resource Type

To complete this procedure, you must supply the name for the resource type that you plan to register. The resource type name is an abbreviation for the data service name. For information about resource type names of data services that are supplied with Sun Cluster, see the release notes for your release of Sun Cluster.

See the scrgadm(1M) man page for additional information.

Note – Perform this procedure from any cluster node.

1. **Become superuser on a cluster member.**
2. **Register the resource type.**

   ```
   # scrgadm -a -t resource-type
   ```

   -a
   Adds the specified resource type.

   -t *resource-type*
   Specifies name of the resource type to add. See the release notes for your release of Sun Cluster to determine the predefined name to supply.

3. **Verify that the resource type has been registered.**

   ```
   # scrgadm -pv -t resource-type
   ```

Example – Registering Resource Types

The following example registers Sun Cluster HA for Sun Java System Web Server (internal name iws).

```
# scrgadm -a -t SUNW.iws
# scrgadm -pv -t SUNW.iws
Res Type name:                                  SUNW.iws
  (SUNW.iws) Res Type description:              None registered
  (SUNW.iws) Res Type base directory:           /opt/SUNWschtt/bin
  (SUNW.iws) Res Type single instance:          False
  (SUNW.iws) Res Type init nodes:               All potential masters
  (SUNW.iws) Res Type failover:                 False
  (SUNW.iws) Res Type version:                  1.0
  (SUNW.iws) Res Type API version:              2
  (SUNW.iws) Res Type installed on nodes:       All
  (SUNW.iws) Res Type packages:                 SUNWschtt
```

Where to Go From Here

After registering resource types, you can create resource groups and add resources to the resource group. See "Creating a Resource Group" on page 38 for details.

Upgrading a Resource Type

As newer versions of resource types are released, you will want to install and register the upgraded resource type. You may also want to upgrade your existing resources to the newer resource type versions. This section provides the following procedures for installing and registering an upgraded resource type and for upgrading an existing resource to a new resource type version.

- "How to Install and Register an Upgrade of a Resource Type" on page 33
- "How to Migrate Existing Resources to a New Version of the Resource Type" on page 34

▼ How to Install and Register an Upgrade of a Resource Type

This procedure can also be performed using the Resource Group option of `scsetup`. For information on `scsetup`, see the `scsetup`(1M) man page.

1. **Install the resource type upgrade package on all cluster nodes.**

   Note – If the resource type package is not installed on all of the nodes, then an additional step will be required (Step 3).

   The upgrade documentation will indicate whether it is necessary to boot a node in non-cluster mode to install the resource type upgrade package. To avoid down time, add the new package in a rolling upgrade fashion on one node at a time, while the node is booted in non-cluster mode and the other nodes are in cluster mode.

2. **Register the new resource type version.**

   ```
   scrgadm -a -t resource_type -f path_to_new_RTR_file
   ```

   The new resource type will have a name in the following format.

   ```
   vendor_id.rtname:version
   ```

   Use `scrgadm -p` or `scrgadm -pv` (verbose) to display the newly registered resource type.

3. **If the new resource type is not installed on all of the nodes, set the `Installed_nodes` property to the nodes on which it is actually installed.**

   ```
   # scrgadm -c -t resource_type -h installed_node_list
   ```

A new version of a resource type may differ from a previous version in the following ways.

- The settings of the resource type properties may change.
- The set of declared resource properties, including standard and extension properties, may change.
- The attributes of resource properties, such as default, min, max, arraymin, arraymax, or tunability may change.
- The set of declared methods may differ.
- The implementation of methods and/or monitor may change.

▼ How to Migrate Existing Resources to a New Version of the Resource Type

This procedure can also be performed using the Resource Group option of `scsetup`. For information on `scsetup`, see the `scsetup`(1M) man page.

The existing resource type version and the changes in the new version determine how to migrate to the new version type. The resource type upgrade documentation will tell you whether the migration can occur. If a migration is not supported, consider deleting the resource and replacing it with a new resource of the upgraded version or leaving the resource at the old version of the resource type.

When you migrate the existing resource, the following values may change.

Default property values

If an upgraded version of the resource type declares a new default value for a defaulted property, the new default value will be inherited by existing resources.

The new resource type version's `VALIDATE` method checks to make sure that existing property settings are appropriate. If the settings are not appropriate, edit the properties of the existing resource to appropriate values. To edit the properties, see Step 3 .

Resource type name

The RTR file contains the following properties that are used to form the fully qualified name of the resource type.

- `Vendor_id`
- `Resource_type`
- `RT_Version`

When you register the upgraded version of the resource type, its name will be stored as `vendor_id.rtname:version`. A resource that has been migrated to a new version will have a new `Type` property, composed of the properties listed above.

Resource `type_version` property

The standard resource property `Type_version` stores the `RT_Version` property of a resource's type. The `Type_Version` property does not appear in the RTR file. Edit the `Type_Version` property using the following command.

```
scrgadm -c -j resource -y Type_version=new_version
```

1. **Before migrating an existing resource to a new version of the resource type, read the upgrade documentation accompanying the new resource type to determine whether the migration can take place.**

   The documentation will specify when the migration must take place.

   - Any time
   - When the resource is unmonitored
   - When the resource is offline
   - When the resource is disabled
   - When the resource group is unmanaged

   Note – After migrating a resource that can be migrated at any time, the resource probe might not display the correct resource type version. In this situation, disable and re-enable the resource's fault monitor to ensure that the resource probe displays the correct resource type version.

   If the migration is not supported, you must delete the resource and replace it with a new resource of the upgraded version, or leave the resource at the old version of the resource type.

2. **For each resource of the resource type that is to be migrated, change the state of the resource or its resource group to the appropriate state as dictated by the upgrade documentation.**

   For example, if the resource needs to be unmonitored

   ```
   scswitch -M -n -j resource
   ```

   If the resource needs to be offline

   ```
   scswitch -n -j resource
   ```

   If the resource needs to be disabled

   ```
   scswitch -n -j resource
   ```

   If the resource group needs to be unmanaged

   ```
   scsswitch -n -j resource-group
   scswitch -F -g resource_group
   scswitch -u -g resource_group
   ```

3. **For each resource of the resource type that is to be migrated, edit the resource, changing its `Type_version` property to the new version.**

   ```
   scrgadm -c -j resource -y Type_version=new_version \
   -x extension_property=new_value -y extension_property=new_value
   ```

If necessary, edit other properties of the same resource to appropriate values in the same command by adding additional -x or -y options on the command line.

4. **Restore the previous state of the resource or resource group by reversing the command typed in Step 2.**

   For example, to make the resource monitored again

   ```
   scswitch -M -e -j resource
   ```

   To re-enable the resource

   ```
   scswitch -e -j resource
   ```

   To make the resource group managed and online

   ```
   scswitch -o -g resource_group
   scswitch -Z -g resource_group
   ```

Example 1 – Migrating an Existing Resource to a New Resource Type Version

This example shows the migration of an existing resource to a new resource type version. Note that the new resource type package contains methods located in new paths. Because the methods will not be overwritten during the installation, the resource does not need to be disabled until after the upgraded resource type is installed.

This examples assumes the following.

- New resource type version is 2.0
- Tunability from previous version is “when_offline”
- Resource name is “myresource”
- Resource type name is “myrt”
- New RTR file is in `/opt/XYZmyrt/etc/XYZ.myrt`
- There are no dependencies on the resource to be migrated
- The resource to be migrated can be taken offline while leaving the containing resource group online

```
(Install the new package on all nodes according to vendor's directions.)
# scrgadm -a -t myrt -f /opt/XYZmyrt/etc/XYZ.myrt
# scswitch -n -j myresource
# scrgadm -c -j myresource -y Type_version=2.0
# scswitch -e -j myresource
```

Example 2 – Migrating an Existing Resource to a New Resource Type Version

This example shows the migration of an existing resource to a new resource type version. Note that the new resource type package contains only the monitor and RTR file. Because the monitor will be overwritten during installation, the resource must be disabled before the upgraded resource type is installed.

This example assumes the following.

- New resource type version is 2.0
- Tunability from previous version is "when_unmonitored"
- Resource name is "myresource"
- Resource type name is "myrt"
- New RTR file is in `/opt/XYZmyrt/etc/XYZ.myrt`

```
# scswitch -M -n -j myresource
(Install the new package according to vendor's directions.)
# scrgadm -a -t myrt -f /opt/XYZmyrt/etc/XYZ.myrt
# scrgadm -c -j myresource -y Type_version=2.0
# scswitch -M -e -j myresourcee
```

Downgrading a Resource Type

You can downgrade a resource to an older version of its resource type. The conditions under which you can downgrade a resource to an older version of the resource type are more restrictive than when you upgrade to a newer version of the resource type. You must first unmanage the resource group. In addition, you can only downgrade a resource to an upgrade-enabled version of the resource type. You can identify upgrade-enabled versions by using the `scrgadm -p` command. In the output, upgrade-enabled versions contain the suffix `:`*version*.

▼ How to Downgrade a Resource to an Older Version of Its Resource Type

You can downgrade a resource to an older version of its resource type. The conditions under which you can downgrade a resource to an older version of the resource type are more restrictive than when you upgrade to a newer version of the resource type. You must first unmanage the resource group. In addition, you can only downgrade a resource to an upgrade-enabled version of the resource type. You can identify upgrade-enabled versions by using the `scrgadm -p` command. In the output, upgrade-enabled versions contain the suffix `:`*version*.

1. **Switch the resource group that contains the resource you want to downgrade offline.**

   ```
   scswitch -F -g resource_group
   ```

2. **Disable the resource that you want to downgrade and all resources in the resource group.**

   ```
   scswitch -n -j resource_to_downgrade
   scswitch -n -j resource1
   scswitch -n -j resource2
   scswitch -n -j resource3
   . . .
   ```

   Note – Disable resources in order of dependency, starting with the most dependent (application resources) and ending with the least dependent (network address resources).

   ---

3. **Unmanage the resource group.**

   ```
   scswitch -u -g resource_group
   ```

4. **Is the old version of the resource type to which you want to downgrade still registered in the cluster?**

   - If yes, go to the next step.
   - If no, reregister the old version that you want.

     ```
     scrgadm -a -t resource_type_name
     ```

5. **Downgrade the resource by specifying the old version that you want for `Type_version`.**

   ```
   scrgadm -c -j resource_to_downgrade -y Type_version=old_version
   ```

   If necessary, edit other properties of the same resource to appropriate values in the same command.

6. **Bring the resource group that contains the resource that you downgraded to a managed state, enable all the resources, and switch the group online.**

   ```
   scswitch -Z -g resource_group
   ```

Creating a Resource Group

A resource group contains a set of resources, all of which are brought online or offline together on a given node or set of nodes. You must create an empty resource group before you place resources into it.

The two resource group types are **failover** and **scalable**. A failover resource group can be online on one node only at any time, while a scalable resource group can be online on multiple nodes simultaneously.

The following procedure describes how to use the `scrgadm`(1M) command to register and configure your data service.

See Chapter 1 and the *Sun Cluster Concepts Guide for Solaris OS* document for conceptual information on resource groups.

▼ How to Create a Failover Resource Group

A failover resource group contains network addresses, such as the built-in resource types `LogicalHostname` and `SharedAddress`, as well as failover resources, such as the data service application resources for a failover data service. The network resources, along with their dependent data service resources, move between cluster nodes when data services fail over or are switched over.

See the `scrgadm`(1M) man page for additional information.

Note – Perform this procedure from any cluster node.

1. **Become superuser on a cluster member.**

2. **Create the failover resource group.**

   ```
   # scrgadm -a -g resource-group [-h nodelist]
   ```

   `-a` Adds the specified resource group.

   `-g` *resource-group* Specifies your choice of the name of the failover resource group to add. This name must begin with an ASCII character.

   `-h` *nodelist* Specifies an optional, ordered list of nodes that can master this resource group. If you do not specify this list, it defaults to all of the nodes in the cluster.

3. **Verify that the resource group has been created.**

   ```
   # scrgadm -pv -g resource-group
   ```

Example – Creating a Failover Resource Group

This example shows the addition of a failover resource group (`resource-group-1`) that two nodes (`phys-schost-1` and `phys-schost-2`) can master.

```
# scrgadm -a -g resource-group-1 -h phys-schost1,phys-schost-2
# scrgadm -pv -g resource-group-1
```

```
Res Group name:                                          resource-group-1
  (resource-group-1) Res Group RG_description:           <NULL>
  (resource-group-1) Res Group management state:         Unmanaged
  (resource-group-1) Res Group Failback:                 False
  (resource-group-1) Res Group Nodelist:                 phys-schost-1
                                                         phys-schost-2

  (resource-group-1) Res Group Maximum_primaries:        1
  (resource-group-1) Res Group Desired_primaries:        1
  (resource-group-1) Res Group RG_dependencies:          <NULL>
  (resource-group-1) Res Group mode:                     Failover
  (resource-group-1) Res Group network dependencies:     True
  (resource-group-1) Res Group Global_resources_used:    All
  (resource-group-1) Res Group Pathprefix:
```

Where to Go From Here

After you create a failover resource group, you can add application resources to this resource group. See "Adding Resources to Resource Groups" on page 42 for the procedure.

▼ How to Create a Scalable Resource Group

A scalable resource group is used with scalable services. The shared address feature is the Sun Cluster networking facility that enables the multiple instances of a scalable service to appear as a single service. You must first create a failover resource group that contains the shared addresses on which the scalable resources depend. Next, create a scalable resource group, and add scalable resources to that group.

See the scrgadm(1M) man page for additional information.

Note – Perform this procedure from any cluster node.

---

1. **Become superuser on a cluster member.**

2. **Create the failover resource group that holds the shared addresses that the scalable resource will use.**

3. **Create the scalable resource group.**

```
# scrgadm -a -g resource-group \
-y Maximum_primaries=m \
-y Desired_primaries=n \
-y RG_dependencies=depend-resource-group \
-h nodelist]
```

`-a`
Adds a scalable resource group.

`-g` *resource-group*
Specifies your choice of the name of the scalable resource group to add.

`-y Maximum_primaries=`*m*
Specifies the maximum number of active primaries for this resource group.

`-y Desired_primaries=`*n*
Specifies the number of active primaries on which the resource group should attempt to start.

`-y RG_dependencies=`*depend-resource-group*
Identifies the resource group that contains the shared address resource on which the resource group that is being created depends.

`-h` *nodelist*
Specifies an optional list of nodes on which this resource group is to be available. If you do not specify this list, the value defaults to all of the nodes.

4. **Verify that the scalable resource group has been created.**

```
# scrgadm -pv -g resource-group
```

Example – Creating a Scalable Resource Group

This example shows the addition of a scalable resource group (`resource-group-1`) to be hosted on two nodes (`phys-schost-1`, `phys-schost-2`). The scalable resource group depends on the failover resource group (`resource-group-2`) that contains the shared addresses.

```
# scrgadm -a -g resource-group-1 \
-y Maximum_primaries=2 \
-y Desired_primaries=2 \
-y RG_dependencies=resource-group-2 \
-h phys-schost-1,phys-schost-2
# scrgadm -pv -g resource-group-1
Res Group name:                                        resource-group-1
  (resource-group-1) Res Group RG_description:         <NULL>
  (resource-group-1) Res Group management state:       Unmanaged
  (resource-group-1) Res Group Failback:               False
  (resource-group-1) Res Group Nodelist:               phys-schost-1
                                                       phys-schost-2
  (resource-group-1) Res Group Maximum_primaries:      2
  (resource-group-1) Res Group Desired_primaries:      2
```

```
(resource-group-1) Res Group RG_dependencies:          resource-group-2
(resource-group-1) Res Group mode:                     Scalable
(resource-group-1) Res Group network dependencies:     True
(resource-group-1) Res Group Global_resources_used:    All
(resource-group-1) Res Group Pathprefix:
```

Where to Go From Here

After you have created a scalable resource group, you can add scalable application resources to the resource group. See "How to Add a Scalable Application Resource to a Resource Group" on page 48 for details.

Adding Resources to Resource Groups

A resource is an instantiation of a resource type. You must add resources to a resource group before the RGM can manage the resources. This section describes the following three resource types.

- logical hostname resources
- shared address resources
- data service (application) resources

Always add logical hostname resources and shared address resources to failover resource groups. Add data service resources for failover data services to failover resource groups. Failover resource groups contain both the logical hostname resources and the application resources for the data service. Scalable resource groups contain only the application resources for scalable services. The shared address resources on which the scalable service depends must reside in a separate failover resource group. You must specify dependencies between the scalable application resources and the shared address resources for the data service to scale across cluster nodes.

See the *Sun Cluster Concepts Guide for Solaris OS* document and Chapter 1 for more information on resources.

▼ How to Add a Logical Hostname Resource to a Resource Group

To complete this procedure, you must supply the following information.

- The name of the failover resource group into which you are adding the resource
- The hostnames that you plan to add to the resource group

Note – When you add a logical hostname resource to a resource group, the extension properties of the resource are set to their default values. To specify a nondefault value, you must modify the resource after you add the resource to a resource group. For more information, see "How to Modify a Logical Hostname Resource or a Shared Address Resource" on page 66.

See the `scrgadm`(1M) man page for additional information.

Note – Perform this procedure from any cluster node.

---

1. **Become superuser on a cluster member.**
2. **Add the logical hostname resource to the resource group.**

```
# scrgadm -a -L [-j resource] -g resource-group -l hostnamelist, ... [-n netiflist]
```

`-a` Adds a logical hostname resource.

`-L` Specifies the logical hostname resource form of the command.

`-j` *resource* Specifies an optional resource name of your choice. If you do not specify this option, the name defaults to the first hostname that is specified with the `-l` option.

`-g` *resource-group* Specifies the name of the resource group in which this resource resides.

`-l` *hostnamelist, …* Specifies a comma-separated list of UNIX hostnames (logical hostnames) by which clients communicate with services in the resource group.

`-n` *netiflist* Specifies an optional, comma-separated list that identifies the IP Networking Multipathing groups that are on each node. Each element in *netiflist* must be in the form of `netif@node`. `netif` can be given as an IP Networking Multipathing group name, such as `sc_ipmp0`. The node can be identified by the node name or node ID, such as `sc_ipmp0@1` or `sc_ipmp@phys-schost-1`.

Note – Sun Cluster does not currently support using the adapter name for `netif`.

---

3. **Verify that the logical hostname resource has been added.**

```
# scrgadm -pv -j resource
```

Adding the resource causes the Sun Cluster software to validate the resource. If the validation succeeds, you can enable the resource, and you can move the resource group into the state where the RGM manages it. If the validation fails, the `scrgadm` command produces an error message and exits. If the validation fails, check the `syslog` on each node for an error message. The message appears on the node that performed the validation, not necessarily the node on which you ran the `scrgadm` command.

Example – Adding a Logical Hostname Resource to a Resource Group

This example shows the addition of logical hostname resource (`resource-1`) to a resource group (`resource-group-1`).

```
# scrgadm -a -L -j resource-1 -g resource-group-1 -l schost-1
# scrgadm -pv -j resource-1
Res Group name: resource-group-1
(resource-group-1) Res name:                              resource-1
  (resource-group-1:resource-1) Res R_description:
  (resource-group-1:resource-1) Res resource type:        SUNW.LogicalHostname
  (resource-group-1:resource-1) Res resource group name:  resource-group-1
  (resource-group-1:resource-1) Res enabled:              False
  (resource-group-1:resource-1) Res monitor enabled:      True
```

Where to Go From Here

After you add logical hostname resources, use the procedure "How to Bring Online Resource Groups" on page 51 to bring them online.

▼ How to Add a Shared Address Resource to a Resource Group

To complete this procedure, you must supply the following information.

- The name of the resource group into which you are adding the resource. This group must be a failover resource group that you created previously.
- The hostnames that you plan to add to the resource group.

Note – When you add a shared address resource to a resource group, the extension properties of the resource are set to their default values. To specify a nondefault value, you must modify the resource after you add the resource to a resource group. For more information, see "How to Modify a Logical Hostname Resource or a Shared Address Resource" on page 66.

See the scrgadm(1M) man page for additional information.

Note – Perform this procedure from any cluster node.

1. **Become superuser on a cluster member.**
2. **Add the shared address resource to the resource group.**

```
# scrgadm -a -S [-j resource] -g resource-group -l hostnamelist, … \
[-X auxnodelist] [-n netiflist]
```

-a
Adds shared address resources.

-S
Specifies the shared address resource form of the command.

-j *resource*
Specifies an optional resource name of your choice. If you do not specify this option, the name defaults to the first hostname that is specified with the -l option.

-g *resource-group*
Specifies the resource group name.

-l *hostnamelist*, …
Specifies a comma-separated list of shared address hostnames.

-X *auxnodelist*
Specifies a comma-separated list of physical node names or IDs that identify the cluster nodes that can host the shared address but never serve as primary if failover occurs. These nodes are mutually exclusive, with the nodes identified as potential masters in the resource group's node list.

-n *netiflist*
Specifies an optional, comma-separated list that identifies the IP Networking Multipathing groups that are on each node. Each element in *netiflist* must be in the form of netif@node. netif can be given as an IP Networking Multipathing group name, such as sc_ipmp0. The node can be identified by the node name or node ID, such as sc_ipmp0@1 or sc_ipmp@phys-schost-1.

Note – Sun Cluster does not currently support using the adapter name for `netif`.

