# 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: 819–0703–10
August 2005, Revision A


050509@11223

# 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.<br>Use `ls -a` to list all files.<br>`machine_name% you have mail.` |
| **`AaBbCc123`** | What you type, contrasted with onscreen computer output | `machine_name%` **`su`**<br>`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*.<br>Perform a *patch analysis*.<br>Do *not* save the file.<br>[Note that some emphasized items appear 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 3.0-3.1 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.

# Documentation, Support, and Training

| Sun Function | URL | Description |
|---|---|---|
| Documentation | `http://www.sun.com/documentation/` | Download PDF and HTML documents, and order printed documents |
| Support and Training | `http://www.sun.com/supportraining/` | Obtain technical support, download patches, and learn about Sun courses |

# Getting 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.



For information about data services, resource types, resources, and resource groups, see *Sun Cluster Concepts Guide for Solaris OS*.

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 a Sun Cluster data service is not provided for your application, consider developing a custom data service for the application. To develop a custom data service, use the Sun Cluster data services APIs. For more information, see *Sun Cluster Data Services Developer's Guide for Solaris OS*.

---

**Note –** Sun Cluster does not provide a data service for the `sendmail`(1M) subsystem. The `sendmail` subsystem can run on the individual cluster nodes, but the `sendmail` functionality is not highly available. This restriction applies to all the `sendmail` functionality, including the functionality of mail delivery and mail routing, queuing, and 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.

---

**Note –** Some Sun Cluster data services are not supported for use in x86 based clusters. For more information, see the release notes for your release of Sun Cluster at `http://docs.sun.com`.

---

## 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 short period of downtime for upgrades, place a single copy of the application and configuration files on the cluster file system.

For information about how to create cluster file systems, see "Planning the Global Devices and Cluster File Systems" in *Sun Cluster Software Installation Guide for Solaris OS*.

- **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 that enable Sun Cluster to fail over local file systems. To fail over, the local file system must reside on global disk groups with affinity switchovers enabled.

  For information about how to use the `HAStoragePlus` resource type, see "Enabling Highly Available Local File Systems" on page 91.

## Verifying the `nsswitch.conf` File Contents

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

- The databases within the Solaris environment to use for name-service lookups
- The order in which the databases are to be consulted

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.

For additional information about how to configure the `nsswitch.conf` file for the Sun Cluster environment, see "Planning the Sun Cluster Environment" in *Sun Cluster Software Installation Guide for Solaris OS*.

## 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. To determine whether any special considerations apply, see the documentation for the data service that you plan to configure.

For information about how to create cluster file systems, see "Planning the Global Devices and Cluster File Systems" in *Sun Cluster Software Installation Guide for Solaris OS*.

The resource type `HAStoragePlus` enables you to use a highly available local file system in a Sun Cluster environment that is configured for failover. For information about setting up the `HAStoragePlus` resource type, see "Enabling Highly Available Local File Systems" on page 91.

## Enabling Solaris SMF Services to Run Under the Control of Sun Cluster

You might require Sun Cluster to make highly available an application other than NFS or DNS that is integrated with the Solaris Service Management Facility (SMF). To ensure that Sun Cluster can restart or fail over the application correctly after a failure, you must disable SMF service instances for the application as follows:

- For any application other than NFS or DNS, disable the SMF service instance on all potential primary nodes for the Sun Cluster resource that represents the application.
- If multiple instances of the application share any component that you require Sun Cluster to monitor, disable all service instances of the application. Examples of such components are daemons, file systems, and devices.

---

**Note –** If you do not disable the SMF service instances of the application, both the Solaris SMF and Sun Cluster might attempt to control the startup and shutdown of the application. As a result, the behavior of the application might become unpredictable.

---

For more information, see the following documentation:

- "How to Disable a Service Instance" in *System Administration Guide: Basic Administration*
- *Sun Cluster Data Service for NFS Guide for Solaris OS*
- *Sun Cluster Data Service for DNS Guide for Solaris OS*

# 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 the behavior of Sun Cluster in response to the following set of conditions:

- A node that has failed and left the cluster rejoins the cluster.
- The node that is rejoining the cluster appears earlier in the node list than the current primary node.

If failback is set to `True`, the device group or resource group is switched off the current primary and switched onto the rejoining node, making the rejoining node the new primary.

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.

For more information about the relationship between disk device groups and resource groups, see "Disk Device Groups" in *Sun Cluster Overview for Solaris OS*.

For information about how to set up disk device groups, see the following documentation:

- "Disk Device Groups" in *Sun Cluster Software Installation Guide for Solaris OS*
- "Example of How to Configure Device Groups and Resource Groups" in *Sun Cluster Software Installation Guide for Solaris OS*

# 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. Other resources in the resource group that contains the `HAStorage` or `HAStoragePlus` resource are brought online *only* after the disk device resources become available.
- With `AffinityOn` set to `True`, enforce colocation of resource groups and disk device groups on the same node. This enforced colocation enhances the performance of disk-intensive data services.

In addition, `HAStoragePlus` is capable of mounting any global file system that is in an unmounted state. For more information, see "Planning the Cluster File System Configuration" on page 17.

**Note –** If the device group is switched to another node while the `HAStorage` or `HAStoragePlus` resource is online, `AffinityOn` has no effect. The resource group does **not** migrate with the device group. However, if the resource group is switched to another node, the setting of `AffinityOn` 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 85 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 91 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`

The following types of data services require `HAStorage` or `HAStoragePlus`:

- Data services with nodes that are not directly connected to storage
- Data services that are disk intensive

### Data Services With Nodes That Are Not Directly Connected to Storage

Some nodes in the node list of a data service's resource group might not be directly connected to the storage. In this situation, you must coordinate the boot order between the storage and the data service. To meet this requirement, configure the resource group as follows:

- Configure `HAStorage` or `HAStoragePlus` resources in the resource group.
- Set the dependency of the other data service resources to the `HAStorage` or `HAStoragePlus` resource.

### Data Services That Are Disk Intensive

Some data services, such as Sun Cluster HA for Oracle and Sun Cluster HA for NFS are disk intensive. If your data service is disk intensive, ensure that the resource groups and disk device groups are colocated on the same node. To meet this requirement, perform the following tasks.

- Adding an `HAStorage` or `HAStoragePlus` resource to your data service resource group
- Switching the `HAStorage` or `HAStoragePlus` resource online
- Setting the dependency of your data service resources to the `HAStorage` resource or `HAStoragePlus` resource
- Setting `AffinityOn` to `True`

---

**Note –** The failback settings must be identical for both the resource group and device groups.

---

Some data services are not disk intensive. For example, Sun Cluster HA for DNS, which reads all of its files at startup, is not disk intensive. If your data service is **not** disk intensive, configuring the `HAStorage` resource type or `HAStoragePlus` resource type is optional.

## Choosing Between `HAStorage` and `HAStoragePlus`

To determine whether to create `HAStorage`resources or `HAStoragePlus` resources in 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.
- Use `HAStorage` if you are using Sun Cluster 3.0 12/01 or earlier.

---

**Note –** To integrate any file system locally into a Sun Cluster environment that is configured for failover, you use the `HAStoragePlus` resource type. For more information, see "Planning the Cluster File System Configuration" on page 17. If you are using Sun Cluster 3.0 12/01 or earlier, you must first upgrade to Sun Cluster 3.0 5/02 or Sun Cluster 3.1.

---

# Considerations for Installing and Configuring a Data Service

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.

- Retries within the I/O subsystem during disk failures might cause applications whose data services are disk intensive to experience delays. Disk-intensive data services are I/O intensive and have a large number of disks configured in the cluster. An I/O subsystem might require several minutes to retry and recover from a disk failure. This delay can cause Sun Cluster to fail over the application to another node, even though the disk might 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 node list properties when configuring data services.

- `installed_nodes` property
- `nodelist` property
- `auxnodelist` property

## `installed_nodes` Property

The `installed_nodes` property is 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.

## `nodelist` Property

The `nodelist` property is a property of a resource group. This property 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 identify 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 identify 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.

## `auxnodelist` Property

The `auxnodelist` property is 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 that are identified in the node list of the resource group. This list pertains to scalable services only. For details, see the `scrgadm`(1M) man page.

# 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 Cluster CD-ROM
  - Sun Cluster Agents CD-ROM
- 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) is to manage. The documentation for the individual data services describes these procedures.

Before you install and configure data services, see *Sun Cluster Software Installation Guide for Solaris OS*, which includes instructions for the following tasks:

- Installing the data service software packages
- Configuring 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. See the SunPlex Manager online help for more information.

---

## Installation and Configuration Task Flow

The following table summarizes the tasks for installing and configuring Sun Cluster data services. The table also provides cross-references to detailed instructions for performing the tasks.

**TABLE 1–1** Tasks for Installing and Configuring Sun Cluster Data Services

| Task | Instructions |
|---|---|
| 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<br>"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 |
| Register and configure the data service | The appropriate Sun Cluster data services book |

# Example of Configuring a Failover Data Service

This example summarizes how to set up the resource types, resources, and resource groups that a failover data service for the Oracle application requires. For complete instructions for configuring the data service for the Oracle application, see *Sun Cluster Data Service for Oracle Guide for Solaris OS*.

The principal difference between this example and an example of a scalable data service is as follows: In addition to the failover resource group that contains the network resources, a scalable data service requires a separate resource group (*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 to run 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.

- Installing a cluster
- Administering a cluster
- Creating and configuring resources and resource groups
- Configuring data services with the Sun Cluster software

For instructions for using use SunPlex Manager to install cluster software, see *Sun Cluster Software Installation Guide for Solaris OS*. 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 for how to register and configure your data service in the book for the data service. If, for example, you are using Sun Cluster HA for Oracle, see "Registering and Configuring Sun Cluster HA for Oracle" in *Sun Cluster Data Service for Oracle Guide for Solaris OS*.

For more information about how to use the `scrgadm` command to administer data service resources, see the following documentation:

- Chapter 2
- The `scrgadm`(1M) man page

## Summary by Task of Tools for Administering Data Service Resources

The following table summarizes by task which tool you can use in addition to the command line for administering data service resources. For more information about these tasks and for details about how to use the command line to complete related procedures, see Chapter 2.

**TABLE 1–2** Tools for Administering Data Service Resources

| Task | SunPlex Manager | SPARC: Sun Management Center | `scsetup` Utility |
|---|---|---|---|
| Register a resource type | Yes | No | Yes |
| Create a resource group | Yes | No | Yes |

**TABLE 1–2** Tools for Administering Data Service Resources *(Continued)*

| Task | SunPlex Manager | SPARC: Sun Management Center | `scsetup` Utility |
|---|---|---|---|
| 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. To determine if you can use other tools to complete a procedure, see "Tools for Data Service Resource Administration" on page 25.

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

This chapter contains the following sections.

# Overview of Tasks for Administering Data Service Resources

The following table summarizes the tasks for installing and configuring Sun Cluster data services. The table also provides cross-references to detailed instructions for performing the tasks.

