Cluster Server 7.4.1 Bundled Agents Reference Guide - Solaris

VERITAS™

Last updated: 2019-02-07

Legal Notice

THE DOCUMENTATION IS PROVIDED "AS IS" AND ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT, ARE DISCLAIMED, EXCEPT TO THE EXTENT THAT SUCH DISCLAIMERS ARE HELD TO BE LEGALLY INVALID. VERITAS TECHNOLOGIES LLC SHALL NOT BE LIABLE FOR INCIDENTAL OR CONSEQUENTIAL DAMAGES IN CONNECTION WITH THE FURNISHING, PERFORMANCE, OR USE OF THIS DOCUMENTATION. THE INFORMATION CONTAINED IN THIS DOCUMENTATION IS SUBJECT TO CHANGE WITHOUT NOTICE.

The Licensed Software and Documentation are deemed to be commercial computer software as defined in FAR 12.212 and subject to restricted rights as defined in FAR Section 52.227-19 "Commercial Computer Software - Restricted Rights" and DFARS 227.7202, et seq. "Commercial Computer Software and Commercial Computer Software Documentation," as applicable, and any successor regulations, whether delivered by Veritas as on premises or hosted services. Any use, modification, reproduction release, performance, display or disclosure of the Licensed Software and Documentation by the U.S. Government shall be solely in accordance with the terms of this Agreement.

Veritas Technologies LLC
500 E Middlefield Road
Mountain View, CA 94043

http://www.veritas.com

Technical Support

Technical Support maintains support centers globally. All support services will be delivered in accordance with your support agreement and the then-current enterprise technical support policies. For information about our support offerings and how to contact Technical Support, visit our website:

https://www.veritas.com/support

You can manage your Veritas account information at the following URL:

https://my.veritas.com

If you have questions regarding an existing support agreement, please email the support agreement administration team for your region as follows:

Worldwide (except Japan)	CustomerCare@veritas.com
Japan	CustomerCare_Japan@veritas.com

Documentation

Make sure that you have the current version of the documentation. Each document displays the date of the last update on page 2. The latest documentation is available on the Veritas website:

https://sort.veritas.com/documents

Documentation feedback

Your feedback is important to us. Suggest improvements or report errors or omissions to the documentation. Include the document title, document version, chapter title, and section title of the text on which you are reporting. Send feedback to:

infoscaledocs@veritas.com

You can also see documentation information or ask a question on the Veritas community site:

http://www.veritas.com/community/

Veritas Services and Operations Readiness Tools (SORT)

Veritas Services and Operations Readiness Tools (SORT) is a website that provides information and tools to automate and simplify certain time-consuming administrative tasks. Depending on the product, SORT helps you prepare for installations and upgrades, identify risks in your datacenters, and improve operational efficiency. To see what services and tools SORT provides for your product, see the data sheet:

https://sort.veritas.com/data/support/SORT_Data_Sheet.pdf

Contents

Chapter 1

Introducing Bundled agents

This chapter includes the following topics:

About Bundled agents

Bundled agents are Cluster Server (VCS) processes that manage resources of predefined resource types according to commands received from the VCS engine, HAD. You install these agents when you install VCS.

A node has one agent per resource type that monitors all resources of that type. For example, a single IP agent manages all IP resources.

When the agent starts, it obtains the necessary configuration information from VCS. The agent then periodically monitors the resources, and updates VCS with the resource status.

Agents can:

- Bring resources online.
- Take resources offline.
- Monitor resources and report state changes.

For a more detailed overview of how agents work, refer to the *Cluster Server Administrator's Guide*.

Resources and their attributes

Resources are parts of a system. They are known by their types, for example: a volume, a disk group, or an IP address. VCS includes a set of resource types. Different attributes define these resource types in the types.cf file. Each type has a corresponding agent that controls the resource.

The VCS configuration file, main.cf, contains the values for the resource attributes and has an include directive to the types.cf file.

An attribute's given value configures the resource to function in a specific way. By modifying the value of a resource attribute, you can change the way the VCS agent manages the resource. For example, the IP agent uses the Address attribute to determine the IP address to monitor.

Modifying agents and their resources

Use the Cluster Manager (Java Console), Veritas Operations Manager, or the command line to dynamically modify the configuration of the resources managed by an agent.

VCS enables you to edit the main.cf file directly. To implement these changes, make sure to restart VCS.

See the *Cluster Server Administrator's Guide* for instructions on how to complete these tasks.

Attributes

Attributes contain data about the cluster, systems, service groups, resources, resource types, and the agent. An attribute has a definition and a value. You change attribute values to configure VCS resources. Attributes are either optional or required, although sometimes attributes that are optional in one configuration might be required in other configurations. Many optional attributes have predefined or default values, which you should change as required.

A variety of internal use only attributes also exist. Do not modify these attributes—modifying them can lead to significant problems for your clusters.

Attributes have type and dimension. Some attribute values can accept numbers, others can accept alphanumeric values or groups of alphanumeric values, while others are simple boolean on/off values.

Table 1-1 Attribute data types

Data Type	Description
string	Enclose strings, which are a sequence of characters, in double quotes ("). Optionally enclose strings in quotes when they begin with a letter, and contains only letters, numbers, dashes (-), and underscores (_). A string can contain double quotes, but the quotes must be immediately preceded by a backslash. In a string, represent a backslash with two backslashes (\\).
integer	Signed integer constants are a sequence of digits from 0 to 9. You can precede them with a dash. They are base 10. Integers cannot exceed the value of a 32-bit signed integer: 2147483647.
boolean	A boolean is an integer with the possible values of 0 (false) and 1 (true).

Table 1-2 Attribute dimensions

Dimension	Description
scalar	A scalar has only one value. This is the default dimension.
vector	A vector is an ordered list of values. Each value is indexed using a positive integer beginning with zero. A set of brackets ([]) denotes that the dimension is a vector. Find the specified brackets after the attribute name on the attribute definition in the types.cf file.
keylist	A keylist is an unordered list of unique strings.
association	An association is an unordered list of name-value pairs. An equal sign separates each pair. A set of braces ({}) denotes that an attribute is an association. Braces are specified after the attribute name on the attribute definition in the types.cf file, for example: str SnmpConsoles{}.

Zone-aware agents

Table 1-3 lists the ContainerOpts attribute default values for resource types. Veritas recommends that you do not modify these values.

Table 1-3 ContainerOpts attribute default values for applications and resource types

Resource Type	RunInContainer	PassCInfo
Application	1	0
Apache	1	0
IP	0	1
IPMultiNICB	0	1
Mount	0	0
NIC	0	1
Process	1	0
Zone	0	1
ProcessOnOnly	1	0
Project	0	1

For more information on using zones in your VCS environment, refer to the *Veritas InfoScale Virtualization Guide*.

IMF-aware agents

With Intelligent Monitoring Framework (IMF), VCS supports intelligent resource monitoring in addition to poll-based monitoring. IMF is an extension to the VCS agent framework.

The following are the IMF-aware agents:

- Apache HTTP server agent. See "Apache HTTP server agent" on page 182.
- Application agent. See "Application agent" on page 196.
- DiskGroup agent. See "DiskGroup agent" on page 21.
- Mount agent. See "Mount agent" on page 56.
- Process agent. See "Process agent" on page 220.
- Zone agent. See "Zone agent" on page 227.
- LDom agent. See "LDom agent" on page 234.

Enabling debug log messages

To help troubleshoot agent issues, you can enable debug log messages in the agent framework as well as the agents.

To enable agent framework debug log messages:

```
# hatype -modify agent_name LogDbg -add DBG_AGDEBUG DBG_AGINFO
DBG_AGTRACE
```

For example:

```
# hatype -modify Mount LogDbg -add DBG_AGDEBUG DBG_AGINFO DBG_AGTRACE
```

To enable agent-specific debug log messages:

```
# hatype -modify agent_name LogDbg -add debug_log_levels
```

For example:

```
# hatype -modify Mount LogDbg -add DBG_1 DBG_2 DBG_3 DBG_4 DBG_5 DBG_6
```

Alternatively, you can also use the following command:

```
# hatype -modify Mount LogDbg -add 1 2 3 4 5 6
```

Agent-specific debug log level information is specified in the agent’s description. For example, for information about the Mount agent, See “Debug log levels for Mount agent” on page 74.

For more information about log behavior, refer to the *VCS Administrator's Guide*.

VCS support for multi-pathing solutions

This section applies to Zpool agent only.

VCS supports Dynamic Multi-Pathing (DMP) that is included as a part of Veritas InfoScale. Veritas does not support multi-pathing solutions that are not explicitly listed in the hardware compatibility list (HCL). You can find the HCL on the SORT web site, under the Documentation tab. However, Veritas supports third-party solutions, which are included as a part of the operating systems.

Veritas aims to thoroughly test and support third-party and native solutions, but it is not possible to test all third-party multi-pathing applications. This is because of complex support matrix and a number of potential product combinations. Hence, Veritas does not officially support multi-pathing solutions that are not explicitly listed in the HCL. Also, advanced functionality such as I/O fencing with SCSI3-PGR is only supported with arrays and multi-pathing solutions listed in the HCL and only with Veritas InfoScale.

If you are using a third-party multi-pathing solution, Veritas understands your need of keeping data paths redundant and does not insist that you uninstall or disable the solution. Veritas does not consider third-party multi-pathing solutions as invalid and continues to troubleshoot any support issues. However, for persisting support issues related to multi-pathing solutions, you need to contact the multi-pathing vendor.

Chapter 2

Storage agents

This chapter includes the following topics:

About the storage agents

Storage agents monitor shared storage and make shared storage highly available. Storage includes shared disks, disk groups, volumes, and mounts.

DiskGroup agent

The DiskGroup agent brings online, takes offline, and monitors Veritas Volume Manager (VxVM) private disk groups. This agent uses VxVM commands to determine the state of disk groups. You can use this agent to monitor or make private disk groups highly available.

Note: The private disk group should not be configured in a parallel service group if the disk group is configured on the same shared disk across nodes.

For important information on this agent, See "Notes for DiskGroup agent" on page 30.

IMF awareness

The DiskGroup agent is Intelligent Monitoring Framework (IMF)-aware and uses Asynchronous Monitoring Framework (AMF) kernel driver for IMF notification. For more information about IMF and intelligent resource monitoring, refer to the *Cluster Server Administrator's Guide*.

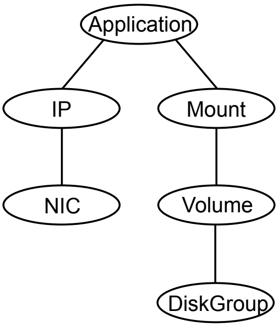
For more information about IMF-related DiskGroup agent functions, see See "Agent functions for DiskGroup agent" on page 22.

For more information about using the DiskGroup agent with IMF, see See "Using the DiskGroup agent with IMF" on page 32.

Dependencies for DiskGroup agent

The DiskGroup resource does not depend on any other resources.

Figure 2-1 Sample service group that includes a DiskGroup resource



Agent functions for DiskGroup agent

Online	Imports the disk group using the `vxdg` command.
Offline	Deports the disk group using the `vxdg` command.

Monitor

Determines if the disk group is online or offline using the `vxdg` command. The Monitor function sets the VxVM noautoimport flag. This action allows VCS to maintain control of importing the disk group. The monitor function uses the following command to set the noautoimport flag:

```
# vxdg -g disk_group set autoimport=no
```

If IMF is enabled for the DiskGroup agent, the resource is monitored asynchronously and any change in the disk group state is immediately sent to the DiskGroup agent for appropriate action.

Clean

Terminates all ongoing resource actions and takes the resource offline—forcibly when necessary.

Info

The Info function gets information from the Volume Manager and displays the type and free size for the DiskGroup resource.

Initiate the Info function by setting the InfoInterval timing to a value greater than 0.

In the following example, the Info function executes every 60 seconds:

```
# haconf -makerw
```

```
# hatype -modify DiskGroup InfoInterval 60
```

Enter the following command to retrieve information about the DiskType and FreeSize of the DiskGroup resource:

```
# hares -value diskgroupres ResourceInfo
```

Output includes:

```
DiskType sliced
FreeSize 35354136
```

The value specified is in kilo bytes.

Action	Different actions follow: ■ license.vfd Checks for valid Veritas Volume manager license–if one is not found use the vxlicinst utility to install a valid license key. ■ disk.vfd Checks if all disks in diskgroup are visible on host—if it fails, check if the path to disks exists from the host and check if LUN masking and zoning are set properly. ■ udid.vfd Checks the UDID (unique disk identifiers) of disks on the cluster nodes—if it fails, ensure that the disks that are used for the disk group are the same on all cluster nodes. ■ verifyplex.vfd Checks if the number of plexes on each site for the Campus Cluster setup are set properly—if it fails, check that the sites, disks, and plexes are set properly for a Campus Cluster setup. ■ volinuse Checks if open volumes are in use or file systems on volumes that are mounted outside of VCS configuration. See “High availability fire drill” on page 31.
imf_init	Initializes the agent to interface with Intelligent monitoring framework (IMF). The function runs when the agent starts up.
imf_getnotification	Waits for notification about disk group state changes. The function runs after the agent initializes with IMF. The function waits for notification. Upon receiving notification, the agent takes action on the resource.
imf_register	Registers the resource entities, which the agent must monitor using IMF. The function runs for each resource after the resource goes into a steady state, either online or offline.

State definitions for DiskGroup agent

ONLINE	Indicates that the disk group is imported.
OFFLINE	Indicates that the disk group is not imported.
FAULTED	Indicates that the disk group has unexpectedly deported or become disabled.

UNKNOWN

Indicates that a problem exists either with the configuration or the ability to determine the status of the resource. One cause of this state is when I/O fencing is not configured—the cluster level attribute UseFence is not set to "SCSI3" but the Reservation attribute value is "SCSI3".

Attributes for DiskGroup agent

Table 2-1 Required attributes

Required attribute	Description
DiskGroup	Name of the disk group that is configured with Veritas Volume Manager. Type and dimension: string-scalar

Table 2-2 Optional attributes

Optional attributes	Description
MonitorReservation	If the value is 1 and SCSI-3 fencing is used, the agent monitors the SCSI reservation on the disk group. If the reservation is missing, the Monitor agent function takes the resource offline. Type and dimension: boolean-scalar Default: 0 **Note:** If the MonitorReservation attribute is set to 0, and the value of the cluster-wide attribute UseFence is set to SCSI3, and the disk group is imported without SCSI reservation, then the monitor agent function takes the service group containing the disk group resource offline.