3. **Verify that the shared address resource has been added and validated.**

   ```
   # scrgadm -pv -j resource
   ```

   Adding the resource causes the Sun Cluster software to validate the resource. If the validation succeeds, you can enable the resource, and you can move the resource group into the state where the RGM manages it. If the validation fails, the `scrgadm` command produces an error message and exits. If the validation fails, check the `syslog` on each node for an error message. The message appears on the node that performed the validation, not necessarily the node on which you ran the `scrgadm` command.

Example – Adding a Shared Address Resource to a Resource Group

This example shows the addition of a shared address resource (`resource-1`) to a resource group (`resource-group-1`).

```
# scrgadm -a -S -j resource-1 -g resource-group-1 -l schost-1
# scrgadm -pv -j resource-1
(resource-group-1) Res name:                              resource-1
    (resource-group-1:resource-1) Res R_description:
    (resource-group-1:resource-1) Res resource type:       SUNW.SharedAddress
    (resource-group-1:resource-1) Res resource group name:  resource-group-1
    (resource-group-1:resource-1) Res enabled:             False
    (resource-group-1:resource-1) Res monitor enabled:     True
```

Where to Go From Here

After you add a shared resource, use the procedure "How to Bring Online Resource Groups" on page 51 to enable the resource.

▼ How to Add a Failover Application Resource to a Resource Group

A failover application resource is an application resource that uses logical hostnames that you previously created in a failover resource group.

To complete this procedure, you must supply the following information.

- The name of the failover resource group into which you are adding the resource
- The name of the resource type for the resource
- The logical hostname resources that the application resource uses, which are the logical hostnames that you previously included in the same resource group

See the `scrgadm`(1M) man page for additional information.

Note – Perform this procedure from any cluster node.

1. **Become superuser on a cluster member.**

2. **Add a failover application resource to the resource group.**

```
# scrgadm -a -j resource -g resource-group -t resource-type \
[-x Extension_property=value, ...] [-y Standard_property=value, ...]
```

`-a` Adds a resource.

`-j` *resource* Specifies your choice of the name of the resource to add.

`-g` *resource-group* Specifies the name of the failover resource group created previously.

`-t` *resource-type* Specifies the name of the resource type for the resource.

`-x` *Extension_property=value, …* Specifies a comma-separated list of extension properties that depend on the particular data service. See the documentation for each data service to determine whether the data service requires this property.

`-y` *Standard_property=value, …* Specifies a comma-separated list of standard properties that depends on the particular data service. See the documentation for each data service and Appendix A to determine whether the data service requires this property.

Note – You can set additional properties. See Appendix A and the documentation in this book on how to install and configure your failover data service for details.

3. **Verify that the failover application resource has been added and validated.**

```
# scrgadm -pv -j resource
```

Adding the resource causes the Sun Cluster software to validate the resource. If the validation succeeds, you can enable the resource, and you can move the resource group into the state where the RGM manages it. If the validation fails, the scrgadm command produces an error message and exits. If the validation fails, check the syslog on each node for an error message. The message appears on the node that performed the validation, not necessarily the node on which you ran the scrgadm command.

Example – Adding a Failover Application Resource to a Resource Group

This example shows the addition of a resource (resource-1) to a resource group (resource-group-1). The resource depends on logical hostname resources (schost-1, schost-2), which must reside in the same failover resource groups that you defined previously.

```
# scrgadm -a -j resource-1 -g resource-group-1 -t resource-type-1 \
-y Network_resources_used=schost-1,schost2 \
# scrgadm -pv -j resource-1
(resource-group-1) Res name:                              resource-1
    (resource-group-1:resource-1) Res R_description:
    (resource-group-1:resource-1) Res resource type:        resource-type-1
    (resource-group-1:resource-1) Res resource group name:  resource-group-1
    (resource-group-1:resource-1) Res enabled:              False
    (resource-group-1:resource-1) Res monitor enabled:      True
```

Where to Go From Here

After you add a failover application resource, use the procedure "How to Bring Online Resource Groups" on page 51 to enable the resource.

▼ How to Add a Scalable Application Resource to a Resource Group

A scalable application resource is an application resource that uses shared addresses in a failover resource group.

To complete this procedure, you must supply the following information:

- The name of the scalable resource group into which you are adding the resource
- The name of the resource type for the resource
- The shared address resources that the scalable service resource uses, which are the shared addresses that you previously included in a failover resource group

See the `scrgadm`(1M) man page for additional information.

Note – Perform this procedure from any cluster node.

---

1. **Become superuser on a cluster member.**
2. **Add a scalable application resource to the resource group.**

```
# scrgadm -a -j resource -g resource-group -t resource-type \
-y Network_resources_used=network-resource[,network-resource...] \
-y Scalable=True
[-x Extension_property=value, ...] [-y Standard_property=value, ...]
```

`-a`
Adds a resource.

`-j` *resource*
Specifies your choice of the name of the resource to add.

`-g` *resource-group*
Specifies the name of a scalable service resource group that you previously created.

`-t` *resource-type*
Specifies the name of the resource type for this resource.

`-y Network_resources_used=` *network-resource*[,*network-resource*...]
Specifies the list of network resources (shared addresses) on which this resource depends.

`-y Scalable=True`
Specifies that this resource is scalable.

`-x` *Extension_property=value, …*
Specifies a comma-separated list of extension properties that depend on the particular data service. See the documentation for each data service to determine whether the data service requires this property.

`-y` *Standard_property=value, …*
Specifies a comma-separated list of standard properties that depends on the particular data service. See the documentation for each data service and Appendix A to determine whether the data service requires this property.

`-y` *Standard_property=value, …*
Specifies a comma-separated list of standard properties that depends on the particular data service. See the documentation for each data service and Appendix A to determine whether the data service requires this property.

Note – You can set additional properties. See Appendix A and the documentation in this book on how to install and configure your scalable data service for information on other configurable properties. Specifically for scalable services, you typically set the Port_list, Load_balancing_weights, and Load_balancing_policy properties, which Appendix A describes.

3. **Verify that the scalable application resource has been added and validated.**

   ```
   # scrgadm -pv -j resource
   ```

   Adding the resource causes the Sun Cluster software to validate the resource. If the validation succeeds, you can enable the resource, and you can move the resource group into the state where the RGM manages it. If the validation fails, the scrgadm command produces an error message and exits. If the validation fails, check the syslog on each node for an error message. The message appears on the node that performed the validation, not necessarily the node on which you ran the scrgadm command.

Example – Adding a Scalable Application Resource to a Resource Group

This example shows the addition of a resource (resource-1) to a resource group (resource-group-1). Note that resource-group-1 depends on the failover resource group that contains the network addresses that are in use (schost-1 and schost-2 in the following example). The resource depends on shared address resources (schost-1, schost-2), which must reside in one or more failover resource groups that you defined previously.

```
# scrgadm -a -j resource-1 -g resource-group-1 -t resource-type-1 \
-y Network_resources_used=schost-1,schost-2 \
-y Scalable=True
# scrgadm -pv -j resource-1
(resource-group-1) Res name:                                resource-1
    (resource-group-1:resource-1) Res R_description:
    (resource-group-1:resource-1) Res resource type:        resource-type-1
    (resource-group-1:resource-1) Res resource group name:  resource-group-1
    (resource-group-1:resource-1) Res enabled:              False
    (resource-group-1:resource-1) Res monitor enabled:      True
```

Where to Go From Here

After you add a scalable application resource, follow the procedure "How to Bring Online Resource Groups" on page 51 to enable the resource.

Bringing Online Resource Groups

To enable resources to begin providing HA services, you must enable the resources in their resource groups, enable the resource monitors, make the resource groups managed, and bring online the resource groups. You can perform these tasks individually or by using the following procedure. See the `scswitch`(1M) man page for details.

Note – Perform this procedure from any cluster node.

▼ How to Bring Online Resource Groups

1. **Become superuser on a cluster member.**

2. **Enable the resources, and bring online the resource groups.**

   ```
   # scswitch -Z -g rg-list
   ```

   If the resource monitors were disabled, they are enabled also.

   Note – If you have intentionally disabled a resource a or a fault monitor that must remain disabled, specify the `-z` instead of the `-Z` option.

   `-Z` Brings resource groups online by first enabling their resources and fault monitors.

   `-g` *rg-list* Specifies a comma-separated list of the names of the resource groups to bring online. The resource groups must exist. The list may contain one resource group name or more than one resource group name.

   You can omit the `-g` *rg-list* option. If you omit this option, all resource groups are brought online.

   Note – If any resource group that you are bringing online declares a strong affinity for other resource groups, this operation might fail. For more information, see "Distributing Online Resource Groups Among Cluster Nodes" on page 97.

   ---

3. **Verify that the resource is online.**

Run the following command on any cluster node, and check the resource group state field to verify that each resource group is online on the nodes that are specified in the node list.

```
# scstat -g
```

Example – Bring a Resource Group Online

This example shows how to bring a resource group (`resource-group-1`) online and verify its status.

```
# scswitch -Z -g resource-group-1
# scstat -g
```

Where to Go From Here

After you bring a resource group online, it is configured and ready for use. If a resource or node fails, the RGM switches the resource group online on alternate nodes to maintain availability of the resource group.

Disabling and Enabling Resource Monitors

The following procedures disable or enable resource fault monitors, not the resources themselves. A resource can continue to operate normally while its fault monitor is disabled. However, if the fault monitor is disabled and a data service fault occurs, automatic fault recovery is not initiated.

See the `scswitch`(1M) man page for additional information.

Note – Run this procedure from any cluster node.

How to Disable a Resource Fault Monitor

1. **Become superuser on a cluster member.**

2. **Disable the resource fault monitor.**

   ```
   # scswitch -n -M -j resource
   ```

   `-n` Disable a resource or resource monitor.

   `-M` Disable the fault monitor for the specified resource.

   `-j` *resource* The name of the resource.

3. **Verify that the resource fault monitor has been disabled.**

   Run the following command on each cluster node, and check for monitored fields (`RS Monitored`).

   ```
   # scrgadm -pv
   ```

Example–Disabling a Resource Fault Monitor

This example shows how to disable a resource fault monitor.

```
# scswitch -n -M -j resource-1
# scrgadm -pv
...
RS Monitored: no...
```

How to Enable a Resource Fault Monitor

1. **Become superuser on a cluster member.**

2. **Enable the resource fault monitor.**

   ```
   # scswitch -e -M -j resource
   ```

   `-e` Enables a resource or resource monitor.

   `-M` Enables the fault monitor for the specified resource.

   `-j` *resource* Specifies the name of the resource.

3. **Verify that the resource fault monitor has been enabled.**

   Run the following command on each cluster node, and check for monitored fields (`RS Monitored`).

   ```
   # scrgadm -pv
   ```

Example–Enabling a Resource Fault Monitor

This example shows how to enable a resource fault monitor.

```
# scswitch -e -M -j resource-1
# scrgadm -pv
...
RS Monitored: yes...
```

Removing Resource Types

You do not need to remove resource types that are not in use. However, if you want to remove a resource type, you can use this procedure to do so.

See the `scrgadm`(1M) and `scswitch`(1M) man pages for additional information.

Note – Perform this procedure from any cluster node.

▼ How to Remove a Resource Type

Before you remove a resource type, you must disable and remove all of the resources of that type in all of the resource groups that are in the cluster. Use the `scrgadm -pv` command to identify the resources and resource groups that are in the cluster.

1. **Become superuser on a cluster member.**

2. **Disable each resource of the resource type that you will remove.**

   ```
   # scswitch -n -j resource
   ```

   `-n` Disables the resource.

   `-j` *resource* Specifies the name of the resource to disable.

3. **Remove each resource of the resource type that you will remove.**

   ```
   # scrgadm -r -j resource
   ```

   `-r` Removes the specified resource.

   `-j` Specifies the name of the resource to remove.

4. **Remove the resource type.**

```
# scrgadm -r -t resource-type
```

-r Removes the specified resource type.

-t *resource-type* Specifies the name of the resource type to remove.

5. **Verify that the resource type has been removed.**

```
# scrgadm -p
```

Example – Removing a Resource Type

This example shows how to disable and remove all of the resources of a resource type (`resource-type-1`) and then remove the resource type itself. In this example, `resource-1` is a resource of the resource type `resource-type-1`.

```
# scswitch -n -j resource-1
# scrgadm -r -j resource-1
# scrgadm -r -t resource-type-1
```

Removing Resource Groups

To remove a resource group, you must first remove all of the resources from the resource group.

See the `scrgadm`(1M) and `scswitch`(1M) man pages for additional information.

Note – Perform this procedure from any cluster node.

▼ How to Remove a Resource Group

1. **Become superuser on a cluster member.**
2. **Run the following command to switch the resource group offline.**

```
# scswitch -F -g resource-group
```

-F Switches a resource group offline.

-g *resource-group* Specifies the name of the resource group to take offline.

3. **Disable all of the resources that are part of the resource group.**

You can use the `scrgadm -pv` command to view the resources in the resource group. Disable all of the resources in the resource group that you will remove.

```
# scswitch -n -j resource
```

-n Disables the resource.

-j *resource* Specifies the name of the resource to disable.

If any dependent data service resources exist in a resource group, you cannot disable the resource until you have disabled all of the resources that depend on it.

4. **Remove all of the resources from the resource group.**

   Use the `scrgadm` command to perform the following tasks.

   - Remove the resources.
   - Remove the resource group.

```
# scrgadm -r -j resource
# scrgadm -r -g resource-group
```

-r Removes the specified resource or resource group.

-j *resource* Specifies the name of the resource to be removed.

-g *resource-group* Specifies the name of the resource group to be removed.

5. **Verify that the resource group has been removed.**

```
# scrgadm -p
```

Example – Removing a Resource Group

This example shows how to remove a resource group (`resource-group-1`) after you have removed its resource (`resource-1`).

```
# scswitch -F -g resource-group-1
# scrgadm -r -j resource-1
# scrgadm -r -g resource-group-1
```

Removing Resources

Disable the resource before you remove it from a resource group.

See the `scrgadm`(1M) and `scswitch`(1M) man pages for additional information.

Note – Perform this procedure from any cluster node.

▼ How to Remove a Resource

1. **Become superuser on a cluster member.**

2. **Disable the resource that you want to remove.**

   ```
   # scswitch -n -j resource
   ```

   `-n` Disables the resource.

   `-j` *resource* Specifies the name of the resource to disable.

3. **Remove the resource.**

   ```
   # scrgadm -r -j resource
   ```

   `-r` Removes the specified resource.

   `-j` *resource* Specifies the name of the resource to remove.

4. **Verify that the resource has been removed.**

   ```
   # scrgadm -p
   ```

Example – Removing a Resource

This example shows how to disable and remove a resource (`resource-1`).

```
# scswitch -n -j resource-1
# scrgadm -r -j resource-1
```

Switching the Current Primary of a Resource Group

Use the following procedure to switch over a resource group from its current primary to another node that will become the new primary.

See the `scrgadm`(1M) and `scswitch`(1M) man pages for additional information.

Note – Perform this procedure from any cluster node.

▼ How to Switch the Current Primary of a Resource Group

To complete this procedure, you must supply the following information.

- The name of the resource group that you will switch over.
- The names of the nodes on which you want the resource group to be brought online or to remain online. These nodes must be cluster nodes that have been set up to be potential masters of the resource group that you will switch. To see a list of potential primaries for the resource group, use the `scrgadm -pv` command.

1. **Become superuser on a cluster member.**

2. **Switch the primary to a potential primary.**

   ```
   # scswitch -z -g resource-group -h nodelist
   ```

   `-z` Switches the specified resource group online.

   `-g` *resource-group* Specifies the name of the resource group to switch.

   `-h` *nodelist* Specifies a comma-separated list of the names of the nodes on which the resource group is to be brought online or is to remain online. The list may contain one node name or more than one node name. This resource group is then switched offline on all of the other nodes.

   Note – If any resource group that you are switching declares a strong affinity for other resource groups, the attempt to switch might fail or be delegated. For more information, see "Distributing Online Resource Groups Among Cluster Nodes" on page 97.

3. **Verify that the resource group has been switched to the new primary.**

   Run the following command, and check the output for the state of the resource group that has been switched over.

   ```
   # scstat -g
   ```

Example – Switching the Resource Group to a New Primary

This example shows how to switch a resource group (resource-group-1) from its current primary (phys-schost-1) to the potential primary (phys-schost-2). First, verify that the resource group is online on phys-schost-1. Next, perform the switch. Finally, verify that the group is switched to be online on phys-schost-2.

```
phys-schost-1# scstat -g
...
Resource Group Name:          resource-group-1
  Status
    Node Name:                phys-schost-1
    Status:                   Online

    Node Name:                phys-schost-2
    Status:                   Offline
...
phys-schost-1# scswitch -z -g resource-group-1 -h phys-schost-2
phys-schost-1# scstat -g
...
Resource Group Name:          resource-group-1
  Status
    Node Name:                phys-schost-2
    Status:                   Online

    Node Name:                phys-schost-1
    Status:                   Offline
...
```

Disabling Resources and Moving Their Resource Group Into the UNMANAGED State

At times, you must bring a resource group into the UNMANAGED state before you perform an administrative procedure on it. Before you move a resource group into the UNMANAGED state, you must disable all of the resources that are part of the resource group and bring the resource group offline.

See the scrgadm(1M) and scswitch(1M) man pages for additional information.

Note – Perform this procedure from any cluster node.

▼ How to Disable a Resource and Move Its Resource Group Into the `UNMANAGED` State

To complete this procedure, you must supply the following information.

- The name of the resources to be disabled
- The name of the resource group to move into the `UNMANAGED` state

To determine the resource and resource group names that you need for this procedure, use the `scrgadm -pv` command.

Note – When a shared address resource is disabled, the resource might still be able to respond to `ping`(1M) commands from some hosts. To ensure that a disabled shared address resource cannot respond to `ping` commands, you must bring the resource's resource group to the `UNMANAGED` state.

---

1. **Become superuser on a cluster member.**

2. **Disable the resource.**

   Repeat this step for all of the resources in the resource group.

   ```
   # scswitch -n -j resource
   ```

   `-n` Disables the resource.

   `-j` *resource* Specifies the name of the resource to disable.

3. **Run the following command to switch the resource group offline.**

   ```
   # scswitch -F -g resource-group
   ```

   `-F` Switches a resource group offline.

   `-g` *resource-group* Specifies the name of the resource group to take offline.

4. **Move the resource group into the `UNMANAGED` state.**

   ```
   # scswitch -u -g resource-group
   ```

   `-u` Moves the specified resource group in the `UNMANAGED` state.

   `-g` *resource-group* Specifies the name of the resource group to move into the `UNMANAGED` state.

5. **Verify that the resources are disabled and the resource group is in the `UNMANAGED` state.**

   ```
   # scrgadm -pv -g resource-group
   ```

Example – Disabling a Resource and Moving the Resource Group Into the UNMANAGED State

This example shows how to disable the resource (resource-1) and then move the resource group (resource-group-1) into the UNMANAGED state.

```
# scswitch -n -j resource-1
# scswitch -F -g resource-group-1
# scswitch -u -g resource-group-1
# scrgadm -pv -g resource-group-1
Res Group name:                                              resource-group-1
  (resource-group-1) Res Group RG_description:               <NULL>
  (resource-group-1) Res Group management state:             Unmanaged
  (resource-group-1) Res Group Failback:                     False
  (resource-group-1) Res Group Nodelist:                     phys-schost-1
                                                             phys-schost-2
  (resource-group-1) Res Group Maximum_primaries:            2
  (resource-group-1) Res Group Desired_primaries:            2
  (resource-group-1) Res Group RG_dependencies:              <NULL>
  (resource-group-1) Res Group mode:                         Failover
  (resource-group-1) Res Group network dependencies:         True
  (resource-group-1) Res Group Global_resources_used:        All
  (resource-group-1) Res Group Pathprefix:

  (resource-group-1) Res name:                               resource-1
    (resource-group-1:resource-1) Res R_description:
    (resource-group-1:resource-1) Res resource type:         SUNW.apache
    (resource-group-1:resource-1) Res resource group name:   resource-group-1
    (resource-group-1:resource-1) Res enabled:               True
    (resource-group-1:resource-1) Res monitor enabled:       False
    (resource-group-1:resource-1) Res detached:              False
```

Displaying Resource Type, Resource Group, and Resource Configuration Information

Before you perform administrative procedures on resources, resource groups, or resource types, use the following procedure to view the current configuration settings for these objects.

See the scrgadm(1M) and scswitch(1M) man pages for additional information.

Note – Perform this procedure from any cluster node.

Displaying Resource Type, Resource Group, and Resource Configuration Information

The scrgadm command provides the following three levels of configuration status information.

- With the -p option, the output shows a very limited set of property values for resource types, resource groups, and resources.
- With the -pv option, the output shows more details on other resource type, resource group, and resource properties.
- With the -pvv option, the output provides a detailed view, including resource type methods, extension properties, and all of the resource and resource group properties.

You can also use the -t, -g, and -j (resource type, resource group, and resource, respectively) options, followed by the name of the object that you want to view, to check status information on specific resource types, resource groups, and resources. For example, the following command specifies that you want to view specific information on the resource apache-1 only.

```
# scrgadm -p[v[v]] -j apache-1
```

See the scrgadm(1M) man page for details.

Changing Resource Type, Resource Group, and Resource Properties

Sun Cluster defines standard properties for configuring resource types, resource groups, and resources. These standard properties are described in the following sections:

- "Resource Type Properties" on page 117
- "Resource Properties" on page 123
- "Resource Group Properties" on page 134

Resources also have extension properties, which are predefined for the data service that represents the resource. For a description of the extension properties of a data service, see the documentation for the data service.