**TABLE 2–1** Tasks for Administering Data Service Resources

| Task | Instructions |
|---|---|
| Register a resource type | "How to Register a Resource Type" on page 33 |
| Upgrade a resource type | "How to Migrate Existing Resources to a New Version of the Resource Type" on page 36<br>"How to Install and Register an Upgrade of a Resource Type" on page 35 |
| Create failover or scalable resource groups | "How to Create a Failover Resource Group" on page 43<br>"How to Create a Scalable Resource Group" on page 44 |
| 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 46<br>"How to Add a Shared Address Resource to a Resource Group" on page 48<br>"How to Add a Failover Application Resource to a Resource Group" on page 50<br>"How to Add a Scalable Application Resource to a Resource Group" on page 52 |

**TABLE 2–1** Tasks for Administering Data Service Resources *(Continued)*

| Task | Instructions |
|---|---|
| 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 54 |
| Disable and enable resource monitors independent of the resource | "How to Disable a Resource Fault Monitor" on page 56<br>"How to Enable a Resource Fault Monitor" on page 57 |
| Remove resource types from the cluster | "How to Remove a Resource Type" on page 58 |
| Remove resource groups from the cluster | "How to Remove a Resource Group" on page 59 |
| Remove resources from resource groups | "How to Remove a Resource" on page 60 |
| Switch the primary for a resource group | "How to Switch the Current Primary of a Resource Group" on page 61 |
| 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 64 |
| Display resource type, resource group, and resource configuration information | "Displaying Resource Type, Resource Group, and Resource Configuration Information" on page 66 |
| Change resource type, resource group, and resource properties | "How to Change Resource Type Properties" on page 67<br>"How to Change Resource Group Properties" on page 69<br>"How to Change Resource Properties" on page 69 |
| Clear error flags for failed Resource Group Manager (RGM) processes | "How to Clear the `STOP_FAILED` Error Flag on Resources" on page 72 |
| Reregister the built-in resource types `LogicalHostname` and `SharedAddress` | "How to Reregister Preregistered Resource Types After Inadvertent Deletion" on page 76 |
| Upgrade the built-in resource types `LogicalHostname` and `SharedAddress` | "Upgrading a Resource Type" on page 34<br>"Upgrading a Preregistered Resource Type" on page 74 |

**TABLE 2–1** Tasks for Administering Data Service Resources *(Continued)*

| Task | Instructions |
|---|---|
| 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 77 |
| Remove a node from a resource group | "Removing a Node From a Resource Group" on page 80 |
| Set up `HAStorage` or `HAStoragePlus` for resource groups to synchronize the startups between those resource groups and disk device groups | "How to Set Up `HAStorage` Resource Type for New Resources" on page 86 |
| Set up `HAStoragePlus` to enable highly available local file systems for failover data services with high I/O disk intensity | "How to Set Up the `HAStoragePlus` Resource Type for an NFS-Exported File System" on page 93 |
| Modify online the resource for a highly available file system | "Modifying Online the Resource for a Highly Available File System" on page 95 |
| Upgrade the `HAStoragePlus` resource type | "Upgrading a Resource Type" on page 34<br>"Upgrading the `HAStoragePlus` Resource Type" on page 102 |
| Distribute online resource groups among cluster nodes | "Distributing Online Resource Groups Among Cluster Nodes" on page 104 |
| Configure a resource type to automatically free up a node for a critical data service. | "How to Set Up an `RGOffload` Resource" on page 112 |
| 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 116 |
| Tune fault monitors for Sun Cluster data services | "Tuning Fault Monitors for Sun Cluster Data Services" on page 119 |

**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 25 for details about these options.

# Configuring and Administering Sun Cluster Data Services

Configuring a Sun Cluster data service involves the following tasks.

- Registering a resource type
- Upgrading a resource type
- Creating resource groups
- Adding resources to resource groups
- Bringing online resources

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 67.

# 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. For details about resource types, see Chapter 1.

## ▼ How to Register a Resource Type

**Note –** Perform this procedure from any cluster node.

**Before You Begin**

Ensure that you have 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.

**Steps**

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 2–1** Registering a Resource Type

The following example registers the `SUNW.iws` resource type, which represents the Sun Java System Web Server application in a Sun Cluster configuration.

```
# 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
```

**Next Steps** After registering resource types, you can create resource groups and add resources to the resource group. For details, see "Creating a Resource Group" on page 42.

**See Also** The `scrgadm`(1M) man page.

# Upgrading a Resource Type

Upgrading a resource type enables you to use new features that are introduced in the new version of the resource type. A new version of a resource type might differ from a previous version in the following ways.

- Default settings of resource type properties might change.
- New extension properties of the resource type might be introduced.
- Existing extension properties of the resource type might be withdrawn.
- The set of standard properties that are declared for the resource type might change.

- The attributes of resource properties such as `min`, `max`, `arraymin`, `arraymax`, `default`, and `tunability` might change.
- The set of declared methods might differ.
- The implementation of methods or the fault monitor might change.

Upgrading a resource type involves the tasks that are explained in the following sections:

1. "How to Install and Register an Upgrade of a Resource Type" on page 35
2. "How to Migrate Existing Resources to a New Version of the Resource Type" on page 36

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

The instructions that follow explain how to use the `scrgadm`(1M) command to perform this task. However, you are not restricted to using the `scrgadm` command for this task. Instead of the `scrgadm` command, you can use SunPlex Manager or the Resource Group option of the `scsetup`(1M) command to perform this task.

**Before You Begin**

Consult the documentation for the resource type to determine what you must do before installing the upgrade package on a node. One action from the following list will be required:

- You must reboot the node in noncluster mode.
- You may leave the node running in cluster mode, but you must turn off monitoring of all instances of the resource type.
- You may leave the node running in cluster mode and leave monitoring turned on for all instances of the resource type.

If you must reboot the node in noncluster mode, prevent a loss of service by performing a rolling upgrade. In a rolling upgrade, you install the package on each node individually while leaving the remaining nodes running in cluster mode.

**Steps**

1. **Become superuser or assume an equivalent role.**
2. **Install the package for the resource type upgrade on all cluster nodes where instances of the resource type are to be brought online.**
3. **Register the new version of the resource type.**

   To ensure that the correct version of the resource type is registered, you must specify the following information:

   - The resource type name

- The resource type registration (RTR) file that defines the resource type

```
# scrgadm -a -t resource-type-name -f path-to-new-rtr-file
```

The format of the resource type name is as follows:

*vendor-id*.*base-rt-name*:*rt-version*

For an explanation of this format, see "Format of Resource Type Names" on page 158.

4. **Display the newly registered resource type.**

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

5. **If necessary, set the `Installed_nodes` property to the nodes where the package for the resource type upgrade is installed.**

You must perform this step if the package for the resource type upgrade is not installed on all cluster nodes.

The `nodelist` property of all resource groups that contain instances of the resource type must be a subset of the `Installed_nodes` property of the resource type.

```
# scrgadm -c -t resource-type -h installed-node-list
```

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

The instructions that follow explain how to use the `scrgadm`(1M) command to perform this task. However, you are not restricted to using the `scrgadm` command for this task. Instead of the `scrgadm` command, you can use SunPlex Manager or the Resource Group option of the `scsetup`(1M) command to perform this task.

**Before You Begin**

Consult the instructions for upgrading the resource type to determine when you can migrate resources to a new version of the resource type.

- Any time
- Only when the resource is unmonitored
- Only when the resource is offline
- Only when the resource is disabled
- Only when the resource group is unmanaged

The instructions might state that you cannot upgrade your existing version of the resource. If you cannot migrate the resource, consider the following alternatives:

- Deleting the resource and replacing it with a new resource of the upgraded version
- Leaving the resource at the old version of the resource type

**Steps**

1. **Become superuser or assume an equivalent role.**
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.**

   - **If you can migrate the resource at any time, no action is required.**
   - **If you can migrate the resource only when the resource is unmonitored, type the following command:**

     # **scswitch -M -n -j** *resource*

   - **If you can migrate the resource only when the resource is offline, type the following command:**

     # **scswitch -n -j** *resource*

     ---

     **Note –** If other resources depend on the resource that you are migrating, this step fails. In this situation, consult the error message that is printed to determine the names of the dependent resources. Then repeat this step, specifying a comma-separated list that contains the resource that you are migrating and any dependent resources.

     ---

   - **If you can migrate the resource only when the resource is disabled, type the following command:**

     # **scswitch -n -j** *resource*

     ---

     **Note –** If other resources depend on the resource that you are migrating, this step fails. In this situation, consult the error message that is printed to determine the names of the dependent resources. Then repeat this step, specifying a comma-separated list that contains the resource that you are migrating and any dependent resources.

     ---

   - **If you can migrate the resource only when the resource group is unmanaged, type the following commands:**

     # **scswitch -n -j** *resource-list*
     # **scswitch -F -g** *resource-group*
     # **scswitch -u -g** *resource-group*

     The replaceable items in these commands are as follows:

     *resource-list* Specifies a comma-separated list of all resources in the resource group that is to be unmanaged.

     *resource-group* Specifies the resource group that is to be unmanaged.

**Note –** You can specify the resources in *resource-list* in any order. The `scswitch` command disables the resources in the order that is required to satisfy dependencies between the resources, regardless of their order in *resource-list*.

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

   If necessary, set other properties of the same resource to appropriate values in the same command. To set these properties, specify additional `-x` options or `-y` options in the command.

   To determine whether you are required to set other properties, consult the instructions for upgrading the resource type. You might be required to set other properties for the following reasons:

   - An extension property has been introduced in the new version of the resource type.
   - The default value of an existing property has been changed in the new version of the resource type.

   ```
   # scrgadm -c -j resource -y Type_version=new-version \
   [-x extension-property=new-value] [ -y standard-property=new-value]
   ```

   **Note –** If the existing version of the resource type does not support upgrades to the new version, this step fails.

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

   - **If you can migrate the resource at any time, no action is required.**

     **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 you can migrate the resource only when the resource is unmonitored, type the following command:**

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

- **If you can migrate the resource only when the resource is offline, type the following command:**

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

  **Note –** If you disabled in Step 2 other resources that depend on the resource that you are migrating, enable the dependent resources also.

- **If you can migrate the resource only when the resource is disabled, type the following command:**

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

  **Note –** If you disabled in Step 2 other resources that depend on the resource that you are migrating, enable the dependent resources also.

- **If you can migrate the resource only when the resource group is unmanaged, type the following commands:**

  ```
  # scswitch -e -j resource-list
  # scswitch -o -g resource-group
  # scswitch -z -g resource-group
  ```

**Example 2–2** Migrating a Resource That Can Be Migrated Only When Offline

This example shows the migration of a resource that can be migrated only when the resource is offline. The new resource type package contains methods that are located in new paths. Because the methods are not overwritten during the installation, the resource does not need to be disabled until after the upgraded resource type is installed.

The characteristics of the resource in this example are as follows:

- The new resource type version is 2.0.
- The resource name is `myresource`.
- The resource type name is `myrt`.
- The new RTR file is in `/opt/XYZmyrt/etc/XYZ.myrt`.
- No dependencies on the resource that is to be migrated exist.
- The resource that is to be migrated can be taken offline while leaving the containing resource group online.

This example assumes that the upgrade package is already installed on all cluster nodes according to the supplier'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 2–3 Migrating a Resource That Can Be Migrated Only When Unmonitored

This example shows the migration of a resource that can be migrated only when the resource is unmonitored. The new resource type package contains only the monitor and RTR file. Because the monitor is overwritten during installation, monitoring of the resource must be disabled before the upgrade package is installed.

The characteristics of the resource in this example are as follows:

- The new resource type version is 2.0.
- The resource name is `myresource` .
- The resource type name is `myrt`.
- The new RTR file is in `/opt/XYZmyrt/etc/XYZ.myrt`.

The following operations are performed in this example.