Table 2-2 Optional attributes *(continued)*

Optional attributes	Description
PanicSystemOnDGLoss	Determines whether to panic the node if the disk group becomes disabled or monitor operation times out. A loss of storage connectivity can cause the disk group to become disabled. VxVM commands not responding properly can cause monitor operation to timeout. **Note:** System administrators may want to set a high value for FaultOnMonitorTimeout to increase system tolerance. This attribute accepts the following values: ■ 0: Do not halt the system ■ 1: halt the system if either disk group goes into disabled state or the disk group resource faults due to monitor timeout ■ 2: halt the system only if disk group goes into disabled state ■ 3: halt the system only if disk group resource faults due to monitor timeout If the value of the attribute is 0, and the disk group becomes disabled, the following occurs: ■ If the cluster has I/O fencing enabled (Cluster level attribute UseFence set to SCSI3), the monitor operation on the DiskGroup resource is reported OFFLINE. This results in the agent to call clean to take the DiskGroup resource offline. As part of bringing the DiskGroup resource offline, the agent attempts to deport the disabled disk group. The clean operation succeeds even if the disabled disk group fails to deport and the DiskGroup resource enters into FAULTED state. This state enables the failover of the service group that contains the DiskGroup resource. To fail back the service group containing the DiskGroup resource, manually deport the disk group after restoring storage connectivity on the failed node. ■ If the cluster does not use I/O fencing, a message is logged and the resource is reported ONLINE. The resource is reported ONLINE so that it does not fail over, which ensures data integrity.

Table 2-2 Optional attributes *(continued)*

Optional attributes	Description
PanicSystemOnDGLoss (Continued)	Notes: - The PanicSystemOnDGLoss attribute does not depend on the MonitorReservation attribute. - Veritas recommends to configure volume resource for each volume in the diskgroup to detect storage connectivity loss more effectively. - If PanicSystemOnDGLoss is set to non-zero value, the system panic is initiated using `uadmin 5 0` command. This command halts the system. An administrator needs to bring up the system. Type and dimension: integer-scalar Default: 0
StartVolumes	If value of this attribute is 1, the DiskGroup online function starts all volumes belonging to that disk group after importing the group. **Note:** If the Veritas Volume Manager default autostartvolumes at system level is set to on, all the volumes of the disk group is started as a part of the import disk group. Type and dimension: boolean-scalar Default: 1
StopVolumes	If value is 1, the DiskGroup offline function stops all volumes belonging to that disk group before it deports the disk group. Type and dimension: boolean-scalar Default: 1

Table 2-2 Optional attributes *(continued)*

Optional attributes	Description
UmountVolumes	This attribute enables the DiskGroup resource to forcefully go offline even if open volumes are mounted outside of VCS control. When the value of this attribute is 1 and the disk group has open volumes, the following occurs: ■ The agent attempts to unmount the file systems on open volumes. If required, the agent attempts to kill all VCS managed and un-managed applications using the file systems on those open volumes. ■ The agent attempts to forcefully unmount the file systems to close the volumes. Type and dimension: integer-scalar Default: 0
Reservation	Determines if you want to enable SCSI-3 reservation. This attribute can have one of the following three values: ■ ClusterDefault—The disk group is imported with SCSI-3 reservation if the value of the cluster-level UseFence attribute is SCSI3. If the value of the cluster-level UseFence attribute is NONE, the disk group is imported without reservation. ■ SCSI3—The disk group is imported with SCSI-3 reservation if the value of the cluster-level UseFence attribute is SCSI3. ■ NONE—The disk group is imported without SCSI-3 reservation. To import a disk group with SCSI-3 reservations, ensure that the disks of the disk group are SCSI-3 persistent reservation (PR) compliant. Type and dimension: string-scalar Default: ClusterDefault Example: "SCSI3"
ForceImport	Enables and disables the forcible import of a disk group. Type and dimension: boolean-scalar Default: 1

Table 2-2 Optional attributes *(continued)*

Optional attributes	Description
ClearClone	If the value of this attribute is 1, the disk group is imported with the '-c' option. While importing the disk group, this option clears the "clone" and "udid_mismatch" flags from the disks of the disk group and also updates the UDID, if required. For more information about the '-c' option, refer to the *VxVM manual page*. **Note:** For hardware cloning devices, do not set this attribute to 1. Type and dimension: boolean-scalar Default: 0
DGOptions	Specifies the options for the `vxdg` import command. The agent uses this attribute only while bringing a DiskGroup resource online. For more information, see the *vxdg(1m) manual page*. Type and dimension: string-scalar Example: "-o noautostart -o updateid"

Table 2-3 Internal attributes

Attribute	Description
tempUseFence	This attribute is reserved for internal use only.
NumThreads	The number of threads that are used within the agent process for managing resources. This number does not include the number of threads that are used for other internal purposes. Setting the NumThreads attribute to a higher value may decrease the time required to go online or the time required to monitor a large number of DiskGroup resources. Type and dimension: static integer-scalar Default: 1 **Note:** If there are many DiskGroup resources and if the resources are taking more time to come online, consider increasing the NumThreads attribute to a value greater than 1.

Resource type definition for DiskGroup agent

The resource definition for this agent on Solaris follows:

```
type DiskGroup (
 static keylist SupportedActions = { "license.vfd", "disk.vfd", "udid.vfd",
 "verifyplex.vfd", "checkudid", "campusplex", "numdisks", "joindg",
 "splitdg", "getvxvminfo", "volinuse" }
 static int OnlineRetryLimit = 1
 static str ArgList[] = { DiskGroup, StartVolumes, StopVolumes, MonitorOnly,
 MonitorReservation, tempUseFence, PanicSystemOnDGLoss, UmountVolumes,
 Reservation, ConfidenceLevel, ClearClone, DGOptions, ForceImport}
 str DiskGroup
    static str IMFRegList[] = { DiskGroup, Reservation }
    static int IMF{} = { Mode = 3, MonitorFreq = 5, RegisterRetryLimit = 3 }
 boolean StartVolumes = 1
 boolean StopVolumes = 1
 static int NumThreads = 1
 boolean MonitorReservation = 0
 temp str tempUseFence = INVALID
 int PanicSystemOnDGLoss = 0
 int UmountVolumes = 0
 str Reservation = ClusterDefault
 boolean ClearClone = 0
 str DGOptions
 boolean ForceImport = 1
```

Notes for DiskGroup agent

The DiskGroup agent has the following notes:

- High availability fire drill
- Using volume sets
- Setting the noautoimport flag for a disk group
- Configuring the Fiber Channel adapter
- Using the DiskGroup agent with IMF
- Using the DiskGroup agent with NFS

High availability fire drill

The high availability fire drill detects discrepancies between the VCS configuration and the underlying infrastructure on a node. These discrepancies might prevent a service group from going online on a specific node.

For DiskGroup resources, the high availability fire drill checks for:

- The Veritas Volume Manager license
- Visibility from host for all disks in the disk group
- The same disks for the disk group on cluster nodes
- Equal number of plexes on all sites for the disk group in a campus cluster setup

For more information about using the high availability fire drill see the *Cluster Server Administrator's Guide*.

Using volume sets

When you use a volume set, set StartVolumes and StopVolumes attributes of the DiskGroup resource that contains a volume set to 1. If a file system is created on the volume set, use a Mount resource to mount the volume set.

See the Mount agent description for more information.

Setting the noautoimport flag for a disk group

VCS requires that the noautoimport flag of an imported disk group be explicitly set to true. This value enables VCS to control the importation and deportation of disk groups as needed when bringing disk groups online and taking them offline.

To check the status of the noautoimport flag for an imported disk group

◆
```
# vxprint -l disk_group | grep noautoimport
```

If the output from this command is blank, the noautoimport flag is set to false and VCS lacks the necessary control.

The Monitor function changes the value of the VxVM noautoimport flag from off to on. It changes the value instead of taking the service group offline. This action allows VCS to maintain control of importing the disk group.

The following command changes the autoimport flag to false:

```
# vxdg -g disk_group set autoimport=no
```

VxVM noautoimport flag usage

The Monitor function changes the value of the VxVM noautoimport flag from off to on. It changes the value instead of taking the service group offline. This action allows VCS to maintain control of importing the disk group.

The following command changes the autoimport flag to false:

```
# vxdg -g disk_group set autoimport=no
```

Configuring the Fiber Channel adapter

Most Fiber Channel (FC) drivers have a configurable parameter called "failover". This configurable parameter is in the FC driver’s configuration file. This parameter is the number of seconds that the driver waits before it transitions a disk target from OFFLINE to FAILED. After the state becomes FAILED, the driver flushes all pending fiber channel commands back to the application with an error code. Veritas recommends that you use a non-zero value that is smaller than any of the MonitorTimeout values of the Disk Group resources. Use this value to avoid excessive waits for monitor timeouts.

For more information, refer to the following guides:

- *Dynamic Multi-Pathing Administrator's Guide*
- *Fiber Channel Adapter's Configuration Guide*

Using the DiskGroup agent with IMF

Considerations to use the DiskGroup agent with IMF:

- You can either set the MonitorFreq to 0 or a high value. Setting the value of the MonitorFreq key to a high value ensures that the agent does not run the monitor function frequently. Setting the MonitorFreq key to 0 disables the traditional monitoring while IMF monitoring is in progress. Traditional monitoring is done after receiving the notification for a resource.
  However, if the disk group is configured with reservation and value of the MonitorReservation attribute is set to 1, then set the MonitorFreq key value to the frequency at which you want the agent to run the monitor function, to verify the reservation on the disk group.

Using the DiskGroup agent with NFS

If the file systems on the VxVM volumes are shared using NFS, you must make sure that the major number of all the volumes across the cluster nodes are consistent. By making the vxio driver numbers consistent for all nodes in a VCS cluster, it makes volume major numbers consistent on all the cluster nodes.

NFS clients know the major and minor numbers of the block device containing the file system exported by the NFS server, so when making the NFS server highly available, it is important to make sure that all nodes in the cluster that can act as NFS servers have the same major and minor numbers for the volume block device.

To determine the current value assigned to the vxio and vxspec drivers, enter:

```
# grep '^vx' /etc/name_to_major
```

The following output is displayed:

```
..
vxio 327
vxspec 328
..
```

To determine the major numbers available on the system, check the `/etc/name_to_major` file and use the unassigned numbers.

To reassign the major numbers to the vxio and vxspec drivers, enter:

```
# haremajor -vx major-number-vxio major-number-vxspec
```

For example:

```
# haremajor -vx 338 339

haremajor 1.1

Using the following major number(s):
  338
  339
Do you want to continue [y/n]? y

Updating /etc/name_to_major

If there are any problems, you can backout the changes by restoring the
following files:

- /etc/name_to_major.off.3409

To complete re-majoring, reboot your machine with the following command:
reboot
```

Note: If you assign a major number, you must reboot the node.

For more information, refer to the `haremajor` command manual page.

Sample configurations for DiskGroup agent

DiskGroup resource configuration

Sample configuration of the DiskGroup resource:

```
DiskGroup dg1 (
    DiskGroup = testdg_1
)
```

Debug log levels for DiskGroup agent

The DiskGroup agent uses the following debug log levels:

DBG_1, DBG_3, DBG_4

DiskGroupSnap agent

Use the DiskGroupSnap agent to perform fire drills in a campus cluster. The DiskGroupSnap agent enables you to verify the configuration and data integrity in a Campus Cluster environment with VxVM stretch mirroring. The agent also supports SCSI-3 fencing.

Note: The DiskGroupSnap agent requires the Global Cluster Option (GCO) license enabled on all systems in the cluster.

For more information on fire drills, refer to the *Cluster Server Administrator’s Guide*.

You must define the DiskGroupSnap agent in a separate FireDrill service group which is similar to the Application service group. The FireDrill service group might contain resources similar to the Application service group, for example Mount, Application, and so on.

The FireDrill service group must also contain a resource of type DiskGroupSnap such that the Mount resource depends on the DiskGroupSnap resource. The main DiskGroup must contain multiple sites registered in it with the value of the "siteconsistent" attribute set to on.

When the DiskGroupSnap agent goes online, the agent detaches one of the sites from the main DiskGroup and imports the detached site on the fire drill host as an independent DiskGroup with a different name. The volumes on the DiskGroup are also imported and mounted with same names on the fire drill host.

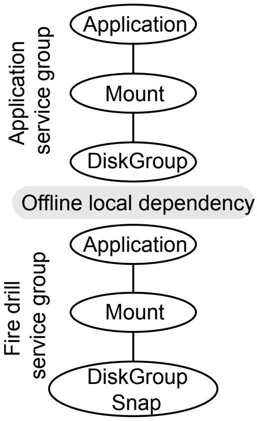
The DiskGroupSnap agent provides Gold and Bronze configurations for the fire drill, which can be specified using the agent's FDType attribute. The configuration decides the site to be detached from the DiskGroup for fire drill testing. The Gold configuration is the default option in which the agent selects a site from the DiskGroup that is neither the local VxVM site nor the site on which the DiskGroup is online. With the Gold configuration, you can also specify an alternate site to detach through the agent's FDSiteName attribute. With the Bronze configuration, the agent uses the local VxVM site name as the site to detach from the DiskGroup.

For important information about this agent, See "Notes for DiskGroupSnap agent" on page 38.

Dependencies for DiskGroupSnap agent

The DiskGroupSnap resource does not depend on any other resources. The service group that contains the DiskGroupSnap agent's resource has an offline local dependency on the application's service group. The offline local dependency is to make sure the firedrill service group and the application service group are not online at the same site at the same time.

Figure 2-2 Sample service group that includes a DiskGroupSnap resource



Agent functions for DiskGroupSnap agent

Online	Verifies that the application's disk group is in a valid campus cluster configuration. It detaches the site that the value of the FDSiteName attribute specifies. It then creates another disk group to be used for the fire drill on the detached site. After the completion of Online function, the agent creates a lock file in the lock directory (/var/VRTSvcs/lock) to indicate that the resource is online.

Offline	This re-attaches the site that the value of the FDSiteName attribute specifies back to the application’s disk group. After the completion of Offline function the agent removes the lock file from the lock directory (/var/VRTSvcs/lock) to indicate that the resource is Offline.
Monitor	Monitors the DiskGroupSnap resource by checking the existence of the Lock file in /var/VRTSvcs/lock directory..
Clean	Takes the DiskGroupSnap resource offline.
Open	If the DiskGroupSnap resource has a parent resource that is not ONLINE, then it deletes the online lock file of the DiskGroupSnap resource. This marks the DiskGroupSnap resource as OFFLINE.

State definitions for DiskGroupSnap agent

ONLINE	The DiskGroupSnap resource functions normally.
OFFLINE	The DiskGroupSnap resource is not running.
UNKNOWN	A configuration error exists.
FAULTED	The DiskGroupSnap resource is taken offline unexpectedly outside of VCS control.

Attributes for DiskGroupSnap agent

Table 2-4 Required attributes

Required attribute	Description
TargetResName	The name of the DiskGroup resource from the application service group. Type-dimension: string-scalar Example: "dgres"

Table 2-4 Required attributes *(continued)*

Required attribute	Description
FDType	Specifies the configuration to be used for the fire drill. The possible values for this attribute are: ■ Bronze ■ Gold (default) The Bronze configuration uses the local host's VxVM site name as the site to be detached from the DiskGroup. This action leaves the DiskGroup vulnerable to site disaster since a copy of the production volume might not be available when the fire drill is in progress. In the Gold configuration there are at least three copies of the parent volume available on different sites, hence, even after detaching one site the volume is not vulnerable to site disaster while the fire drill is in progress.