To determine whether you can change a property, see the Tunable entry for the property in the description of the property.

The following procedures describe how to change properties for configuring resource types, resource groups, and resources.

▼ How to Change Resource Type Properties

To complete this procedure, you must supply the following information.

- The name of the resource type to change.
- The name of the resource type property to change. For resource types, you can change only certain properties. To determine whether you can change a property, see the Tunable entry for the property in "Resource Type Properties" on page 117.

  Note – You cannot change the `Installed_nodes` property explicitly. To change this property, specify the `-h` *installed-node-list* option of the `scrgadm` command.

Note – Perform this procedure from any cluster node.

---

1. **Become superuser on a cluster member.**
2. **Run the `scrgadm` command to determine the name of the resource type that you need for this procedure.**

   ```
   # scrgadm -pv
   ```

3. **Change the resource type property.**

   For resource types, you can change only certain properties. To determine whether you can change a property, see the Tunable entry for the property in "Resource Type Properties" on page 117.

```
# scrgadm -c -t resource-type [-h installed-node-list] [-y property=new-value]
```

`-c` Changes the specified resource type property.

`-t` *resource-type* Specifies the name of the resource type.

-h *installed-node-list* Specifies the names of nodes on which this resource type is installed.

-y *property=new-value* Specifies the name of the standard property to change and the new value of the property.

You cannot change the Installed_nodes property explicitly. To change this property, specify the -h *installed-node-list* option of the scrgadm command.

4. **Verify that the resource type property has been changed.**

```
# scrgadm -pv -t resource-type
```

Example – Changing a Resource Type Property

This example shows how to change the SUNW.apache property to define that this resource type is installed on two nodes (phys-schost-1 and phys-schost-2).

```
# scrgadm -c -t SUNW.apache -h phys-schost-1,phys-schost-2
# scrgadm -pv -t SUNW.apache
Res Type name:                              SUNW.apache
  (SUNW.apache) Res Type description:       Apache Resource Type
  (SUNW.apache) Res Type base directory:    /opt/SUNWscapc/bin
  (SUNW.apache) Res Type single instance:   False
  (SUNW.apache) Res Type init nodes:        All potential masters
  (SUNW.apache) Res Type failover:          False
  (SUNW.apache) Res Type version:           1.0
  (SUNW.apache) Res Type API version:       2
  (SUNW.apache) Res Type installed on nodes: phys-schost1 phys-schost-2
  (SUNW.apache) Res Type packages:          SUNWscapc
```

▼ How to Change Resource Group Properties

To complete this procedure, you must supply the following information.

- The name of the resource group to change
- The name of the resource group property to change and its new value

This procedure describes the steps to change resource group properties. See Appendix A for a complete list of resource group properties.

Note – Perform this procedure from any cluster node.

1. **Become superuser on a cluster member.**

2. **Change the resource group property.**

```
# scrgadm -c -g resource-group -y property=new_value
```

`-c`	Changes the specified property.
`-g` *resource-group*	Specifies the name of the resource group.
`-y` *property*	Specifies the name of the property to change.

3. **Verify that the resource group property has been changed.**

```
# scrgadm -pv -g resource-group
```

Example – Changing a Resource Group Property

This example shows how to change the `Failback` property for the resource group (`resource-group-1`).

```
# scrgadm -c -g resource-group-1 -y Failback=True
# scrgadm -pv -g resource-group-1
```

▼ How to Change Resource Properties

To complete this procedure, you must supply the following information.

- The name of the resource with the property to change
- The name of the property to change

This procedure describes the steps to change resource properties. See Appendix A for a complete list of resource group properties.

Note – Perform this procedure from any cluster node.

---

1. **Become superuser on a cluster member.**

2. **Run the `scrgadm -pvv` command to view the current resource property settings.**

```
# scrgadm -pvv -j resource
```

3. **Change the resource property.**

```
# scrgadm -c -j resource -y property=new_value | -x extension_property=new_value
```

`-c`	Changes the specified property.
`-j` *resource*	Specifies the name of the resource.

`-y` *property=new_value*

Specifies the name of the standard property to change.

`-x` *extension_property=new_value*

Specifies the name of the extension property to change. For a description of the extension properties of a data service, see the documentation for the data service.

4. **Verify that the resource property has been changed.**

```
# scrgadm pvv -j resource
```

Example – Changing a Standard Resource Property

This example shows how to change the system-defined `Start_timeout` property for the resource (`resource-1`).

```
# scrgadm -c -j resource-1 -y start_timeout=30
# scrgadm -pvv -j resource-1
```

Example – Changing an Extension Resource Property

This example shows how to change an extension property (`Log_level`) for the resource (`resource-1`).

```
# scrgadm -c -j resource-1 -x Log_level=3
# scrgadm -pvv -j resource-1
```

▼ How to Modify a Logical Hostname Resource or a Shared Address Resource

By default, logical hostname resources and shared address resources use name services for name resolution. You might configure a cluster to use a name service that is running on the same cluster. During the failover of a logical hostname resource or a shared address resource, a name service that is running on the cluster might also be failing over. If the logical hostname resource or the shared address resource uses the name service that is failing over, the resource fails to fail over.

Note – Configuring a cluster to use a name server that is running on the same cluster might impair the availability of other services on the cluster.

To prevent such a failure to fail over, modify the logical hostname resource or the shared address resource to bypass name services. To modify the resource to bypass name services, set the `CheckNameService` extension property of the resource to `false`. You can modify the `CheckNameService` property at any time.

Note – If your version of the resource type is earlier than 2, you must upgrade the resource type before you attempt to modify the resource. For more information, see "Upgrading a Preregistered Resource Type" on page 69.

---

1. **Become superuser on a cluster member.**
2. **Change the resource property.**

```
# scrgadm -c -j resource -x CheckNameService=false
```

`-j` *resource*
Specifies the name of the logical hostname resource or shared address resource that you are modifying

`-y CheckNameService=false`
Sets the `CheckNameService` extension property of the resource to `false`

Clearing the STOP_FAILED Error Flag on Resources

When the `Failover_mode` resource property is set to `NONE` or `SOFT` and the `STOP` of a resource fails, the individual resource goes into the `STOP_FAILED` state, and the resource group goes into the `ERROR_STOP_FAILED` state. You cannot bring a resource group in this state on any node online, nor can you edit the resource group (create or delete resources, or change resource group or resource properties).

▼ How to Clear the STOP_FAILED Error Flag on Resources

To complete this procedure, you must supply the following information.

- The name of the node where the resource is `STOP_FAILED`
- The name of the resource and resource group that are in `STOP_FAILED` state

See the scswitch(1M) man page for additional information.

Note – Perform this procedure from any cluster node.

1. **Become superuser on a cluster member.**
2. **Identify which resources have gone into the STOP_FAILED state and on which nodes.**

   ```
   # scstat -g
   ```

3. **Manually stop the resources and their monitors on the nodes on which they are in STOP_FAILED state.**

   This step might require that you kill processes or run commands that are specific to resource types or other commands.

4. **Manually set the state of these resources to OFFLINE on all of the nodes on which you manually stopped the resources.**

   ```
   # scswitch -c -h nodelist -j resource -f STOP_FAILED
   ```

   -c Clears the flag.

   -h *nodelist* Specifies a comma-separated list of the names of the nodes where the resource is in the STOP_FAILED state. The list may contain one node name or more than one node name.

   -j *resource* Specifies the name of the resource to switch offline.

   -f STOP_FAILED Specifies the flag name.

5. **Check the resource group state on the nodes where you cleared the STOP_FAILED flag in Step 4.**

   The resource group state should now be OFFLINE or ONLINE.

   ```
   # scstat -g
   ```

   The command scstat -g indicates whether the resource group remains in the ERROR_STOP_FAILED state. If the resource group is still in the ERROR_STOP_FAILED state, then run the following scswitch command to switch the resource group offline on the appropriate nodes.

   ```
   # scswitch -F -g resource-group
   ```

   -F Switches the resource group offline on all of the nodes that can master the group.

   -g *resource-group* Specifies the name of the resource group to switch offline.

This situation can occur if the resource group was being switched offline when the STOP method failure occurred and the resource that failed to stop had a dependency on other resources in the resource group. Otherwise, the resource group reverts to the ONLINE or OFFLINE state automatically after you have run the command in Step 4 on all of the STOP_FAILED resources.

Now you can switch the resource group to the ONLINE state.

Upgrading a Preregistered Resource Type

In Sun Cluster 3.1 9/04, the following preregistered resource types are enhanced:

- SUNW.LogicalHostname, which represents a logical hostname
- SUNW.SharedAddress, which represents a shared address

The purpose of these enhancements is to enable you to modify logical hostname resources and shared address resources to bypass name services for name resolution.

Upgrade these resource types if all conditions in the following list apply:

- You are upgrading from an earlier version of Sun Cluster.
- You need to use the new features of the resource types.

For general instructions that explain how to upgrade a resource type, see "Upgrading a Resource Type" on page 33. The information that you need to complete the upgrade of the preregistered resource types is provided in the subsections that follow.

Information for Registering the New Resource Type Version

The relationship between the version of each preregistered resource type and the release of Sun Cluster is shown in the following table. The release of Sun Cluster indicates the release in which the version of the resource type was introduced.

Resource Type	Resource Type Version	Sun Cluster Release
SUNW.LogicalHostname	1.0	3.0
	2	3.1 9/04

Resource Type	Resource Type Version	Sun Cluster Release
SUNW.SharedAddress	1.0	3.0
	2	3.1 9/04

To determine the version of the resource type that is registered, use one command from the following list:

- scrgadm -p
- scrgadm -pv

EXAMPLE 2–1 Registering a New Version of the SUNW.LogicalHostname Resource Type

This example shows the command for registering version 2 of the SUNW.LogicalHostname resource type during an upgrade.

```
# scrgadm -a -t SUNW.LogicalHostname:2
```

Information for Migrating Existing Instances of the Resource Type

The information that you need to migrate an instance of a preregistered resource type is as follows:

- You can perform the migration at any time.
- If you need to use the new features of the preregistered resource type, the required value of the Type_version property is 2.
- If you are modifying the resource to bypass name services, set the CheckNameService extension property of the resource to false.

EXAMPLE 2–2 Migrating a Logical Hostname Resource

This example shows the command for migrating the logical hostname resource lhostrs. As a result of the migration, the resource is modified to bypass name services for name resolution.

```
# scrgadm -c -j lhostrs -y Type_version=2 -x CheckNameService=false
```

Reregistering Preregistered Resource Types After Inadvertent Deletion

Two preregistered resource types are `SUNW.LogicalHostname` and `SUNW.SharedAddress`. All of the logical hostname and shared address resources use these resource types. You never need to register these two resource types, but you might accidentally delete them. If you have deleted resource types inadvertently, use the following procedure to reregister them.

Note – If you are upgrading a preregistered resource type, follow the instructions in "Upgrading a Preregistered Resource Type" on page 69 to register the new resource type version.

See the `scrgadm`(1M) man page for additional information.

Note – Perform this procedure from any cluster node.

▼ How to Reregister Preregistered Resource Types After Inadvertent Deletion

- **Reregister the resource type.**

```
# scrgadm -a -t SUNW.resource-type
```

`-a` Adds a resource type.

`-t SUNW.`*resource-type* Specifies the resource type to add (reregister). The resource type can be either `SUNW.LogicalHostname` or `SUNW.SharedAddress`.

Example – Reregistering a Preregistered Resource Type After Inadvertent Deletion

This example shows how to reregister the `SUNW.LogicalHostname` resource type.

```
# scrgadm -a -t SUNW.LogicalHostname
```

Adding or Removing a Node to or From a Resource Group

The procedures in this section enable you to perform the following tasks.

- Configure a cluster node to be an additional master of a resource group.
- Remove a node from a resource group.

The procedures are slightly different, depending on whether you plan to add or remove the node to or from a failover or scalable resource group.

Failover resource groups contain network resources that both failover and scalable services use. Each IP subnetwork connected to the cluster has its own network resource that is specified and included in a failover resource group. The network resource is either a logical hostname or a shared address resource. Each network resource includes a list of IP Networking Multipathing groups that it uses. For failover resource groups, you must update the complete list of IP Networking Multipathing groups for each network resource that the resource group includes (the `netiflist` resource property).

For scalable resource groups, in addition to changing the scalable group to be mastered on the new set of hosts, you must repeat the procedure for failover groups that contain the network resources that the scalable resource uses.

See the `scrgadm`(1M) man page for additional information.

Note – Run either of these procedures from any cluster node.

Adding a Node to a Resource Group

The procedure to follow to add a node to a resource group depends on whether the resource group is a scalable resource group or a failover resource group. For detailed instructions, see the following sections:

- "How to Add a Node to a Scalable Resource Group" on page 73
- "How to Add a Node to a Failover Resource Group" on page 73

You must supply the following information to complete the procedure.

- The names and node IDs of all of the cluster nodes
- The names of the resource groups to which you are adding the node

- The name of the IP Networking Multipathing group that will host the network resources that are used by the resource group on all of the nodes

Also, be sure to verify that the new node is already a cluster member.

▼ How to Add a Node to a Scalable Resource Group

1. **For each network resource that a scalable resource in the resource group uses, make the resource group where the network resource is located run on the new node.**

   See Step 1 through Step 4 in the following procedure for details.

2. **Add the new node to the list of nodes that can master the scalable resource group (the `nodelist` resource group property).**

   This step overwrites the previous value of `nodelist`, and therefore you must include all of the nodes that can master the resource group here.

   ```
   # scrgadm -c -g resource-group -h nodelist
   ```

   `-c` Changes a resource group.

   `-g` *resource-group* Specifies the name of the resource group to which the node is being added.

   `-h` *nodelist* Specifies a comma-separated list of the names of the nodes that can master the resource group.

3. **(Optional) Update the `Load_balancing_weights` property of the scalable resource to assign a weight to the node that you want to add to the resource group.**

   Otherwise, the weight defaults to `1`. See the `scrgadm`(1M) man page for more information.

▼ How to Add a Node to a Failover Resource Group

1. **Display the current node list and the current list of IP Networking Multipathing groups that are configured for each resource in the resource group.**

   ```
   # scrgadm -pvv -g resource-group | grep -i nodelist
   # scrgadm -pvv -g resource-group | grep -i netiflist
   ```

   Note – The output of the command line for `nodelist` and `netiflist` identifies the nodes by node name. To identify node IDs, run the command `scconf -pv | grep -i` *node_id*.

   ---

2. **Update `netiflist` for the network resources that the node addition affects.**

   This step overwrites the previous value of `netiflist`, and therefore you must include all of the IP Networking Multipathing groups here.

   ```
   # scrgadm -c -j network-resource -x netiflist=netiflist
   ```

`-c`	Changes a network resource.
`-j` *network-resource*	Specifies the name of the network resource (logical hostname or shared address) that is being hosted on the *netiflist* entries.
`-x netiflist=`*netiflist*	Specifies a comma-separated list that identifies the IP Networking Multipathing groups that are on each node. Each element in *netiflist* must be in the form of `netif@node`. `netif` can be given as an IP Networking Multipathing group name, such as `sc_ipmp0`. The node can be identified by the node name or node ID, such as `sc_ipmp0@1` or `sc_ipmp@phys-schost-1`.

3. **Update the node list to include all of the nodes that can now master this resource group.**

   This step overwrites the previous value of `nodelist`, and therefore you must include all of the nodes that can master the resource group here.

   ```
   # scrgadm -c -g resource-group -h nodelist
   ```

`-c`	Changes a resource group.
`-g` *resource-group*	Specifies the name of the resource group to which the node is being added.
`-h` *nodelist*	Specifies a comma-separated list of the names of the nodes that can master the resource group.

4. **Verify the updated information.**

   ```
   # scrgadm -pvv -g resource-group | grep -i nodelist
   # scrgadm -pvv -g resource-group | grep -i netiflist
   ```

Example – Adding a Node to a Resource Group

This example shows how to add a node (`phys-schost-2`) to a resource group (`resource-group-1`) that contains a logical hostname resource (`schost-2`).

```
# scrgadm -pvv -g resource-group-1 | grep -i nodelist
(resource-group-1) Res Group Nodelist:    phys-schost-1 phys-schost-3
# scrgadm -pvv -g resource-group-1 | grep -i netiflist
(resource-group-1:schost-2) Res property name: NetIfList
(resource-group-1:schost-2:NetIfList) Res property class: extension
```

```
(resource-group-1:schost-2:NetIfList) List of IP Networking Multipathing
interfaces on each node
(resource-group-1:schost-2:NetIfList) Res property type: stringarray
(resource-group-1:schost-2:NetIfList) Res property value: sc_ipmp0@1 sc_ipmp0@3
```

(Only nodes 1 and 3 have been assigned IP Networking Multipathing groups.
You must add a IP Networking Multipathing group
for node 2.)

```
# scrgadm -c -j schost-2 -x netiflist=sc_ipmp0@1,sc_ipmp0@2,sc_ipmp0@3
# scrgadm -c -g resource-group-1 -h phys-schost-1,phys-schost-2,phys-schost-3
# scrgadm -pvv -g resource-group-1 | grep -i nodelist
(resource-group-1) Res Group Nodelist:     phys-schost-1 phys-schost-2
                                           phys-schost-3
# scrgadm -pvv -g resource-group-1 | grep -i netiflist
(resource-group-1:schost-2:NetIfList) Res property value: sc_ipmp0@1 sc_ipmp0@2
                                                          sc_ipmp0@3
```

Removing a Node From a Resource Group

The procedure to follow to remove a node from a resource group depends on whether the resource group is a scalable resource group or a failover resource group. For detailed instructions, see the following sections:

- "How to Remove a Node From a Scalable Resource Group" on page 76
- "How to Remove a Node From a Failover Resource Group" on page 77
- "How to Remove a Node From a Failover Resource Group That Contains Shared Address Resources" on page 78

For an example, see "Example – Removing a Node From a Resource Group" on page 79.

To complete the procedure, you must supply the following information.

- Node names and node IDs of all of the cluster nodes

  ```
  # scconf -pv | grep "Node ID"
  ```
- Name(s) of the resource group or groups from which you plan to remove the node

  ```
  # scrgadm -pv | grep "Res Group Nodelist"
  ```
- Names of the IP Networking Multipathing groups that will host the network resources that are used by the resource group(s) on all of the nodes

  ```
  # scrgadm -pvv | grep "NetIfList.*value"
  ```

Additionally, be sure to verify that the resource group **is not mastered** on the node that you will remove. If the resource group **is mastered** on the node that you will remove, run the `scswitch` command to switch the resource group offline from that node. The following `scswitch` command will bring the resource group offline from a given node, provided that *new-masters* does not contain that node.

```
# scswitch -z -g resource-group -h new-masters
```

`-g` *resource-group* Specifies the name of the resource group (mastered on the node that you will remove) that you are switching offline.

`-h` *new-masters* Specifies the node(s) that will now master the resource group.

See the `scswitch`(1M) man page for additional information.

Caution – If you plan to remove a node from all of the resource groups, and you use a scalable services configuration, first remove the node from the scalable resource group(s). Then, remove the node from the failover group(s).

▼ How to Remove a Node From a Scalable Resource Group

A scalable service is configured as two resource groups, as follows.

- One resource group is a scalable group that contains the scalable service resource.
- One resource group is a failover group that contains the shared address resources that the scalable service resource uses.

Additionally, the `RG_dependencies` property of the scalable resource group is set to configure the scalable group with a dependency on the failover resource group. See Appendix A for details on this property.

See the *Sun Cluster Concepts Guide for Solaris OS* document for details about scalable service configuration.

Removing a node from the scalable resource group causes the scalable service to no longer be brought online on that node. To remove a node from the scalable resource group, perform the following steps.

1. **Remove the node from the list of nodes that can master the scalable resource group (the `nodelist` resource group property).**

   ```
   # scrgadm -c -g scalable-resource-group -h nodelist
   ```

   `-c` Changes a resource group.

   `-g` *scalable-resource-group* Specifies the name of the resource group from which the node is being removed.

-h *nodelist*	Specifies a comma-separated list of the names of the nodes that can master this resource group.

2. **(Optional) Remove the node from the failover resource group that contains the shared address resource.**

   See "How to Remove a Node From a Failover Resource Group That Contains Shared Address Resources" on page 78 for details.

3. **(Optional) Update the `Load_balancing_weights` property of the scalable resource to remove the weight of the node that you want to remove from the resource group.**

   See the `scrgadm`(1M) man page for more information.

How to Remove a Node From a Failover Resource Group

Perform the following steps to remove a node from a failover resource group.

Caution – If you plan to remove a node from all of the resource groups, and you use a scalable services configuration, first remove the node from the scalable resource group(s). Then, use this procedure to remove the node from the failover group(s).

Note – If the failover resource group contains shared address resources that scalable services use, see "How to Remove a Node From a Failover Resource Group That Contains Shared Address Resources" on page 78.

---

1. **Update the node list to include all of the nodes that can now master this resource group.**

   This step removes the node and overwrites the previous value of the node list. Be sure to include all of the nodes that can master the resource group here.

   ```
   # scrgadm -c -g failover-resource-group -h nodelist
   ```

-c	Changes a resource group.
-g *failover-resource-group*	Specifies the name of the resource group from which the node is being removed.
-h *nodelist*	Specifies a comma-separated list of the names of the nodes that can master this resource group.

2. **Display the current list of IP Networking Multipathing groups that are configured for each resource in the resource group.**

```
# scrgadm -pvv -g failover-resource-group | grep -i netiflist
```

3. **Update `netiflist` for network resources that the removal of the node affects.**

   This step overwrites the previous value of `netiflist`. Be sure to include all of the IP Networking Multipathing groups here.