1. Before the upgrade package is installed, the following command is run to disable monitoring of the resource:

   ```
   # scswitch -M -n -j myresource
   ```

2. The upgrade package is installed on all cluster nodes according to the supplier's directions.
3. To register the new version of the resource type, the following command is run:

   ```
   # scrgadm -a -t myrt -f /opt/XYZmyrt/etc/XYZ.myrt
   ```

4. To change the `Type_version` property to the new version, the following command is run:

   ```
   # scrgadm -c -j myresource -y Type_version=2.0
   ```

5. To enable monitoring of the resource after its migration, the following command is run:

   ```
   # scswitch -M -e -j myresource
   ```

# Downgrading a Resource Type

You can downgrade a resource to an older version of its resource type. The conditions for downgrading a resource to an older version of the resource type are more restrictive than the conditions for upgrading to a newer version of the resource type. The resource group that contains the resource must be unmanaged.

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

The instructions that follow explain how to use the `scrgadm`(1M) command to perform this task. However, you are not restricted to using the `scrgadm` command for this task. Instead of the `scrgadm` command, you can use SunPlex Manager or the Resource Group option of the `scsetup`(1M) command to perform this task.

**Steps**

1. **Become superuser or assume an equivalent role.**

2. **Switch offline the resource group that contains the resource that you are downgrading.**

   **`scswitch -F -g`** *resource-group*

3. **Disable all resources in the resource group that contains the resource that you are downgrading.**

   **`scswitch -n -j`** *resource-list*

   ---

   **Note –** You can specify the resources in *resource-list* in any order. The `scswitch` command disables the resources in the order that is required to satisfy dependencies between the resources, regardless of their order in *resource-list*.

   If other resources depend on any resource in *resource-list*, this step fails. In this situation, consult the error message that is printed to determine the names of the dependent resources. Then repeat this step, specifying a comma-separated list that contains the resources that you originally specified and any dependent resources.

   ---

4. **Unmanage the resource group that contains the resource that you are downgrading.**

   **`scswitch -u -g`** *resource-group*

5. **If necessary, reregister the old version of the resource type to which you are downgrading.**

   Perform this step only if the version to which you are downgrading is no longer registered. If the version to which you are downgrading is still registered, omit this step.

   **scrgadm -a -t** *resource-type-name*

6. **For the resource that you are downgrading, set the `Type_version` property to old version to which you are downgrading.**

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

   **scrgadm -c -j** *resource-todowngrade* **-y Type_version=***old-version*

7. **Enable all the resources that you disabled in Step 3.**

   # **scswitch -e -j** *resource-list*

8. **Bring to a managed state the resource group that contains the resource that you downgraded.**

   # **scswitch -o -g** *resource-group*

9. **Bring online the resource group that contains the resource that you downgraded.**

   # **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.

For conceptual information about resource groups, see Chapter 1 and *Sun Cluster Concepts Guide for Solaris OS*.

## ▼ How to Create a Failover Resource Group

A failover resource group contains the following types of resources:

- Network address resources, which are instances of the built-in resource types `LogicalHostname` and `SharedAddress`
- Failover resources, which are the data service application resources for a failover data service

The network address resources and their dependent data service resources move between cluster nodes when data services fail over or are switched over.

---

**Note –** Perform this procedure from any cluster node.

---

**Steps**

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 2–4** 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:
```

**Next Steps** After you create a failover resource group, you can add application resources to this resource group. See "Adding Resources to Resource Groups" on page 46 for the procedure.

**See Also** The `scrgadm`(1M) man page.

## ▼ 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.

---

**Note –** Perform this procedure from any cluster node.

---

**Steps**

1. **Become superuser on a cluster member.**
2. **Create the failover resource group that holds the shared addresses that the scalable resource is to 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 2–5** 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:
```

**Next Steps** 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 52 for details.

**See Also** The `scrgadm`(1M) man page.

# 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.

For more information about resources, see *Sun Cluster Concepts Guide for Solaris OS* and Chapter 1.

## ▼ How to Add a Logical Hostname Resource to a 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 71.

**Note –** Perform this procedure from any cluster node.

**Before You Begin**

Ensure that you have the following information.

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

**Steps**

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 support the use of the adapter name for `netif`.

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

   # **scrgadm -pv -j** *resource*

**Example 2–6** 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
```

**Example 2–7** Adding Logical Hostname Resources That Identify IP Networking Multipathing Groups

This example shows the addition of the following logical host name resources to the resource group `nfs-fo-rg`:

- A resource that is named `cs23-rs`, which identifies the IP Networking Multipathing group `sc_ipmp0` on node 1 and node 2
- A resource that is named `cs24-rs`, which identifies the IP Networking Multipathing group `sc_ipmp1` on node 1 and node 2

```
# scrgadm -a -L -j cs23-rs -g nfs-fo-rg -l cs23-rs -n sc_ipmp0@1,sc_ipmp0@2
# scrgadm -a -L -j cs24-rs -g nfs-fo-rg -l cs24-rs -n sc_ipmp1@1,sc_ipmp1@2
```

**Next Steps** After you add logical hostname resources, use the procedure "How to Bring Online Resource Groups" on page 54 to bring them online.

**Troubleshooting** Adding a resource causes the Sun Cluster software to validate the resource. If the validation fails, the `scrgadm` command prints an error message and exits. To determine why the validation failed, 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.

**See Also** The `scrgadm`(1M) man page.

## ▼ How to Add a Shared Address Resource to a 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 71.

---

---

**Note –** Perform this procedure from any cluster node.

---

**Before You Begin** Ensure that you have 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.

**Steps**

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 support the use of the adapter name for `netif`.

---

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

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

**Example 2–8** 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
```

**Next Steps** After you add a shared-address resource, use the procedure "How to Bring Online Resource Groups" on page 54 to enable the resource.

**Troubleshooting** Adding a resource causes the Sun Cluster software to validate the resource. If the validation fails, the `scrgadm` command prints an error message and exits. To determine why the validation failed, 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.

**See Also** The `scrgadm`(1M) man page.

## ▼ 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.

---

**Note –** Perform this procedure from any cluster node.

---

**Before You Begin** Ensure that you have the following information.

- The name of the failover resource group to 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

**Steps** **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 a failover resource group. This resource group must already exist. |
| `-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 you are setting for the resource. The extension properties that you can set depend on the resource type. To determine which extension properties to set, see the documentation for the resource type. |
| `-y` *standard-property=value, …* | Specifies a comma-separated list of standard properties that you are setting for the resource. The standard properties that you can set depend on the resource type. To determine which standard properties to set, see the documentation for the resource type and Appendix A. |

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

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

**Example 2–9** 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
```

**Next Steps** After you add a failover application resource, use the procedure "How to Bring Online Resource Groups" on page 54 to enable the resource.

**Troubleshooting** Adding a resource causes the Sun Cluster software to validate the resource. If the validation fails, the `scrgadm` command prints an error message and exits. To determine why the validation failed, 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.

**See Also** The `scrgadm`(1M) man page.

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

A scalable application resource is an application resource that uses shared-address resources. The shared-address resources are in a failover resource group.

---

**Note –** Perform this procedure from any cluster node.

---

**Before You Begin** Ensure that you have the following information.

- The name of the scalable resource group to 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

**Steps**

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 you are setting for the resource. The extension properties that you can set depend on the resource type. To determine which extension properties to set, see the documentation for the resource type.

`-y` *standard-property=value*, …
Specifies a comma-separated list of standard properties that you are setting for the resource. The standard properties that you can set depend on the resource type. For scalable services, you typically set the `Port_list`, `Load_balancing_weights`, and `Load_balancing_policy` properties. To determine which standard properties to set, see the documentation for the resource type and Appendix A.

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

   # **scrgadm -pv -j** *resource*

**Example 2–10** 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
```

**Next Steps** After you add a scalable application resource, follow the procedure "How to Bring Online Resource Groups" on page 54 to enable the resource.

**Troubleshooting** Adding a resource causes the Sun Cluster software to validate the resource. If the validation fails, the `scrgadm` command prints an error message and exits. To determine why the validation failed, 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.

**See Also** The `scrgadm`(1M) man page.

# Bringing Online Resource Groups

To enable resources to begin providing HA services, you must perform the following operations:

- Enabling the resources in their resource groups
- Enabling the resource monitors
- Making the resource groups managed
- Bringing online the resource groups

You can perform these tasks individually or by using a single command.

After you bring online a resource group, 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.

## ▼ How to Bring Online Resource Groups

Perform this task from any cluster node.

**Steps**

1. **On a cluster member, become superuser or assume an equivalent role.**
2. **Type the command to bring online the resource groups.**
   - **If you have intentionally disabled a resource or a fault monitor that must remain disabled, type the following command:**

     `# scswitch -z -g` *rg-list*

     `-z` Brings online resource groups *without* 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.

- **If you require the resources and their fault monitors to be enabled when the resource groups are brought online, type the following command:**

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

`-Z` Brings online resource groups *after* 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 104.

---

3. **Verify that each resource group that you specified in Step 2 is online.**

   The output from this command indicates on which nodes each resource group is online.

```
# scstat -g
```

**Example 2–11** Bringing Online a Resource Group

This example shows how to bring online the resource group `resource-group-1` and verify its status. All resources in this resource and their fault monitors are also enabled.

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

**Next Steps** If you brought resource groups online *without* enabling their resources and fault monitors, enable the fault monitors of any resources that you require to be enabled. For more information, see "How to Enable a Resource Fault Monitor" on page 57.

**See Also** The `scswitch`(1M) man page.

# 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

**Steps**

1. **Become superuser on a cluster member.**
2. **Disable the resource fault monitor.**

   # **scswitch -n -M -j** *resource*

   `-n` Disables a resource or resource monitor

   `-M` Disables the fault monitor for the specified resource

   `-j` *resource* Specifies 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 2–12** 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

**Steps**

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 2–13** 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, follow this procedure.

---

**Note –** Perform this procedure from any cluster node.

---

## ▼ How to Remove a Resource Type

Removing a resource type involves disabling and removing all resources of that type in the cluster before unregistering the resource type.

**Before You Begin** To identify all instances of the resource type that you are removing, type the following command:

```
# scrgadm -pv
```

**Steps**

1. **Become superuser on a cluster member.**
2. **Disable each resource of the resource type that you are removing.**

   # **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 are removing.**

   # **scrgadm -r -j** *resource*

   | | |
   |---|---|
   | `-r` | Removes the specified resource |
   | `-j` | Specifies the name of the resource to remove |

4. **Unregister the resource type.**

   # **scrgadm -r -t** *resource-type*

   | | |
   |---|---|
   | `-r` | Unregisters 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 2–14** 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 unregister the resource type. 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
```

**See Also** The following man pages:

- `scrgadm`(1M)
- `scswitch`(1M)

# Removing Resource Groups

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

**Note –** Perform this procedure from any cluster node.

## ▼ How to Remove a Resource Group

**Before You Begin**

To identify all resources in the resource group that you are removing, type the following command:

```
# scrgadm -pv
```

**Steps**

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 in the resource group that you are removing.**

   ```
   # 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.**

   For each resource, type the following command.

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

   `-r` Removes the specified resource

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

5. **Remove the resource group.**

   # **scrgadm -r -g** *resource-group*

   `-r` Removes the specified resource group

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

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

   # **scrgadm -p**

**Example 2–15** 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
```

**See Also** The following man pages:

- `scrgadm`(1M)
- `scswitch`(1M)

# Removing Resources

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

**Note –** Perform this procedure from any cluster node.

## ▼ How to Remove a Resource

**Steps**

1. **Become superuser on a cluster member.**
2. **Disable the resource that you are removing.**

   # **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 2–16** 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
```

**See Also** The following man pages:

- `scrgadm`(1M)
- `scswitch`(1M)

# 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 is to become the new primary.

## ▼ How to Switch the Current Primary of a Resource Group

**Note –** Perform this procedure from any cluster node.

**Before You Begin** Ensure that the following conditions are met:

- You have the following information:
  - The name of the resource group that you are switching over
  - The names of the nodes on where the resource group is to be brought online or to remain online
- The nodes where the resource group is to be brought online or to remain online are cluster nodes.
- These nodes have been set up to be potential masters of the resource group that you are switching.