Table 2-5 Optional attributes

Optional attribute	Description
FDSiteName	The unique VxVM site name tag for the fire drill disks. The value of this attribute is used in conjunction with the FDType attribute and it must be set to one of the sites registered in the main DiskGroup. ■ When FDType is set to the Bronze configuration, the value of FDSiteName should either be empty or the name of the local host VxVM site for the fire drill host. ■ When FDType is set to the Gold configuration, FDSiteName identifies a site in the DiskGroup to detach as a part of the fire drill. If FDSiteName is left blank, the agent will choose a site to detach based on the DiskGroup configuration. The agent chooses a site name from the DiskGroup which is neither the production server's site name nor the fire drill host's site name. Table 2-6 shows the possible values of the attributes FDType and FDSiteName and the decision taken by the agent.

Consider a configuration where the Production DiskGroup contains three sites: A, B, and C, and the Application service group is online on a node with local VxVM site ID is A. Fire drill is being done on another node Application service group is online on a node where local VxVM site ID is B.

Table 2-6 Example FDType configurations

FDType	Bronze			Gold/Empty		
FDSitename	Empty	B	C	Empty	B	C
Result	Use B as the site to detach and proceed	Detach site B from DiskGroup	Error	Check if there is another site other than A and B and select it. Else, it is an error	Error	Detach site C from the DiskGroup

Notes for DiskGroupSnap agent

The DiskGroupSnap agent has the following notes:

- See “Fire drill configuration after upgrading VCS” on page 38.
- See “Configuring the SystemZones attribute for the fire drill service group” on page 38.
- See “Configuring the FireDrill service group” on page 39.
- See “Adding the ReuseMntPt attribute to the ArgList attribute for the Mount agent type” on page 39.
- See “Configuration considerations” on page 40.
- See “Agent limitations” on page 41.

Fire drill configuration after upgrading VCS

After upgrading VCS from any earlier version to 6.0, delete all resources of type DiskGroupSnap and recreate them again using the new definitions of the attributes. Failure to perform this step might result in an unexpected behavior of the agent.

Configuring the SystemZones attribute for the fire drill service group

You must assign the local system values to the SystemZones attribute of the application’s service group. You set these values so that the service group fails over in the same zone before it tries to fail over across zones.

For more information about campus cluster setup, refer to the *Cluster Server Administrator’s Guide*.

For example, you set up the service group's SystemZones attribute for two zones: 0 and 1. You want the service group on Node_A and Node_B to fail over between the two nodes before it comes up on Node_C and Node_D. The application and its fire drill service group both have the following values for the SystemZones attribute:

```
SystemZones = { Node_A = 0, Node_B = 0, Node_C = 1, Node_D = 1 }
```

Configuring the FireDrill service group

In the FireDrill service group, the application-level resources (for example, process resources, application resources, or Oracle resources, and so on) can have the same attribute values in the firedrill service group and the application service group. The reuse of the same values for the attributes can result in VCS reporting the wrong resources as online.

Set the FireDrill type-level attribute to 1 for those types. For example, if the Oracle and Listener resources are configured identically, set the FireDrill attribute for Oracle and Netlsnr to 1:

```
# haconf –makerw
# hatype –modify Oracle FireDrill 1
# hatype –modify Netlsnr FireDrill 1
# haconf -dump –makero
```

Adding the ReuseMntPt attribute to the ArgList attribute for the Mount agent type

If you plan to use a Mount resource in a firedrill service group, you must add the ReuseMntPt attribute to ArgList and set its value to 1.

To add the ReuseMntPt attribute to the ArgList attribute and set its value to 1

1 Make the configuration read and write.

```
# haconf -makerw
```

2 Add the ReuseMntPt attribute to the ArgList attribute.

```
# hatype -modify Mount ArgList -add ReuseMntPt
```

3 Change the value of the ReuseMntPt attribute to 1 for the firedrill's Mount resource.

```
# hares -modify firedrill_mount_resource_name ReuseMntPt 1
```

4. Change the value of the ReuseMntPt attribute to 1 for the original Mount resource.

   ```
   # hares -modify original_mount_resource_name ReuseMntPt 1
   ```

5. Make the configuration read only.

   ```
   # haconf -dump -makero
   ```

Configuration considerations

Keep the following recommendations in mind:

- You must install Veritas Volume Manager 5.1 or later with the FMR license and the Site Awareness license.
- Do not bring the DiskGroupSnap resource online in the SystemZone where the application service group is online.
- Make sure that the firedrill service group and the application service group both use the same values for the SystemZones attribute.
- Do not use Volume resources in the firedrill service group. The DiskGroupSnap agent internally uses the `vxvol` command to start all the volumes in the firedrill disk group.
- In large setups, you may need to tweak the various timer values so that the timers do not time out while waiting for VxVM commands to complete. The timers you need to tweak are the OfflineTimeout for the DiskGroupSnap resource and MonitorInterval and ActionTimeout for the associated DiskGroup resource, for example:

  ```
  # haconf –makerw
  # hares -override dgsres OfflineTimeout
  # hares -modify dgsres OfflineTimeout 600
  # hares -override dgres MonitorInterval
  # hares -modify dgres MonitorInterval 1200 (this has to be twice
    the value intended for ActionTimeout below)
  # hares -override dgres ActionTimeout
  # hares -modify dgres ActionTimeout 600
  # haconf –dump –makero
  ```

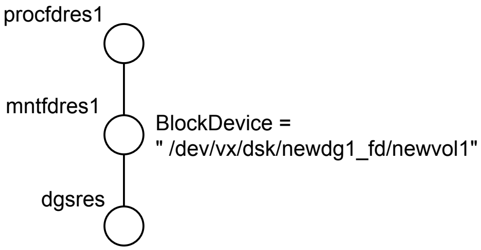
- When you create the firedrill service group, in general use the same attribute values that you use in the application service group.

The BlockDevice attribute of the Mount resource changes between the application service group and the firedrill service group. In the BlockDevice path, you must append an _fd to the disk group name portion, for example, /dev/vx/dsk/newdg1/newvol1 becomes /dev/vx/dsk/newdg1_fd/newvol1.

See Figure 2-3 on page 41. shows the changes to resource values for the firedrill service group; note that the Volume resource is not included.

- Before commencing the fire drill, make sure that all the sites registered in the application DiskGroup are in ACTIVE state.

Figure 2-3 Sample resource values for a DiskGroupSnap resource



Agent limitations

The following limitations apply to the DiskGroupSnap agent:

- The DiskGroupSnap agent does not support Volume Sets.
- The DiskGroupSnap agent cannot be used in a Storage Foundation for Oracle RAC environment.
- The online and offline operations of the DiskGroupSnap resource invokes VCS action entry points to run VxVM commands to detach/reattach the fire drill site. Since VxVM requires that these commands are run on the node where the disk group is imported, the disk group has to be imported on some node in the cluster before these operations.
- Take the firedrill service group offline before you shut down VCS on any node. If you fail to take the firedrill service group offline before you shut down VCS, you must manually reattach the fire drill site to the disk group to continue to perform fire drills.
- Use the enclosures that have the ASL/APM libraries that are supported in the Veritas Volume Manager. To view the supported enclosures, use the `vxddladm listsupport` command.
- Do not switch the Application service group when fire drill is in progress.

Resource type definition for DiskGroupSnap agent

The resource type definition for this agent follows:

```
type DiskGroupSnap (
static int ActionTimeout = 120
static int MonitorInterval = 300
static int NumThreads = 1
static str ArgList[] = { TargetResName, FDSiteName, FDType }
str TargetResName
str FDSiteName
str FDType
)
```

Sample configurations for DiskGroupSnap agent

In Figure 2-4, the Primary site is in the Bronze configuration and the Disaster recovery site is in a Gold configuration.

Since the Primary site does not have dedicated fire drill disks, it is in a Bronze configuration. In the Bronze configuration, you re-purpose the mirror disks in the disaster recovery site to serve as fire drill test disks. The drawback with the Bronze configuration is that if a disk failure occurs when the fire drill is online at the Primary site, it results in a site failure.

The FDSiteName value in a bronze configuration is the VxVM site name. For this configuration, the FDSiteName attribute values for the nodes at the Primary site follow:

```
FDSiteName@Node_A = pri
 FDSiteName@Node_B = pri
```

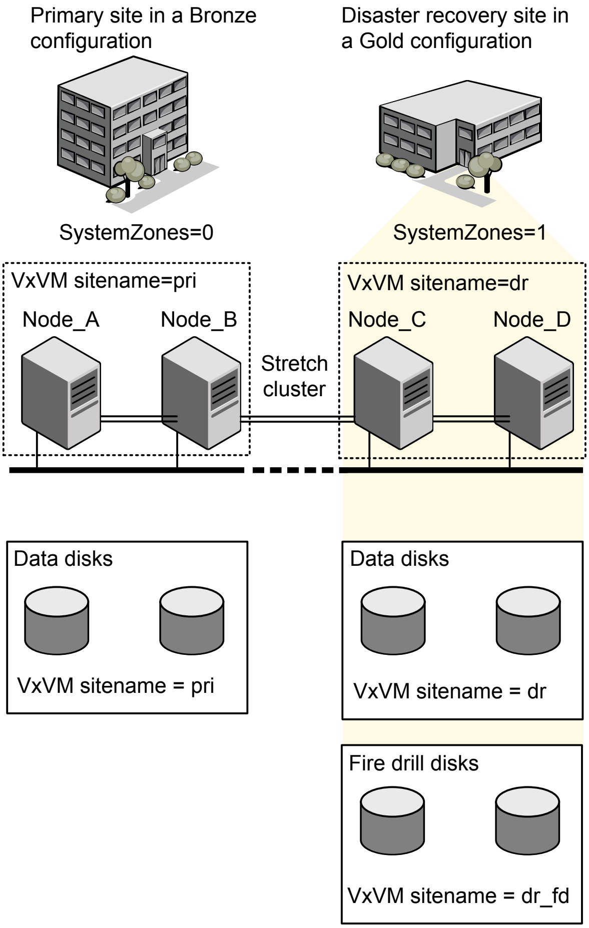
The Disaster Recovery site is in a Gold configuration as it has dedicated fire drill disks at the site. For the FDSiteName attribute, use the VxVM site tag given to the fire drill disks. For this configuration, the FDSiteName attribute values for the nodes at the Disaster recovery site follow:

```
FDSiteName@Node_C = dr_fd
 FDSiteName@Node_D = dr_fd
```

Set values for the SystemZones attribute to zero for Node_A and Node_B, and one for Node_C and Node_D. For example:

```
SystemZones = { Node_A = 0, Node_B = 0, Node_C = 1, Node_D = 1 }
```

Figure 2-4 Primary site with the Bronze configuration and the disaster recovery site with the Gold configuration



Typical main.cf configuration for DiskGroupSnap agent

The following sample configuration shows the fire drill’s service group and its corresponding application service group. The fire drill’s service group follows:

```
group dgfdsg (
    SystemList = { Node_A = 0, Node_B = 1, Node_C = 2, Node_D = 3 }
    SystemZones = { Node_A = 0, Node_B = 0, Node_C = 1, Node_D = 1}
    )
```

```
DiskGroupSnap dgsres (
TargetResName = dgres
        )

FDtype = "Gold"
        FDSiteName @Node_A = pri
        FDSiteName @Node_B = pri
        FDSiteName @Node_C = dr_fd
        FDSiteName @Node_D = dr_fd
        )

    Mount mntfdres1 (
        MountPoint = "/dgsfs1"
        BlockDevice = "/dev/vx/dsk/newdg1_fd/newvol1"
        FSType = vxfs
        FsckOpt = "-y"
        ReuseMntPt = 1
        )

    Mount mntfdres2 (
        MountPoint = "/dgsfs2"
        BlockDevice = "/dev/vx/dsk/newdg1_fd/newvol2"
        FSType = vxfs
        FsckOpt = "-y"
        ReuseMntPt = 1
        )

    Process procfdres1 (
        PathName = "/usr/bin/ksh"
        Arguments = "/scrib.sh /dgsfs1"
        )

    Process procfdres2 (
        PathName = "/usr/bin/ksh"
        Arguments = "/scrib.sh /dgsfs2"
        )

    requires group dgsg offline local

    mntfdres1 requires dgsres
    mntfdres2 requires dgsres
```

```
 procfdres1 requires mntfdres1
 procfdres2 requires mntfdres2
```

The application’s service group (the actual service group) follows:

```
group dgsg (
     SystemList = { Node_A = 0, Node_B = 1, Node_C = 2, Node_D = 3 }
     SystemZones = { Node_A = 0, Node_B = 0, Node_C = 1, Node_D = 1}
     )

 DiskGroup dgres (
     DiskGroup = newdg1
     )

 Mount mntres1 (
     MountPoint = "/dgsfs1"
     BlockDevice = "/dev/vx/dsk/newdg1/newvol1"
     FSType = vxfs
     FsckOpt = "-y"
     ReuseMntPt = 1
     )

 Mount mntres2 (
     MountPoint = "/dgsfs2"
     BlockDevice = "/dev/vx/dsk/newdg1/newvol2"
     FSType = vxfs
     FsckOpt = "-y"
      ReuseMntPt = 1
     )

 Process procres1 (
     PathName = "/usr/bin/ksh"
     Arguments = "/scrib.sh /dgsfs1"
     )

 Process procres2 (
     PathName = "/usr/bin/ksh"
     Arguments = "/scrib.sh /dgsfs2"
     )

 mntres1 requires dgres
 mntres2 requires dgres
```

```
    procres1 requires mntres1
    procres2 requires mntres2
```

Sample main.cf of DiskGroupSnap with Oracle resource

The following Oracle configuration has been simplified for presentation within this guide.

```
group fd_oragrp (
        SystemList = { Node_A = 0, Node_B = 1 }
        AutoStart = 0
        SystemZones = { Node_A = 0, Node_B = 1 }
        )

        DiskGroupSnap dgres (
                FDSiteName @Node_A = siteA
                FDSiteName @Node_B = siteB
                TargetResName = oradg_res
                FDType = "Bronze"
                )

        IP fd_oraip (
        Device = bge0
        Address = "10.198.95.191"
        NetMask = "255.255.255.0"
                    )

        Mount fd_archmnt (
                FsckOpt = "-y"
                ReuseMntPt = 1
                BlockDevice = "/dev/vx/dsk/oradg_fd/archive_vol"
                MountPoint = "/ora_archive"
                FSType = vxfs
                )

        Mount fd_datamnt (
                FsckOpt = "-y"
                ReuseMntPt = 1
                BlockDevice = "/dev/vx/dsk/oradg_fd/data_vol"
                MountPoint = "/ora_data"
                FSType = vxfs
                )

        NIC fd_oranic (
                Device = bge0
        NetworkHosts = { "10.198.95.1" }
                )

        Netlsnr fd_LSNR (
                Home = "/opt/oracle/ora_home"
                Owner = oracle
                )

        Oracle fd_Ora_01 (
                Owner = oracle
                Home = "/opt/oracle/ora_home"
                Sid = Ora_01
                )
      requires group oragrp offline local
      fd_LSNR requires fd_Ora_01
      fd_LSNR requires fd_oraip
      fd_Ora_01 requires fd_archmnt
      fd_Ora_01 requires fd_datamnt
      fd_archmnt requires dgres
      fd_datamnt requires dgres
      fd_oraip requires fd_oranic

      group oragrp (
        SystemList = { Node_A = 0, Node_B = 1 }
        AutoStartList = { Node_A, Node_B }
        SystemZones = { Node_A = 0, Node_B = 1 }
        )

        DiskGroup oradg_res (
                DiskGroup = oradg
                )

        IP Node_A4vip (
               Device = bge0
               Address = "10.198.95.192"
               Netmask = "255.255.252.0"
                )

        Mount arch_mnt (
                FsckOpt = "-y"
                ReuseMntPt = 1
                BlockDevice = "/dev/vx/dsk/oradg/archive_vol"
```

```
                MountPoint = "/ora_archive"
                FSType = vxfs
                )

        Mount data_mnt (
                FsckOpt = "-y"
                ReuseMntPt = 1
                BlockDevice = "/dev/vx/dsk/oradg/data_vol"
                MountPoint = "/ora_data"
                FSType = vxfs
                )

        NIC nic_Node_A4vip (
                Device = bge0
                )

        Netlsnr LSNR (
                Home = "/opt/oracle/ora_home"
                Owner = oracle
                )

        Oracle Ora_01 (
                Owner = oracle
                Home = "/opt/oracle/ora_home"
                Sid = Ora_01
                )

        Volume arch_vol (
                Volume = archive_vol
                DiskGroup = oradg
                )

        Volume data_vol (
                Volume = data_vol
                DiskGroup = oradg
                )
LSNR requires Ora_01
LSNR requires Node_A4vip
Ora_01 requires arch_mnt
Ora_01 requires data_mnt
arch_mnt requires arch_vol
arch_vol requires oradg_res
data_mnt requires data_vol
```

```
data_vol requires oradg_res
Node_A4vip requires nic_Node_A4vip
```

Debug log levels for DiskGroupSnap agent

The DiskGroupSnap agent uses the following debug log levels:

DBG_1

Disk agent

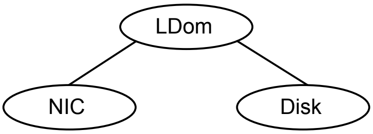
Monitors a physical disk or a partition.