   ```
   # scrgadm -c -j network-resource -x netiflist=netiflist
   ```

   Note – The output of the preceding command line identifies the nodes by node name. Run the command line `scconf -pv | grep "Node ID"` to find the node ID.

   `-c`

   Changes a network resource.

   `-j` *network-resource*

   Specifies the name of the network resource that is hosted on the `netiflist` entries.

   `-x netiflist=`*netiflist*

   Specifies a comma-separated list that identifies the IP Networking Multipathing groups that are on each node. Each element in *netiflist* must be in the form of `netif@node`. `netif` can be given as an IP Networking Multipathing group name, such as `sc_ipmp0`. The node can be identified by the node name or node ID, such as `sc_ipmp0@1` or `sc_ipmp@phys-schost-1`.

   Note – Sun Cluster does not currently support using the adapter name for `netif`.

   ---

4. **Verify the updated information.**

   ```
   # scrgadm -pvv -g failover-resource-group | grep -i nodelist
   # scrgadm -pvv -g failover-resource-group | grep -i netiflist
   ```

▼ How to Remove a Node From a Failover Resource Group That Contains Shared Address Resources

In a failover resource group that contains shared address resources that scalable services use, a node can appear in the following locations.

- the node list of the failover resource group
- the `auxnodelist` of the shared address resource

To remove the node from the node list of the failover resource group, follow the procedure "How to Remove a Node From a Failover Resource Group" on page 77.

To modify the auxnodelist of the shared address resource, you must remove and recreate the shared address resource.

If you remove the node from the failover group's node list, you can continue to use the shared address resource on that node to provide scalable services. To do so, you must add the node to the auxnodelist of the shared address resource. To add the node to the auxnodelist, perform the following steps.

Note – You can also use the following procedure to **remove** the node from the auxnodelist of the shared address resource. To remove the node from the auxnodelist, you must delete and recreate the shared address resource.

---

1. **Switch the scalable service resource offline.**
2. **Remove the shared address resource from the failover resource group.**
3. **Create the shared address resource.**

   Add the node ID or node name of the node that you removed from the failover resource group to the auxnodelist.

   ```
   # scrgadm -a -S -g failover-resource-group \
    -l shared-address -X new-auxnodelist
   ```

   failover-resource-group The name of the failover resource group that used to contain the shared address resource.

   shared-address The name of the shared address.

   new-auxnodelist The new, modified auxnodelist with the desired node added or removed.

Example – Removing a Node From a Resource Group

This example shows how to remove a node (phys-schost-3) from a resource group (resource-group-1), which contains a logical hostname resource (schost-1).

```
# scrgadm -pvv -g resource-group-1 | grep -i nodelist
(resource-group-1) Res Group Nodelist:       phys-schost-1 phys-schost-2
                                             phys-schost-3
# scrgadm -c -g resource-group-1 -h phys-schost-1,phys-schost-2
# scrgadm -pvv -g resource-group-1 | grep -i netiflist
(resource-group-1:schost-1) Res property name: NetIfList
(resource-group-1:schost-1:NetIfList) Res property class: extension
(resource-group-1:schost-1:NetIfList) List of IP Networking Multipathing
interfaces on each node
(resource-group-1:schost-1:NetIfList) Res property type: stringarray
(resource-group-1:schost-1:NetIfList) Res property value: sc_ipmp0@1 sc_ipmp0@2
                                                          sc_ipmp0@3
```

(*sc_ipmp0@3 is the IP Networking Multipathing group to be removed.)*

```
# scrgadm -c  -j schost-1 -x  netiflist=sc_ipmp0@1,sc_ipmp0@2
# scrgadm -pvv -g resource-group-1 | grep -i nodelist
(resource-group-1) Res Group Nodelist:       phys-schost-1 phys-schost-2
# scrgadm -pvv -g resource-group-1 | grep -i netiflist
(resource-group-1:schost-1:NetIfList) Res property value: sc_ipmp0@1 sc_ipmp0@2
```

Synchronizing the Startups Between Resource Groups and Disk Device Groups

After a cluster boots up or services fail over to another node, global devices and cluster file systems might require time to become available. However, a data service can run its START method before global devices and cluster file systems—on which the data service depends—come online. In this instance, the START method times out, and you must reset the state of the resource groups that the data service uses and restart the data service manually. The resource types HAStorage and HAStoragePlus monitor the global devices and cluster file systems and cause the START method of the other resources in the same resource group to wait until they become available. (To determine which resource type to create, see "Choosing Between HAStorage and HAStoragePlus" on page 20.) To avoid additional administrative tasks, set up HAStorage or HAStoragePlus for all of the resource groups whose data service resources depend on global devices or cluster file systems.

To create a HAStorage resource type, see "How to Set Up HAStorage Resource Type for New Resources" on page 80.

To create a HAStoragePlus resource type, see "How to Set Up HAStoragePlus Resource Type" on page 86.

▼ How to Set Up HAStorage Resource Type for New Resources

HAStorage might not be supported in a future release of Sun Cluster. Equivalent functionality is supported by HAStoragePlus. To upgrade from HAStorage to HAStoragePlus, see "Upgrading from HAStorage to HAStoragePlus" on page 83.

In the following example, the resource group `resource-group-1` contains three data services.

- Sun Java System Web Server, which depends on `/global/resource-group-1`
- Oracle, which depends on `/dev/global/dsk/d5s2`
- NFS, which depends on `dsk/d6`

To create a `HAStorage` resource `hastorage-1` for new resources in `resource-group-1`, read "Synchronizing the Startups Between Resource Groups and Disk Device Groups" on page 80 and then perform the following steps.

To create a HAStoragePlus resource type, see "Enabling Highly Available Local File Systems" on page 86.

1. **Become superuser on a cluster member.**

2. **Create the resource group `resource-group-1`.**

   ```
   # scrgadm -a -g resource-group-1
   ```

3. **Determine whether the resource type is registered.**

   The following command prints a list of registered resource types.

   ```
   # scrgadm -p | egrep Type
   ```

4. **If you need to, register the resource type.**

   ```
   # scrgadm -a -t SUNW.HAStorage
   ```

5. **Create the `HAStorage` resource `hastorage-1`, and define the service paths.**

```
# scrgadm -a -j hastorage-1 -g resource-group-1 -t SUNW.HAStorage \
-x ServicePaths=/global/resource-group-1,/dev/global/dsk/d5s2,dsk/d6
```

   `ServicePaths` can contain the following values.

   - global device group names, such as `nfs-dg`
   - paths to global devices, such as `/dev/global/dsk/d5s2` or `dsk/d6`
   - cluster file system mount points, such as `/global/nfs`

   Note – Global device groups might not be colocated with the resource groups that correspond to them if `ServicePaths` contains cluster file system paths.

6. **Enable the `hastorage-1` resource.**

   ```
   # scswitch -e -j hastorage-1
   ```

7. **Add the resources (Sun Java System Web Server, Oracle, and NFS) to `resource-group-1`, and set their dependency to `hastorage-1`.**

   For example, for Sun Java System Web Server, run the following command.

```
# scrgadm -a -j resource \-g resource-group-1 -t SUNW.iws \
-x Confdir_list=/global/iws/schost-1 -y Scalable=False \
-y Network_resources_used=schost-1 -y Port_list=80/tcp \
-y Resource_dependencies=hastorage-1
```

8. **Verify that you have correctly configured the resource dependencies.**

   ```
   # scrgadm -pvv -j resource | egrep strong
   ```

9. **Set resource-group-1 to the MANAGED state, and bring resource-group-1 online.**

   ```
   # scswitch -Z -g resource-group-1
   ```

The HAStorage resource type contains another extension property, AffinityOn, which is a Boolean that specifies whether HAStorage must perform an affinity switchover for the global devices and cluster file systems that are defined in ServicePaths. See the SUNW.HAStorage(5) man page for details.

Note – HAStorage and HAStoragePlus do not permit AffinityOn to be set to TRUE if the resource group is scalable. HAStorage and HAStoragePlus checks the AffinityOn value and internally resets the value to FALSE for a scalable resource group.

▼ How to Set Up HAStorage Resource Type for Existing Resources

HAStorage might not be supported in a future release of Sun Cluster. Equivalent functionality is supported by HAStoragePlus. To upgrade from HAStorage to HAStoragePlus, see "Upgrading from HAStorage to HAStoragePlus" on page 83.

To create a HAStorage resource for existing resources, read "Synchronizing the Startups Between Resource Groups and Disk Device Groups" on page 80, and then perform the following steps.

1. **Determine whether the resource type is registered.**

   The following command prints a list of registered resource types.

   ```
   # scrgadm -p | egrep Type
   ```

2. **If you need to, register the resource type.**

   ```
   # scrgadm -a -t SUNW.HAStorage
   ```

3. **Create the HAStorage resource hastorage-1.**

   ```
   # scrgadm -a -g resource-group -j hastorage-1 -t SUNW.HAStorage \
   -x ServicePaths= ... -x AffinityOn=True
   ```

4. **Enable the `hastorage-1` resource.**

   ```
   # scswitch -e -j hastorage-1
   ```

5. **Set up the dependency for each of the existing resources, as required.**

   ```
   # scrgadm -c -j resource -y Resource_Dependencies=hastorage-1
   ```

6. **Verify that you have correctly configured the resource dependencies.**

   ```
   # scrgadm -pvv -j resource | egrep strong
   ```

Upgrading from HAStorage to HAStoragePlus

HAStorage might not be supported in a future release of Sun Cluster. Equivalent functionality is supported by HAStoragePlus. To upgrade from HAStorage to HAStorage, see the following sections.

How to Upgrade from HAStorage to HAStoragePlus When Using Device Groups or CFS

HAStorage might not be supported in a future release of Sun Cluster. Equivalent functionality is supported by HAStoragePlus. To upgrade from HAStorage to HAStoragePlus when using device groups or CFS, complete the following steps.

The following example uses a simple HA-NFS resource active with HAStorage. The ServicePaths are the diskgroup `nfsdg` and the AffinityOn property is TRUE. Furthermore, the HA-NFS resource has `Resource_Dependencies` set to the HAStorage resource.

1. **Remove the dependencies the application resources has on HAStorage.**

   ```
   # scrgadm -c -j nfsserver-rs -y Resource_Dependencies=""
   ```

2. **Disable the HAStorage resource.**

   ```
   # scswitch -n -j nfs1storage-rs
   ```

3. **Remove the HAStorage resource from the application resource group.**

   ```
   # scrgadm -r -j nfs1storage-rs
   ```

4. **Unregister the HAStorage resource type.**

```
# scrgadm -r -t SUNW.HAStorage
```

5. **Register the HAStoragePlus resource type.**

```
# scrgadm -a -t SUNW.HAStoragePlus
```

6. **Create the HAStoragePlus resource.**

   To specify a filesystem mount point, input the following text.

```
# scrgadm -a -j nfs1-hastp-rs -g nfs1-rg -t \
SUNW.HAStoragePlus -x FilesystemMountPoints=/global/nfsdata -x \
AffinityOn=True
```

   To specify global device paths, input the following text.

```
# scrgadm -a -j nfs1-hastp-rs -g nfs1-rg -t \
SUNW.HAStoragePlus -x GlobalDevicePaths=nfsdg -x AffinityOn=True
```

Note – Instead of using the ServicePaths property for HAStorage, you must use the GlobalDevicePaths or FilesystemMountPoints property for HAStoragePlus. The FilesystemMountPoints extension property must match the sequence specified in `/etc/vfstab`.

---

7. **Enable the HAStoragePlus resource.**

```
# scswitch -e -j nfs1-hastp-rs
```

8. **Set up the dependencies between the application server and HAStoragePlus.**

```
# scrgadm -c -j nfsserver-rs -y \
Resource_Depencencies=nfs1=hastp-rs
```

How to Upgrade from HAStorage With CFS to HAStoragePlus With Failover Filesystem

HAStorage might not be supported in a future release of Sun Cluster. Equivalent functionality is supported by HAStoragePlus. To upgrade from HAStorage with CFS to HAStoragePlus with Failover Filesystem (FFS), complete the following steps.

The following example uses a simple HA-NFS resource active with HAStorage. The ServicePaths are the diskgroup `nfsdg` and the AffinityOn property is TRUE. Furthermore, the HA-NFS resource has `Resource_Dependencies` set to HAStorage resource.

1. **Remove the dependencies the application resource has on HAStorage resource.**

```
# scrgadm -c -j nfsserver-rs -y Resource_Dependencies=""'
```

2. **Disable the HAStorage resource.**

```
# scswitch -n -j nfs1storage-rs
```

3. **Remove the HAStorage resource from the application resource group.**

```
# scrgadm -r -j nfs1storage-rs
```

4. **Unregister the HAStorage resource type.**

```
# scrgadm -r -t SUNW.HAStorage
```

5. **Modify `/etc/vfstab` to remove the global flag and change "mount at boot" to "no".**

6. **Create the HAStoragePlus resource.**

   To specify a filesystem mount point, input the following text.

```
# scrgadm -a -j nfs1-hastp-rs -g nfs1-rg -t \
SUNW.HAStoragePlus -x FilesystemMountPoints=/global/nfsdata -x \
AffinityOn=True
```

   To specify global device paths, input the following text.

```
# scrgadm -a -j nfs1-hastp-rs -g nfs1-rg -t \
SUNW.HAStoragePlus -x GlobalDevicePaths=nfsdg -x AffinityOn=True
```

Note – Instead of using the ServicePaths property for HAStorage, you must use the GlobalDevicePaths or FilesystemMountPoints property for HAStoragePlus. The FilesystemMountPoints extension property must match the sequence specified in `/etc/vfstab`.

---

7. **Enable the HAStoragePlus resource.**

```
# scswitch -e -j nfs1-hastp-rs
```

8. **Set up the dependencies between the application server and HAStoragePlus.**

```
# scrgadm -c -j nfsserver-rs -y \
Resource_Depencencies=nfs1=hastp-rs
```

Enabling Highly Available Local File Systems

The HAStoragePlus resource type can be used to make a local file system highly available within a Sun Cluster environment. The local file system partitions must reside on global disk groups with affinity switchovers enabled and the Sun Cluster environment must be configured for failover. This enables the user to make any file system on multi-host disks accessible from any host directly connected to those multi-host disks. (You cannot use HAStoragePlus to make a root file system highly available.) The failback settings must be identical for both the resource group and device group(s).

Using a highly available local file system is strongly recommended for some I/O intensive data services, and a procedure on how to configure the HAStoragePlus resource type has been added to the Registration and Configuration procedures for these data services. For procedures on how to set up the HAStoragePlus resource type for these data services, see the following sections.

- "Registering and Configuring Sun Cluster HA for Oracle" in *Sun Cluster Data Service for Oracle Guide for Solaris OS*
- "Registering and Configuring Sun Cluster HA for Sybase ASE" in *Sun Cluster Data Service for Sybase ASE Guide for Solaris OS*

For the procedure to set up HAStoragePlus resource type for other data services, see "How to Set Up HAStoragePlus Resource Type" on page 86.

Note – The instructions in this section explain how to use the HAStoragePlus resource type with the UNIX file system. For information about using the HAStoragePlus resource type with the Sun StorEdge™ QFS file system, see your Sun StorEdge QFS documentation.

▼ How to Set Up HAStoragePlus Resource Type

The HAStoragePlus resource type was introduced in Sun Cluster 3.0 5/02. This new resource type performs the same functions as HAStorage, and synchronizes the startups between resource groups and disk device groups. The HAStoragePlus resource type has an additional feature to make a local file system highly available. (For background information on making a local file system highly available, see "Enabling Highly Available Local File Systems" on page 86.) To use both of these features, set up the HAStoragePlus resource type.

To set up HAStoragePlus, the local file system partitions must reside on global disk groups with affinity switchovers enabled and the Sun Cluster environment must be configured for failover.

The following example uses a simple NFS service that shares out home directory data from a locally mounted directory /global/local-fs/nfs/export/ home. The example assumes the following:

- The mount point /global/local-fs/nfs will be used to mount a UFS local file system on a Sun Cluster global device partition.
- The /etc/vfstab entry for the /global/local-fs/nfs file system should specify that it is a local file system and the mount boot flag is no.
- The PathPrefix directory (the directory used by HA-NFS to maintain administrative and status information) is on the root directory of the same file system to be mounted (for example, /global/local-fs/nfs).

1. **Become superuser on a cluster member.**
2. **Determine whether the resource type is registered.**

   The following command prints a list of registered resource types.

   ```
   # scrgadm -p | egrep Type
   ```

3. **If you need to, register the resource type.**

   ```
   # scrgadm -a -t SUNW.nfs
   ```

4. **Create the failover resource group nfs-r**

```
# scrgadm -a -g nfs-rg -y PathPrefix=/global/local-fs/nfs
```

5. **Create a logical host resource of type SUNW.LogicalHostname.**

   ```
   # scrgadm -a -j nfs-lh-rs -g nfs-rg -L -l log-nfs
   ```

6. **Register the HAStoragePlus resource type with the cluster.**

```
# scrgadm -a -t SUNW.HAStoragePlus
```

7. **Create the resource nfs-hastp-rs of type HAStoragePlus.**

```
# scrgadm -a -j nfs-hastp-rs -g nfs-rg -t SUNW.HAStoragePlus \
-x FilesystemMountPoints=/global/local-fs/nfs \
-x AffinityOn=TRUE
```

Note – The FilesystemMountPoints extension property can be used to specify a list of one or more file system mount points. This list can consist of both local and global file system mount points. The mount at boot flag is ignored by HAStoragePlus for global file systems.

8. **Bring the resource group nfs-rg online on a cluster node.**

   This node will become the primary node for the `/global/local-fs/nfs` file system's underlying global device partition. The file system `/global/local-fs/nfs` will then be locally mounted on this node

   ```
   # scswitch -Z -g nfs-rg
   ```

9. **Register the SUNW.nfs resource type with the cluster. Create the resource nfs-rs of type SUNW.nfs and specify its resource dependency on the resource `nfs-hastp-rs`.**

   `dfstab.nfs-rs` will be present in `/global/local-fs/nfs/SUNW.nfs`.

   ```
   # scrgadm -a -t SUNW.nfs
   # scrgadm -a -g nfs-rg -j nfs-rs -t SUNW.nfs \
   -y Resource_dependencies=nfs-hastp-rs
   ```

   Note – The `nfs-hastp-rs` resource must be online before you can set the dependency in the nfs resource.

   ---

10. **Bring the resource nfs-rs online.**

    ```
    # scswitch -Z -g nfs-rg
    ```

Caution – Be sure to switch only at the resource group level. Switching at the device group level will confuse the resource group causing it to fail over.

Now whenever the service is migrated to a new node, the primary I/O path for `/global/local-fs/nfs` will always be online and colocated with the NFS servers. The file system `/global/local-fs/nfs` will be locally mounted before starting the NFS server.

Modifying Online the Resource for a Highly Available File System

You might need a highly available file system to remain available while you are modifying the resource that represents the file system. For example, you might need the file system to remain available because storage is being provisioned dynamically. In this situation, modify the resource that represents the highly available file system while the resource is online.

In the Sun Cluster environment, a highly available file system is represented by an HAStoragePlus resource. Sun Cluster enables you to modify an online HAStoragePlus resource as follows:

- Adding file systems to the HAStoragePlus resource
- Removing file systems from the HAStoragePlus resource

Note – Sun Cluster does not enable you to rename a file system while the file system is online.

▼ How to Add File Systems to an Online HAStoragePlus Resource

When you add a file system to an HAStoragePlus resource, the HAStoragePlus resource treats a local file system differently from a global file system.

- The HAStoragePlus resource always automatically mounts a local file system.
- The HAStoragePlus resource automatically mounts a global file system *only* if the AffinityOn extension property of the HAStoragePlus resource is True.

For information about the AffinityOn extension property, see "Synchronizing the Startups Between Resource Groups and Disk Device Groups" on page 80.

1. **On one node of the cluster, become superuser.**

2. **In the /etc/vfstab file on each node of the cluster, add an entry for the mount point of each file system that you are adding.**

   For each entry, set the mount at boot field and the mount options field as follows:

   - Set the mount at boot field to no.
   - If the file system is a global file system, set the mount options field to contain the global option.

3. **Retrieve the list of mount points for the file systems that the HAStoragePlus resource already manages.**

   ```
   # scha_resource_get -O extension -R hasp-resource -G hasp-rg \
   FileSystemMountPoints
   ```

   -R *hasp-resource* Specifies the HAStoragePlus resource to which you are adding file systems

   -G *hasp-rg* Specifies the resource group that contains the HAStoragePlus resource

4. **Modify the `FileSystemMountPoints` extension property of the `HAStoragePlus` resource to contain the following mount points:**

   - The mount points of the file systems that the `HAStoragePlus` resource already manages
   - The mount points of the file systems that you are adding to the `HAStoragePlus` resource

   ```
   # scrgadm -c -j hasp-resource -x FileSystemMountPoints="mount-point-list"
   ```

   `-j` *hasp-resource*
   Specifies the `HAStoragePlus` resource to which you are adding file systems

   `-x FileSystemMountPoints="`*mount-point-list*`"`
   Specifies a comma-separated list of mount points of the file systems that the `HAStoragePlus` resource already manages and the mount points of the file systems that you are adding

5. **Confirm that you have a match between the mount point list of the `HAStoragePlus` resource and the list that you specified in Step 4.**

   ```
   # scha_resource_get -O extension -R hasp-resource -G hasp-rg \
    FileSystemMountPoints
   ```

   `-R` *hasp-resource*
   Specifies the `HAStoragePlus` resource to which you are adding file systems

   `-G` *hasp-rg*
   Specifies the resource group that contains the `HAStoragePlus` resource

6. **Confirm that the `HAStoragePlus` resource is online and not faulted.**

   If the `HAStoragePlus` resource is online and faulted, validation of the resource succeeded, but an attempt by `HAStoragePlus` to mount a file system failed.