To see a list of potential primaries for the resource group, type the following command:

```
# scrgadm -pv
```

**Steps**

1. **On a cluster member, become superuser or assume an equivalent role.**

2. **Switch the resource group to a new set of primaries.**

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

   | | |
   |---|---|
   | `-z` | Switches the specified resource group to a new set of primaries. |
   | `-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 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 104.

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

   The output from this command indicates the state of the resource group that has been switched over.

   ```
   # scstat -g
   ```

**Example 2–17** Switching a Resource Group to a New Primary

This example shows how to switch the resource group `resource-group-1` from its current primary `phys-schost-1` to the potential primary `phys-schost-2`.

1. To verify that the resource group is online on `phys-schost-1`, the following command is run:

```
phys-schost-1# scstat -g
...-- Resource Groups --

            Group Name          Node Name           State
            ----------          ---------           -----
     Group: resource-group-1    phys-schost-1       Online
     Group: resource-group-1    phys-schost-2       Offline...
```

2. To perform the switch, the following command is run:

```
phys-schost-1# scswitch -z -g resource-group-1 -h phys-schost-2
```

3. To verify that the group is switched to be online on `phys-schost-2`, the following command is run:

```
phys-schost-1# scstat -g
...-- Resource Groups --

            Group Name          Node Name           State
            ----------          ---------           -----
     Group: resource-group-1    phys-schost-1       Offline
     Group: resource-group-1    phys-schost-2       Online...
```

**See Also** The following man pages:

- `scrgadm`(1M)
- `scswitch`(1M)

# 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

**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.

**Before You Begin**

Ensure that you have 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, type the following command:

```
# scrgadm -pv
```

**Steps**

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

2. **Disable all resources in the resource group.**

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

   | | |
   |---|---|
   | `-n` | Disables the resources |
   | `-j` *resource-list* | Specifies a comma-separated list of the resources in the resource group |

   **Note –** You can specify the resources in *resource-list* in any order. The `scswitch` command disables the resources in the order that is required to satisfy dependencies between the resources, regardless of their order in *resource-list*.

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 2–18** 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
```

**See Also** The following man pages:

- `scrgadm`(1M)
- `scswitch`(1M)

# Displaying Resource Type, Resource Group, and Resource Configuration Information

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

---

**Note –** You can view configuration settings for resources, resource groups, and resource types from any cluster node.

---

The `scrgadm` command provides the following 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 about 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 about specific resource types, resource groups, and resources. For example, the following command specifies that you want to view specific information about the resource `apache-1` only.

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

For more information, see the `scrgadm`(1M) man page.

# 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 123
- "Resource Properties" on page 131
- "Resource Group Properties" on page 146

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

**Note –** Perform this procedure from any cluster node.

**Before You Begin**

Ensure that you have 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 123.

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

**Steps**

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 123.

   ```
   # 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.<br><br>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 2–19** 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

This procedure explains how to change resource group properties. For a description of resource group properties, see "Resource Group Properties" on page 146.

---

**Note –** Perform this procedure from any cluster node.

---

**Before You Begin**

Ensure that you have the following information.

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

**Steps**

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 2–20** 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

This procedure explains how to change extension properties and standard properties of a resource.

- For a description of standard resource properties, see "Resource Properties" on page 131.
- For a description of the extension properties of a resource, see the documentation for the resource's resource type.

**Note –** Perform this procedure from any cluster node.

**Before You Begin**

Ensure that you have the following information.

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

**Steps**

1. **Become superuser on a cluster member.**
2. **View the current resource property settings.**

   # **scrgadm -pvv -j** *resource*
3. **Change the resource property.**

   # **scrgadm -c -j** *resource* **-y** *standard-property=new-value* | **-x** *extension-property=new-value*

   | | |
   |---|---|
   | `-c` | Changes the specified property |
   | `-j` *resource* | Specifies the name of the resource |
   | `-y` *standard-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 |
4. **Verify that the resource property has been changed.**

   # **scrgadm pvv -j** *resource*

### Example 2–21 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 2–22 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 74.

---

**Steps**

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, a failure of the resource's STOP method causes the following effects:

- The individual resource goes into the STOP_FAILED state.
- The resource group that contains the resource goes into the ERROR_STOP_FAILED state.

In this situation, you cannot perform the following operations:

- Bringing online the resource group on any node
- Adding resources to the resource group
- Removing resources from the resource group
- Changing the properties of the resource group
- Changing the properties of resources in the resource group

## ▼ How to Clear the STOP_FAILED Error Flag on Resources

**Note –** Perform this procedure from any cluster node.

**Before You Begin**

Ensure that you have 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

**Steps**

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.**

   `# scstat -g`

   The resource group state should now be `OFFLINE` or `ONLINE`.

   The resource group remains in the `ERROR_STOP_FAILED` state in the following combination of circumstances:

   - The resource group was being switched offline when the `STOP` method failure occurred.
   - The resource that failed to stop had a dependency on other resources in the resource group.

6. **If the resource group remains in the `ERROR_STOP_FAILED` state, correct the error as follows.**

   a. **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 |

   b. **Switch the resource group to the `ONLINE` state.**

**See Also** The `scswitch`(1M) man page.

# 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 34. 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 ClusterRelease |
|---|---|---|
| `SUNW.LogicalHostname` | 1.0 | 3.0 |
| | 2 | 3.1 9/04 |
| `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 2–23** 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 2–24** 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

The resource types `SUNW.LogicalHostname` and `SUNW.SharedAddress` are preregistered. 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 inadvertently 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 74 to register the new resource type version.

---

**Note –** Perform this procedure from any cluster node.

## ▼ How to Reregister Preregistered Resource Types After Inadvertent Deletion

**Step** ● **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 2–25** Reregistering a Preregistered Resource Type After Inadvertent Deletion

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

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

**See Also** The `scrgadm`(1M) man page.

# Adding or Removing a Node to or From a Resource Group

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

- Configuring a cluster node to be an additional master of a resource group
- Removing 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).

The procedure for scalable resource groups involves the following steps:

1. Repeating the procedure for failover groups that contain the network resources that the scalable resource uses
2. Changing the scalable group to be mastered on the new set of hosts

For more information, see the `scrgadm`(1M) man page.

---

**Note –** Run either procedure 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 77
- "How to Add a Node to a Failover Resource Group" on page 78

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 is to 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

**Steps**

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 5 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 scalable resource's `Load_balancing_weights` property to assign a weight to the node that you are adding 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

**Steps**

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. **If the `HAStorage` or `HAStoragePlus` `AffinityOn` extension property equals `True`, add the node to the appropriate disk set or device group.**

   - **If you are using Solstice DiskSuite or Solaris Volume Manager, use the `metaset` command.**

     `# metaset -s` *disk-set-name* `-a -h` *node-name*

     | | |
     |---|---|
     | `-s` *disk-set-name* | Specifies the name of the disk set on which the `metaset` command is to work |
     | `-a` | Adds a drive or host to the specified disk set |
     | `-h` *node-name* | Specifies the node to be added to the disk set |

   - **SPARC: If you are using VERITAS Volume Manager, use the `scsetup` utility.**

     a. **On any active cluster member, start the `scsetup` utility.**

        `# scsetup`

        The Main Menu is displayed.

     b. **On the Main Menu, type the number that corresponds to the option for device groups and volumes.**

     c. **On the Device Groups menu, type the number that corresponds to the option for adding a node to a VxVM device group.**

     d. **Respond to the prompts to add the node to the VxVM device group.**

4. **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 |

5. **Verify the updated information.**

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

**Example 2–26** 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 an IP Networking Multipathing group for node 2.)*

```
# scrgadm -c -j schost-2 -x netiflist=sc_ipmp0@1,sc_ipmp0@2,sc_ipmp0@3
# metaset -s red -a -h phys-schost-2
# 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 81
- "How to Remove a Node From a Failover Resource Group" on page 82
- "How to Remove a Node From a Failover Resource Group That Contains Shared Address Resources" on page 83

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 are to host the network resources that are used by the resource groups 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 are removing. If the resource group **is mastered** on the node that you are removing, run the `scswitch` command to switch the resource group offline from that node. The following `scswitch` command brings 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 that you are switching offline. This resource group is mastered on the node that you are removing |
| `-h` *new-masters* | Specifies the nodes that is now to master the resource group. |

For more information, see the `scswitch`(1M) man page.

**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 groups. Then, remove the node from the failover groups.

## ▼ 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. For information about this property, see Appendix A.

For details about scalable service configuration, see *Sun Cluster Concepts Guide for Solaris OS*.

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.

**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.**

   For details, see "How to Remove a Node From a Failover Resource Group That Contains Shared Address Resources" on page 83.

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

**See Also** The `scrgadm`(1M) man page.

## ▼ 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 groups. Then use this procedure to remove the node from the failover groups.

**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 83.

**Steps**

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 support the use of 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 82.

To modify the `auxnodelist` of the shared address resource, you must remove and re-create 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 continue to use the shared address resource, 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 re-create the shared address resource.

---

**Steps**

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`) that 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 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 come online. If the data service depends on global devices or cluster file systems that are not yet online, the `START` method times out. In this situation, you must reset the state of the resource groups that the data service uses and restart the data service manually.

To avoid these additional administrative tasks, use the `HAStorage` resource type or the `HAStoragePlus` resource type. Add an instance of `HAStorage` or `HAStoragePlus` to all resource groups whose data service resources depend on global devices or cluster file systems. Instances of these resource types perform the following operations:

- Monitoring global devices and cluster file systems
- Forcing the `START` method of the other resources in the same resource group to wait until global devices and cluster file systems become available

To determine which resource type to use, see "Choosing Between `HAStorage` and `HAStoragePlus`" on page 21.

To create an `HAStorage` resource, see "How to Set Up `HAStorage` Resource Type for New Resources" on page 86.

To create an `HAStoragePlus` resource, see "How to Set Up the `HAStoragePlus` Resource Type for an NFS-Exported File System" on page 93.

## ▼ How to Set Up HAStorage Resource Type for New Resources

HAStorage might not be supported in a future release of Sun Cluster software. Equivalent functionality is supported by HAStoragePlus. For instructions for upgrading from HAStorage to HAStoragePlus, see "Upgrading From HAStorage to HAStoragePlus" on page 88.

In the following example, the resource group resource-group-1 contains the following 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 an HAStorage resource hastorage-1 for new resources in resource-group-1, read "Synchronizing the Startups Between Resource Groups and Disk Device Groups" on page 85 and then perform the following steps.

To create an HAStoragePlus resource, see "Enabling Highly Available Local File Systems" on page 91.

**Steps**

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
   - Mount points for cluster file systems, such as /global/nfs

**Note –** If `ServicePaths` contains paths for cluster file systems, global device groups might not be colocated with the resource groups that correspond to them.

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
```

**More Information**

## Affinity Switchovers

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`. For details, see the `SUNW.HAStorage`(5) man page.

**Note –** `HAStorage` and `HAStoragePlus` do not permit `AffinityOn` to be set to `True` if the resource group is scalable. `HAStorage` and `HAStoragePlus` check the `AffinityOn` value and internally reset 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 software. Equivalent functionality is supported by HAStoragePlus. For instructions for upgrading from HAStorage to HAStoragePlus, see "Upgrading From HAStorage to HAStoragePlus" on page 88.