You can use the Disk agent to monitor a physical disk or a slice that is exported to LDoms.

Dependencies

Disk resources have no dependencies.

Figure 2-5 Sample service group that includes a Disk resource on Solaris



Agent functions

Monitor	Performs read I/O operations on the raw device to determine if a physical disk or a partition is accessible.

State definitions

The state definitions for this agent follow:

ONLINE	Indicates that the disk is working normally.
FAULTED	Indicates that the disk has stopped working or is inaccessible.
UNKNOWN	Indicates that a problem exists either with the configuration or the ability to determine the status of the resource.

Attributes

Table 2-7 Required attributes

Required attribute	Description
Partition	Indicates which partition to monitor. Specify the partition with the full path beginning with a slash (/). If absolute path is not specified, the name is assumed to reside in /dev/rdsk/. Example: "/dev/rdsk/c2t0d0s2" or "/dev/vx/dmp/c2t0d0s2" Type and dimension: string-scalar **Note:** If DMP device name is provided, ensure that persistence of the naming scheme is set to Yes to avoid path name changes by reboot or device re-configuration. Refer to the *Dynamic Multi-Pathing Administrator's guide* for details on setting persistence for naming scheme. **Note:** If the disk is Extensible Firmware Interface (EFI) labeled, use the pathname without slice number.

Resource type definition

The resource definition for this agent on Solaris follows:

```
type Disk (
    static int OfflineMonitorInterval = 60
    static str ArgList[] = { Partition }
    static str Operations = None
    str Partition
)
```

Debug log levels

The Disk agent uses the following debug log levels:

DBG_1

Volume agent

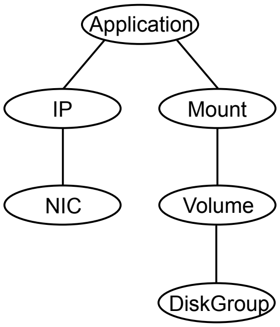
The Volume agent brings online, takes offline, and monitors a Veritas Volume Manager (VxVM) volume. Use the agent to make a volume highly available.

Note: Do not use the Volume agent for volumes created for replication.

Dependencies for Volume agent

Volume resources depend on DiskGroup resources.

Figure 2-6 Sample service group that includes a Volume resource



Agent functions for Volume agent

Online	Uses the `vxrecover` command to start the volume.
Offline	Uses the `vxvol` command to stop the volume.
Monitor	Attempts to read a block from the raw device interface to the volume to determine if the volume is online, offline, or unknown.
Clean	Terminates all ongoing resource actions and takes the resource offline—forcibly when necessary.

State definitions for Volume agent

ONLINE	Indicates that the specified volume is started and that I/O is permitted.
OFFLINE	Indicates that the specified volume is not started and that I/O is not permitted.

FAULTED	Indicates the volume stopped unexpectedly and that I/O is not permitted.
UNKNOWN	Indicates that the agent could not determine the state of the resource or that the resource attributes are configured incorrectly.

Attributes for Volume agent

Table 2-8 Required attributes

Required attribute	Description
DiskGroup	Name of the disk group that contains the volume. Type and dimension: string-scalar Example: "`DG1`"
Volume	Name of the volume from disk group specified in DiskGroup attribute. Type and dimension: string-scalar Example: "DG1Vol1"

Table 2-9 Internal attribute

Optional attribute	Description
NumThreads	Number of threads used within the agent process for managing resources. This number does not include threads used for other internal purposes. Setting the NumThreads attribute to a higher value may decrease the time required to go online or the time required to monitor a large number of DiskGroup resources. If there are many DiskGroup resources and if the resources are taking more time to come online, consider increasing the NumThreads attribute to a value greater than 1. Veritas recommends that you should not modify this attribute. Default: 1

Resource type definition for Volume agent

The resource type definition for this agent follows:

```
type Volume (
    static int NumThreads = 1
    static str ArgList[] = { Volume, DiskGroup }
    str Volume
    str DiskGroup
)
```

Sample configuration for Volume agent

The sample configuration for the Volume agent follows:

```
Volume sharedg_vol3 (
    Volume = vol3
    DiskGroup = sharedg
)
```

Debug log levels for Volume agent

The Volume agent uses the following debug log levels:

DBG_1

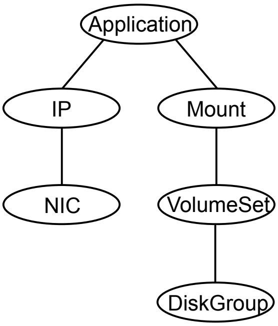
VolumeSet agent

The VolumeSet agent brings online, takes offline, and monitors a Veritas Volume Manager (VxVM) volume set. Use the agent to make a volume set highly available.

Dependencies for VolumeSet agent

VolumeSet resources depend on DiskGroup resources.

Figure 2-7 Sample service group that includes a VolumeSet resource



Agent functions for VolumeSet agent

Online	Uses the vxrecover command to start the volume set.
Offline	Uses the vxvset command to stop the volume set.
Monitor	Attempts to read a block from the raw device interface to the volumes inside the volume set to determine if the volume set is online, offline, or unknown.
Clean	Terminates all ongoing resource actions and takes the resource offline—forcibly when necessary.

State definitions for VolumeSet agent

ONLINE	Indicates that all the volumes in the volume set are started and that I/O is permitted for all the volumes.
OFFLINE	Indicates that at least one of the volume is not started in the volume set and that I/O is not permitted for that volume.
FAULTED	Indicates the volumes that are inside the volume set have stopped unexpectedly and that I/O is not permitted.
UNKNOWN	Indicates that the agent could not determine the state of the resource or that the resource attributes are configured incorrectly.

Attributes for VolumeSet agent

Table 2-10 Required attributes

Required attribute	Description
DiskGroup	The name of the disk group that contains the volume set. Type and dimension: string-scalar Example: "DG1"
VolumeSet	The name of the volume set from the disk group that you specified in the DiskGroup attribute. Type and dimension: string-scalar Example: "DG1VolSet1"

Resource type definition for VolumeSet agent

```
type VolumeSet (
    static str ArgList[] = { DiskGroup, VolumeSet }
    str VolumeSet
    str DiskGroup
)
```

Sample configurations for VolumeSet agent

This sections contains sample configurations for this agent.

A configured VolumeSet that is dependent on a DiskGroup resource

The VolumeSet's shared_vset3 resource is configured and is dependent on DiskGroup resource with a shared diskgroup.

```
VolumeSet sharedg_vset3 (
VolumeSet = vset3
DiskGroup = sharedg
)
```

Agent notes for VolumeSet agent

This sections contains notes about this agent.

Inaccessible volumes prevent the VolumeSet agent from coming online

The VolumeSet agent does not come online if any volume is inaccessible in its volume set.

To remove a volume from volume set

◆ Enter the following commands to remove a volume from a volume set mounted on mountpoint.

```
# fsvoladm remove mountpoint volume_name
# vxvset -g diskgroup rmvol volumeset volume_name
```

Debug log levels for VolumeSet agent

The VolumeSet agent uses the following debug log levels:

DBG_1, DBG_4

Mount agent

The Mount agent brings online, takes offline, and monitors a file system or an NFS client mount point. You can use the agent to make file systems or NFS client mount points highly available.

This agent is zone-aware.

This agent also supports high availability fire drills.

The ContainerName and ContainerType attributes are deprecated.

Refer to the *Veritas InfoScale Virtualization Guide* for information on using mount resources inside local zone.

For mounting the NFS file system, the Mount agent supports the IPv6 protocol.

For important information about this agent, See “Notes for Mount agent” on page 68.

IMF awareness

The Mount agent is IMF-aware and uses Asynchronous Monitoring Framework (AMF) kernel driver for IMF notification. For more information about IMF and intelligent resource monitoring, refer to the *Cluster Server Administrator’s Guide*.

Note: IMF for mounts is supported only for VxFS and NFS file system types.

For more information about IMF-related Mount agent functions, see Agent functions for Mount agent.

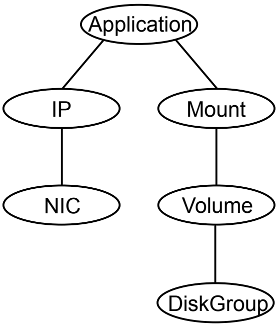
For more information about using the Mount agent with IMF, see IMF usage notes.

Dependencies for Mount agent

The Mount resource does not depend on any other resources.

In some scenarios, the Mount agent depends on Volume or DiskGroup resources.

Figure 2-8 Sample service group that includes a Mount resource



Agent functions for Mount agent

Online	Mounts a block device on the directory. If the mount process fails for non-NFS mounts, the agent attempts to run the fsck command on the device before attempting to mount the file system again. If file system type is NFS, agent mounts the remote file system to a specified directory. The remote NFS file system is specified in the BlockDevice attribute. **Note:** If you enable writeback mode for the SmartIO feature, the files that have pending dirty data can become inaccessible in the event of a node failure or Solid State Drive (SSD) device failure. For more information about recovering the writeback cache, refer to the *Veritas InfoScale SmartIO for Solid State Drives Solutions Guide*.
Offline	Unmounts the mounted file system.
Monitor	Determines if the file system is mounted. If IMF is enabled for the Mount agent, the resource is monitored asynchronously and any change in the resource state is immediately sent to VCS for appropriate action.
imf_init	Initializes the agent to interface with the asynchronous monitoring framework (AMF) kernel driver. This function runs when the agent starts up.

imf_getnotification

Waits for notification about resource state changes. This function runs after the agent initializes with the AMF kernel driver. The agent continuously waits for notification and takes action on the resource upon notification.

imf_register

Registers the resource entities, which the agent must monitor, with the AMF kernel driver. This function runs for each resource after the resource goes into steady state (online or offline). This action entry point registers the mount point, block device, and file system type for mount agent.

Clean

Unmounts the mounted file system forcefully.

Info

The Mount agent info function executes the command:

```
# df -k mount_point
```

The output displays Mount resource information:

```
Size Used Avail Use%
```

To initiate the info agent function, set the InfoInterval timing to a value greater than 0. In this example, the info agent function executes every 60 seconds:

```
# haconf -makerw
```

```
# hatype -modify Mount InfoInterval 60
```

The command to retrieve information about the Mount resource is:

```
# hares -value mountres ResourceInfo
```

Output includes:

```
Size 2097152
Used 139484
Available 1835332
Used% 8%
```

Action

- chgmntlock

  Resets the VxFS file system lock to a VCS-defined lock.
- mountpoint.vfd

  Checks if the specified mount point exists on the offline node. If it fails and you request that VCS fixes it, it creates the mount point directory using `mkdir` command.
- mounted.vfd

  Checks if the mount point is already mounted on the offline node. If it fails, you need to unmount all the file systems from the specified mount point directory.
- vxfslic.vfd

  Checks for valid Veritas File System (VxFS) licenses. If it fails, you need to update the license for VxFS.
- mountentry.vfd

  Checks that the mount point is not listed in auto file system tables.

  For example,

  /etc/vfstab

  If this action fails, you need to remove the mount point from auto file system tables.

attr_changed

Unlocks the mounts when you change the value of the VxFSMountLock attribute from 1 or 2 to 0.

State definitions for Mount agent

The state definitions for this agent follow:

ONLINE

For the local file system, indicates that the block device is mounted on the specified mount point.

For an NFS client, indicates that the NFS remote file system is mounted on the specified mount directory.

OFFLINE

For the local file system, indicates that the block device is not mounted on the specified mount point.

For an NFS client, indicates that the NFS remote file system is not mounted on the specified mount directory.

FAULTED — For the local file system, indicates that the block device has unexpectedly unmounted.

For the NFS client, indicates that the NFS remote file system has unexpectedly unmounted.

UNKNOWN — Indicates that a problem exists either with the configuration or the ability to determine the status of the resource.