   ```
   # scstat -g
   ```

EXAMPLE 2–3 Adding a File System to an Online `HAStoragePlus` Resource

This example shows how to add a file system to an online `HAStoragePlus` resource.

- The `HAStoragePlus` resource is named `rshasp` and is contained in the resource group `rghasp`.
- The `HAStoragePlus` resource named `rshasp` already manages the file system whose mount point is `/global/global-fs/fs1`.
- The mount point of the file system that is to be added is `/global/global-fs/fs2`.

The example assumes that the `/etc/vfstab` file on each cluster node already contains an entry for the file system that is to be added.

EXAMPLE 2–3 Adding a File System to an Online HAStoragePlus Resource *(Continued)*

```
# scha_resource_get -O extension -R rshasp -G rghasp FileSystemMountPoints
STRINGARRAY
/global/global-fs/fs1
# scrgadm -c -j rshasp \
-x FileSystemMountPoints="/global/global-fs/fs1,/global/global-fs/fs2"
# scha_resource_get -O extension -R rshasp -G rghasp FileSystemMountPoints
STRINGARRAY
/global/global-fs/fs1
/global/global-fs/fs2
# scstat -g

 -- Resource Groups and Resources --

             Group Name      Resources
             ----------      ---------
  Resources: rghasp          rshasp


 -- Resource Groups --

             Group Name      Node Name    State
             ----------      ---------    -----
      Group: rghasp          node46       Offline
      Group: rghasp          node47       Online


 -- Resources --

             Resource Name   Node Name    State     Status Message
             -------------   ---------    -----     --------------
   Resource: rshasp          node46       Offline   Offline
   Resource: rshasp          node47       Online    Online
```

▼ How to Remove File Systems From an Online HAStoragePlus Resource

When you remove a file system from an HAStoragePlus resource, the HAStoragePlus resource treats a local file system differently from a global file system.

- The HAStoragePlus resource always automatically unmounts a local file system.
- The HAStoragePlus resource automatically unmounts a global file system *only* if the AffinityOn extension property of the HAStoragePlus resource is True.

For information about the AffinityOn extension property, see "Synchronizing the Startups Between Resource Groups and Disk Device Groups" on page 80.

Caution – Before removing a file system from an online `HAStoragePlus` resource, ensure that no applications are using the file system. When you remove a file system from an online `HAStoragePlus` resource, the file system might be forcibly unmounted. If a file system that an application is using is forcibly unmounted, the application might fail or hang.

---

1. **On one node of the cluster, become superuser.**

2. **Retrieve the list of mount points for the file systems that the `HAStoragePlus` resource already manages.**

   ```
   # scha_resource_get -O extension -R hasp-resource -G hasp-rg \
   FileSystemMountPoints
   ```

`-R` *hasp-resource*	Specifies the `HAStoragePlus` resource from which you are removing file systems
`-G` *hasp-rg*	Specifies the resource group that contains the `HAStoragePlus` resource

3. **Modify the `FileSystemMountPoints` extension property of the `HAStoragePlus` resource to contain *only* the mount points of the file systems that are to remain in the `HAStoragePlus` resource.**

   ```
   # scrgadm -c -j hasp-resource -x FileSystemMountPoints="mount-point-list"
   ```

   `-j` *hasp-resource*
   Specifies the `HAStoragePlus` resource from which you are removing file systems.

   `-x FileSystemMountPoints="`*mount-point-list*`"`
   Specifies a comma-separated list of mount points of the file systems that are to remain in the `HAStoragePlus` resource. This list must *not* include the mount points of the file systems that you are removing.

4. **Confirm that you have a match between the mount point list of the `HAStoragePlus` resource and the list that you specified in Step 3.**

   ```
   # scha_resource_get -O extension -R hasp-resource -G hasp-rg \
   FileSystemMountPoints
   ```

`-R` *hasp-resource*	Specifies the `HAStoragePlus` resource from which you are removing file systems
`-G` *hasp-rg*	Specifies the resource group that contains the `HAStoragePlus` resource

5. **Confirm that the `HAStoragePlus` resource is online and not faulted.**

   If the `HAStoragePlus` resource is online and faulted, validation of the resource succeeded, but an attempt by `HAStoragePlus` to unmount a file system failed.

```
# scstat -g
```

6. **(Optional) From the `/etc/vfstab` file on each node of the cluster, remove the entry for the mount point of each file system that you are removing.**

EXAMPLE 2–4 Removing a File System From an Online `HAStoragePlus` Resource

This example shows how to remove a file system from an online `HAStoragePlus` resource.

- The `HAStoragePlus` resource is named `rshasp` and is contained in the resource group `rghasp`.
- The `HAStoragePlus` resource named `rshasp` already manages the file systems whose mount points are as follows:
  - `/global/global-fs/fs1`
  - `/global/global-fs/fs2`
- The mount point of the file system that is to be removed is `/global/global-fs/fs2`.

```
# scha_resource_get -O extension -R rshasp -G rghasp FileSystemMountPoints
STRINGARRAY
/global/global-fs/fs1
/global/global-fs/fs2
# scrgadm -c -j rshasp -x FileSystemMountPoints="/global/global-fs/fs1"
# scha_resource_get -O extension -R rshasp -G rghasp FileSystemMountPoints
STRINGARRAY
/global/global-fs/fs1
 # scstat -g

 -- Resource Groups and Resources --

             Group Name      Resources
             ----------      ---------
  Resources: rghasp          rshasp


 -- Resource Groups --

             Group Name      Node Name    State
             ----------      ---------    -----
      Group: rghasp          node46       Offline
      Group: rghasp          node47       Online


 -- Resources --

             Resource Name   Node Name    State     Status Message
             -------------   ---------    -----     --------------
   Resource: rshasp          node46       Offline   Offline
   Resource: rshasp          node47       Online    Online
```

▼ How to Recover From a Fault After Modifying an HAStoragePlus Resource

If a fault occurs during a modification of the FileSystemMountPoints extension property, the status of the HAStoragePlus resource is online and faulted. After the fault is corrected, the status of the HAStoragePlus resource is online.

1. **Determine the fault that caused the attempted modification to fail.**

   ```
   # scstat -g
   ```

   The status message of the faulty HAStoragePlus resource indicates the fault. Possible faults are as follows:

   - The device on which the file system should reside does not exist.
   - An attempt by the fsck command to repair a file system failed.
   - The mount point of a file system that you attempted to add does not exist.
   - A file system that you attempted to add cannot be mounted.
   - A file system that you attempted to remove cannot be unmounted.

2. **Correct the fault that caused the attempted modification to fail.**

3. **Repeat the step to modify the FileSystemMountPoints extension property of the HAStoragePlus resource.**

   ```
   # scrgadm -c -j hasp-resource -x FileSystemMountPoints="mount-point-list"
   ```

   -j *hasp-resource*
   Specifies the HAStoragePlus resource that you are modifying

   -x FileSystemMountPoints="*mount-point-list*"
   Specifies a comma-separated list of mount points that you specified in the unsuccessful attempt to modify the highly available file system

4. **Confirm that the HAStoragePlus resource is online and not faulted.**

   ```
   # scstat -g
   ```

EXAMPLE 2–5 Status of a Faulty HAStoragePlus Resource

This example shows the status of a faulty HAStoragePlus resource. This resource is faulty because an attempt by the fsck command to repair a file system failed.

EXAMPLE 2–5 Status of a Faulty `HAStoragePlus` Resource *(Continued)*

```
# scstat -g
 -- Resource Groups and Resources --

             Group Name      Resources
             ----------      ---------
  Resources: rghasp          rshasp


 -- Resource Groups --

             Group Name      Node Name    State
             ----------      ---------    -----
      Group: rghasp          node46       Offline
      Group: rghasp          node47       Online


 -- Resources --

           Resource Name   Node Name    State   Status Message
           -------------   ---------    -----   --------------
 Resource: rshasp          node46       Offline Offline
 Resource: rshasp          node47       Online  Online Faulted - Failed
to fsck: /mnt.
```

Upgrading the `HAStoragePlus` Resource Type

In Sun Cluster 3.1 9/04, the `HAStoragePlus` resource type is enhanced to enable you to modify highly available file systems online. Upgrade the `HAStoragePlus` resource type if all conditions in the following list apply:

- You are upgrading from an earlier version of Sun Cluster.
- You need to use the new features of the `HAStoragePlus` resource type.

For general instructions that explain how to upgrade a resource type, see "Upgrading a Resource Type" on page 33. The information that you need to complete the upgrade of the `HAStoragePlus` resource type is provided in the subsections that follow.

Information for Registering the New Resource Type Version

The relationship between a resource type version and the release of Sun Cluster is shown in the following table. The release of Sun Cluster indicates the release in which the version of the resource type was introduced.

Resource Type Version	Sun Cluster Release
1.0	3.0 5/02
2	3.1 9/04

To determine the version of the resource type that is registered, use one command from the following list:

- `scrgadm -p`
- `scrgadm -pv`

The resource type registration (RTR) file for this resource type is `/usr/cluster/lib/rgm/rtreg/SUNW.HAStoragePlus`.

Information for Migrating Existing Instances of the Resource Type

The information that you need to migrate instances of the `HAStoragePlus` resource type is as follows:

- You can perform the migration at any time.
- If you need to use the new features of the `HAStoragePlus` resource type, the required value of the `Type_version` property is `2`.

Distributing Online Resource Groups Among Cluster Nodes

For maximum availability or optimum performance, some combinations of services require a specific distribution of online resource groups among cluster nodes. Distributing online resource groups involves creating affinities between resource groups for the following purposes:

- Enforcing the required distribution when the resource groups are first brought online
- Preserving the required distribution after an attempt to fail over or switch over a resource group

This section provides the following examples of how to use resource group affinities to distribute online resource groups among cluster nodes:

- Enforcing colocation of a resource group with another resource group
- Specifying a preferred colocation of a resource group with another resource group
- Balancing the load of a set of resource groups
- Specifying that a critical service has precedence
- Delegating the failover or switchover of a resource group
- Combining affinities between resource groups to specify more complex behavior

Resource Group Affinities

An affinity between resource groups restricts on which nodes the resource groups may be brought online simultaneously. In each affinity, a source resource group declares an affinity for a target resource group or several target resource groups. To create an affinity between resource groups, set the `RG_affinities` resource group property of the source as follows:

`-y RG_affinities=`*operator target-rg-list*

Note – Do not include a space between *operator* and *target-rg-list*.

operator Specifies the type of affinity that you are creating. For more information, see Table 2–2.

target-rg-list Specifies a comma-separated list of resource groups that are the target of the affinity that you are creating. You may specify a single resource group in the list.

TABLE 2–2 Types of Affinities Between Resource Groups

Operator	Affinity Type	Effect
+	Weak positive	If possible, the source is brought online on a node or on nodes where the target is online or starting. However, the source and the target are allowed to be online on different nodes.
++	Strong positive	The source is brought online only on a node or on nodes where the target is online or starting. The source and the target are *not* allowed to be online on different nodes.
-	Weak negative	If possible, the source is brought online on a node or on nodes where the target is *not* online or starting. However, the source and the target are allowed to be online on the same node.
--	Strong negative	The source is brought online only on a node or on nodes where the target is not online. The source and the target are *not* allowed to be online on the same node.
+++	Strong positive with failover delegation	Same as strong positive, except that an attempt by the source to fail over is delegated to the target. For more information, see "Delegating the Failover or Switchover of a Resource Group" on page 102.

Weak affinities take precedence over `Nodelist` preference ordering.

The current state of other resource groups might prevent a strong affinity from being satisfied on any node. In this situation, the resource group that is the source of the affinity remains offline. If other resource groups' states change to enable the strong affinities to be satisfied, the resource group that is the source of the affinity comes back online.

Note – Use caution when declaring a strong affinity on a source resource group for more than one target resource group. If all declared strong affinities cannot be satisfied, the source resource group remains offline.

Enforcing Colocation of a Resource Group With Another Resource Group

A service that is represented by one resource group might depend so strongly on a service in a second resource group that both services must run on the same node. For example, an application that is comprised of multiple interdependent service daemons might require that all daemons run on the same node.

In this situation, force the resource group of the dependent service to be colocated with the resource group of the other service. To enforce colocation of a resource group with another resource group, declare on the resource group a strong positive affinity for the other resource group.

```
# scrgadm -c|-a -g source-rg -y RG_affinities=++target-rg
```

`-g` *source-rg*
Specifies the resource group that is the source of the strong positive affinity. This resource group is the resource group *on* which you are declaring a strong positive affinity for another resource group.

`-y RG_affinities=++`*target-rg*
Specifies the resource group that is the target of the strong positive affinity. This resource group is the resource group *for* which you are declaring a strong positive affinity.

A resource group follows the resource group for which it has a strong positive affinity. However, a resource group that declares a strong positive affinity is prevented from failing over to a node on which the target of the affinity is not already running.

Note – Only failovers that are initiated by a resource monitor are prevented. If a node on which the source resource group and target resource group are running fails, both resource groups are restarted on the same surviving node.

For example, a resource group `rg1` declares a strong positive affinity for resource group `rg2`. If `rg2` fails over to another node, `rg1` also fails over to that node. This failover occurs even if all the resources in `rg1` are operational. However, if a resource in `rg1` attempts to fail over `rg1` to a node where `rg2` is not running, this attempt is blocked.

If you require a resource group that declares a strong positive affinity to be allowed to fail over, you must delegate the failover. For more information, see "Delegating the Failover or Switchover of a Resource Group" on page 102.

EXAMPLE 2–6 Enforcing Colocation of a Resource Group With Another Resource Group

This example shows the command for modifying resource group `rg1` to declare a strong positive affinity for resource group `rg2`. As a result of this affinity relationship, `rg1` is brought online only on nodes where `rg2` is running. This example assumes that both resource groups exist.

```
# scrgadm -c -g rg1 -y RG_affinities=++rg2
```

Specifying a Preferred Colocation of a Resource Group With Another Resource Group

A service that is represented by one resource group might use a service in a second resource group. As a result, these services run most efficiently if they run on the same node. For example, an application that uses a database runs most efficiently if the application and the database run on the same node. However, the services can run on different nodes because the reduction in efficiency is less disruptive than additional failovers of resource groups.

In this situation, specify that both resource groups should be colocated if possible. To specify preferred colocation of a resource group with another resource group, declare on the resource group a weak positive affinity for the other resource group.

```
# scrgadm -c|-a -g source-rg -y RG_affinities=+target-rg
```

`-g` *source-rg*
: Specifies the resource group that is the source of the weak positive affinity. This resource group is the resource group *on* which you are declaring a weak positive affinity for another resource group.

`-y RG_affinities=+`*target-rg*
: Specifies the resource group that is the target of the weak positive affinity. This resource group is the resource group *for* which you are declaring a weak positive affinity.

By declaring a weak positive affinity on one resource group for another resource group, you increase the probability of both resource groups running on the same node. The source of a weak positive affinity is first brought online on a node where the target of the weak positive affinity is already running. However, the source of a weak positive affinity does not fail over if a resource monitor causes the target of the affinity to fail over. Similarly, the source of a weak positive affinity does not fail over if the target of the affinity is switched over. In both situations, the source remains online on the node where the source is already running.

Note – If a node on which the source resource group and target resource group are running fails, both resource groups are restarted on the same surviving node.

EXAMPLE 2–7 Specifying a Preferred Colocation of a Resource Group With Another Resource Group

This example shows the command for modifying resource group `rg1` to declare a weak positive affinity for resource group `rg2`. As a result of this affinity relationship, `rg1` and `rg2` are first brought online on the same node. But if a resource in `rg2` causes `rg2` to fail over, `rg1` remains online on the node where the resource groups were first brought online. This example assumes that both resource groups exist.

```
# scrgadm -c -g rg1 -y RG_affinities=+rg2
```

Distributing a Set of Resource Groups Evenly Among Cluster Nodes

Each resource group in a set of resource groups might impose the same load on the cluster. In this situation, by distributing the resource groups evenly among cluster nodes, you can balance the load on the cluster.

To distribute a set of resource groups evenly among cluster nodes, declare on each resource group a weak negative affinity for the other resource groups in the set.

```
# scrgadm -c|-a -g source-rg -y RG_affinities=-target-rg-list
```

`-g` *source-rg*
Specifies the resource group that is the source of the weak negative affinity. This resource group is the resource group *on* which you are declaring a weak negative affinity for other resource groups.

`-y RG_affinities=-`*target-rg-list*
Specifies a comma-separated list of resource groups that are the target of the weak negative affinity. These resource groups are the resource groups *for* which you are declaring a weak negative affinity.

By declaring a weak negative affinity on one resource group for other resource groups, you ensure that a resource group is always brought online on the most lightly loaded node in the cluster. The fewest other resource groups are running on that node. Therefore, the smallest number of weak negative affinities are violated.

EXAMPLE 2–8 Distributing a Set of Resource Groups Evenly Among Cluster Nodes

This example shows the commands for modifying resource groups `rg1`, `rg2`, `rg3`, and `rg4` to ensure that these resource groups are evenly distributed among the available nodes in the cluster. This example assumes that resource groups `rg1`, `rg2`, `rg3`, and `rg4` exist.

```
# scrgadm -c -g rg1 RG_affinities=-rg2,-rg3,-rg4
# scrgadm -c -g rg2 RG_affinities=-rg1,-rg3,-rg4
# scrgadm -c -g rg3 RG_affinities=-rg1,-rg2,-rg4
# scrgadm -c -g rg4 RG_affinities=-rg1,-rg2,-rg3
```

Specifying That a Critical Service Has Precedence

A cluster might be configured to run a combination of mission-critical services and noncritical services. For example, a database that supports a critical customer service might run in the same cluster as noncritical research tasks.

To ensure that the noncritical services do not affect the performance of the critical service, specify that the critical service has precedence. By specifying that the critical service has precedence, you prevent noncritical services from running on the same node as the critical service.

When all nodes are operational, the critical service runs on a different node from the noncritical services. However, a failure of the critical service might cause the service to fail over to a node where the noncritical services are running. In this situation, the noncritical services are taken offline immediately to ensure that the computing resources of the node are fully dedicated to the mission-critical service.

To specify that a critical service has precedence, declare on the resource group of each noncritical service a strong negative affinity for the resource group that contains the critical service.

```
# scrgadm -c|-a -g noncritical-rg -y RG_affinities=--critical-rg
```

`-g` *noncritical-rg*
: Specifies the resource group that contains a noncritical service. This resource group is the resource group *on* which you are declaring a strong negative affinity for another resource group.

`-y RG_affinities=--`*critical-rg*
: Specifies the resource group that contains the critical service. This resource group is the resource group *for* which you are declaring a strong negative affinity.

A resource group moves away from a resource group for which it has a strong negative affinity.

EXAMPLE 2–9 Specifying That a Critical Service Has Precedence

This example shows the commands for modifying the noncritical resource groups `ncrg1` and `ncrg2` to ensure that the critical resource group `mcdbrg` has precedence over these resource groups. This example assumes that resource groups `mcdbrg`, `ncrg1`, and `ncrg2` exist.

```
# scrgadm -c -g ncrg1 RG_affinities=--mcdbrg
# scrgadm -c -g ncrg2 RG_affinities=--mcdbrg
```

Delegating the Failover or Switchover of a Resource Group

The source resource group of a strong positive affinity cannot fail over or be switched over to a node where the target of the affinity is not running. If you require the source resource group of a strong positive affinity to be allowed to fail over or be switched over, you must delegate the failover to the target resource group. When the target of the affinity fails over, the source of the affinity is forced to fail over with the target.

Note – You might need to switch over the source resource group of a strong positive affinity that is specified by the ++ operator. In this situation, switch over the target of the affinity and the source of the affinity at the same time.

To delegate failover or switchover of a resource group to another resource group, declare on the resource group a strong positive affinity with failover delegation for the other resource group.

```
# scrgadm -c|-a -g source-rg -y RG_affinities=+++target-rg
```

`-g` *source-rg*

Specifies the resource group that is delegating failover or switchover. This resource group is the resource group *on* which you are declaring a strong positive affinity with failover delegation for another resource group.

`-y RG_affinities=+++`*target-rg*

Specifies the resource group to which *source-rg* delegates failover or switchover. This resource group is the resource group *for* which you are declaring a strong positive affinity with failover delegation.

A resource group may declare a strong positive affinity with failover delegation for at most one resource group. However, a given resource group may be the target of strong positive affinities with failover delegation that are declared by any number of other resource groups.

A strong positive affinity with failover delegation is not fully symmetric. The target can come online while the source remains offline. However, if the target is offline, the source cannot come online.

If the target declares a strong positive affinity with failover delegation for a third resource group, failover or switchover is further delegated to the third resource group. The third resource group performs the failover or switchover, forcing the other resource groups to fail over or be switched over also.