**Before You Begin** Read "Synchronizing the Startups Between Resource Groups and Disk Device Groups" on page 85.

**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 software. Equivalent functionality is supported by HAStoragePlus. For instructions for upgrading from HAStorage to HAStorage, see the subsections that follow.

## ▼ How to Upgrade From HAStorage to HAStoragePlus When Using Device Groups or CFS

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

**Steps**

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.**

---

**Note –** Instead of using the ServicePaths property of HAStorage, you must use the FilesystemMountPoints property or GlobalDevicePaths property of HAStoragePlus.

---

- **To specify the mount point of a file system, type the following command.**

  The FilesystemMountPoints extension property must match the sequence that is specified in /etc/vfstab.

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

- **To specify global device paths, type the following command.**

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

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 File System

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

**Steps**

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.**

---

**Note –** Instead of using the `ServicePaths` property of `HAStorage`, you must use the `FilesystemMountPoints` property or `GlobalDevicePaths` property of `HAStoragePlus`.

---

- **To specify the mount point of a file system, type the following command.**

The `FilesystemMountPoints` extension property must match the sequence that is specified in `/etc/vfstab`.

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

- **To specify global device paths, type the following command.**

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

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

Using a highly available local file system improves the performance of I/O intensive data services. To make a local file system highly available in a Sun Cluster environment, use the `HAStoragePlus` resource type.

The instructions for each Sun Cluster data service that is I/O intensive explain how to configure the data service to operate with the `HAStoragePlus` resource type. For more information, see the individual Sun Cluster data service guides.

For instructions for setting up the `HAStoragePlus` resource type for an NFS-exported file system, see "How to Set Up the `HAStoragePlus` Resource Type for an NFS-Exported File System" on page 93.

---

**Note –** Do *not* use the `HAStoragePlus` resource type to make a root file system highly available.

---

## Configuration Requirements for Highly Available Local File Systems

Any file system on multihost disks must be accessible from any host that is directly connected to those multihost disks. To meet this requirement, configure the highly available local file system as follows:

- Ensure that the disk partitions of the local file system reside on global devices.
- Set the `AffinityOn` extension property of the `HAStoragePlus` resource that specifies these global devices to `True`.
- Create the `HAStoragePlus` resource in a failover resource group.
- Ensure that the failback settings for the device groups and the resource group that contains the `HAStoragePlus` resource are identical.

---

**Note –** The use of a volume manager with the global devices for a highly available local file system is optional.

---

## Format of Device Names for Devices Without a Volume Manager

If you are not using a volume manger, use the appropriate format for the name of the underlying storage device. The format to use depends on the type of storage device as follows:

- For block devices: `/dev/global/dsk/d`*D*`s`*S*
- For raw devices: `/dev/global/rdsk/d`*D*`s`*S*

The replaceable items in these names are as follows:

- *D* is an integer that specifies the device ID (DID) instance number.
- *S* is an integer that specifies the slice number.

## Sample Entries in `/etc/vfstab` for Highly Available Local File Systems

The following examples show entries in the `/etc/vfstab` file for global devices that are to be used for highly available local file systems.

**EXAMPLE 2–27** Entries in `/etc/vfstab` for a Global Device Without a Volume Manager

This example shows entries in the `/etc/vfstab` file for a global device on a physical disk without a volume manager.

```
/dev/global/dsk/d1s0       /dev/global/rdsk/d1s0
/global/local-fs/nfs  ufs     5  no     logging
```

**EXAMPLE 2–28** Entries in `/etc/vfstab` for a Global Device With Solaris Volume Manager

This example shows entries in the `/etc/vfstab` file for a global device that uses Solaris Volume Manager.

```
/dev/md/kappa-1/dsk/d0   /dev/md/kappa-1/rdsk/d0
/global/local-fs/nfs ufs     5  no     logging
```

**EXAMPLE 2–29** Entries in `/etc/vfstab` for a Global Device With VxVM

This example shows entries in the `/etc/vfstab` file for a global device that uses VxVM.

```
/dev/vx/dsk/kappa-1/appvol    /dev/vx/rdsk/kappa-1/appvol
/global/local-fs/nfs vxfs     5 no     log
```

## ▼ How to Set Up the `HAStoragePlus` Resource Type for an NFS-Exported File System

The `HAStoragePlus` 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 about making a local file system highly available, see "Enabling Highly Available Local File Systems" on page 91. To use both of these features, set up the `HAStoragePlus` resource type.

---

**Note –** These instructions 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.

---

The following example uses a simple NFS service that exports 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` is 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 omit the global option and specify that the mount at boot flag is no.
- The path-prefix directory is on the root directory of the same file system that is to be mounted, for example, `/global/local-fs/nfs`. The path-prefix directory is the directory that HA-NFS uses to maintain administrative information and status information.

**Steps**

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

2. **Determine whether the `HAStoragePlus` resource type and the `SUNW.nfs` resource type are registered.**

   The following command prints a list of registered resource types.

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

3. **If necessary, register the `HAStoragePlus` resource type and the `SUNW.nfs` resource type.**

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

4. **Create the failover resource group `nfs-rg`.**

   ```
   # 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. **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 –** You can use the `FilesystemMountPoints` extension property to specify a list of one or more mount points for file systems. This list can consist of mount points for both local file systems and global file systems. The mount at boot flag is ignored by `HAStoragePlus` for global file systems.

   ---

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

   The node where the resource group is brought online becomes the primary node for the `/global/local-fs/nfs` file system's underlying global device partition. The file system `/global/local-fs/nfs` is then locally mounted on this node.

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

8. **Create the resource `nfs-rs` of type `SUNW.nfs` and specify its resource dependency on the resource `nfs-hastp-rs`.**

   The file `dfstab.nfs-rs` must be present in `/global/local-fs/nfs/SUNW.nfs`.

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

   **Note –** Before you can set the dependency in the `nfs-rs` resource, the `nfs-hastp-rs` resource must be online.

9. **Take offline the resource group `nfs-rg`.**

   ```
   # scswitch -F -g nfs-rg
   ```

10. **Bring online the `nfs-rg` group on a cluster node.**

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

**Caution –** Ensure that you switch only the resource group. Do *not* attempt to switch the device group. If you attempt to switch the device group, the states of the resource group and the device group become inconsistent, causing the resource group to fail over.

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` is locally mounted before the NFS server is started.

# 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 85.

**Steps**

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 2–30** 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.

```
# 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 85.

**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.

**Steps**

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 2–31 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.

**Steps**

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 2–32** 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.

```
# 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 34. 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 ClusterRelease |
|---|---|
| 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 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=`***affinity-list*

*affinity-list* Specifies a comma-separated list of affinities between the source resource group and a target resource group or several target resource groups. You may specify a single affinity or more than one affinity in the list.

Specify each affinity in the list as follows:

*operator target-rg*

**Note –** Do not include a space between *operator* and *target-rg*.

*operator* Specifies the type of affinity that you are creating. For more information, see Table 2–2.

*target-rg* Specifies the resource group that is the target of the affinity that you are creating.

**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 110. |

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 110.

**EXAMPLE 2–33** 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.

**EXAMPLE 2–33** Enforcing Colocation of a Resource Group With Another Resource Group *(Continued)*

```
# 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 2–34** 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=neg-affinity-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=`*neg-affinity-list*
: Specifies a comma-separated list of weak negative affinities between the source resource group and the resource groups that are the target of the weak negative affinity. The target 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 2–35** 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 2–36** 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 2–37** 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.

**EXAMPLE 2–37** Delegating the Failover or Switchover of a Resource Group *(Continued)*

```
# 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 2–38** 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 104.

Prioritized Service Management (`RGOffload`) enables your cluster automatically to free a node's resources for critical data services. Use `RGOffload` when the startup of a critical failover data service requires a noncritical, scalable or failover data service to be brought offline. Use `RGOffload` to offload resource groups that contain 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

**Steps**

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 necessary, register the resource type.**

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

4. **Set the `Desired_primaries` property to zero in each resource group that the `RGOffload` resource is to be offload.**

   ```
   # 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 might cause the resource group to be taken offline and brought back online repeatedly.

For a description of `RGOffload` extension properties, see "Configuring `RGOffload` Extension Properties" on page 114.

```
# 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
   ```

   You can also use `Resource_dependencies_weak`. Using `Resource_dependencies_weak` on the `RGOffload` resource type allows the critical failover resource to start even if errors are encountered during offload of `offload-rg`.

8. **Bring online the resource group that is to be offloaded**

   ```
   # 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.

   In Step 4, `Desired_primaries` for resource groups that are to be offloaded was set to 0. Therefore, the `-Z`option cannot bring these resource groups online.

9. **If the critical failover resource group is not online, bring it online.**

   ```
   # scswitch -Z -g critical-rg
   ```

**Example 2–39** Configuring an `RGOffload` Resource

This example shows how to configure the `RGOffload` resource `rgofl` as follows:

- The critical resource group `oracle_rg` contains the `RGOffload` resource.
- The critical resource is `oracle-server-rs`.
- The scalable resource groups `IWS-SC` and `IWS-SC-2` are to be offloaded when the critical resource group comes online.

- The resource groups `oracle_rg`, `IWS-SC`, and `IWS-SC-2` can be mastered on any node of cluster `triped`, namely: `phys-triped-1`, `phys-triped-2`, or `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 online on all nodes the resource groups that are to be offloaded.*]
```
# 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

This section lists the extension properties that you can configure for `RGOffload`. The Tunable entry indicates when you can update the property.

Typically, you use the command line `scrgadm -x` *parameter=value* to configure extension properties when you create the `RGOffload` resource.

`continue_to_offload` (Boolean)
Specifies whether to continue offloading the remaining resource groups in the `rg_to_offload` list after an error in offloading a resource group.

This property is used only by the `START` method.

**Default:** `True`

**Tunable:** Any time

`max_offload_retry` (integer)
Specifies the number of attempts to offload a resource group during startup if cluster reconfiguration or resource group reconfiguration causes a failure to offload. The interval between successive retries is 10 seconds.

If `max_offload_retry` is too high, the `START` method of the `RGOffload` resource might time out before the maximum offload attempts are completed. To avoid this possibility, use the following formula to calculate `max_offload_retry`:

*max-offload-retry* < *start-timeout* / (*num-rg* × *offload-retry-interval*)

| | |
|---|---|
| *max-offload-retry* | The value of the `max_offload_retry` extension property |
| *start-timeout* | The value of the `Start_timeout` of property the `RGOffload` resource |
| *num-rg* | The number of resource groups to that are to be offloaded |
| *offload-retry-interval* | The interval between successive retries, which is 10 seconds |

This property is used only by the `START` method.

**Default:** 15

**Tunable:** Any time

`rg_to_offload` (string)
Specifies a comma-separated list of resource groups that are to be offloaded on a node when a critical failover resource group starts on that node. This property has no default and must be set.

This list should not contain resource groups that depend upon each other. `RGOffload` does not check for dependency loops in the list of resource groups that are set in the `rg_to_offload` extension property.

For example, if resource group `RG-B` depends in some way on `RG-A`, do not include both resource groups in `rg_to_offload`.

**Default:** None

**Tunable:** Any time

# Fault Monitor

The `RGOffload` fault monitor prevents noncritical resource groups from being brought online on the node that masters the critical resource. The fault monitor might detect that a noncritical resource group is online on the node that masters the critical resource. In this situation, the fault monitor attempts to start the resource group on other nodes. The fault monitor also brings offline the resource group on the node that masters the critical resource.