Attributes for Mount agent

Table 2-11 Required attributes

Required attribute	Description
BlockDevice	Block device for mount point. When you specify the block device to mount, enclose IPv6 addresses in square brackets. The mount command requires square brackets around the IPv6 address to differentiate between the colons in the address and the colon that separates the remote host and remote directory. Type and dimension: string-scalar Examples: ■ "/dev/vx/dsk/myvcs_dg/myvol" ■ IPv4 ■ "10.209.70.90:/dirname/anotherdir" ■ IPv6 "[fe80::1:2:3]:/dirname/anotherdir"

Table 2-11 Required attributes *(continued)*

Required attribute	Description
FsckOpt	Mandatory for the following file system types: - ufs - vxfs Use this attribute to specify options for the `fsck` command. You must correctly set this attribute for local mounts. If the mount process fails, the `fsck` command is executed with the specified options before it attempts to remount the block device. Its value must include either `-y` or `-n`. Refer to the fsck manual page for more information. For NFS mounts, the value of this attribute is not applicable and is ignored. Type and dimension: string-scalar VxFS example: -y **Note:** When you use the command line, add the % sign to escape '-'. For example: `hares -modify MntRes FsckOpt %-y`
FSType	Type of file system. Supports ufs, nfs, zfs, lofs, or vxfs. Type and dimension: string-scalar Example: "vxfs"
MountPoint	Directory for mount point Type and dimension: string-scalar Example: "/tmp/mnt"

Table 2-12 Optional attributes

Optional attribute	Description
MountOpt	Options for the `mount` command. Refer to the `mount` manual page for more information. Do not set the VxFS mount option "mntlock=*key*". The agent uses this option only when bringing a Mount resource online. Type and dimension: string-scalar Example: "rw"

Table 2-12 Optional attributes *(continued)*

Optional attribute	Description
SnapUmount	If the value of this attribute is 1, this attribute automatically unmounts VxFS snapshots when the file system is unmounted. If the value of this attribute is 0, and snapshots are mounted, the resource cannot be brought offline. In this case, failover does not occur. Type and dimension: integer-scalar Default: 0
CkptUmount	If the value of this attribute is 1, this attribute automatically unmounts VxFS Storage Checkpoints when file system is unmounted. If the value of this attribute is 0, and Storage Checkpoints are mounted, then failover does not occur. Type and dimension: integer-scalar Default: 1
AccessPermissionChk	If the value of this attribute is 1 or 2, the monitor verifies that the values of the MntPtPermission, MntPtOwner, and MntPtGroup attributes are the same as the actual mounted file system values. If any of these do not match the values that you have defined, a message is logged. If the value of this attribute is 2, and if the mounted file system permissions do not match the attribute values, the Monitor agent function returns the state as OFFLINE. Type and dimension: integer-scalar Default: 0

Table 2-12 Optional attributes *(continued)*

Optional attribute	Description
CreateMntPt	If the value of this attribute is 0, no mount point is created. The mount can fail if the mount point does not exist with suitable permissions. If the value of this attribute is 1 or 2, and a mount point does not exist, the agent creates a mount point with system default permissions when the resource is brought online. If the permissions for the mount point are less than 555, a warning message is logged. If the value of this attribute is 2, and the mount point does not exist, the agent creates a mount point with system default permissions when the resource is brought online. If the permissions for the mount point are less than 555, a warning message is logged. In addition, VCS deletes the mount point and any recursively created directories when the resource is brought offline. The mount point gets deleted only if it is empty, which is also true for recursive mount points. Type and dimension: integer-scalar Default: 0
MntPtGroup	This attribute specifies the group ownership of the mounted file system. The agent verifies the group ownership of the mounted file system every monitor cycle if the value of the AccessPermissionChk attribute is not 0. Type and dimension: string-scalar Example: "grp1"
MntPtOwner	This attribute specifies the user ownership of the mounted file system. The agent verifies the user ownership of the mounted file system every monitor cycle if the value of the AccessPermissionChk attribute is not 0. Type and dimension: string-scalar Example: "usr1"

Table 2-12 Optional attributes *(continued)*

Optional attribute	Description
MntPtPermission	This attribute specifies the permissions of the mounted file system in an absolute format of a four-digit octal. The agent verifies the mode of the mounted file system every monitor cycle if the value of the AccessPermissionChk attribute is not 0. Type and dimension: string-scalar Example: "0755"
OptCheck	The value of this attribute determines if VCS should verify the mount options. The state of the resource is determined based on the result of the verification. If the value of this attribute is 0 (default), the mount options are not checked. If the value of the OptCheck attribute is 1, 2 or 3, a check is performed to see if the mount command options that you have specified for VCS are set in the MountOpt attribute. The MountOpt attributes should be the same as the actual mount command options. If the actual mount options differ from the MountOpt attribute, a message is logged. The state of the resource depends on the value of this attribute. If the value of the attribute is 1, the state of the resource is unaffected. If the value is 2, the state of the resource is set to offline. If the value is 3, state of the resource is set to unknown. Type and dimension: integer-scalar Default: 0
RecursiveMnt	If the value of this attribute is 1, VCS creates all the parent directories of the mount point if necessary. All directories in the path are created with system default permissions. **Note:** Permissions on mount points must be a minimum of 555 for the operating system commands to work correctly. Type and dimension: boolean-scalar Default: 0

Table 2-12 Optional attributes *(continued)*

Optional attribute	Description
ReuseMntPt	If the same mount point needs to be specified in more than one mount resource, set the value of this attribute to 1. Note that this attribute only accepts a value of 1 or 0. To use this attribute, the cluster administrator needs to add this attribute to the arglist of the agent. Set the appropriate group and resource dependencies such that only one resource can come online on a system at a time. Type and dimension: integer-scalar Default: 0

Table 2-12 Optional attributes *(continued)*

Optional attribute	Description
VxFSMountLock	This attribute is only applicable to Veritas File System (VxFS). This attribute controls a file system locking feature to prevent accidental unmounts. This attribute can take three values: 0, 1, or 2. VxFSMountLock=0 The resource does not detect any changes to the lock when VCS reports that it is online after you set the value to zero. ■ If the mount point is initially locked with the mntlock="VCS", the monitor agent function unlocks it. ■ If the mount point is initially locked with a key that is not equal to "VCS", the agent logs a message once. ■ If the mount point is initially not locked, no action is performed. VxFSMountLock=1 The resource does not detect changes to the lock when VCS reports it online after the value was set to one. VCS does not monitor the lock. ■ If the mount point is initially locked with the mntlock="VCS", no action is performed. ■ If the mount point is initially locked with a key that is not equal to "VCS", the agent logs a message once. ■ If the mount point is initially not locked, the monitor agent function locks it with the mntlock="VCS". VxFSMountLock=2 When the value of the VxFSMountLock is 2, the file system is locked and the agent monitors any change to mntlock. ■ If the mount point is locked with the mntlock="VCS", no action is performed. ■ If the mount point is initially locked with a key that is not equal to "VCS", the monitor agent function logs a message whenever a change in mntlock is detected. ■ If the mount point is not locked, the agent locks it with the mntlock="VCS". Type and dimension: integer-scalar Default: 1

Table 2-12 Optional attributes *(continued)*

Optional attribute	Description
CacheRestoreAccess	This attribute is applicable only if: ■ File system type is VxFS. ■ Writeback caching is enabled for the SmartIO feature. The value of this attribute determines whether to perform restore access operation or not. Following are the values: ■ 0: Does not perform restore access operation. ■ 1: Performs restore access operation. Type and dimension: boolean-scalar Default: 0 **Note:** For the procedure and implications of enabling writeback caching, refer to the *Veritas InfoScale SmartIO for Solid State Drives Solutions Guide*.

Table 2-13 Internal attribute

Internal attribute	Description
AEPTimeout	This is an internal attribute. Do not modify this attribute. This attribute is used to pass the timeout value of agent entry points.

Resource type definition for Mount agent

The resource definition for this agent on Solaris follows:

```
type Mount (
        static keylist RegList = { VxFSMountLock }
        static int IMF{} = { Mode = 3, MonitorFreq = 1,
         RegisterRetryLimit = 3 }
        static str IMFRegList[] = { MountPoint, BlockDevice, FSType }
        static boolean AEPTimeout = 1
        static keylist SupportedActions = { "mountpoint.vfd",
         "mounted.vfd", "vxfslic.vfd" , "chgmntlock", "mountentry.vfd" }
        static str ArgList[] = { MountPoint, BlockDevice, FSType,
         MountOpt, FsckOpt, SnapUmount, CkptUmount, OptCheck,
         CreateMntPt, MntPtPermission, MntPtOwner, MntPtGroup,
         AccessPermissionChk, RecursiveMnt, VxFSMountLock,
         CacheRestoreAccess }
        str MountPoint
        str BlockDevice
        str FSType
        str MountOpt
        str FsckOpt
        boolean SnapUmount = 0
        boolean CkptUmount = 1
        int OptCheck = 0
        int CreateMntPt = 0
        int ReuseMntPt = 0
        str MntPtPermission
        str MntPtOwner
        str MntPtGroup
        int AccessPermissionChk = 0
        boolean RecursiveMnt = 0
        boolean VxFSMountLock = 1
        boolean CacheRestoreAccess = 0
)
```

Notes for Mount agent

The Mount agent has the following notes:

- High availability fire drill
- VxFS file system lock
- IMF usage notes
- IPv6 usage notes
- Support for loopback file system
- Enabling Level two monitoring for the Mount agent
- ZFS file system and pool creation example
- Support for VxFS direct mount inside non-global zones

High availability fire drill

The high availability fire drill detects discrepancies between the VCS configuration and the underlying infrastructure on a node; discrepancies that might prevent a service group from going online on a specific node.

For Mount resources, the high availability drill performs the following, it:

- Checks if the specified mount point directory exists

- Checks if the mount point directory is already used
- Checks for valid Veritas File System (VxFS) licenses
- Checks if the mount point exists in the /etc/vfstab file

For more information about using the high availability fire drill, see the *Cluster Server Administrator's Guide*.

VxFS file system lock

If the mount option in the mount table output has the option mntlock="*key*", then it is locked with the key "*key*". To verify if mount locking is in use and has the value of "*key*", run the `mount` command and review its output.

```
# mount
```

If the VxFS file system has mntlock="*key*" in its mount options, then unmounting the file system fails.

You can unlock the file system with the `fsadm` command and then unmount it. To unlock a locked mount, run the following command where "*key*" is the lock identifier and *mount_point_name* is the file system mount point.

```
# /opt/VRTS/bin/fsadm -o mntunlock="key" mount_point_name
```

To unmount a file system mounted with locking, run the `vxumount`command with the option mntunlock="*key*", for example:

```
# /opt/VRTS/bin/umount -o mntunlock="key" mount_point_name
```

IMF usage notes

If you use IMF for intelligent resource monitoring, depending on the value of the FSType attribute, you must set the MonitorFreq key value of the IMF attribute.

See the *Cluster Server Administrator's Guide* for the IMF attribute description.

IPv6 usage notes

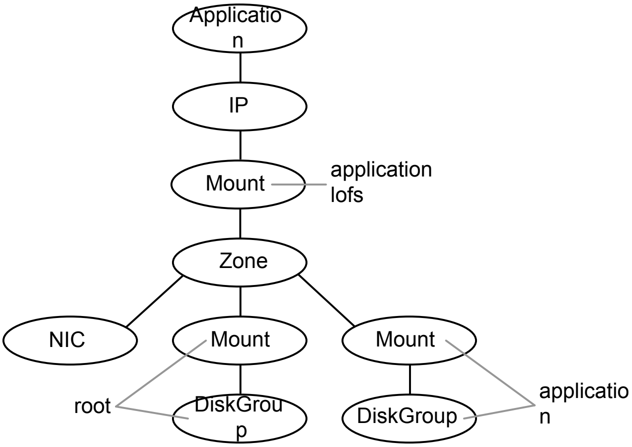
Review the following information for IPv6 use:

- For IPv6 functionality for NFS, you must use NFS version 4 in order to make the mount reachable. Note that NFSv4 requires several configuration steps in the operating system and NFS-related resources in VCS to enable it on the client and the exporting server.

Support for loopback file system

The Mount agent provides loopback file system support. You can manage the loopback file system as a Mount resource. You can use this loopback support to mount a file system in the global zone and share it in non-global zones. For loopback support, configure the FSType attribute to use a value of lofs.

Figure 2-9 Sample service group for the zone root on shared storage with a loopback file system when VCS manages the loopback file system as a Mount resource



The following is a sample configuration where you use the Mount resource to manage the lofs file system:

```
group loopbacksg (
SystemList = { sysA = 0, sysB = 1 }
ContainerInfo@sysA = { Name = zone1, Type = Zone, Enabled = 1 }
ContainerInfo@sysB = { Name = zone1, Type = Zone, Enabled = 1 }
)
Mount lofs_mnt_global_to_local (
MountPoint = "/export/home/zone1/root/lofs_mnt"
BlockDevice = "/mnt1/m1"
FSType = lofs
)
Zone z1 (
)
```

```
Mount base_mnt (
MountPoint = "/mnt1"
BlockDevice = "/dev/vx/dsk/tdg/tvol1"
FSType = vxfs
FsckOpt = "-y"
)
lofs_mnt_global_to_local requires z1
lofs_mnt_global_to_local requires base_mnt
```

Sample configurations for Physical-to-Virtual (P2V) scenarios

In the following sample configuration, the local zone (zone1) runs only on system 1 (sys1) and it does not exist in system 2 (sys2). The mount resource comes online inside the zone. On system 1, the "/export/home/zone1/root/mnt" mount point exists inside the zone. On system 1, the "/export/home" mount point is mounted on "/export/home/zone1/root/mnt" inside the zone. On system 2, the "/export/home" mount point is mounted on "/mnt" in the global zone.

```
group mountgrp (
         SystemList = { sys1 = 0, sys2 = 1 }
         ContainerInfo @sys1 = { Name = zone1, Type = Zone, Enabled = 1 }
         ContainerInfo @sys2 = { Name = zone1, Type = Zone, Enabled = 2 }
         Administrators = { z_mountres_sys1, z_zoneres_sys1 }
         )



         Mount loopback (
                 MountPoint @sys1 = "/export/home/zone1/root/mnt"
                 MountPoint @sys2 = "/mnt"
                 BlockDevice = "/export/home"
                 FSType = lofs
                 FsckOpt = "-n"
                 )

         Zone zoneres (
                 )

         loopback requires zoneres
```

In the following sample configuration, there are two mount resources. The first mount resource (mountres) mounts the actual shared volume "/dev/vx/dsk/vdg/nfs_vol" on "/lockinfo". Next, "/lockinfo" is loop-backed on "/export/home/zone1/root/mnt" inside the zone on system 1. On system 2, the "/dev/vx/dsk/vdg/nfs_vol" volume is mounted on "/lockinfo" and then "/lockinfo" is loop-backed and mounted on "/mnt" in the global zone. In the following example, localization has been done for the mount resource.:

```
group mountgrp (
         SystemList = { sys1 = 0, sys2 = 1 }
         ContainerInfo @sys1 = { Name = zone1, Type = Zone, Enabled = 1 }
         ContainerInfo @sys2 = { Name = zone1, Type = Zone, Enabled = 2 }
         Administrators = { z_mountres_sys1, z_zoneres_sys1 }
         )

         DiskGroup dgres (
                 DiskGroup = vdg
                 )

         Mount loopback (
                 MountPoint @sys1 = "/export/home/zone1/root/mnt"
                 MountPoint @sys2 = "/mnt"
                 BlockDevice = "/lockinfo"
                 FSType = lofs
                 FsckOpt = "-n"
                 )

         Mount mountres (
                 MountPoint = "/lockinfo"
                 BlockDevice = "/dev/vx/dsk/vdg/nfs_vol"
                 FSType = vxfs
                 FsckOpt = "-y"
                 )

         Zone zoneres (
                 )

         loopback requires mountres
         loopback requires zoneres
         mountres requires dgres
```

Enabling Level two monitoring for the Mount agent

Level two monitoring can be enabled for the Mount agent only if FSType is set to "nfs".