EXAMPLE 2–10 Delegating the Failover or Switchover of a Resource Group

This example shows the command for modifying resource group `rg1` to declare a strong positive affinity with failover delegation for resource group `rg2`. As a result of this affinity relationship, `rg1` delegates failover or switchover to `rg2`. This example assumes that both resource groups exist.

```
# scrgadm -c -g rg1 -y RG_affinities=+++rg2
```

Combining Affinities Between Resource Groups

You can create more complex behaviors by combining multiple affinities. For example, the state of an application might be recorded by a related replica server. The node selection requirements for this example are as follows:

- The replica server must run on a different node from the application.
- If the application fails over from its current node, the application should fail over to the node where the replica server is running.
- If the application fails over to the node where the replica server is running, the replica server must fail over to a different node. If no other node is available, the replica server must go offline.

You can satisfy these requirements by configuring resource groups for the application and the replica server as follows:

- The resource group that contains the application declares a weak positive affinity for the resource group that contains the replica server.
- The resource group that contains the replica server declares a strong negative affinity for the resource group that contains the application.

EXAMPLE 2–11 Combining Affinities Between Resource Groups

This example shows the commands for combining affinities between the following resource groups.

- Resource group `app-rg` represents an application whose state is tracked by a replica server.
- Resource group `rep-rg` represents the replica server.

In this example, the resource groups declare affinities as follows:

- Resource group `app-rg` declares a weak positive affinity for resource group `rep-rg`.
- Resource group `rep-rg` declares a strong negative affinity for resource group `app-rg`.

This example assumes that both resource groups exist.

```
# scrgadm -c -g app-rg RG_affinities=+rep-rg
# scrgadm -c -g rep-rg RG_affinities=--app-rg
```

Freeing Node Resources by Offloading Noncritical Resource Groups

Note – The use of strong negative affinities between resource groups provides a simpler method for offloading noncritical resource groups. For more information, see "Distributing Online Resource Groups Among Cluster Nodes" on page 97.

Prioritized Service Management (`RGOffload`) allows your cluster to automatically free a node's resources for critical data services. `RGOffload` is used when the startup of a critical failover data service requires a Non-Critical, scalable or failover data service to be brought offline. `RGOffload` is used to offload resource groups containing noncritical data services.

Note – **The critical data service must be a failover data service.** The data service to be offloaded can be a failover or scalable data service.

▼ How to Set Up an `RGOffload` Resource

1. **Become superuser on a cluster member.**

2. **Determine whether the `RGOffload` resource type is registered.**

   The following command prints a list of resource types.

   ```
   # scrgadm -p|egrep SUNW.RGOffload
   ```

3. **If needed, register the resource type**

   ```
   # scrgadm -a -t SUNW.RGOffload
   ```

   .

4. **Set the Desired_primaries to zero in each resource group to be offloaded by the `RGOffload` resource.**

   ```
   # scrgadm -c -g offload-rg -y Desired_primaries=0
   ```

5. **Add the `RGOffload` resource to the critical failover resource group and set the extension properties.**

   Do not place a resource group on more than one resource's `rg_to_offload` list. Placing a resource group on multiple `rg_to_offload` lists may cause the resource group to be taken offline and brought back online repeatedly.

   See "Configuring `RGOffload` Extension Properties" on page 107 for extension property descriptions.

   ```
   # scrgadm -aj rgoffload-resource \
   -t SUNW.RGOffload -g critical-rg \
   -x rg_to_offload=offload-rg-1, offload-rg-2, ... \
   -x continue_to_offload=TRUE \
   -x max_offload_retry=15
   ```

   Note – Extension properties other than `rg_to_offload` are shown with default values here. `rg_to_offload` is a comma-separated list of resource groups that are not dependent on each other. This list cannot include the resource group to which the `RGOffload` resource is being added.

   ---

6. **Enable the `RGOffload` resource.**

   ```
   # scswitch -ej rgoffload-resource
   ```

7. **Set the dependency of the critical failover resource on the `RGOffload` resource.**

```
# scrgadm -c -j critical-resource \
-y Resource_dependencies=rgoffload-resource
```

`Resource_dependencies_weak` may also be used. Using `Resource_dependencies_weak` on the `RGOffload` resource type will allow the critical failover resource to start up even if errors are encountered during offload of `offload-rg`.

8. **Bring the resource group to be offloaded online.**

   ```
   # scswitch -z -g offload-rg, offload-rg-2, ... -h [nodelist]
   ```

   The resource group remains online on all nodes where the critical resource group is offline. The fault monitor prevents the resource group from running on the node where the critical resource group is online.

   Because `Desired_primaries` for resource groups to be offloaded is set to 0 (see Step 4), the "-Z" option will not bring these resource groups online.

9. **If the critical failover resource group is not online, bring it online.**

   ```
   # scswitch -Z -g critical-rg
   ```

SPARC: Example – Configuring an `RGOffload` Resource

This example describes how to configure an `RGOffload` resource (`rgofl`), the critical resource group that contains the `RGOffload` resource (`oracle_rg)`, and scalable resource groups that are offloaded when the critical resource group comes online (`IWS-SC`, `IWS-SC-2`). The critical resource in this example is `oracle-server-rs`.

In this example, `oracle_rg`, `IWS-SC`, and `IWS-SC-2` can be mastered on any node of cluster "triped", phys-triped-1, phys-triped-2, phys-triped-3.

```
[Determine whether the SUNW.RGOffload resource type is registered.]
# scrgadm -p|egrep SUNW.RGOffload

[If needed, register the resource type.]
# scrgadm -a -t SUNW.RGOffload

[Set the Desired_primaries to zero in each resource group to be
offloaded by the RGOffload resource.]
# scrgadm -c -g IWS-SC-2 -y Desired_primaries=0
# scrgadm -c -g IWS-SC -y Desired_primaries=0

[Add the RGOffload resource to the critical resource group and set
the extension properties.]
# scrgadm -aj rgofl -t SUNW.RGOffload -g oracle_rg \
-x rg_to_offload=IWS-SC,IWS-SC-2 -x continue_to_offload=TRUE \
-x max_offload_retry=15

[Enable the RGOffload resource.]
# scswitch -ej rgofl

[Set the dependency of the critical failover resource to the RGOffload resource.]
```

```
# scrgadm -c -j oracle-server-rs -y Resource_dependencies=rgofl
```

[*Bring the resource groups to be offloaded online on all nodes.*]

```
# scswitch -z -g IWS-SC,IWS-SC-2 -h phys-triped-1,phys-triped-2,phys-triped-3
```

[*If the critical failover resource group is not online, bring it online.*]

```
# scswitch -Z -g oracle_rg
```

Configuring RGOffload Extension Properties

Typically, you use the command line scrgadm -x *parameter=value* to configure extension properties when you create the RGOffload resource. See Appendix A for details on all of the Sun Cluster standard properties.

Table 2–3 describes extension properties that you can configure for RGOffload. The Tunable entries indicate when you can update the property.

TABLE 2–3 RGOffload Extension Properties

Name/Data Type	Default
rg_to_offload (string)	A comma-separated list of resource groups that need to be offloaded on a node when a critical failover resource group starts up on that node. This list should not contain resource groups that depend upon each other. This property has no default and must be set. RGOffload does not check for dependency loops in the list of resource groups set in the rg_to_offload extension property. For example, if resource group RG-B depends in some way on RG-A, then both RG-A and RG-B should not be included in rg_to_offload. **Default:** None **Tunable:** Any time
continue_to_offload (Boolean)	A Boolean to indicate whether to continue offloading the remaining resource groups in the rg_to_offload list after an error in offloading a resource group occurs. This property is only used by the START method. **Default:** True **Tunable:** Any time

TABLE 2–3 RGOffload Extension Properties *(Continued)*

Name/Data Type	Default
max_offload_retry (integer)	The number of attempts to offload a resource group during startup in case of failures due to cluster or resource group reconfiguration. There is an interval of 10 seconds between successive retries. Set the max_offload_retry so that (the number of resource groups to be offloaded * max_offload_retry * 10 seconds) is less than the Start_timeout for the RGOffload resource. If this number is close to or more than the Start_timeout number, the START method of RGOffload resource may time out before maximum offload attempts are completed. This property is only used by the START method. **Default:** 15 **Tunable:** Any time

Fault Monitor

The Fault Monitor probe for RGOffload resource is used to keep resource groups specified in the rg_to_offload extension property offline on the node mastering the critical resource. During each probe cycle, Fault Monitor verifies that resource groups to be offloaded (offload-rg) are offline on the node mastering the critical resource. If the offload-rg is online on the node mastering the critical resource, the Fault Monitor attempts to start offload-rg on nodes other than the node mastering the critical resource, thereby bringing offload-rg offline on the node mastering the critical resource.

Because desired_primaries for offload-rg is set to 0, offloaded resource groups are not restarted on nodes that become available later. Therefore, the RGOffload Fault Monitor attempts to start up offload-rg on as many primaries as possible, until maximum_primaries limit is reached, while keeping offload-rg offline on the node mastering the critical resource.

RGOffload attempts to start up all offloaded resource groups unless they are in the MAINTENANCE or UNMANAGED state. To place a resource group in an UNMANAGED state, use the scswitch command.

```
# scswitch -u -g resourcegroup
```

The Fault Monitor probe cycle is invoked after every Thorough_probe_interval.

Replicating and Upgrading Configuration Data for Resource Groups, Resource Types, and Resources

If you require identical resource configuration data on two clusters, you can replicate the data to the second cluster to save the laborious task of setting it up again. Use `scsnapshot` to propagate the resource configuration information from one cluster to another cluster. To save effort, ensure that your resource-related configuration is stable and you do not need to make any major changes to the resource configuration, before copying the information to a second cluster.

Configuration data for resource groups, resource types, and resources can be retrieved from the Cluster Configuration Repository (CCR) and formatted as a shell script. The script can be used to perform the following tasks:

- Replicate configuration data on a cluster that does not have configured resource groups, resource types, or resources
- Upgrade configuration data on a cluster that has configured resource groups, resource types, and resources

The `scsnapshot` tool retrieves configuration data that is stored in the CCR. Other configuration data are ignored. The `scsnapshot` tool ignores the dynamic state of different resource groups, resource types, and resources.

▼ How to Replicate Configuration Data on a Cluster Without Configured Resource Groups, Resource Types, and Resources

This procedure replicates configuration data on a cluster that does not have configured resource groups, resource types, and resources. In this procedure, a copy of the configuration data is taken from one cluster and used to generate the configuration data on another cluster.

1. **Using the system administrator role, log in to any node in the cluster from which you want to copy the configuration data.**

   For example, `node1`.

   The system administrator role gives you the following role-based access control (RBAC) rights:

   - `solaris.cluster.resource.read`
   - `solaris.cluster.resource.modify`

2. **Retrieve the configuration data from the cluster.**

   ```
   node1 % scsnapshot -s scriptfile
   ```

   The scsnapshot tool generates a script called *scriptfile*. For more information about using the scsnapshot tool, see the scsnapshot(1m) man page.

3. **Edit the script to adapt it to the specific features of the cluster where you want to replicate the configuration data.**

   For example, you might have to change the IP addresses and host names that are listed in the script.

4. **Launch the script from any node in the cluster where you want to replicate the configuration data.**

   The script compares the characteristics of the local cluster to the cluster where the script was generated. If the characteristics are not the same, the script writes an error and ends. A message asks whether you want to rerun the script, using the -f option. The -f option forces the script to run, despite any difference in characteristics. If you use the -f option, ensure that you do not create inconsistencies in your cluster.

   The script verifies that the Sun Cluster resource type exists on the local cluster. If the resource type does not exist on the local cluster, the script writes an error and ends. A message asks whether you want to install the missing resource type before running the script again.

▼ How to Upgrade Configuration Data on a Cluster With Configured Resource Groups, Resource Types, and Resources

This procedure upgrades configuration data on a cluster that already has configured resource groups, resource types, and resources. This procedure can also be used to generate a configuration template for resource groups, resource types, and resources.

In this procedure, the configuration data on cluster1 is upgraded to match the configuration data on cluster2.

1. **Using the system administrator role, log on to any node in `cluster1`.**

   For example, node1.

   The system administrator role gives you the following RBAC rights:

   - solaris.cluster.resource.read
   - solaris.cluster.resource.modify

2. **Retrieve the configuration data from the cluster by using the image file option of the `scsnapshot` tool:**

   ```
   node1% scsnapshot -s scriptfile1 -o imagefile1
   ```

When run on node1, the scsnapshot tool generates a script that is called *scriptfile1*. The script stores configuration data for the resource groups, resource types, and resources in an image file that is called *imagefile1*. For more information about using the scsnapshot tool, see the scsnapshot(1M) man page.

3. **Repeat Step 1 through Step 2 on a node in cluster2:**

   ```
   node2 % scsnapshot -s scriptfile2 -o imagefile2
   ```

4. **On node1, generate a script to upgrade the configuration data on cluster1 with configuration data from cluster2:**

   ```
   node1 % scsnapshot -s scriptfile3 imagefile1 imagefile2
   ```

   This step uses the image files that you generated in Step 2 and Step 3, and generates a new script that is called *scriptfile3*.

5. **Edit the script that you generated in Step 4 to adapt it to the specific features of the cluster1, and to remove data specific to cluster2.**

6. **From node1, launch the script to upgrade the configuration data.**

   The script compares the characteristics of the local cluster to the cluster where the script was generated. If the characteristics are not the same, the script writes an error and ends. A message asks whether you want to rerun the script, using the -f option. The -f option forces the script to run, despite any difference in characteristics. If you use the -f option, ensure that you do not create inconsistencies in your cluster.

   The script verifies that the Sun Cluster resource type exists on the local cluster. If the resource type does not exist on the local cluster, the script writes an error and ends. A message asks whether you want to install the missing resource type before running the script again.

Tuning Fault Monitors for Sun Cluster Data Services

Each data service that is supplied with the Sun Cluster product has a built-in fault monitor. The fault monitor performs the following functions:

- Detecting the unexpected termination of processes for the data service server
- Checking the health of the data service

The fault monitor is contained in the resource that represents the application for which the data service was written. You create this resource when you register and configure the data service. For more information, see the documentation for the data service.

System properties and extension properties of this resource control the behavior of the fault monitor. The default values of these properties determine the preset behavior of the fault monitor. The preset behavior should be suitable for most Sun Cluster installations. Therefore, you should tune a fault monitor *only* if you need to modify this preset behavior.

Tuning a fault monitor involves the following tasks:

- Setting the interval between fault monitor probes
- Setting the timeout for fault monitor probes
- Defining the criteria for persistent faults
- Specifying the failover behavior of a resource

Perform these tasks when you register and configure the data service. For more information, see the documentation for the data service.

Note – A resource's fault monitor is started when you bring online the resource group that contains the resource. You do not need to start the fault monitor explicitly.

Setting the Interval Between Fault Monitor Probes

To determine whether a resource is operating correctly, the fault monitor probes this resource periodically. The interval between fault monitor probes affects the availability of the resource and the performance of your system as follows:

- The interval between fault monitor probes affects the length of time that is required to detect a fault and respond to the fault. Therefore, if you decrease the interval between fault monitor probes, the time that is required to detect a fault and respond to the fault is also decreased. This decrease enhances the availability of the resource.
- Each fault monitor probe consumes system resources such as processor cycles and memory. Therefore, if you decrease the interval between fault monitor probes, the performance of the system is degraded.

The optimum interval between fault monitor probes also depends on the time that is required to respond to a fault in the resource. This time depends on how the complexity of the resource affects the time that is required for operations such as restarting the resource.

To set the interval between fault monitor probes, set the `Thorough_probe_interval` system property of the resource to the interval in seconds that you require.

Setting the Timeout for Fault Monitor Probes

The timeout for fault monitor probes specifies the length of time that a fault monitor waits for a response from a resource to a probe. If the fault monitor does not receive a response within this timeout, the fault monitor treats the resource as faulty. The time that a resource requires to respond to a fault monitor probe depends on the operations that the fault monitor performs to probe the resource. For information about operations that a data service's fault monitor performs to probe a resource, see the documentation for the data service.

The time that is required for a resource to respond also depends on factors that are unrelated to the fault monitor or the application, for example:

- System configuration
- Cluster configuration
- System load
- Amount of network traffic

To set the timeout for fault monitor probes, set the `Probe_timeout` extension property of the resource to the timeout in seconds that you require.

Defining the Criteria for Persistent Faults

To minimize the disruption that transient faults in a resource cause, a fault monitor restarts the resource in response to such faults. For persistent faults, more disruptive action than restarting the resource is required:

- For a failover resource, the fault monitor fails over the resource to another node.
- For a scalable resource, the fault monitor takes the resource offline.

A fault monitor treats a fault as persistent if the number of complete failures of a resource exceeds a specified threshold within a specified retry interval. Defining the criteria for persistent faults enables you to set the threshold and the retry interval to accommodate the performance characteristics of your cluster and your availability requirements.

Complete Failures and Partial Failures of a Resource

A fault monitor treats some faults as a *complete failure* of a resource. A complete failure typically causes a complete loss of service. The following failures are examples of a complete failure:

- Unexpected termination of the process for a data service server
- Inability of a fault monitor to connect to a data service server

A complete failure causes the fault monitor to increase by 1 the count of complete failures in the retry interval.

A fault monitor treats other faults as a *partial failure* of a resource. A partial failure is less serious than a complete failure, and typically causes a degradation of service, but not a complete loss of service. An example of a partial failure is an incomplete response from a data service server before a fault monitor probe is timed out.

A partial failure causes the fault monitor to increase by a fractional amount the count of complete failures in the retry interval. Partial failures are still accumulated over the retry interval.

The following characteristics of partial failures depend on the data service:

- The types of faults that the fault monitor treats as partial failure
- The fractional amount that each partial failure adds to the count of complete failures

For information about faults that a data service's fault monitor detects, see the documentation for the data service.

Dependencies of the Threshold and the Retry Interval on Other Properties

The maximum length of time that is required for a single restart of a faulty resource is the sum of the values of the following properties:

- `Thorough_probe_interval` system property
- `Probe_timeout` extension property

To ensure that you allow enough time for the threshold to be reached within the retry interval, use the following expression to calculate values for the retry interval and the threshold:

retry-interval ≥ *threshold* × (*thorough-probe-interval* + *probe-timeout*)

System Properties for Setting the Threshold and the Retry Interval

To set the threshold and the retry interval, set the following system properties of the resource:

- To set the threshold, set the `Retry_count` system property to the maximum allowed number of complete failures.
- To set the retry interval, set the `Retry_interval` system property to the interval in seconds that you require.

Specifying the Failover Behavior of a Resource

The failover behavior of a resource determines how the RGM responds to the following faults:

- Failure of the resource to start
- Failure of the resource to stop
- Failure of the resource's fault monitor to stop

To specify the failover behavior of a resource, set the `Failover_mode` system property of the resource. For information about the possible values of this property, see the description of the `Failover_mode` system property in "Resource Properties" on page 123.

APPENDIX **A**

Standard Properties

This appendix describes the standard resource type, resource group, and resource properties. This appendix also describes the resource property attributes that are available for changing system-defined properties and creating extension properties.

This appendix contains the following sections:

Resource Type Properties

The following information describes the resource type properties that are defined by Sun Cluster. The property values are categorized as follows (after Category):

- **Required** – The property requires an explicit value in the Resource Type Registration (RTR) file. Otherwise, the object to which the property belongs cannot be created. A blank or the empty string is not allowed as a value.
- **Conditional** – To exist, the property must be declared in the RTR file. Otherwise, the RGM does not create the property and the property is not available to administrative utilities. A blank or the empty string is allowed. If the property is declared in the RTR file but no value is specified, the RGM supplies a default value.
- **Conditional/Explicit** – To exist, the property must be declared in the RTR file with an explicit value. Otherwise, the RGM does not create the property and the property is not available to administrative utilities. A blank or the empty string is not allowed.

- **Optional** – The property can be declared in the RTR file. If the property is not declared in the RTR file, the RGM creates it and supplies a default value. If the property is declared in the RTR file but no value is specified, the RGM supplies the same default value as if the property were not declared in the RTR file.

Resource type properties cannot be updated by administrative utilities with the exception of Installed_nodes and RT_system, which cannot be declared in the RTR file and must be set by the administrator.

Property names are shown first, followed by a description.

API_version (integer)
The version of the resource management API that is used by this resource type implementation.

The following information summarizes the maximum API_version that is supported by each release of Sun Cluster.

Before and up to 3.1	2
3.1 10/03	3
3.1 4/04	4
3.1 9/04	5

Declaring a value for API_version that is greater than 2 in the RTR file prevents that resource type from being installed on a version of Sun Cluster that supports a lower maximum version. For example, if you declare API_version=5 for a resource type, that resource type cannot be installed on any version of Sun Cluster that was released before 3.1 9/04.

Category: Optional

Default: 2

Tunable: No

Boot (string)
An optional callback method: the path to the program that the RGM invokes on a node, which joins or rejoins the cluster when a resource of this type is already managed. This method is expected to do initialization actions for resources of this type that are similar to the Init method.

Category: Conditional/Explicit

Default: None

Tunable: No

Failover (boolean)
TRUE indicates that resources of this type cannot be configured in any group that can be online on multiple nodes at once.

Category: Optional

Default: FALSE

Tunable: No

Fini (string)

An optional callback method: the path to the program that the RGM invokes when a resource of this type is removed from RGM management.

Category: Conditional/Explicit

Default: None

Tunable: No

Init (string)

An optional callback method: the path to the program that the RGM invokes when a resource of this type becomes managed by the RGM.

Category: Conditional/Explicit

Default: None

Tunable: No

Init_nodes (enum)

The values can be RG_primaries (just the nodes that can master the resource) or RT_installed_nodes (all nodes on which the resource type is installed). Indicates the nodes on which the RGM is to call the Init, Fini, Boot, and Validate methods.