Because `desired_primaries` for noncritical resource groups is set to 0, offloaded resource groups are not restarted on nodes that become available later. Therefore, the `RGOffload` fault monitor attempts to start noncritical resource groups on as many primaries as possible, until `maximum_primaries` limit is reached. However, the fault monitor keeps noncritical resource groups offline on the node that masters the critical resource.

`RGOffload` attempts to start all offloaded resource groups unless the resource groups 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 value of the `RGOffload` resource's `Thorough_probe_interval` property specifies the interval between fault monitor probes.

# 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.

**Steps**

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`.

**Steps**

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 131.

APPENDIX **A**

# Standard Properties

This appendix describes the standard resource type, resource, and resource group properties. This appendix also describes the resource property attributes that are available for changing system-defined properties and creating extension properties.

**Note –** Property names for resource types, resources, and resource groups are *not* case sensitive. You can use any combination of uppercase and lowercase letters when you specify property names.

This appendix covers the following topics:



## Resource Type Properties

The following information describes the resource type properties that are defined by the Sun Cluster software. The property values are categorized as follows:

- **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 space 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 space 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 or 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 space 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 was not declared in the RTR file.
- **Query-only** – Cannot be set directly by an administrative tool.

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 cluster administrator.

Property names are shown first, followed by a description.

---

**Note –** Resource type property names, such as `API_version` and `Boot`, are *not* case sensitive. You can use any combination of uppercase and lowercase letters when you specify property names.

---

`API_version` (`integer`)
The minimum version of the resource management API that is required to support 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` |
| 3.1 3/05 | `6` |

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.

**Note –** If you do not declare this property or set this property to the default value (2), the data service can be installed on any version of Sun Cluster starting with Sun Cluster 3.0.

**Category:** Optional

**Default:** 2

**Tunable:** NONE

`Boot` (`string`)
An optional callback method: the path to the program that the RGM runs on a node, which joins or rejoins the cluster when a resource of this type is already managed. This method initializes resources of this type as the `Init` method does.

**Category:** Conditional or Explicit

**Default:** No default

**Tunable:** NONE

`Failover` (`boolean`)
`TRUE` indicates that resources of this type cannot be configured in any group that can be online on multiple nodes at the same time.

The following table shows how to use this resource type property in combination with the `Scalable` resource property.

| Value of the `Failover` Resource Type | Value of the `Scalable` Resource | Description |
|---|---|---|
| `TRUE` | `TRUE` | Do not specify this illogical combination. |
| `TRUE` | `FALSE` | Specify this combination for a failover service. |
| `FALSE` | `TRUE` | Specify this combination for a scalable service that uses a `SharedAddress` resource for network load balancing.<br><br>The *Sun Cluster Concepts Guide for Solaris OS* describes `SharedAddress` in more detail. |
| `FALSE` | `FALSE` | Although it is an unusual combination, you can use this combination to select a multi-master service that does not use network load balancing. |

The description of `Scalable` in the `r_properties`(5) man page and Chapter 3, "Key Concepts for System Administrators and Application Developers," in *Sun Cluster Concepts Guide for Solaris OS* contain additional information.

**Category:** Optional

**Default:** `FALSE`

**Tunable:** `NONE`

`Fini` (`string`)
An optional callback method: the path to the program that the RGM runs when a resource of this type is removed from RGM management.

**Category:** Conditional or Explicit

**Default:** No default

**Tunable:** `NONE`

`Init` (`string`)
An optional callback method: the path to the program that the RGM runs when a resource of this type becomes managed by the RGM.

**Category:** Conditional or Explicit

**Default:** No default

**Tunable:** `NONE`

`Init_nodes` (`enum`)
Indicates the nodes on which the RGM is to call the `Init`, `Fini`, `Boot`, and `Validate` methods. 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).

**Category:** Optional

**Default:** `RG_PRIMARIES`

**Tunable:** `NONE`

`Installed_nodes` (`string_array`)
A list of the cluster node names on which the resource type can 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:** `ANYTIME`

`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:** NONE

`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:** NONE

`Monitor_check` (`string`)
An optional callback method: the path to the program that the RGM runs before performing a monitor-requested failover of a resource of this type.

**Category:** Conditional or Explicit

**Default:** No default

**Tunable:** NONE

`Monitor_start` (`string`)
An optional callback method: the path to the program that the RGM runs to start a fault monitor for a resource of this type.

**Category:** Conditional or Explicit

**Default:** No default

**Tunable:** NONE

`Monitor_stop` (`string`)
A callback method that is required if `Monitor_start` is set: the path to the program that the RGM runs to stop a fault monitor for a resource of this type.

**Category:** Conditional or Explicit

**Default:** No default

**Tunable:** NONE

`Pkglist` (`string_array`)
An optional list of packages that are included in the resource type installation.

**Category:** Conditional or Explicit

**Default:** No default

**Tunable:** NONE

`Postnet_stop` (`string`)
An optional callback method: the path to the program that the RGM runs 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 or Explicit

**Default:** No default

**Tunable:** NONE

`Prenet_start` (`string`)
An optional callback method: the path to the program that the RGM runs before the RGM calls the `Start` method of any network-address resources on which a resource of this type depends. This method performs `Start` actions that must be performed before network interfaces are configured.

**Category:** Conditional or Explicit

**Default:** No default

**Tunable:** NONE

`Resource_list` (`string_array`)
The list of all resources of the resource type. The cluster administrator does not set this property directly. Rather, the RGM updates this property when the cluster administrator adds or removes a resource of this type to or from any resource group.

**Category:** Query-only

**Default:** Empty list

**Tunable:** NONE

`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*`:`*rt-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, use the stock symbol of the company that is 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:** NONE

`RT_basedir` (`string`)
The directory path that is used to complete relative paths for callback methods. This path must be set to the directory in which the resource type packages are installed. The path must be a complete path, that is, it must start with a forward slash (`/`).

**Category:** Required unless all method path names are absolute

**Default:** No default

**Tunable:** `NONE`

`RT_description` (`string`)
A brief description of the resource type.

**Category:** Conditional

**Default:** Empty string

**Tunable:** `NONE`

`RT_system` (`boolean`)
If the `RT_system` property is `TRUE` for a resource type, you cannot delete the resource type (`scrgadm -r -t` *resource-type-name*). This property prevents the accidental deletion of resource types, such as `LogicalHostname`, that are used to support the cluster infrastructure. However, you can apply the `RT_system` property to any resource type.

To delete a resource type whose `RT_system` property is set to `TRUE`, you must first set the property to `FALSE`. Use care when you delete a resource type whose resources support cluster services.

**Category:** Optional

**Default:** `FALSE`

**Tunable:** `ANYTIME`

`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:** Conditional/Explicit or Required

**Default:** No default

**Tunable:** `NONE`

`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:** NONE

`Start` (`string`)
A callback method: the path to the program that the RGM runs to start a resource of this type.

**Category:** Required unless the RTR file declares a `Prenet_start` method

**Default:** No default

**Tunable:** NONE

`Stop` (`string`)
A callback method: the path to the program that the RGM runs to stop a resource of this type.

**Category:** Required unless the RTR file declares a `Postnet_stop` method

**Default:** No default

**Tunable:** NONE

`Update` (`string`)
An optional callback method: the path to the program that the RGM runs when properties of a running resource of this type are changed.

**Category:** Conditional or Explicit

**Default:** No default

**Tunable:** NONE

`Validate` (`string`)
An optional callback method: the path to the program that the RGM runs to check values for properties of resources of this type.

**Category:** Conditional or Explicit

**Default:** No default

**Tunable:** NONE

`Vendor_ID` (`string`)
See the `Resource_type` property.

**Category:** Conditional

**Default:** No default

**Tunable:** NONE

# Resource Properties

This section describes the resource properties that are defined by the Sun Cluster software. The property values are categorized as follows:

- **Required**. The cluster administrator must specify a value when he or she creates a resource with an administrative utility.
- **Optional**. If the cluster administrator does not specify a value when he or she creates 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 cluster 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.

The `Tunable` attribute, which is described in "Resource Property Attributes" on page 154, lists whether and when you can update resource properties, as follows:

| | |
|---|---|
| `FALSE` or `NONE` | 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`.

This property is only used for scalable services.

**Category:** Optional

**Default:** No default

**Tunable:** `ANYTIME`

`Boot_timeout` for each callback method in the Type (`integer`)
A time lapse, in seconds, after which the RGM concludes that an invocation of this method has failed. For a given resource type, timeout properties are defined only for those methods that are declared in the RTR file.

**Category:** Conditional or Optional

**Default:** `3600` (one hour), if the method itself is declared in the RTR file

**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 is available to the cluster 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 RTR file, the `Tunable` value for the property is `WHEN_DISABLED`.

**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 154 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`)
Modifies the recovery actions that the RGM takes when a resource fails to start or to stop successfully, or when a resource monitor finds a resource to be unhealthy and consequently requests a restart or failover.

`NONE`, `SOFT`, or `HARD` (method failures)

These settings affect only failover behavior when a start or stop method (`Prenet_start`, `Start`, `Monitor_stop`, `Stop`, `Postnet_stop`) fails. Once the resource has started successfully, `NONE`, `SOFT`, and `HARD` have no effect on subsequent resource restart or giveover behavior that the resource monitor initiates with the `scha_control` command or the `scha_control()` function. See the `scha_control`(1HA) and the `scha_control`(3HA) man pages. `NONE` indicates that the RGM is not to take any recovery action when one of the previously listed start or stop methods fails. `SOFT` or `HARD` indicates that if a `Start` or `Prenet_start` method fails, the RGM is to relocate the resource's group to a different node. For `Start` or `Prenet_start` failures, `SOFT` and `HARD` are the same.

For failure of a stop method (Monitor_stop, Stop, or Postnet_stop), SOFT is the same as NONE. If Failover_mode is set to HARD when one of these stop methods fails, the RGM reboots the node to force the resource group offline. The RGM might then attempt to start the group on another node.

RESTART_ONLY or LOG_ONLY

Unlike NONE, SOFT, and HARD, which affect failover behavior when a start or stop method fails, RESTART_ONLY and LOG_ONLY affect all failover behavior. Failover behavior includes 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 or a resource group. The RGM allows Retry_count restarts within Retry_interval. If Retry_count is exceeded, no further 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.

RESTART_ONLY or LOG_ONLY (method failures)

If a Prenet_start, Start, Monitor_stop, Stop, or Postnet_stop method fails, RESTART_ONLY and LOG_ONLY are the same as NONE. That is, the node is neither failed over nor rebooted.

Effect of Failover_mode settings on a data service

The effect that each setting for Failover_mode has on a data service depends on whether the data service is monitored or unmonitored and whether it is based on the Data Services Development Library (DSDL).

- A data service is monitored if it implements a Monitor_start method and monitoring of the resource is enabled. The RGM starts a resource monitor by executing the Monitor_start method after starting the resource itself. The resource monitor probes the health of the resource. If the probes fail, the resource monitor might request a restart or a failover by calling the scha_control() function. For DSDL-based resources, probes might reveal partial failure (degradation) or a complete failure of the data service. Repeated partial failures accumulate to a complete failure.
- A data service is unmonitored if it does not provide a Monitor_start method or monitoring of the resource has been disabled.
- DSDL-based data services include those that are developed with Agent Builder, through the GDS, or by using the DSDL directly. Some data services, HA Oracle for example, were developed without using the DSDL.