To enable Level two monitoring, run the following commands:

- `# haconf -makerw`

- `# hares -override resource_name LevelTwoMonitorFreq`
- `# hares -modify resource_name LevelTwoMonitorFreq 1`
- `# haconf -dump -makero`

For more details about the LevelTwoMonitorFreq attribute, refer to the *Cluster Server Agent Developer's Guide*.

ZFS file system and pool creation example

If you want to use the Mount resource to monitor the ZFS file system, perform the following steps.

Create the tank storage pool and file system on the disk device c1t0d0 for example.

```
# zpool create tank c1t0d0
```

Create the home file system in tank.

```
# zfs create tank/home
```

Set the value of the MountPoint attribute to legacy.

```
# zfs set mountpoint=legacy tank/home
```

Set the Mount agent's attributes. The following is an example of this configuration's main.cf file.

```
Mount m1 (
    MountPoint = "/mp1"
    BlockDevice = "tank/home"
    FSType = zfs
    MountOpt = rw
    FsckOpt = "-n"
)
```

Support for VxFS direct mount inside non-global zones

The Mount agent supports VxFS direct mount inside non-global zones from the global zone. You can mount VxFS directly inside a non-global zone. To mount VxFS inside a non-global zone, override the ContainerOpts attribute at the resource level and set the value of the RunInContainer attribute to 1. For example, enter the following commands sequentially:

```
# hares –override res_direct_mount ContainerOpts
# hares –modify res_direct_mount ContainerOpts RunInContainer 1
PassCInfo 0
```

The following is a sample configuration for VxFs direct mount inside a non-global zone:

```
group sg_direct_mount (
        SystemList = { sysA = 0, sysB = 1 }
        ContainerInfo = { Name = zone1, Type = Zone, Enabled = 1 }
        Administrators = { z_zoneres_sysA, z_zoneres_sysB }
        )

        Mount res_direct_mount (
                BlockDevice = "/dev/vx/dsk/data_dg/data_vol"
                MountPoint = "/mymount/mnt1"
                FSType = vxfs
                FsckOpt = "-y"
                ContainerOpts = { RunInContainer = 1, PassCInfo = 0 }
                )
```

For more details, refer to the *Veritas InfoScale Virtualization Guide*.

Sample configurations for Mount agent

VxFS configuration example for Mount agent

Configuration for VxFS follows:

```
Mount mnt-fs1 (
    MountPoint= "/mnt1"
    BlockDevice = "/dev/vx/dsk/mnt-dg1/mnt-vol1"
    FSType = "vxfs"
    FsckOpt = "-n"
    MountOpt = "rw"
)
```

Debug log levels for Mount agent

The Mount agent uses the following debug log levels:

DBG_1, DBG_2, DBG_3, DBG_4, DBG_5

Zpool agent

The Zpool agent brings online, takes offline, and monitors ZFS storage pools. It exports ZFS storage pools (which reside on shared storage) from one node. It then imports the pool onto another node as required.

The automount feature of ZFS mounts all its file systems by setting the mountpoint property to something other than legacy. To find the value of the mountpoint property, use the `zfs get` command. For example, from the command line for the tank mountpoint, enter:

```
# zfs get mountpoint tank
NAME         PROPERTY        VALUE        SOURCE
tank         mountpoint      /tank        default
```

As another example, to find the value of the mountpoint property for the legacypool storage pool, enter:

```
# zfs get mountpoint legacypool
NAME         PROPERTY        VALUE        SOURCE
tank         mountpoint      legacy       default
```

The Zpool agent checks the mountpoint and canmount properties and the ChkZFSMounts attribute to decide whether the mounted file system should be checked in the Zpool agent or not.

If the value of the mountpoint property is `legacy`, it does not check the file system mount status. The agent assumes that you plan to use Mount resources to manage and monitor the ZFS file systems.

If the value of the mountpoint property is not `legacy` or `none` and if the canmount property is not set to `off`, the agent checks the mount status of the ZFS file systems. If the value of the mountpoint property is `none` or if the canmount property is set to `off`, the agent assumes that you do not want the file system to be mounted.

See “VCS support for multi-pathing solutions” on page 19.

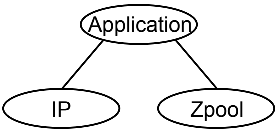
Limitations for Zpool agent

The agent does not support the use of logical volumes in ZFS. If ZFS logical volumes are in use in the pool, the pool cannot be exported, even with the `-f` option. Oracle does not recommend the use of logical volumes in ZFS due to performance and reliability issues.

Dependencies for Zpool agent

No dependencies exist for the Zpool resource for a pool that has a non-legacy value for its mountpoint property.

Figure 2-10 Sample service group for a Zpool resource



Agent functions for Zpool agent

Online

Imports the ZFS storage pool.

Offline

Exports the ZFS storage pool.

Monitor

Checks the online status of the ZFS pool.

If the mountpoint property of the ZFS file system is set and if the value is not `legacy` or `none`; if the canmount property is not set to `off`; and if the ChkZFSMounts attribute is enabled, the agent checks whether all the ZFS file systems under the same ZFS storage pool are mounted.

If the ZFS pool contains a ZFS file system that a non-global zone uses, then you need to import the pool before the zone boots up. After the zone boots up, if the mountpoint property for this ZFS file system that the non-global zone uses is not set to `legacy` or `none` and if the canmount property is not set to `off`, the file system is mounted after the zone boots up.

If you have enabled the ChkZFSMounts attribute in the Zpool resource, the Monitor function skips checking the availability of the ZFS file system because unless the zone resource is up, the file systems are not mounted.

The Zone resource depends on the Zpool resource for the non-global zone scenario. In this case, you need to provide the ZoneResName attribute, which indicates the name of the Zone resource. When the Zone resource is in an ONLINE state, then ChkZFSMounts starts to check the mount status of the ZFS file system pool that the non-global zone uses.

Clean

Exports the ZFS storage pool forcefully.

State definitions for Zpool agent

ONLINE

Reports an ONLINE state when the ZFS file systems that share a common storage pool are mounted, and the zpool command `zpool list -H -o health $Poolname` indicates that the pool is online.

OFFLINE

Reports an OFFLINE state when all of the ZFS file systems that share a common storage pool are unmounted. It also reports an OFFLINE state when the zpool command `zpool list -H -o health $Poolname` command’s status indicates that the pool is offline.

UNKNOWN

Reports an UNKNOWN state in the following situations:

- If the status of the storage pool cannot be detected.
- If the storage pool is online but the path of the mounted file system does not match the path that is specified in the AltRootPath attribute of this agent.
- If the storage pool is online but altrootpath setting of the pool is not set. Export the pool and bring the resource online to correct this.

Attributes for Zpool agent

Table 2-14 Required attributes

Required attribute	Description
PoolName	The name of the ZFS storage pool name. Type and dimension: string-scalar Default: n/a Example: tank

Table 2-14 Required attributes *(continued)*

Required attribute	Description
AltRootPath	Provides the alternate root path that is necessary to prevent the etc/zfs/zpool.cache file from being populated. Supplying this value keeps a node from importing the ZFS storage pool automatically when it restarts after a crash. Not importing the ZFS storage prevents concurrency violations and file system corruption. If you do not provide a value for the AltRootPath attribute, VCS sets the $AltRootPath to "/" as a workaround. This workaround makes sure that the ZFS command `zpool import -R` *`$AltRootPath $PoolName`* does not populate the zpool.cache file. Type and dimension: string-scalar Default: / Example: /mnt
ChkZFSMounts	The ChkZFSMounts attribute enables the check to determine whether all the file systems are properly mounted for that ZFS storage pool when the mountpoint property is not set to legacy. The default value is enabled (set to 1). Type and dimension: boolean-scalar Default: 1

Table 2-15 Optional Attributes

Optional Attribute	Description
FailMode	The FailMode attribute controls the system behavior in the event of a catastrophic pool failure. The value of this attribute is used as the failmode option while importing the ZFS storage pool. The possible values are wait, continue, or panic. ■ wait: Blocks all I/O access to the pool until the device connectivity is recovered and the errors are cleared. ■ continue: Returns EIO to any new write I/O requests but allows reads to any of the remaining healthy devices. ■ panic: Halt the system and generates a system crash dump. For more information about this attribute, refer to the *zpool manual pages*. Type and dimension: string-scalar Default: continue Example: panic **Note:** Veritas recommends to set FailMode to panic to effectively detect storage connectivity loss to the underlying disks of the pool.
ForceOpt	If this attribute is enabled (the value is set to 1), and if the following commands fail, then the commands are re-invoked with the `-f` option ■ zpool export in the offline entry point ■ zpool export in the clean entry point ■ zpool import in the online entry point For more information about this attribute, refer to the *zpool manual pages*. Type and dimension: boolean-scalar Default: 1 Example: 1

Table 2-15 Optional Attributes *(continued)*

Optional Attribute	Description
ForceRecoverOpt	If this attribute is enabled (the value is set to 1), and if the zpool import command fails, then the zpool import command is reinvoked with the `-F` option. For more information about this attribute, refer to the *zpool manual pages.* Type and dimension: boolean-scalar Default: 0 Example: 1
DeviceDir	Specifies the directories or block device files that the zpool import command searches for devices or files. Each directory or block device file is passed to the zpool import command with the `-d` option. The directory or block device file path must exist and be absolute. For more information, refer to the *zpool manual pages*. To reduce the time taken for zpool imports, provide the block device file path or create links to the devices constituting the zpool in a directory and provide that directory path in the DeviceDir attribute. Type and dimension: string-vector Default: n/a Example 1: "/dev/vx/dmp" Example 2: "/dev/vx/dmp/disk_1s2"
ZoneResName	Use the ZoneResName attribute when a non-global zone needs the Zpool resource. In this case, supply the ZoneResName attribute with the name of the Zone resource. Type and dimension: string-scalar Default: n/a Example: zone1

Resource type definition for Zpool agent

```
type Zpool (
static str ArgList[] = { PoolName, AltRootPath, ChkZFSMounts,
ZoneResName, "ZoneResName:State", DeviceDir, FailMode,
ForceOpt, ForceRecoverOpt }
str PoolName
str AltRootPath
boolean ChkZFSMounts = 1
str ZoneResName
str DeviceDir[]
str FailMode = continue
boolean ForceOpt = 1
boolean ForceRecoverOpt = 0
)
```

Sample configurations for Zpool agent

A main.cf example that shows the Zpool agent configuration.

```
include "types.cf"
cluster clus1 (
UserNames = { admin = dqrJqlQnrMrrPzrLqo,
z_zone_res = dOMoOTnNMlMSlVPnOT,
z_dummy_res = bIJbIDiFJeJJhRJdIG }
Administrators = { admin }
)
system sysA (
)
system sysB (
)
group tstzfs (
SystemList = { sysA = 0, sysB = 1 }
AutoStartList = { sysA, sysB }
ContainerInfo = { Name = z1, Type = Zone, Enabled = 1 }
Administrators = { z_zone_res }
)
Zone zone_res (
)
Zpool legacy_res (
PoolName = legacypool
ZoneResName = zone_res
)
Zpool tstzone_res (
PoolName = tstzonepool
ZoneResName = zone_res
)
Zpool zpool_res (
PoolName = tank
AltRootPath = "/mytank"
)
Application custom_app (
StartProgram = "/mytank/tank/startapp"
StopProgram = "/mytank/tank/stopapp"
MonitorProcesses = { "/mytank/tank/mondaemon" }
)
custom_app requires zpool_res
zone_res requires legacy_res
zone_res requires tstzone_res
```

Debug log levels for Zpool agent

The Zpool agent uses following debug log levels:

DBG_1, DBG_2, and DBG_5

VMwareDisks agent

The VMwareDisks agent enables vMotion and VMware Distributed Resource Scheduler (DRS) in VCS clusters configured and deployed on virtual machines in VMware environment.

When a VCS cluster with a shared disk is configured on virtual machines, VMware does not support VMware Distributed Resource Scheduler (DRS) and vMotion. Thus the vMotion and DRS capabilities are compromised. The solution to this issue is to attach the disk(s) to a single virtual machine at a time in a VCS cluster. In case of a user-initiated failover or a fault-induced failover, these disks failover (detach-attach) to the target virtual machine along with the service group. VMwareDisks agent manages the attaching and detaching of the disks to the virtual machines.

To ensure proper functioning of the VMwareDisks agent, verify the following:

- Ensure that the VMware disks are in persistent mode. If the VMware disks are in independent mode, VMWareDisks agent will revert them to persistent mode in case of a failover.
- The ESX/ESXi host or vCenter user account has administrative privileges or is a root user. If you do not want to use the administrator user account or the root user, create a role with the required privileges for the VMwareDisks resource functionality and assign this role to one or more users. This user must have the ability to:
  - Perform low-level file operations

- Add an existing disk
- Modify resources
- Remove disks

Note that the above list is only illustrative, you can add additional privileges as required.

If the user is a vCenter user, you must assign the requisite privileges so that the user can access the datastore.

To assign role and privileges

1. Log on to the vCenter Server and navigate to **Home > Inventory > Datastores and Datastore Clusters**.
2. From the vCenter inventory tree view, select the appropriate datacenter.
3. Right-click the datacenter and select **Add Permission**.

   Or

   Go to the **Permissions** tab. On the Permissions pane, right-click and select **Add Permission**.
4. In the Assign Permissions window, add the user, select the role, and assign privileges.

   For more information, refer to *VMware vSphere ESXi and vCenter Server Documentation*.

Agent functions

Online	Attaches the disks to the virtual machine.
Offline	Detaches the disks from the virtual machine.
Monitor	Verifies that the disks are attached to the virtual machine.

State definitions

ONLINE	Indicates that the disks are attached to the virtual machine.
OFFLINE	Indicates that the disks are not attached to the virtual machine.
UNKNOWN	Indicates that the agent could not detect the state of the disks.

Attributes

This section summarizes the required and optional attributes of the VMwareDisks agent.