Category: Optional

Default: RG_primaries

Tunable: No

Installed_nodes (string_array)

A list of the cluster node names on which the resource type is allowed to be run. The RGM automatically creates this property. The cluster administrator can set the value. You cannot declare this property in the RTR file.

Category: Can be configured by the cluster administrator

Default: All cluster nodes

Tunable: Yes

Is_logical_hostname (boolean

TRUE indicates that this resource type is some version of the LogicalHostname resource type that manages failover Internet Protocol (IP) addresses.

Category: Query-only

Default: No default

Tunable: No

Is_shared_address (boolean)
TRUE indicates that this resource type is some version of the SharedAddress resource type that manages failover Internet Protocol (IP) addresses.

Category: Query-only

Default: No default

Tunable: No

Monitor_check (string)
An optional callback method: the path to the program that the RGM invokes before doing a monitor-requested failover of a resource of this type.

Category: Conditional/Explicit

Default: No default

Tunable: No

Monitor_start (string)
An optional callback method: the path to the program that the RGM invokes to start a fault monitor for a resource of this type.

Category: Conditional/Explicit

Default: No default

Tunable: No

Monitor_stop (string)
A callback method that is required if Monitor_start is set: the path to the program that the RGM invokes to stop a fault monitor for a resource of this type.

Category: Conditional/Explicit

Default: No default

Tunable: No

Pkglist (string_array)
An optional list of packages that are included in the resource type installation.

Category: Conditional/Explicit

Default: No default

Tunable: No

Postnet_stop (string)
An optional callback method: the path to the program that the RGM invokes after calling the Stop method of any network-address resources on which a resource of this type depends. After the network interfaces are configured down, this method must perform Stop actions.

Category: Conditional/Explicit

Default: No default

Tunable: No

`Prenet_start` (`string`)
An optional callback method: the path to the program that the RGM invokes before calling the `Start` method of any network-address resources on which a resource of this type depends. This method is expected to perform `Start` actions that must be performed before network interfaces are configured.

Category: Conditional/Explicit

Default: No default

Tunable: No

`Resource_type` (`string`)
The name of the resource type. To view the names of the currently registered resource types, use:

```
scrgadm -p
```

In Sun Cluster 3.1 and later releases, a resource type name includes the version, which is mandatory:

```
vendor_id.resource_type:version
```

The three components of the resource type name are properties that are specified in the RTR file as *Vendor_id*, *Resource_type*, and *RT_version*. The `scrgadm` command inserts the period (.) and colon (:) delimiters. The `RT_version` suffix of the resource type name is the same value as the `RT_version` property. To ensure that the *Vendor_id* is unique, the recommended approach is to use the stock symbol for the company creating the resource type. Resource type names that were created before Sun Cluster 3.1 continue to use the syntax:

```
vendor_id.resource_type
```

Category: Required

Default: Empty string

Tunable: No

`RT_basedir` (`string`)
The directory path that is used to complete relative paths for callback methods. This path is expected to be set to the installation location for the resource type packages. The path must be a complete path, that is, the path must start with a forward slash (`/`). This property is not required if all the method path names are absolute.

Category: Required unless all method path names are absolute

Default: No default

Tunable: No

`RT_description` (`string`)
A brief description of the resource type.

Category: Conditional

Default: Empty string

Tunable: No

RT_system (boolean)
Indicates, when set to TRUE on a resource type, that the permitted scrgadm(1M) operations on that resource type are restricted. A resource type whose RT_system value is set to TRUE is called a system resource type. Editing the RT_system property itself is never restricted, regardless of the current value of RT_system.

Category: Optional

Default: FALSE

Tunable: Yes

RT_version (string)
Starting in Sun Cluster 3.1, a required version string of this resource type implementation. The RT_version is the suffix component of the full resource type name. The RT_version property, which was optional in Sun Cluster 3.0, is mandatory in Sun Cluster 3.1 and later releases.

Category: Optional/Explicit or Required

Default: No default

Tunable: No

Single_instance (boolean)
If TRUE, indicates that only one resource of this type can exist in the cluster. The RGM allows only one resource of this type to run cluster wide at one time.

Category: Optional

Default: FALSE

Tunable: No

Start (string)
A callback method: the path to the program that the RGM invokes to start a resource of this type.

Category: Required unless the RTR file declares a Prenet_start method

Default: No default

Tunable: No

Stop (string)
A callback method: the path to the program that the RGM invokes to stop a resource of this type.

Category: Required unless the RTR file declares a Postnet_stop method

Default: No default

Tunable: No

Update (string)
An optional callback method: the path to the program that the RGM invokes when properties of a running resource of this type are changed.

Category: Conditional/Explicit

Default: No default

Tunable: No

Validate (string)
An optional callback method: the path to the program that will be invoked to check values for properties of resources of this type.

Category: Conditional/Explicit

Default: No default

Tunable: No

Vendor_ID (string)
See the Resource_type property.

Category: Conditional

Default: No default

Tunable: No

Resource Properties

This section describes the resource properties that are defined by Sun Cluster. The property values are categorized as follows (after Category):

- **Required** – The administrator must specify a value when creating a resource with an administrative utility.
- **Optional** – If the administrator does not specify a value when creating a resource group, the system supplies a default value.
- **Conditional** – The RGM creates the property only if the property is declared in the RTR file. Otherwise, the property does not exist and is not available to system administrators. A conditional property that is declared in the RTR file is optional or required, depending on whether a default value is specified in the RTR file. For details, see the description of each conditional property.
- **Query-only** – Cannot be set directly by an administrative tool.

Tunable lists whether and when you can update resource properties, as follows:

NONE or FALSE	Never
TRUE or ANYTIME	Any time
AT_CREATION	When the resource is added to a cluster
WHEN_DISABLED	When the resource is disabled

Property names are shown first, followed by a description.

Affinity_timeout (integer)
Length of time in seconds during which connections from a given client IP address for any service in the resource are sent to the same server node.

This property is relevant only when Load_balancing_policy is either Lb_sticky or Lb_sticky_wild. In addition, Weak_affinity must be set to FALSE (the default value).

This property is only used for scalable services.

Category: Optional

Default: No default

Tunable: ANYTIME

Cheap_probe_interval (integer)
The number of seconds between invocations of a quick fault probe of the resource. This property is created by the RGM and available to the administrator only if it is declared in the RTR file.

This property is optional if a default value is specified in the RTR file. If the Tunable attribute is not specified in the resource type file, the Tunable value for the property is WHEN_DISABLED.

This property is required if it is declared in the RTR file and the Default attribute is not specified.

Category: Conditional

Default: No default

Tunable: WHEN_DISABLED

Extension properties
Extension properties as declared in the RTR file of the resource's type. The implementation of the resource type defines these properties. "Resource Property Attributes" on page 140 contains information about the individual attributes that you can set for extension properties.

Category: Conditional

Default: No default

Tunable: Depends on the specific property

Failover_mode (enum)

NONE, SOFT, and HARD affect only failover behavior when a start method (Prenet_start or Start) fails. Once the resource has started successfully, however, NONE, SOFT, and HARD have no effect on subsequent resource restart or giveover behavior that the resource monitor initiates with scha_control(1HA) or scha_control(3HA). NONE (the default) indicates that the RGM is to set the resource state on method failure and wait for the user to intervene. SOFT indicates that, if a Start method fails, the RGM is to relocate the resource's group to a different node. If a Stop or Monitor_stop method fails, the RGM sets the resource to Stop_failed state and the resource group to Error_stop_failed state. The RGM then waits for the user to intervene. For Stop or Monitor_stop failures, NONE and SOFT are the same. HARD indicates that, if a Start method fails, the RGM is to relocate the group. If a Stop or Monitor_stop method fails, the RGM is to stop the resource by aborting the cluster node. HARD, NONE, SOFT affect failover behavior when a Start or Prenet_start method fails.

Unlike NONE, SOFT, and HARD, RESTART_ONLY and LOG_ONLY affect all failover behavior, including monitor-initiated (scha_control) restarts of resources and resource groups, and giveovers that are initiated by the resource monitor (scha_control). RESTART_ONLY indicates that the monitor can run scha_control to restart a resource, but attempts to perform a resource group restart or giveover with scha_control subsequently fail. The RGM allows Retry_count restarts within Retry_interval. If Retry_count is exceeded, no resource restarts are permitted. If Failover_mode is set to LOG_ONLY, no resource restarts or giveovers are permitted. Setting Failover_mode to LOG_ONLY is the same as setting Failover_mode to RESTART_ONLY with Retry_count set to zero. If a start method fails, RESTART_ONLY and LOG_ONLY are the same as NONE: no failover occurs and the resource moves to Start_failed state.

Category:	Optional
Default:	No default
Tunable:	ANYTIME

Load_balancing_policy (string)

A string that defines the load-balancing policy in use. This property is used only for scalable services. The RGM automatically creates this property if the Scalable property is declared in the RTR file. Load_balancing_policy can take the following values:

Lb_weighted (the default). The load is distributed among various nodes according to the weights set in the Load_balancing_weights property.

Lb_sticky. A given client (identified by the client IP address) of the scalable service is always sent to the same node of the cluster.

Lb_sticky_wild. A given client's IP address that connects to an IP address of a wildcard sticky service is always sent to the same cluster node, regardless of the port number to which the IP address is coming.

Category:	Conditional/Optional

Default: `Lb_weighted`

Tunable: `AT_CREATION`

`Load_balancing_weights` (`string_array`)
For scalable resources only. The RGM automatically creates this property if the `Scalable` property is declared in the RTR file. The format is *weight@node,weight@node*, where *weight* is an integer that reflects the relative portion of load distributed to the specified *node*. The fraction of load distributed to a node is the weight for this node divided by the sum of all weights. For example, `1@1,3@2` specifies that node 1 receives 1/4 of the load and node 2 receives 3/4. The empty string (“”), the default, sets a uniform distribution. Any node that is not assigned an explicit weight receives a default weight of 1.

If the `Tunable` attribute is not specified in the resource type file, the `Tunable` value for the property is `ANYTIME`. Changing this property revises the distribution for new connections only.

Category: Conditional/Optional

Default: The empty string (“”)

Tunable: `ANYTIME`

method`_timeout` for each callback method in the Type (`integer`)
A time lapse, in seconds, after which the RGM concludes that an invocation of the method has failed.

Category: Conditional/Optional

Default: 3600 (one hour), if the method itself is declared in the RTR file

Tunable: `ANYTIME`

`Monitored_switch` (`enum`)
Set to `Enabled` or `Disabled` by the RGM if the cluster administrator enables or disables the monitor with an administrative utility. If `Disabled`, the monitor does not have its `Start` method called until it is enabled again. If the resource does not have a monitor callback method, this property does not exist.

Category: Query-only

Default: No default

Tunable: Never

`Network_resources_used` (`string_array`)
A list of logical host name or shared address network resources that are used by the resource. For scalable services, this property must refer to shared address resources that exist in a separate resource group. For failover services, this property refers to logical host name or shared address resources that exist in the same resource group. The RGM automatically creates this property if the `Scalable` property is declared in the RTR file. If `Scalable` is not declared in the RTR file, `Network_resources_used` is unavailable unless it is explicitly declared in the RTR file.

If the Tunable attribute is not specified in the resource type file, the Tunable value for the property is AT_CREATION.

Note – The SUNW.Event(5) man page describes how to set up this property for CRNP.

Category: Conditional/Required

Default: No default

Tunable: AT_CREATION

Num_resource_restarts on each cluster node (integer)

You cannot directly set this property, as it is set by the RGM to the number of scha_control, Resource_restart, or Resource_is_restarted calls that have been made for this resource on this node within the past *n* seconds. *n* is the value of the Retry_interval property of the resource. The resource restart counter is reset to zero by the RGM whenever a scha_control giveover is executed by this resource, whether the giveover attempt succeeds or fails.

If a resource type does not declare the Retry_interval property, then the Num_resource_restarts property is not available for resources of that type.

Category: Query-only

Default: No default

Tunable: No

Num_rg_restarts on each cluster node (integer)

You cannot directly set this property, which is set by the RGM to the number of scha_control Restart calls that have been made by this resource for its containing resource group on this node within the past *n* seconds, where *n* is the value of the Retry_interval property of the resource. If a resource type does not declare the Retry_interval property, then the Num_rg_restarts property is not available for resources of that type.

Category: See description

Default: No default

Tunable: No

On_off_switch (enum)

Set to Enabled or Disabled by the RGM if the cluster administrator enables or disables the resource with an administrative utility. If disabled, a resource is brought offline and then has no callbacks invoked until it is enabled again.

Category: Query-only

Default: No default

Tunable: Never

Port_list (string_array)

A list of port numbers on which the server is listening. Appended to each port number is a slash (/) followed by the protocol that is being used by that port, for example, Port_list=80/tcp or Port_list=80/tcp6,40/udp6. You can specify the following protocol values:

- tcp, for TCP IPv4
- tcp6, for TCP IPv6
- udp, for UDP IPv4
- udp6, for UDP IPv6 If the Scalable property is declared in the RTR file, the RGM automatically creates Port_list. Otherwise, this property is unavailable unless it is explicitly declared in the RTR file.

Setting up this property for Apache is described in the *Sun Cluster Data Service for Apache Guide for Solaris OS*.

Category:	Conditional/Required
Default:	No default
Tunable:	AT_CREATION

R_description (string)

A brief description of the resource.

Category:	Optional
Default:	The empty string
Tunable:	ANYTIME

Resource_dependencies (string_array)

A list of resources in the same or in different groups on which this resource has a strong dependency. This resource cannot be started if any resource in the list is not online. If this resource and one of the resources in the list start at the same time, the RGM waits until the resource in the list starts before the RGM starts this resource. If the resource in this resource's Resource_dependencies list does not start, this resource also remains offline. The resource in this resource's list might not start because the resource group for the resource in the list remains offline or because the resource in the list is in a Start_failed state. If this resource remains offline because of a dependency on a resource in a different resource group that fails to start, this resource's group enters a Pending_online_blocked state.

If this resource is brought offline at the same time as those resources in the list, this resource stops before those in the list. However, if this resource remains online or fails to stop, a resource in the list that is in a different resource group stops anyway. Resources in the list cannot be disabled unless this resource is disabled first.

By default in a resource group, application resources have an implicit strong resource dependency on network address resources. Implicit_network_dependencies in "Resource Group Properties" on page 134 contains more information.

Within a resource group, Prenet_start methods are run in dependency order before Start methods. Postnet_stop methods are run in dependency order after Stop methods. In different resource groups, the dependent waits for the depended-on resource to finish Prenet_start and Start before it runs Prenet_start. The depended-on resource waits for the dependent to finish Stop and Postnet_stop before it runs Stop.

Category: Optional

Default: The empty list

Tunable: ANYTIME

Resource_dependencies_restart (string_array)
A list of resources in the same or in different groups on which this resource has a restart dependency.

This property works just as Resource_dependencies does, except that, if any resource in the restart dependency list is restarted, this resource is restarted. The restart of this resource occurs after the resource in the list comes back online.

Category: Optional

Default: The empty list

Tunable: ANYTIME

Resource_dependencies_weak (string_array)
A list of resources in the same or in different groups on which this resource has a weak dependency. A weak dependency determines the order of method calls The RGM calls the Start methods of the resources in this list before the Start method of this resource. The RGM calls the Stop methods of this resource before the Stop methods of those in the list. The resource can still start if those in the list fail to start or remain offline.

If this resource and a resource in its Resource_dependencies_weak list start concurrently, the RGM waits until the resource in the list starts before the RGM starts this resource. If the resource in the list does not start—for example, if the resource group for the resource in the list remains offline or the resource in the list is in a Start_failed state—this resource starts. This resource's resource group might enter a Pending_online_blocked state temporarily as resources in this resource's Resource_dependencies_weak list start. When all resources in the list have started or failed to start, this resource starts and its group reenters the Pending_online state.

If this resource is brought offline at the same time as those in the list, this resource stops before those in the list. If this resource remains online or fails to stop, a resource in the list stops anyway. You cannot disable resources in the list unless this resource is disabled first.

Within a resource group, Prenet_start methods are run in dependency order before Start methods. Postnet_stop methods are run in dependency order after Stop methods. In different resource groups, the dependent waits for the depended-on resource to finish Prenet_start and Start before it runs Prenet_start. The depended-on resource waits for the dependent to finish Stop and Postnet_stop before it runs Stop.

Category: Optional

Default: The empty list

Tunable: ANYTIME

Resource_name (string)

The name of the resource instance. This name must be unique within the cluster configuration and cannot be changed after a resource has been created.

Category: Required

Default: No default

Tunable: Never

Resource_project_name (string)

The Solaris project name that is associated with the resource. Use this property to apply Solaris resource management features such as CPU shares and resource pools to cluster data services. When the RGM brings resources online, it launches the related processes under this project name. If this property is not specified, the project name is taken from the RG_project_name property of the resource group that contains the resource (see rg_properties (5)). If neither property is specified, the RGM uses the predefined project name default. The specified project name must exist in the projects database and the user root must be configured as a member of the named project. This property is supported in Solaris 9 and later versions of Solaris only.

Note – Changes to this property take effect the next time the resource is started.

Category: Optional

Default: Null

Tunable: ANYTIME

Resource_state on each cluster node (enum)

The RGM-determined state of the resource on each cluster node. Possible states are Online, Offline, Start_failed, Stop_failed, Monitor_failed, Online_not_monitored, Starting, and Stopping.

You cannot configure this property.

Category: Query-only

Default: No default

Tunable: Never

Retry_count (integer)

The number of times a monitor attempts to restart a resource if it fails. This property is created by the RGM and available to the administrator only if it is declared in the RTR file. Retry_count is optional if a default value is specified in the RTR file.

If the Tunable attribute is not specified in the resource type file, the Tunable value for the property is WHEN_DISABLED.

This property is required if it is declared in the RTR file and the Default attribute is not specified.

Category: Conditional

Default: No default

Tunable: WHEN_DISABLED

Retry_interval (integer)

The number of seconds over which to count attempts to restart a failed resource. The resource monitor uses this property in conjunction with Retry_count. This property is created by the RGM and available to the administrator only if it is declared in the RTR file. Retry_interval is optional if a default value is specified in the RTR file.

If the Tunable attribute is not specified in the resource type file, the Tunable value for the property is WHEN_DISABLED.

This property is required if it is declared in the RTR file and the Default attribute is not specified.

Category: Conditional

Default: No default

Tunable: WHEN_DISABLED

Scalable (boolean)

Indicates whether the resource is scalable, that is, whether the resource uses the networking load-balancing features of Sun Cluster.

If this property is declared in the RTR file, the RGM automatically creates the following scalable service properties for resources of that type: Affinity_timeout, Load_balancing_policy, Load_balancing_weights, Network_resources_used, Port_list, UDP_affinity, and Weak_affinity. These properties have their default values unless they are explicitly declared in the RTR file. The default for Scalable, when it is declared in the RTR file, is TRUE.

If this property is declared in the RTR file, it is not permitted to be assigned a Tunable attribute other than AT_CREATION.

If this property is not declared in the RTR file, the resource is not scalable, you cannot tune this property, and no scalable service properties are set by the RGM. However, you can explicitly declare the Network_resources_used and Port_list properties in the RTR file, because these properties can be useful in a nonscalable service as well as in a scalable service.

You use this resource property in combination with the Failover resource type property, as follows:

Using this resource property in combination with the Failover resource type property is described in more detail in r_properties(5).

Category: Optional

Default: No default

Tunable: AT_CREATION

Status on each cluster node (enum)

Set by the resource monitor with scha_resource_setstatus(1HA) or scha_resource_setstatus(3HA). Possible values are OK, degraded, faulted, unknown, and offline. When a resource is brought online or offline, the RGM automatically sets the Status value if the Status value is not set by the resource's monitor or methods.

Category: Query-only

Default: No default

Tunable: Never

Status_msg on each cluster node (string)

Set by the resource monitor at the same time as the Status property. When a resource is brought online or offline, the RGM automatically resets this property to the empty string if this property is not set by the resource's methods.

Category: Query-only

Default: No default

Tunable: Never

Thorough_probe_interval (integer)

The number of seconds between invocations of a high-overhead fault probe of the resource. This property is created by the RGM and available to the administrator only if it is declared in the RTR file. Thorough_probe_interval is optional if a default value is specified in the RTR file.

If the Tunable attribute is not specified in the resource type file, the Tunable value for the property is WHEN_DISABLED.

This property is required if the Default attribute is not specified in the property declaration in the RTR file.

Category: Conditional

Default: No default

Tunable: WHEN_DISABLED

Type (string)
The resource type of which this resource is an instance.

Category: Required

Default: No default

Tunable: Never

Type_version (string)
Specifies which version of the resource type is currently associated with this resource. The RGM automatically creates this property, which cannot be declared in the RTR file. The value of this property is equal to the RT_version property of the resource's type. When a resource is created, the Type_version property is not specified explicitly, though it might appear as a suffix of the resource type name. When a resource is edited, the Type_version can be changed to a new value.

The tunability of this property is derived from the following sources:

- The current version of the resource type
- The #$upgrade_from directive in the RTR file

Category: See description

Default: No default

Tunable: See description

UDP_affinity (boolean)
If true, all UDP traffic from a given client is sent to the same server node that currently handles all TCP traffic for the client.

This property is relevant only when Load_balancing_policy is either Lb_sticky or Lb_sticky_wild. In addition, Weak_affinity must be set to FALSE (the default value).