NONE, SOFT, or HARD (probe failures)

If you set Failover_mode to NONE, SOFT, or HARD and the data service is a monitored DSDL-based service, and if the probe fails completely, the monitor calls the scha_control() function to request a restart of the resource. If probes

continue to fail, the resource is restarted up to a maximum of `Retry_count` number of times within `Retry_interval`. If the probes fail again after the `Retry_count` number of restarts is reached, the monitor requests a failover of the resource's group to another node.

If you set `Failover_mode` to `NONE`, `SOFT`, or `HARD` and the data service is an unmonitored DSDL-based service, the only failure that is detected is the death of the resource's process tree. If the resource's process tree dies, the resource is restarted.

If the data service is a not a DSDL-based service, the restart or failover behavior depends on how the resource monitor is coded. For example, the Oracle resource monitor recovers by restarting the resource or the resource group, or by failing over the resource group.

`RESTART_ONLY` (probe failures)

If you set `Failover_mode` to `RESTART_ONLY` and the data service is a monitored DSDL-based service, and if the probe fails completely, the resource is restarted `Retry_count` times within `Retry_interval`. However, if `Retry_count` is exceeded, the resource monitor exits, sets the resource status to `FAULTED`, and generates the status message "Application faulted, but not restarted. Probe quitting." At this point, although monitoring is still enabled, the resource is effectively unmonitored until it is repaired and restarted by the cluster administrator.

If you set `Failover_mode` to `RESTART_ONLY` and the data service is an unmonitored DSDL-based service, and if the process tree dies, the resource is *not* restarted.

If a monitored data service is not DSDL-based, the recovery behavior depends on how the resource monitor is coded. If you set `Failover_mode` to `RESTART_ONLY`, the resource or resource group can be restarted by a call to the `scha_control()` function `Retry_count` times within `Retry_interval`. If the resource monitor exceeds `Retry_count`, the attempt to restart fails. If the monitor calls the `scha_control()` function to request a failover, that request fails as well.

`LOG_ONLY` (probe failures)

If you set `Failover_mode` to `LOG_ONLY` for any data service, all `scha_control()` requests either to restart the resource or resource group or to fail over the group are precluded. If the data service is DSDL-based, a message is logged when a probe completely fails, but the resource is not restarted. If a probe fails completely more than `Retry_count` times within `Retry_interval`, the resource monitor exits, sets the resource status to `FAULTED`, and generates the status message "Application faulted, but not restarted. Probe quitting." At this point, although monitoring is still enabled, the resource is effectively unmonitored until it is repaired and restarted by the cluster administrator.

If you set `Failover_mode` to `LOG_ONLY` and the data service is an unmonitored DSDL-based service, and if the process tree dies, a message is logged but the resource is not restarted.

If a monitored data service is not DSDL-based, the recovery behavior depends on how the resource monitor is coded. If you set `Failover_mode` to `LOG_ONLY`, all `scha_control()` requests either to restart the resource or resource group or to fail over the group fail.

**Category:** Optional

**Default:** `NONE`

**Tunable:** `ANYTIME`

`Fini_timeout` for each callback method in the Type (`integer`)
A time lapse, in seconds, after which the RGM concludes that an invocation of this method has failed. For a given resource type, timeout properties are defined only for those methods that are declared in the RTR file.

**Category:** Conditional or Optional

**Default:** `3600` (one hour), if the method itself is declared in the RTR file

**Tunable:** `ANYTIME`

`Init_timeout` for each callback method in the Type (`integer`)
A time lapse, in seconds, after which the RGM concludes that an invocation of this method has failed. For a given resource type, timeout properties are defined only for those methods that are declared in the RTR file.

**Category:** Conditional or Optional

**Default:** `3600` (one hour), if the method itself is declared in the RTR file

**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 or 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 that is distributed to the specified *node*. The fraction of load that is 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 one-fourth of the load and node 2 receives three-fourths of the load. 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 RTR file, the `Tunable` value for the property is `ANYTIME`. Changing this property revises the distribution for new connections only.

**Category:** Conditional or Optional

**Default:** The empty string ("")

**Tunable:** ANYTIME

`Monitor_check_timeout` for each callback method in the Type (`integer`)
A time lapse, in seconds, after which the RGM concludes that an invocation of this method has failed. For a given resource type, timeout properties are defined only for those methods that are declared in the RTR file.

**Category:** Conditional or Optional

**Default:** `3600` (one hour), if the method itself is declared in the RTR file

**Tunable:** ANYTIME

`Monitor_start_timeout` for each callback method in the Type (`integer`)
A time lapse, in seconds, after which the RGM concludes that an invocation of this method has failed. For a given resource type, timeout properties are defined only for those methods that are declared in the RTR file.

**Category:** Conditional or Optional

**Default:** `3600` (one hour), if the method itself is declared in the RTR file

**Tunable:** ANYTIME

`Monitor_stop_timeout` for each callback method in the Type (`integer`)
A time lapse, in seconds, after which the RGM concludes that an invocation of this method has failed. For a given resource type, timeout properties are defined only for those methods that are declared in the RTR file.

**Category:** Conditional or 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`, monitoring on the resource is stopped, although the resource itself remains online. The `Monitor_start` method is not called until monitoring is re-enabled. If the resource does not have a monitor callback method, this property does not exist.

**Category:** Query-only

**Default:** No default

**Tunable:** `NONE`

`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 RTR 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 the CRNP.

---

**Category:** Conditional or Required

**Default:** No default

**Tunable:** `AT_CREATION`

`Num_resource_restarts` on each cluster node (`integer`)
You cannot directly set this property. This property 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 (`0`) 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, the `Num_resource_restarts` property is not available for resources of that type.

**Category:** Query-only

**Default:** No default

**Tunable:** `NONE`

`Num_rg_restarts` on each cluster node (`integer`)
You cannot directly set this property. This property is set by the RGM to the number of `scha_control Restart` calls that the resource has made for its containing resource group on this node within the past *n* seconds. *n* is the value of the `Retry_interval` property of the resource. If a resource type does not declare the `Retry_interval` property, the `Num_rg_restarts` property is not available for resources of that type.

**Category:** See description

**Default:** No default

**Tunable:** `NONE`

`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 has no callbacks run until it is re-enabled.

**Category:** Query-only

**Default:** No default

**Tunable:** `NONE`

`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 or Required

**Default:** No default

**Tunable:** `ANYTIME`

`Postnet_stop_timeout` for each callback method in the Type (`integer`)
A time lapse, in seconds, after which the RGM concludes that an invocation of this method has failed. For a given resource type, timeout properties are defined only for those methods that are declared in the RTR file.

**Category:** Conditional or Optional

**Default:** `3600` (one hour), if the method itself is declared in the RTR file

**Tunable:** `ANYTIME`

`Prenet_start_timeout` for each callback method in the Type (`integer`)
A time lapse, in seconds, after which the RGM concludes that an invocation of this method has failed. For a given resource type, timeout properties are defined only for those methods that are declared in the RTR file.

**Category:** Conditional or Optional

**Default:** `3600` (one hour), if the method itself is declared in the RTR file

**Tunable:** `ANYTIME`

`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 the `Resource_dependencies` 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 to start this resource until the resource in the list starts. If the resource in this resource's `Resource_dependencies` list does not start, this resource 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 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 146 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 resource 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 resource 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 the `Resource_dependencies_restart` resource has a restart dependency.

This property works as `Resource_dependencies` does, with one exception. If any resource in the restart dependency list is restarted, this resource is restarted. The RGM restarts this resource 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 the `Resource_dependencies_weak` 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 to start this resource until the resource in the list starts. 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 resource 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 resource 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:** `NONE`

`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 starts 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 the `rg_properties`(5) man page). If neither property is specified, the RGM uses the predefined project name `default`. The specified project name must exist in the projects database(see the `projects`(1) man page and *System Administration Guide: Solaris Containers-Resource Management and Solaris Zones*).

This property is only supported starting in Solaris 9.

---

**Note –** Changes to this property take effect the next time that 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:** `NONE`

`Retry_count` (`integer`)
The number of times that a monitor attempts to restart a resource if it fails. If the `Retry_count` is exceeded, depending on the particular data service and the setting of the `Failover_mode` property, the monitor might perform one of the following actions:

- Allow the resource group to remain on the current primary node, even though the resource is in a faulted state
- Request a failover of the resource group onto a different node

This property is created by the RGM and is made available to the cluster administrator only if this property 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 RTR file, the `Tunable` value for the property is `WHEN_DISABLED`.

---

**Note –** If you specify a negative value for this property, the monitor attempts to restart the resource an unlimited number of times.

---

**Category:** Conditional

**Default:** See above

**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 is available to the cluster 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 RTR file, the `Tunable` value for the property is `WHEN_DISABLED`.

**Category:** Conditional

**Default:** No default (see above)

**Tunable:** `WHEN_DISABLED`

`Scalable` (`boolean`)
Indicates whether the resource is scalable, that is, whether the resource uses the networking load-balancing features of the Sun Cluster software.

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 cannot 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. These properties can be useful in a nonscalable service as well as in a scalable service.

Using this resource property in combination with the `Failover` resource type property is described in more detail in the `r_properties`(5) man page.

**Category:** Optional

**Default:** No default

**Tunable:** `AT_CREATION`

`Start_timeout` for each callback method in the Type (`integer`)
A time lapse, in seconds, after which the RGM concludes that an invocation of this method has failed. For a given resource type, timeout properties are defined only for those methods that are declared in the RTR file.

**Category:** Conditional or Optional

**Default:** `3600` (one hour), if the method itself is declared in the RTR file

**Tunable:** `ANYTIME`

`Status` on each cluster node (`enum`)
Set by the resource monitor with the `scha_resource_setstatus` command or the `scha_resource_setstatus()` function. 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:** `NONE`

`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:** `NONE`

`Stop_timeout` for each callback method in the Type (`integer`)
A time lapse, in seconds, after which the RGM concludes that an invocation of this method has failed. For a given resource type, timeout properties are defined only for those methods that are declared in the RTR file.

**Category:** Conditional or Optional

**Default:** `3600` (one hour), if the method itself is declared in the RTR file

**Tunable:** `ANYTIME`

`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 is available to the cluster 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 RTR file, the `Tunable` value for the property is `WHEN_DISABLED`.

**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:** `NONE`

`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` property 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 this property is set to `TRUE`, sends all UDP traffic from a given client 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`.

This property is only used for scalable services.

**Category:** Optional

**Default:** No default

**Tunable:** `WHEN_DISABLED`

`Update_timeout` for each callback method in the Type (`integer`)
A time lapse, in seconds, after which the RGM concludes that an invocation of this method has failed. For a given resource type, timeout properties are defined only for those methods that are declared in the RTR file.

**Category:** Conditional or Optional

**Default:** `3600` (one hour), if the method itself is declared in the RTR file

**Tunable:** `ANYTIME`

`Validate_timeout` for each callback method in the Type (`integer`)
A time lapse, in seconds, after which the RGM concludes that an invocation of this method has failed. For a given resource type, timeout properties are defined only for those methods that are declared in the RTR file.

**Category:** Conditional or Optional

**Default:** `3600` (one hour), if the method itself is declared in the RTR file

**Tunable:** `ANYTIME`

`Weak_affinity` (`boolean`)
If this property is set to `TRUE`, this property enables 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 the cluster administrator performed 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 the Sun Cluster software. The property values are categorized as follows:

- **Required**. The cluster administrator must specify a value when creating a resource group with an administrative utility.
- **Optional**. If the cluster 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.

Property names are shown first, followed by a description.