Table 2-16 Required attributes

Required attribute	Description
ESXDetails	The list of hostnames and credentials of the ESX hosts or vCenter on which the virtual machines are configured. The key is ESX or vCenter hostname or IP address. The value must be in the format: ■ ESX or IP address—'User name'='Encrypted password' ■ vCenter hostname—'Domain\User Name'='Encrypted password' Example — IPv6: ESXDetails = { "[2620:128:f0a2:900d::150]" = "root=GMIsMUlSKsNSlSLuJUo" } Example — IPv4: ESXDetails = { "192.168.0.100" = "root=GMIsMUlSKsNSlSLuJUo" } Type and dimension: string-association
DiskPaths	The list of disks paths to be managed by the VMwareDisks resource. The key is of the form '[Data store name] Disk path'. The value is of the form 'SCSI controller key: Target ID (unit number of the disk)'. For example, 0:2. For RDM, add prefix 'RDM:' to the disk path. Alternatively, the key can be of the form 'Disk_UUID: [Data store name] Disk_path'. In case of RDM, 'RDM:Disk_UUID:[Data store name] Disk_path'. If the disk UUID is not provided, the agent will discover the UUID and modify the attribute to have UUID of the disk. Type and dimension: string-association

Table 2-17 Optional attributes

Optional attribute	Description
HAInfoDetails	Determines whether or not vSphere HA is enabled. This attribute uses the vCenter Server hostname or IP address to determine the status of vSphere HA. The value must be specified in the format: Key=Value. Where: ■ Key= vCenter Server hostname or IP address ■ Value=vCenter Server logon user credentials. This must be specified in the format: 'Domain\User Name'='Encrypted password' Example — IPv6: HAInfoDetails = {"[2001::15]" = "Administrator@IAG=JPLvPXoVNvQVo"} Example — IPv4: HAInfoDetails = {"192.168.0.100" = "Administrator@IAG=JPLvPXoVNvQVo"} If you do not specify a value for this attribute, the agent considers the vSphere HA setting based on the IsVMHAEnabled attribute value. Type and dimension: string-association
IsVMHAEnabled	Defines whether vSphere HA is enabled. The value 1 indicates that vSphere HA is enabled. This attribute value should match the vSphere HA settings in the VMware vSphere HA cluster. If vSphere HA is turned on, set this attribute value to 1 (True). If vSphere HA is turned off, set this attribute value to 0 (False). vSphere HA settings are considered based on this attribute only if the HAInfoDetails attribute is not set, or if the agent fails to retrieve the vSphere HA details based on the HAInfoDetails attribute. Default: 1 (True) Type and dimension: boolean-scalar
PanicVMOnESXLoss	Set this attribute value to 1 (True) to trigger panic on the virtual machine when the ESX host loses network connectivity. Default: 0 (False) Type and dimension: boolean-scalar
ForceRegister	For internal use only.

Table 2-17 Optional attributes *(continued)*

Optional attribute	Description
VMRegisterWait	The specified time interval, in seconds, during which VMware HA registers the virtual machine on any other ESX host when an ESX host fails. This is applicable only if IsVMHAEnabled is set to true. Default value: 120 seconds Type and dimension: integer
VirtualDiskMode	Specifies the mode to be used when the disk is attached. You can set the value of this attribute to one of the following: ■ `persistent` ■ `independent_persistent` ■ `independent_nonpersistent` You must modify the value after you configure application monitoring. **Note:** The VMwareDisks agent does not detect the mode in which the disk is configured. After a failover, the disk is attached in the mode that is defined in the attribute value. For details about the disk modes, refer to the VMware documentation. Default value: `persistent` Type and dimension: string-association

Resource type definition

```
type VMwareDisks (
 static keylist RegList = { ESXDetails }
 static keylist SupportedActions = { checkESXconn }
 static int InfoInterval = 60
 static int NumThreads = 1
 static str ArgList[] = { ESXDetails, DiskPaths, ResourceInfo,
  IsVMHAEnabled, VMRegisterWait, VirtualDiskMode, PanicVMOnESXLoss,
  ForceRegister, HAInfoDetails }
 str ESXDetails{}
 str DiskPaths{}
 str HAInfoDetails{}
 boolean IsVMHAEnabled = 1
 int VMRegisterWait = 120
 str VirtualDiskMode = persistent
 boolean PanicVMOnESXLoss = 0
 temp boolean ForceRegister = 0
)
```

Sample configurations

Sample configuration where UUID is not provided for the DiskPaths attribute:

```
VMwareDisks VMwareDisks_1 (
 ESXDetails = { "192.168.0.100" = "root=HVJtWTwVLnINjNK",
  "192.168.0.101" = "root=HVJtWTwVLnINjNK",
  "192.168.0.102" = "root=HVJtWTwVLnINjNK" }
 DiskPaths = {
  "[SharedStorage2] VxSwapHost2_1/VxSwapHost1_1.vmdk" = "0:1",
  "[SharedStorage2] VxSwapHost2_1/VxSwapHost1_2.vmdk" = "0:2",
  "RDM:[SharedStorage2] VxSwapHost2_1/VxSwapHost1_3.vmdk" = "0:3" }
 VirtualDiskMode = independent_persistent
)
```

Sample configuration for vCenter:

```
VMwareDisks VMwareDisks_1 (
ESXDetails = { "192.168.0.100" = "administrator=HVJtWTwVLnINjNK" }
DiskPaths = {
 "[SharedStorage2] VxSwapHost2_1/VxSwapHost1_1.vmdk" = "0:1",
 "[SharedStorage2] VxSwapHost2_1/VxSwapHost1_2.vmdk" = "0:2",
 "RDM:[SharedStorage2] VxSwapHost2_1/VxSwapHost1_3.vmdk" = "0:3" }
VirtualDiskMode = independent_persistent
)
```

Sample configuration where UUID is provided for the DiskPaths attribute:

```
VMwareDisks VMwareDisks_1 (
ESXDetails = { "192.168.0.100" = "root=HVJtWTwVLnINjNK",
 "192.168.0.101" = "root=HVJtWTwVLnINjNK",
 "192.168.0.102" = "root=HVJtWTwVLnINjNK" }
DiskPaths = {"RDM:6000C29a-11a3-7845-029d-10737a83ced7:
 [SharedStorage2] VxSwapHost2_1/VxSwapHost1_3.vmdk" = "0:3" }
VirtualDiskMode = independent_persistent
)
```

SFCache agent

The SmartIO feature of Veritas InfoScale enables data efficiency on your SSDs through I/O caching. Using SmartIO to improve efficiency, you can optimize the cost per I/O per second (IOPS). SmartIO uses advanced, customizable heuristics to determine what data to cache and how that data gets removed from the cache. The heuristics take advantage of Veritas InfoScale's knowledge of the characteristics of the workload.

SmartIO uses a cache area on the target device or devices. The cache area is the storage space that SmartIO uses to store the cached data and the metadata about the cached data. The type of the cache area determines whether it supports VxFS caching or VxVM caching. To start using SmartIO, you can create a cache area with a single command, while the application is online.

For more information about SmartIO, see *Veritas InfoScale SmartIO for Solid State Drives Solutions Guide*.

The SFCache agent enables, disables, and monitors cache. In case of a cache fault, the application still runs without any issues on the very same system, but with degraded I/O performance. Considering this, the SFCache agent provides an attribute to control the agent behavior. You can either choose to "IGNORE” or initiate “FAILOVER” in case of cache fault.

The SmartIO feature allows more than one cache area for VxFS and one cache area for VxVM on a single node; all object-level caches are created in these cache areas. An SFCache resource is configured per object (either mount point or volume) for which the SmartIO feature needs to be enabled. For VxFS caching, the SFCache resource depends on the Mount or CFSMount resource. For VxVM caching, the SFCache resource depends on the DiskGroup, Volume, VolumeSet, or CVMVolDg resource.

If the SmartIO feature is not enabled on a node, the SFCache agent will work as a simple FileOnOff agent. The SFCache resource state will be reported as ONLINE/OFFLINE, but caching-related operations will not be performed.

Resource dependency

Figure 2-11 shows sample SFCache resource dependency for VxFS caching. The SFCache resource can depend on the Mount or CFSMount resource.

Figure 2-11 Sample SFCache resource dependency (VxFS caching)

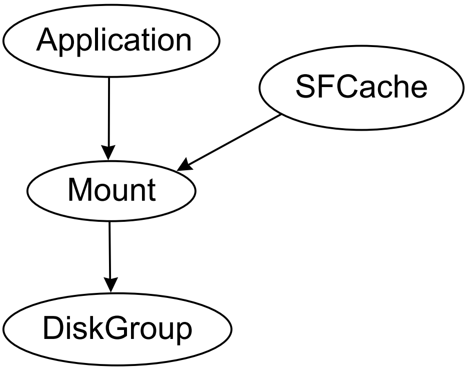
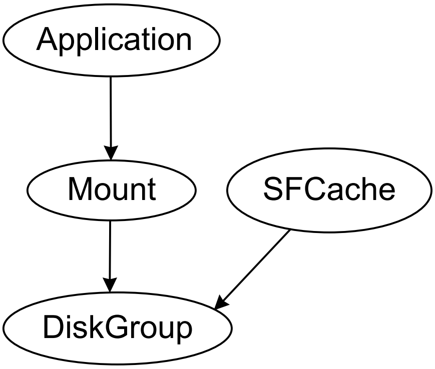


Figure 2-12 shows sample SFCache resource dependency for VxVM caching. The SFCache resource can depend on the DiskGroup, Volume, VolumeSet, or CVMVolDg resource.

Figure 2-12 Sample SFCache resource dependency (VxVM caching)



Agent functions

Online	Sets the cache mode as configured in the CacheMode attribute and finds out the type of cache area. Following are the types: ▪ VxFS ▪ VxVM
Offline	Disables the cache for a configure object.
Clean	Disable the cache for a configured object.

Monitor — Monitors the state of the caching for a configured cache object.

Attr_changed — Validates and makes necessary changes if the CacheMode attribute is reconfigured.

State definitions

The state definitions for this agent follow:

ONLINE — Indicates that caching is enabled for the configured cache object using the appropriate mode. If caching cannot be done on a node, the resource reports ONLINE even if there is no caching.

OFFLINE — Indicates that caching is disabled or the configured cache object is not active.

FAULTED — Indicates that caching is disabled for the specified cache object.

UNKNOWN — Indicates that invalid caching mode is specified for a cache object or the specified resource configuration is invalid.

Attributes

This section describes the attributes of the SFCache agent.

Table 2-18 Required attributes

Attribute	Description
CacheObjectName	Specifies the cache object name; it can be a mount point or disk group/volume. Type and dimension: string-scalar

CacheArea	Specifies the name of the cache areas.
	The writeback cache area is valid only when `CacheMode = writeback` and if it is a VxFS cache area.
	Type and dimension: string-association
	Example:
	`CacheArea @sys1 = { Read = sfcachearea_11, Writeback = sfcachearea_12 }`
	`CacheArea @sys2 = { Read = sfcachearea_21, Writeback = sfcachearea_22 }`

Table 2-19 Optional attributes

Attribute	Description
CacheMode	Specifies the caching mode. Following are the caching modes: ■ read ■ writeback ■ nocache Type and dimension: string-scalar Default: read
CacheFaultPolicy	Specifies the action to be performed in case of a cache fault. Following are the values: ■ IGNORE ■ FAILOVER Type and dimension: string-scalar Default: IGNORE

Table 2-20 Internal attributes

Attribute	Description
FaultOnMonitorTimeouts	Defines whether VCS interprets the Monitor timeout as a resource fault. By default, the FaultOnMonitorTimeouts attribute is set to 4, but the SFCache agent overrides this value and sets it to 0. The Monitor function must timeout four times in a row before the resource is marked as faulted. The first monitor timeout timer and the counter of timeouts are reset after one hour of the first monitor timeout. If the attribute is set to 0, VCS does not treat Monitor timeout as a resource fault. If the attribute is set to 1, VCS interprets the Monitor timeout as a resource fault and the agent calls the Clean function to shutdown the resource. **Note:** If the CacheFaultPolicy attribute is set to FAILOVER, Veritas recommends to set the FaultOnMonitorTimeouts attribute to 4. Default: 0

Table 2-20 Internal attributes *(continued)*

Attribute	Description
NumThreads	Number of threads that are used within the agent process for managing resources. This number does not include the number of threads that are used for other internal purposes. Setting the NumThreads attribute to a higher value may decrease the time required to go online or the time required to monitor a large number of SFCache resources. **Note:** If the NumThreads value is greater than 1, then there is a possibility that the `sfcache` command may fail. Type and dimension: static integer-scalar Default: 1

Resource type definition

```
type SFCache (
        static boolean IntentionalOffline = 1
        static int NumThreads = 1
        static int FaultOnMonitorTimeouts = 0
        static keylist RegList = { CacheMode }
        static str ArgList[] = { CacheObjectName,
         CacheArea, CacheMode, CacheFaultPolicy }
        str CacheObjectName
        str CacheArea{} = { Read=NONE, Writeback=NONE }
        str CacheMode = read
        str CacheFaultPolicy = IGNORE
)
```

Notes for SFCache agent

The SFCache agent has the following note:

- Configuring SFCache resource with CVM/CFS

Configuring SFCache resource with CVM/CFS

To use the SFCache agent with CFS setup, you must configure the SFCache resource in a separate parallel service group with online local soft dependency between the SFCache service group and CFSMount service group.

The following is a sample configuration:

```
group cfssg (
        SystemList = { sysA = 0, sysB = 1 }
        Parallel = 1
        AutoStartList = { sysA, sysB }
        )

        CFSMount cfsmount1 (
                MountPoint = "/cfsmnt"
                BlockDevice = "/dev/vx/dsk/cfsdg01/cfsvol01"
                MountOpt @sysA = rw
                MountOpt @sysB = rw
                NodeList = { sysA, sysB }
                )

        CVMVolDg cvmvoldg1 (
                CVMDiskGroup = cfsdg01
                CVMVolume = { cfsvol01 }
                CVMActivation @sysA = sw
                CVMActivation @sysB = sw
                )

        requires group cvm online local firm
        cfsmount1 requires cvmvoldg1

group cfs_sfcache_sg (
        SystemList = { sysA = 0, sysB = 1 }
        Parallel = 1
        AutoStartList = { sysA, sysB }
        )

        SFCache sfcache1 (
                CacheMode = read
                CacheFaultPolicy = IGNORE
                CacheObjectName = "/cfsmnt"
                CacheArea @sysA = { Read = sfcachearea_11 }
                CacheArea @sysB = { Read = sfcachearea_21 }
                )

        requires group cfssg online local soft
```

Debug log levels

The SFCache agent uses the following debug log levels:

DBG_3, DBG_4

Chapter 3

Network agents

This chapter includes the following topics:

About the network agents

Use network agents to provide high availability for networking resources.

All networking agents IP, NIC, IPMultiNICB, and MultiNICB agents support IPv4 as well as IPv6 protocols.

Agent comparisons

Agent comparisons may be made as described in the following sections.

IP and NIC agents

The IP and NIC agents:

- Monitor a single NIC

IPMultiNICB and MultiNICB agents

The IPMultiNICB and MultiNICB agents:

- Monitor single or multiple NICs
- Check the backup NICs as soon as it comes up
- Require a pre-assigned base IP address for each NIC
- Do not fail over the original base IP address
- Have more than one active NIC at a time

802.1Q trunking

The IP/NIC and IPMultiNICB/MultiNICB agents support 802.1Q trunking.

On Solaris, VLAN is not supported on the Fast Ethernet interfaces. (for example, hme/qfe interfaces).

You need to specify the VLAN interfaces, for example: bge20001 , bge30001, as the base interfaces in the device list in the main.cf file. You also must make sure that the IP addresses that are assigned to the interfaces of a particular VLAN are in the same subnet.

Link aggregation support

The link aggregation feature aggregates multiple network interfaces so that they appear as a single interface. For example, you can combine bge0 and bge1 and name the combined interface aggr100.

You can use the NIC or MultiNICB agents to monitor an aggregated interface. You can use the IP or IPMultiNICB agent respectively to configure and monitor an IP address on the aggregated interface.

All VCS networking agents support link aggregation. However, VCS has no control over the local adapter swapping performed by the link aggregation module. For guidelines on creating and managing link aggregations, refer to the topic Overview of Link Aggregations in the System Administration Guide: IP Services guide available on the Oracle Solaris documentation website.