This property is only used for scalable services.

Category: Optional

Default: No default

Tunable: WHEN_DISABLED

Weak_affinity (boolean)
If true, enable the weak form of the client affinity. The weak form of the client affinity allows connections from a given client to be sent to the same server node except when the following conditions occur:

- A server listener starts, in response to, for example, a fault monitor's restarting, a resource's failing over or switching over, or a node's rejoining a cluster after failing
- Load_balancing_weights for the scalable resource changes because of an administrative action

Weak affinity provides a low-overhead alternative to the default form, both in terms of memory consumption and processor cycles.

This property is relevant only when Load_balancing_policy is either Lb_sticky or Lb_sticky_wild.

This property is only used for scalable services.

Category: Optional

Default: No default

Tunable: WHEN_DISABLED

Resource Group Properties

The following information describes the resource group properties that are defined by Sun Cluster. The property values are categorized as follows (after Category):

- **Required** – The administrator must specify a value when creating a resource group with an administrative utility.
- **Optional** – If the administrator does not specify a value when creating a resource group, the system supplies a default value.
- **Query-only** – Cannot be set directly by an administrative tool.

Each description states whether the property can be updated (Yes) or not (No) after it is initially set.

Property names are shown first, followed by a description.

Auto_start_on_new_cluster (boolean)

This property disallows automatic startup of the Resource Group when a new cluster is forming.

If set to TRUE, the Resource Group Manager attempts to start the resource group automatically to achieve Desired_primaries when the cluster is rebooted. If set to FALSE, the Resource Group does not start automatically when the cluster reboots.

Category: Optional

Default: `TRUE`

Tunable: Yes

`Desired_primaries` (`integer`)

The number of nodes where the group is desired to be online at once.

If the `RG_mode` property is `Failover`, the value of this property must be no greater than 1. If the `RG_mode` property is `Scalable`, a value greater than 1 is allowed.

Category: Optional

Default: `1`

Tunable: Yes

`Failback` (`boolean`)

A Boolean value that indicates whether to recalculate the set of nodes where the group is online when the cluster membership changes. A recalculation can cause the RGM to bring the group offline on less preferred nodes and online on more preferred nodes.

Category: Optional

Default: `FALSE`

Tunable: Yes

`Global_resources_used` (`string_array`)

Indicates whether cluster file systems are used by any resource in this resource group. Legal values that the administrator can specify are an asterisk (*) to indicate all global resources, and the empty string (“”) to indicate no global resources.

Category: Optional

Default: All global resources

Tunable: Yes

`Implicit_network_dependencies` (`boolean`)

A Boolean value that indicates, when `TRUE`, that the RGM should enforce implicit strong dependencies of non-network-address resources on network-address resources within the group. Network-address resources include the logical host name and shared address resource types.

In a scalable resource group, this property has no effect because a scalable resource group does not contain any network-address resources.

Category: Optional

Default: `TRUE`

Tunable: Yes

`Maximum_primaries` (`integer`)

The maximum number of nodes where the group might be online at once.

If the RG_mode property is Failover, the value of this property must be no greater than 1. If the RG_mode property is Scalable, a value greater than 1 is allowed.

Category: Optional

Default: 1

Tunable: Yes

Nodelist (string_array)

A list of cluster nodes where the group can be brought online in order of preference. These nodes are known as the potential primaries or masters of the resource group.

Category: Optional

Default: The list of all cluster nodes

Tunable: Yes

Pathprefix (string)

A directory in the cluster file system in which resources in the group can write essential administrative files. Some resources might require this property. Make Pathprefix unique for each resource group.

Category: Optional

Default: The empty string

Tunable: Yes

Pingpong_interval (integer)

A nonnegative integer value (in seconds) that is used by the RGM to determine where to bring the resource group online. Conditions under which this property might be needed are as follows:

- If a reconfiguration occurs
- A scha_control -O GIVEOVER command or a scha_control() function with the SCHA_GIVEOVER argument is executed If a reconfiguration occurs, if

the resource group fails to come online more than once within the past Pingpong_interval seconds on a particular node, that node is considered ineligible to host the resource group and the RGM looks for another master. The resource group fails to come online because the resource's Start or Prenet_start method exited nonzero or timed out.

If a call to a resource's scha_control command or function causes the resource group to be brought offline on a particular node within the past Pingpong_interval seconds, that node is ineligible to host the resource group as the result of a subsequent call to scha_control() that originates from another node.

Category: Optional

Default: 3600 (one hour)

Tunable: Yes

`Resource_list` (`string_array`)

The list of resources that are contained in the group. The administrator does not set this property directly. Rather, the RGM updates this property as the administrator adds or removes resources from the resource group.

Category:	Query-only
Default:	No default
Tunable:	No

`RG_affinities` (`string`)

The RGM is to try to locate a resource group on a node that is a current master of another given resource group (positive affinity), or to locate a resource group on a node that is not a current master of a given resource group (negative affinity).

You can set `RG_affinities` to the following strings:

- ++, or strong positive affinity
- +, or weak positive affinity
- -, or weak negative affinity
- --, or strong negative affinity
- +++, or strong positive affinity with failover delegation For example, `RG_affinities=+RG2,--RG3` indicates that this resource group has a weak positive affinity for RG2 and a strong negative affinity for RG3.

Using `RG_affinities` is described in "Administering Data Service Resources" in *Sun Cluster Data Services Planning and Administration Guide for Solaris OS*.

Category:	Optional
Default:	The empty string
Tunable:	Yes

`RG_dependencies` (`string_array`)

Optional list of resource groups that indicate a preferred ordering for bringing other groups online or offline on the same node. The graph of all strong `RG_affinities` (positive and negative) together with `RG_dependencies` is not allowed to contain cycles.

For example, suppose that resource group RG2 is listed in the `RG_dependencies` list of resource group RG1. In other words, suppose that RG1 has a resource group dependency on RG2. The following list summarizes the effects of this resource group dependency:

- When a node joins the cluster, `Boot` methods on that node are not run on resources in RG1 until all `Boot` methods on that node have completed on resources in RG2.
- If RG1 and RG2 are both in the `Pending_online` state on the same node at the same time, the starting methods (`Prenet_start` or `Start`) are not run on any resources in RG1 until all the resources in RG2 have completed their starting methods.

- If RG1 and RG2 are both in the Pending_offline state on the same node at the same time, the stopping methods (Stop or Postnet_stop) are not run on any resources in RG2 until all the resources in RG1 have completed their stopping methods.
- An attempt to switch the primaries of RG1 or RG2 fails if switching the primaries would leave RG1 online on any node and RG2 offline on all nodes. scswitch(1M) and scsetup(1M) contain more information.
- Setting the Desired_primaries property to a value that is greater than zero on RG1 is not permitted if Desired_primaries is set to zero on RG2.
- Setting the Auto_start_on_new_cluster property to TRUE on RG1 is not permitted if Auto_start_on_new_cluster is set to FALSE on RG2.

Category: Optional

Default: The empty list

Tunable: Yes

RG_description (string)
A brief description of the resource group.

Category: Optional

Default: The empty string

Tunable: Yes

RG_is_frozen (boolean)
A Boolean value that indicates whether a global device on which a resource group depends is being switched over. If this property is set to TRUE, the global device is being switched over. If this property is set to FALSE, no global device is being switched over. A resource group depends on global devices as indicated by its Global_resources_used property.

You do not set the RG_is_frozen property directly. The RGM updates the RG_is_frozen property when the status of the global devices changes.

Category: Optional

Default: No default

Tunable: No

RG_mode (enum)
Indicates whether the resource group is a failover or scalable group. If the value is Failover, the RGM sets the Maximum_primaries property of the group to 1 and restricts the resource group to being mastered by a single node.

If the value of this property is Scalable, the RGM allows the Maximum_primaries property to have a value greater than 1. As a result, the group can be mastered by multiple nodes simultaneously. The RGM does not allow a resource whose Failover property is TRUE to be added to a resource group whose RG_mode is Scalable.

If `Maximum_primaries` is 1, the default is `Failover`. If `Maximum_primaries` is greater than 1, the default is `Scalable`.

Category: Optional

Default: Depends on the value of `Maximum_primaries`

Tunable: No

`RG_name` (`string`)

The name of the resource group. This name must be unique within the cluster.

Category: Required

Default: No default

Tunable: No

`RG_project_name` (`string`)

The Solaris project name that is associated with the resource group. Use this property to apply Solaris resource management features such as CPU shares and resource pools to cluster data services. When the RGM brings resource groups online, it launches the related processes under this project name for resources that do not have the `Resource_project_name` property set. The specified project name must exist in the projects database and the user `root` must be configured as a member of the named project.

This property is supported in Solaris 9 and later versions of Solaris only.

Note – Changes to this property take effect the next time the resource is started.

Category: Optional

Default: The text string "`default`"

Tunable: `ANYTIME`

`RG_state` on each cluster node (`enum`)

Set by the RGM to `Unmanaged`, `Online`, `Offline`, `Pending_online`, `Pending_offline`, `Pending_online_blocked`, `Error_stop_failed`, `Online_faulted`, or `Pending_online_blocked` to describe the state of the group on each cluster node.

You cannot configure this property. However, you can indirectly set this property by invoking `scswitch`(1M), or by using the equivalent `scsetup`(1M) or SunPlex Manager commands.

Category: Query-only

Default: No default

Tunable: No

RG_system (boolean)

If the RG_system property is TRUE for a resource group, then particular operations are restricted for the resource group and for the resources that the resource group contains. This restriction is intended to help prevent accidental modification or deletion of critical resource groups and resources. Only scrgadm(1M) and scswitch(1M) commands are affected by this property. Operations for scha_control(1HA) and scha_control(3HA) are not affected.

Before performing a restricted operation on a resource group (or a resource group's resources), you must first set the RG_system property of the resource group to FALSE. Use care when you modify or delete a resource group that supports cluster services, or when you modify or delete the resources that such a resource group contains.

A resource group whose RG_system value is set to TRUE is called a system resource group. Editing the RG_system property itself is never restricted, regardless of the current value of RG_system. The rg_properties(5) man page describes these restrictions in more detail.

Category:	Optional
Default:	FALSE
Tunable:	Yes

Resource Property Attributes

The following information describes the resource property attributes that can be used to change system-defined properties or create extension properties.

Caution – You cannot specify Null or the empty string ("") as the default value for boolean, enum, or int types.

Property names are shown first, followed by a description.

Array_maxsize

For stringarray type, the maximum number of array elements permitted.

Array_minsize

For stringarray type, the minimum number of array elements permitted.

Default

Indicates a default value for the property.

`Description`
A string annotation that is intended to be a brief description of the property. The `Description` attribute cannot be set in the RTR file for system-defined properties.

`Enumlist`
For an `enum` type, a set of string values that are permitted for the property.

`Extension`
If used, indicates that the RTR file entry declares an extension property that is defined by the resource type implementation. Otherwise, the entry is a system-defined property.

`Max`
For an `int` type, the maximum value permitted for the property.

`Maxlength`
For `string` and `stringarray` types, the maximum string length permitted.

`Min`
For an `int` type, the minimal value permitted for the property.

`Minlength`
For `string` and `stringarray` types, the minimum string length permitted.

`Property`
The name of the resource property.

`Tunable`
Indicates when the cluster administrator can set the value of this property in a resource. Can be set to `NONE` or `FALSE` to prevent the administrator from setting the property. Values that allow administrator tuning are `TRUE` or `ANYTIME` (at any time), `AT_CREATION` (only when the resource is created), or `WHEN_DISABLED` (when the resource is offline). To establish other tunability conditions, such as "when monitoring is disabled" or "when offline", set this attribute to `ANYTIME` and validate the state of the resource in the `Validate` method.

The default differs for each standard resource property, as shown in the following section. The default setting for tuning an extension property, if not otherwise specified in the RTR file, is `TRUE` (`ANYTIME`).

Type of the property
Allowable types are: `string`, `boolean`, `int`, `enum`, and `stringarray`. You cannot set the type attribute in an RTR file entry for system-defined properties. The type determines acceptable property values and the type-specific attributes that are allowed in the RTR file entry. An `enum` type is a set of string values.

APPENDIX B

Legal RGM Names and Values

This appendix lists the requirements for legal characters for Resource Group Manager (RGM) names and values.

RGM Legal Names

RGM names fall into the following categories.

- Resource group names
- Resource type names
- Resource names
- Property names
- Enumeration literal names

Rules for Names Except Resource Type Names

Except for resource type names, all names must comply with the following rules.

- Must be in ASCII
- Must start with a letter
- Can contain uppercase and lowercase letters, digits, dashes (-), and underscores (_)
- Must not exceed 255 characters

Format of Resource Type Names

The format of the complete name of a resource type depends on the resource type as follows:

- If the resource type's resource type registration (RTR) file contains the `#$upgrade` directive, the format is as follows:

  vendor-id.*base-rt-name*:*version*

- If the resource type's RTR file does *not* contain the `#$upgrade` directive, the format is as follows:

  vendor-id.*base-rt-name*

A period separates *vendor-id* and *base-rt-name*. A colon separates *base-rt-name* and *version*.

The variable items in this format are as follows:

vendor-id Specifies the vendor ID prefix. The vendor ID prefix is the value of the `Vendor_id` resource type property in the RTR file.

base-rt-name Specifies the base resource type name. The base resource type name is the value of the `Resource_type` resource type property in the RTR file.

version Specifies the version suffix. The version suffix is the value of the `RT_version` resource type property in the RTR file. The version suffix is *only* part of the complete resource type name if the RTR file contains the `#$upgrade` directive. The `#$upgrade` directive was introduced in release 3.1 of Sun Cluster.

Note – If only one version of a base resource type name is registered, you do not have to use the complete name in `scrgadm`(1M) commands. You can omit the vendor ID prefix, the version number suffix, or both the prefix and the suffix.

For more information about resource type properties, see "Resource Type Properties" on page 117.

EXAMPLE B–1 Complete Name of a Resource Type With the `#$upgrade` Directive

This example shows the complete name of a resource type for which properties in the RTR file are set as follows:

- `Vendor_id=SUNW`
- `Resource_type=sample`
- `RT_version=2.0`

The complete name of the resource type that is defined by this RTR file is as follows:

EXAMPLE B–1 Complete Name of a Resource Type With the `#$upgrade` Directive *(Continued)*

```
SUNW.sample:2.0
```

EXAMPLE B–2 Complete Name of a Resource Type Without the `#$upgrade` Directive

This example shows the complete name of a resource type for which properties in the RTR file are set as follows:

- `Vendor_id=SUNW`
- `Resource_type=nfs`

The complete name of the resource type that is defined by this RTR file is as follows:

```
SUNW.nfs
```

RGM Values

RGM values fall into two categories: property values and description values. Both categories share the same rules, as follows.

- Values must be in ASCII.
- The maximum length of a value is 4 megabytes minus 1, that is, 4,194,303 bytes.
- Values cannot contain any of the following characters:
  - Null
  - Newline
  - Comma
  - Semicolon

APPENDIX C

Data Service Configuration Worksheets and Examples

This appendix provides worksheets for planning resource-related components of your cluster configuration and examples of completed worksheets for your reference. See "Sun Cluster Installation and Configuration Worksheets" in *Sun Cluster Software Installation Guide for Solaris OS* for worksheets for other components of your cluster configuration.

Configuration Worksheets

If necessary, make additional copies of a worksheet to accommodate all the resource-related components in your cluster configuration. Follow planning guidelines in the *Sun Cluster Software Installation Guide for Solaris OS* and Chapter 1 to complete these worksheets. Then refer to your completed worksheets during cluster installation and configuration.

Note – The data used in the worksheet examples is intended as a guide only. The examples do not represent a complete configuration of a functional cluster.

Resource Types Worksheet

Use this worksheet for resource types other than logical host or shared address.

TABLE C–1 Resource Types Worksheet

Resource Type Name	Nodes on Which the Resource Type Runs

Example: Resource Types Worksheet

TABLE C–2 Example: Resource Types Worksheet

Resource Type Name	Nodes on Which the Resource Type Runs
`SUNW.nshttp`	`phys-schost-1, phys-schost-2`
`SUNW.oracle_listener`	`phys-schost-1, phys-schost-2`
`SUNW.oracle_server`	`phys-schost-1, phys-schost-2`

Network Resources Worksheet

TABLE C–3 Network Resources Worksheet

Component	Name	
Resource name		
Resource group name		
Resource type (circle one)	Logical hostname | Shared address	
Resource type name		
Dependencies		
Hostnames used		
Extension properties	**Name**	**Value**

Example: Network Resources—Shared Address Worksheet

TABLE C–4 Example: Network Resources—Shared Address Worksheet

Component	Name	
Resource name	`sh-galileo`	
Resource group name	`rg-shared`	
Resource type (circle one)	`Shared address`	
Resource type name	`SUNW.SharedAddress`	
Dependencies	`none`	
Hostnames used	`sh-galileo`	
Extension properties	**Name**	**Value**
	`netiflist`	`ipmp0@1, ipmp0@2`

Example: Network Resources—Logical Hostname Worksheet

TABLE C–5 Example: Network Resources—Logical Hostname Worksheet

Component	Name	
Resource name	`relo-galileo`	
Resource group name	`rg-oracle`	
Resource type (circle one)	`Logical hostname`	
Resource type name	`SUNW.LogicalHostname`	
Dependencies	`none`	
Hostnames used	`relo-galileo`	
Extension properties	**Name**	**Value**
	`netiflist`	`ipmp0@1, ipmp0@2`

Application Resources—Failover Worksheet

TABLE C–6 Application Resources—Failover Worksheet

Component	Name	
Resource name		
Resource group name		
Resource type name		
Dependencies		
Extension Properties	**Name**	**Value**

Example: Application Resources—Failover Worksheet

TABLE C–7 Example: Application Resources—Failover Worksheet

Component	Name	
Resource name	`oracle-listener`	
Resource group name	`rg-oracle`	
Resource type name	`SUNW.oracle_listener`	
Dependencies	`hasp_resource`	
Extension Properties	**Name**	**Value**
	`ORACLE_HOME`	`/global/oracle/orahome/`
	`LISTENER_NAME`	`lsnr1`

Application Resources—Scalable Worksheet

TABLE C–8 Application Resources—Scalable Worksheet

Component	Name	
Resource name		
Logical-host resource group name		
Shared-address resource group name		
Logical-host resource type name		
Shared-address resource type name		
Dependencies		
Extension Properties	**Name**	**Value**

Example: Application Resources—Scalable Worksheet

TABLE C–9 Example: Application Resources—Scalable Worksheet

Component	Name	
Resource name	**sh-galileo**	
Logical-host resource group name		
Shared-address resource group name	**rg-shared**	
Logical-host resource type name		
Shared-address resource type name		
Dependencies		
Extension Properties	**Name**	**Value**

Resource Groups—Failover Worksheet

TABLE C–10 Resource Groups—Failover Worksheet

Component	Notes	Name
Resource group name	The name must be unique within the cluster	
Function	Describe the function of this resource group	
Failback? (circle one)	Will this resource group switch back to the primary node after the primary node has failed and been restored?	Yes | No
Node list	Indicate the cluster nodes that may host this resource group. The first node in this list should be the primary, with others being the secondaries. The order of the secondaries will indicate preference for becoming primaries.	
Dependent disk device groups	List the disk device groups upon which this resource group depends.	
Configuration directory	If the resources in this resource group need to create files for administrative purposes, include the subdirectory they use.	

Example: Resource Groups—Failover Worksheet

TABLE C–11 Example: Resource Groups—Failover Worksheet

Component	Notes	Name
Resource group name	The name must be unique within the cluster	**`rg-oracle`**
Function	Describe the function of this resource group	**`Contains the Oracle resources`**
Failback? (circle one)	Will this resource group switch back to the primary node after the primary node has failed and been restored?	**`No`**

Node list	Indicate the cluster nodes that may host this resource group. The first node in this list should be the primary, with others being the secondaries. The order of the secondaries will indicate preference for becoming primaries.	`1)` **`phys-schost-1`**
		`2)` **`phys-schost-2`**

Dependent disk device groups	List the disk device groups upon which this resource group depends.	**`schost1-dg`**
Configuration directory	If the resources in this resource group need to create files for administrative purposes, include the subdirectory they use.	

Resource Groups—Scalable Worksheet

TABLE C–12 Resource Groups—Scalable Worksheet

Component	Notes	Name
Resource group name	The name must be unique within the cluster.	
Function		
Maximum number of primaries		
Desired number of primaries		
Failback? (circle one)	Will this resource group switch back to the primary node, after the primary node has failed?	Yes | No
Node list	Indicate the cluster nodes that may host this resource group. The first node in this list should be the primary, with others being the secondaries. The order of the secondaries will indicate preference for becoming primaries.	
Dependencies	List any resource groups this resource depends on.	

Example: Resource Groups—Scalable Worksheet

TABLE C–13 Example: Resource Groups—Scalable Worksheet

Component	Notes	Name
Resource group name	The name must be unique within the cluster.	**`rg-http`**
Function		**`Contains the web server resources`**
Maximum number of primaries		**`2`**
Desired number of primaries		**`2`**
Failback? (circle one)	Will this resource group switch back to the primary node, after the primary node has failed?	**`No`**

Node list	Indicate the cluster nodes that may host this resource group. The first node in this list should be the primary, with others being the secondaries. The order of the secondaries will indicate preference for becoming primaries.	**`1) phys-schost-1`**
		`2) phys-schost-2`

Dependencies	List any resource groups this resource depends on.	**`rg-shared`**

Index