`Auto_start_on_new_cluster` (`boolean`)
This property controls whether the Resource Group Manager (RGM) starts the resource group automatically when a new cluster is forming. The default is `TRUE`.

If set to `TRUE`, the RGM attempts to start the resource group automatically to achieve `Desired_primaries` when all the nodes of the cluster are simultaneously rebooted.

If set to `FALSE`, the resource group does not start automatically when the cluster is rebooted. The resource group remains offline until the first time that the resource group is manually switched online by using the `scswitch` command or the equivalent GUI instruction. After that, the resource group resumes normal failover behavior.

**Category:** Optional

**Default:** `TRUE`

**Tunable:** `ANYTIME`

`Desired_primaries` (`integer`)
The preferred number of nodes that the group can run on simultaneously.

The default is 1. 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:** `ANYTIME`

`Failback` (`boolean`)
A Boolean value that indicates whether to recalculate the set of nodes on which 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:** `ANYTIME`

`Global_resources_used` (`string_array`)
Indicates whether cluster file systems are used by any resource in this resource group. Legal values that the cluster 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:** `ANYTIME`

`Implicit_network_dependencies` (`boolean`)
A Boolean value that indicates, when `TRUE`, that the RGM should enforce implicit strong dependencies of nonnetwork address resources on network address resources within the group. This means that the RGM starts all network address resources before all other resources and stops network address resources after all other 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:** `ANYTIME`

`Maximum_primaries` (`integer`)
The maximum number of nodes on which the group might be online at the same time.

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:** `ANYTIME`

`Nodelist` (`string_array`)
A list of cluster nodes on which 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 in arbitrary order

**Tunable:** `ANYTIME`

`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:** `ANYTIME`

`Pingpong_interval` (`integer`)
A nonnegative integer value (in seconds) that is used by the RGM to determine where to bring the resource group online in these instances:

- In the event of a reconfiguration
- As the result of the execution of a `scha_control GIVEOVER` command or function

In the event of a reconfiguration, the resource group might fail more than once to come online within the past `Pingpong_interval` seconds on a particular node. This failure occurs because the resource's `Start` or `Prenet_start` method exited with a nonzero status or timed out. As a result, that node is considered ineligible to host the resource group, and the RGM looks for another master.

If a `scha_control` command or `scha_control GIVEOVER` command is executed on a given node by a resource, thereby causing its resource group to fail over to another node, the first node (on which `scha_control` was run) cannot be the destination of another `scha_control GIVEOVER` by the same resource until `Pingpong_interval` seconds have elapsed.

**Category:** Optional

**Default:** `3600` (one hour)

**Tunable:** `ANYTIME`

`Resource_list` (`string_array`)
The list of resources that are contained in the group. The cluster administrator does not set this property directly. Rather, the RGM updates this property as the cluster administrator adds or removes resources from the resource group.

**Category:** Query-only

**Default:** No default

**Tunable:** NONE

`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 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 Chapter 2.

**Category:** Optional

**Default:** The empty string

**Tunable:** ANYTIME

`RG_dependencies` (`string_array`)
Optional list of resource groups that indicates 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, that is, 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. The `scswitch`(1M) and `scsetup`(1M) man pages 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:** `ANYTIME`

`RG_description` (`string`)
A brief description of the resource group.

**Category:** Optional

**Default:** The empty string

**Tunable:** `ANYTIME`

`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:** `NONE`

`RG_mode` (`enum`)
Indicates whether the resource group is a failover or a 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 be set to a value that is 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:** `NONE`

RG_name (string)
The name of the resource group. This property is required and must be unique within the cluster.

**Category:** Required

**Default:** No default

**Tunable:** NONE

RG_project_name (string)
The Solaris project name (see the projects(1) man page) 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 starts the related processes under this project name for resources that do not have the Resource_project_name property set (see the r_properties(5) man page). The specified project name must exist in the projects database (see the projects(1) man page and *System Administration Guide: Solaris Containers-Resource Management and Solaris Zones*).

This property is only supported starting in Solaris 9.

---

**Note –** Changes to this property take affect the next time that 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, 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 running the scswitch command or by using the equivalent scsetup or SunPlex Manager commands. A group can exist in an UNMANAGED state when that group is not under the control of the RGM.

The following descriptions summarize each state.

---

**Note –** States apply to individual nodes only, except the UNMANAGED state, which applies across all nodes. For example, a resource group might be OFFLINE on node A, but PENDING_ONLINE on node B.

---

| | |
|---|---|
| UNMANAGED | The initial state of a newly created resource group, or the state of a previously managed resource group. Either `Init` methods have not yet been run on resources in the group, or `Fini` methods have been run on resources in the group.<br><br>The group is not managed by the RGM. |
| ONLINE | The resource group has been started on the node. In other words, the starting methods `Prenet_start`, `Start`, and `Monitor_start`, as applicable to each resource, have executed successfully on all enabled resources in the group. |
| OFFLINE | The resource group has been stopped on the node. In other words, the stopping methods `Monitor_stop`, `Stop`, and `Postnet_stop`, as applicable to each resource, have executed successfully on all enabled resources in the group. This state also applies before a resource group has started for the first time on the node. |
| PENDING_ONLINE | The resource group is starting on the node. The starting methods `Prenet_start`, `Start`, and `Monitor_start`, as applicable to each resource, are being executed on enabled resources in the group. |
| PENDING_OFFLINE | The resource group is stopping on the node. The stopping methods `Monitor_stop`, `Stop`, and `Postnet_stop`, as applicable to each resource, are being executed on enabled resources in the group. |
| ERROR_STOP_FAILED | One or more resources within the resource group failed to stop successfully and are in the `Stop_failed` state. Other resources in the group might remain online or offline. This resource group is not permitted to start on any node until the `ERROR_STOP_FAILED` state is cleared. |

| | |
|---|---|
| | You must use an administrative command, such as `scswitch -c`, to manually kill the `Stop_failed` resource and reset its state to `OFFLINE`. |
| `ONLINE_FAULTED` | The resource group was `PENDING_ONLINE` and has finished starting on this node. However, one or more resources ended up in the `Start_failed` state or with `Faulted` status. |
| `PENDING_ONLINE_BLOCKED` | The resource group failed to start fully because one or more resources within that resource group have an unsatisfied strong resource dependency on a resource in a different resource group. Such resources remain `OFFLINE`. When the resource dependencies are satisfied, the resource group automatically moves back to the `PENDING_ONLINE` state. |

**Category:** Query-only

**Default:** No default

**Tunable:** `NONE`

`RG_system` (`boolean`)

If the `RG_system` property is `TRUE` for a resource group, 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 the `scrgadm` and `scswitch` 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.

| Operation | Example |
|---|---|
| Delete a resource group | **`scrgadm -r -g RG1`** |
| Edit a resource group property (except for `RG_system`) | **`scrgadm -c -t RG1 -y nodelist=...`** |
| Add a resource to a resource group | **`scrgadm -a -j R1 -g RG1`** |

| Operation | Example |
|---|---|
| Delete a resource from a resource group | **scrgadm -r -j R1 -g RG1** |
| Edit a property of a resource that belongs to a resource group | **scrgadm -c -j R1** |
| Switch a resource group offline | **scswitch -F -g RG1** |
| Manage a resource group | **scswitch -o -g RG1** |
| Unmanage a resource group | **scswitch -u -g RG1** |
| Enable a resource | **scswitch -e -j R1** |
| Enable monitoring for a resource | **scswitch -e -M -j R1** |
| Disable a resource | **scswitch -n -j R1** |
| Disable monitoring for a resource | **scswitch -n -M -j R1** |

If the `RG_system` property is `TRUE` for a resource group, the only property of the resource group that you can edit is the `RG_system` property itself. In other words, editing the `RG_system` property is never restricted.

**Category:** Optional

**Default:** `FALSE`

**Tunable:** `ANYTIME`

# Resource Property Attributes

This section describes the resource property attributes that you can use to change system-defined properties or to 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 that are permitted.

`Array_minsize`
For `stringarray` type, the minimum number of array elements that are 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 that is permitted for the property.

`Maxlength`
For `string` and `stringarray` types, the maximum string length that is permitted.

`Min`
For an `int` type, the minimal value that is permitted for the property.

`Minlength`
For `string` and `stringarray` types, the minimum string length that is permitted.

`Property`
The name of the resource property.

`Tunable`
Indicates when the cluster administrator can set the value of this property in a resource. Set to `NONE` or `FALSE` to prevent the cluster administrator from setting the property. Values that enable a cluster administrator to tune a property are `TRUE` or `ANYTIME` (at any time), `AT_CREATION` (only when the resource is created), or `WHEN_DISABLED` (when the resource is disabled). To establish other 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 entry. 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`, `integer`, `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 these rules:

- Names must be in ASCII.
- Names must start with a letter.
- Names can contain uppercase and lowercase letters, digits, dashes (-), and underscores (_).
- The maximum number of characters that you can use in a name is 255.

# 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*`:`*rt-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 *rt-version*.

The variable elements in this format are as follows:

*vendor-id* — Specifies the vendor ID prefix, which is the value of the `Vendor_id` resource type property in the RTR file. If you are developing a resource type, choose a vendor ID prefix that uniquely identifies the vendor, such as your company's stock ticker symbol. For example, the vendor ID prefix of resource types that are developed by Sun Microsystems, Inc. is `SUNW`.

*base-rt-name* — Specifies the base resource type name, which is the value of the `Resource_type` resource type property in the RTR file.

*rt-version* — Specifies the version suffix, which 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 the Sun Cluster product.

---

**Note –** If only one version of a base resource type name is registered, you do not have to use the complete name in `scrgadm` commands. You can omit the vendor ID prefix, the version number suffix, or both.

---

For more information, see "Resource Type Properties" on page 123.

**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`

**EXAMPLE B–1** Complete Name of a Resource Type With the `#$upgrade` Directive *(Continued)*

The complete name of the resource type that is defined by this RTR file is as follows:

```
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:

- 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 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. For worksheets for other components of your cluster configuration. see Appendix A, "Sun Cluster Installation and Configuration Worksheets," in *Sun Cluster Software Installation Guide for Solaris OS*.

If necessary, make additional copies of a worksheet to accommodate all the resource-related components in your cluster configuration. To complete these worksheets, follow the planning guidelines in *Sun Cluster Software Installation Guide for Solaris OS* and Chapter 1. Then refer to your completed worksheets during cluster installation and configuration.

**Note –** The data that is used in the worksheet examples is intended as a guide only. The examples do not represent a complete configuration of a functional cluster.

## Configuration Worksheets

This appendix contains the following worksheets.

# Resource Types Worksheet

Use this worksheet for resource types other than logical host or shared address.

| Resource Type Name | Nodes on Which the Resource Type Runs |
|---|---|
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |

**EXAMPLE C–1** 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

| 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 C–2** 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 C–3** 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

| Component | Name | |
|---|---|---|
| Resource name | | |
| Resource group name | | |
| Resource type name | | |
| Dependencies | | |
| Extension Properties | **Name** | **Value** |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |

**EXAMPLE C–4** 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

| 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 C–5** 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

| 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 can 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 on 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 C–6** 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 can 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`<br>`2) phys-schost-2` |
| Dependent disk device groups | List the disk device groups on 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

| 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 can 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 on which this resource depends. | |

**EXAMPLE C–7** 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 can 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`<br>`2) phys-schost-2` |
| Dependencies | List any resource groups on which this resource depends. | `rg-shared` |

# Index

## D



## E



## F

## G



## H



## I



## L



## M

## N



## O



## P

## R

## S