Note: After you set up an aggregated interface, the constituting interfaces are no longer configurable. Hence, you must specify aggregated interfaces while configuring the Device attribute of IP, NIC, or MultiNICB resources.

IP agent

The IP agent manages the process of configuring a virtual IP address and its subnet mask on an interface. The virtual IP address must not be in use. You can use this agent when you want to monitor a single IP address on a single adapter.

The interface must be enabled with a physical (or administrative) base IP address before you can assign it a virtual IP address.

This agent is zone-aware. The ContainerOpts resource type attribute for this type has a default value of 0 for RunInContainer and a default value of 1 for PassCInfo. Veritas recommends that you do not change these values.

Refer to the *Veritas InfoScale Virtualization Guide*.

High availability fire drill for IP agent

The high availability fire drill detects discrepancies between the VCS configuration and the underlying infrastructure on a node. These discrepancies might prevent a service group from going online on a specific node. For IP resources, the high availability fire drill:

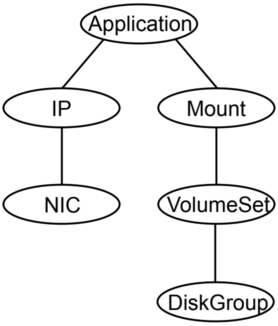
- Checks for the existence of a route to the IP from the specified NIC
- Checks for the existence of the interface configured in the IP resource

For more information about using the high availability fire drill see the *Cluster Server Administrator's Guide*.

Dependencies for IP agent

IP resources depend on NIC resources.

Figure 3-1 Sample service group that includes an IP resource



Agent functions for IP agent

Online

Configures the IP address to the NIC. Checks if another system is using the IP address. Uses the `ifconfig` command to set the IP address on a unique alias on the interface.

Action

- route.vfd

  Checks for the existence of a route to the IP from the specified NIC.
- device.vfd

  Checks for the existence of the interface configured in the Device attribute.

Offline

Brings down the IP address that is specified in the Address attribute.

Monitor

Monitors the interface to test if the IP address that is associated with the interface is alive.

Clean

Brings down the IP address that is specified in the Address attribute.

State definitions for IP agent

The state definitions for this agent follow:

ONLINE

Indicates that the device is up and the specified IP address is assigned to the device.

OFFLINE

Indicates that the device is down or the specified IP address is not assigned to the device.

UNKNOWN

Indicates that the agent could not determine the state of the resource or that the resource attributes are invalid.

FAULTED

Indicates that the IP address could not be brought online, usually because the NIC configured in the IP resource is faulted or the IP address was removed out of VCS control.

Attributes for IP agent

Table 3-1 Required attributes

Required attribute	Description
Address	A virtual IP address that is associated with the interface. Note that the address you specify must not be the same as the configured physical IP address, but should be on the same network. Type and dimension: string-scalar Example: "192.203.47.61" IPv6: "2001::10"
Device	The name of the NIC device that is associated with the IP address. Requires the device name without an alias. Type and dimension: string-scalar Example: "e1000g0"
One of the following attributes: ■ NetMask: Mandatory only if you configure an IPv4 address. ■ PrefixLen: Mandatory only if you configure an IPv6 address.	
NetMask	The subnet mask that is associated with the IP address of the resource. Specify the value of the netmask in decimal (base 10) or hexadecimal (base 16). You must configure this attribute if the IP address is an IPv4 address. Type and dimension: string-scalar Example: "255.255.248.0"
PrefixLen	This is the prefix for the IPv6 address represented as the CIDR value. When you use the IPv6 protocol, you must configure values for this attribute and the corresponding NIC agent’s Device and Protocol attributes. Type-dimension: integer-scalar Range: 1 - 128 Example: 64

Table 3-2 Optional attributes

Optional attribute	Description
RouteOptions	Specifies the routing options that are passed to the `route add` command when the agent configures an interface. The RouteOptions attribute value is generally formed like this: "*`destination gateway metric`*". For details about the `route` command, refer to the man page for your operating system. When the value of this string is null, the agent does not add routes. Type and dimension: string-scalar Example: "192.100.201.0 192.100.201.7" In this example, the agent executes the "`route add 192.100.201.0 192.100.201.7`" command when it configures an interface.
ExclusiveIPZone	Specifies that a resource is configured for an exclusive IP zone. Set this value to 1 if resource is configured for exclusive IP zone. When set to 1, it requires a valid ContainerInfo to be configured in the service group. Type-dimension: boolean-scalar Default: 0
Options	Options for the `ifconfig` command. Refer to the *Solaris ifconfig(1M) manual page* for the complete list of ifconfig options. Veritas recommends setting the IpadmIfProperties and IpadmAddrProperties attributes for Solaris 11. If the Options attribute is specified for Solaris 11, then ifconfig options are set as specified in the Options attribute. If IpadmIfProperites or IpadmAddrProperties attribute is also specified along with the Options attribute, then ipadm properties are set first and then ifconfig options are set as specified in the Options attribute. Type and dimension: string-scalar Example: "failover"

Table 3-2 Optional attributes *(continued)*

Optional attribute	Description
IpadmIfProperties	IpadmIfProperties attribute is applicable for Solaris 11 only. Interface properties for the `ipadm set-ifprop` command. Refer to the *Solaris ipadm(IM) manual page* for properties that can be set using the `ipadm set-ifprop` command. In this attribute, you can specify all the properties you want to set using the `ipadm set-ifprop` command as "key=value" strings. The agent passes the key-value pair one at a time to the `ipadm set-ifprop` command. Type and dimension: string-vector Example: "mtu=1400" "forwarding=on"
IpadmAddrProperties	IpadmAddrProperties attribute is applicable for Solaris 11 only. Address properties for the `ipadm set-addrprop` command. Refer to the *Solaris ipadm(IM) manual page* for properties that can be set using the `ipadm set-addrprop` command. In this attribute, you can specify all the properties you want to set using the `ipadm set-addrprop` command "key=value" strings. The agent passes the key-value pair one at a time to the `ipadm set-addrprop` command. Type and dimension: string-vector Example: "deprecated=on" "private=on"

Resource type definition for IP agent

The resource definition for this agent on Solaris follows:

```
type IP (
        static keylist RegList = { NetMask }
        static keylist SupportedActions = { "device.vfd",
         "route.vfd" }
        static str ArgList[] = { Device, Address, NetMask,
         Options, ArpDelay, IfconfigTwice, RouteOptions,
         PrefixLen, ExclusiveIPZone, IpadmIfProperties,
         IpadmAddrProperties }
        static int ContainerOpts{} = { RunInContainer=0,
         PassCInfo=1 }
        str Device
        str Address
        str NetMask
        str Options
        str IpadmIfProperties[]
        str IpadmAddrProperties[]
        int ArpDelay = 1
        int IfconfigTwice
        str RouteOptions
        int PrefixLen
        boolean ExclusiveIPZone = 0
)
```

Sample configurations for IP agent

The sample configurations for this agent follow:

Configuration 1

Configuration 1 for Solaris follows:

```
IP IP_192_203_47_61 (
Device = le0
Address = "192.203.47.61"
NetMask = "255.255.248.0"
)
```

NetMask in decimal (base 10)

The NetMask for this agent follows:

```
IP IP_192_203_47_61 (
Device = e1000g0
Address = "192.203.47.61"
NetMask = "255.255.248.0"
)
```

Configuration of NetMask in hexadecimal (base 16)

The NetMask for this agent follows:

```
IP IP_192_203_47_61 (
Device = e1000g0
Address = "192.203.47.61"
NetMask = "0xfffff800"
)
```

Debug log levels for IP agent

The IP agent uses the following debug log levels:

DBG_1, DBG_3, DBG_5

NIC agent

The NIC agent monitors the configured NIC. If a network link fails, or if a problem arises with the NIC, the resource is marked FAULTED. You can use the agent to make a single IP address on a single adapter highly available. This resource's Operation value is None.

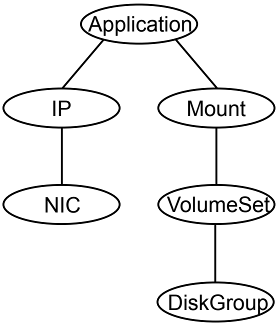
The ContainerName and ContainerType attributes are deprecated.

This agent is zone-aware. The ContainerOpts resource type attribute for this type has a default value of 0 for RunInContainer and a default value of 1 for PassCInfo. Veritas recommends that you do not change these values. Refer to the *Veritas InfoScale Virtualization Guide* for more information.

Dependencies for NIC agent

The NIC resource does not depend on any other resources.

Figure 3-2 Sample service group that includes a NIC resource



The NIC listed in the Device attribute must have a base IP address. The base IP address is the default IP address that is assigned to the physical interface of a host on a network. This agent does not configure network routes or base IP addresses.

Before you use this agent:

- Verify that the NIC has the correct base IP address and subnet mask.
- Verify that the NIC does not have built-in failover support. If it does, disable it.

Agent functions for NIC agent

Monitor	■ Tests the network card and network link. Pings the network hosts or broadcast address of the interface to generate traffic on the network. Counts the number of packets passing through the device before and after the address is pinged. If the count decreases or remains the same, the resource is marked FAULTED. If the NetworkHosts list is empty, or the ping test fails, the agent sends a ping to the device’s broadcast address to generate network traffic. The agent checks for any response to the broadcast request. If there is no reply to the broadcast ping, the resource faults.
Action	■ clearNICFaultInZone This action entry point clears the fault on the NIC device that is configured for Exclusive IP zone. ■ device.vfd Checks for the existence of the interface configured in the Device attribute.

State definitions for NIC agent

The state definitions for this agent follow:

ONLINE	Indicates that the NIC resource is working.
FAULTED	Indicates that the NIC has failed.
UNKNOWN	Indicates the agent cannot determine the interface state. It may be due to an incorrect configuration.

Attributes for NIC agent

Table 3-3 Required attributes

Required attribute	Description
Device	Name of the NIC that you want to monitor. Type and dimension: string-scalar Example: "e1000g0"

Table 3-4 Optional attributes

Optional attribute	Description
NetworkHosts	List of hosts on the network that are pinged to determine if the network connection is alive. You can use this attribute to help to save network capacity and reduce monitor time. Veritas recommends that you use the outgoing gateway routers for this value. Enter the IP address of the host, instead of the host name, to prevent the monitor from timing out. DNS causes the ping to hang. If more than one network host is listed, the monitor returns ONLINE if at least one of the hosts is alive. If an invalid network host address is specified, the resource enters an UNKNOWN state. If you do not specify network hosts, the monitor tests the NIC by sending pings to the broadcast address on the NIC. Type and dimension: string-vector Example: {"166.93.2.1", "166.99.1.2", "2620:128:f0a2:9005::1" }
NetworkType	Type of network. VCS supports only Ethernet. Type and dimension: string-scalar Default: "ether"

Table 3-4 Optional attributes *(continued)*

Optional attribute	Description
PingOptimize	Determines whether to ping during every monitor cycle. The value can be: ■ 0: The agent pings either the network host or the broadcast address during every monitor cycle. It pings during each cycle to determine the state of the network interface. ■ 1: The agent uses the device statistics from the output of the `netstat` command to determine the state of the network interface. If no activity exists on the interface, the agent pings the broadcast address to double-check the state of the network interface. Type and dimension: integer-scalar Default: 1
ExclusiveIPZone	ExclusiveIPZone Specifies that a resource is configured for an exclusive IP zone. Set this value to 1 if a resource is configured for exclusive IP zone. When set to 1, it requires a valid ContainerInfo to be configured in the service group. Type-dimension: boolean-scalar Default: 0

Resource type definition for NIC agent

The resource definition for this agent on Solaris follows:

```
type NIC (
static keylist SupportedActions = { "device.vfd",
"clearNICFaultInZone" }
static str ArgList[] = { Device, PingOptimize, NetworkHosts,
NetworkType, ExclusiveIPZone }
static int ContainerOpts{} = { RunInContainer=0, PassCInfo=1 }
static int OfflineMonitorInterval = 60
static str Operations = None
str Device
int PingOptimize = 1
str NetworkHosts[]
str NetworkType
boolean ExclusiveIPZone = 0
)
```

Notes for the NIC agent

The NIC agent has the following notes:

- High availability fire drill for NIC agent
- Solaris 11: Change of behavior in IPv4 and IPv6 interface state

High availability fire drill for NIC agent

The high availability fire drill detects discrepancies between the VCS configuration and the underlying infrastructure on a node. These discrepancies might prevent a service group from going online on a specific node. For NIC resources, the high availability fire drill checks for the existence of the NIC on the host.

For more information about using the high availability fire drill, see the *Cluster Server Administrator's Guide*.

Solaris 11: Change of behavior in IPv4 and IPv6 interface state

On Solaris 11, `ipadm` and `ifconfig` commands are supported, but the OS recommends to use the `ipadm` command for managing IP and NIC. The `ifconfig` command allows IPv4 and IPv6 IPs to be plumbed only after the NIC is plumbed for the respective protocol. Therefore, to plumb an IPv4 address, NIC has to be plumbed for IPv4 and to plumb an IPv6 address, NIC has to be plumbed for IPv6.

Prior to VCS 6.2, on Solaris 11, the NIC agent used to monitor the IPv4-specific-NIC or IPv6-specific-NIC configured using the Protocol attribute of the NIC resource. In VCS 6.2 and going forward, the NIC agent will use the `ipadm` command to monitor the network interface.

The `ipadm` command does not distinguish between the NIC plumbed for IPv4 or IPv6. If any IP (IPv4 or IPv6) is brought online on the NIC interface, the `ipadm` command shows the interface as online. This change of behavior is also reflected in the NIC agent. Hence, VCS 6.2 onwards, if the NIC has an active IPv4 IP and no IPv6 IP, the IPv6 NIC resource detects the NIC as online. Similarly, if the NIC has an active IPv6 IP and no IPv4 IP, the IPv4 NIC resource detects the NIC as online.

Sample configurations for NIC agent

Configuration without network hosts (using default ping mechanism) for NIC agent

```
NIC groupx_e1000g0 (
    Device = e1000g0
    PingOptimize = 1
    )
```

Configuration with network hosts for NIC agent

Network hosts for Solaris configuration follows:

```
NIC groupx_e1000g0 (
    Device = e1000g0
    NetworkHosts = { "166.93.2.1", "166.99.1.2" }
    )
```

IPv6 configuration for NIC agent

The following is a basic configuration for IPv6 with IP and NIC resources.

```
group nic_group (
SystemList = { sysA = 0, sysB = 1 }
Parallel = 1
)
NIC nic_resource (
Device@sysA = bge0
Device@sysB = bge1
PingOptimize = 0
NetworkHosts@sysA = { "2001:db8:c18:2:214:4fff:fe96:11",
"2001:db8:c18:2:214:4fff:fe96:1" }
NetworkHosts@sysB = { "2001:db8:c18:2:214:4fff:fe96:1111",
"2001:db8:c18:2:214:4fff:fe96:111" }
)
Phantom phantom_resource (
)
group ip_group (
SystemList = { sysA = 0, sysB = 1 }
)
IP ip_resource (
Device@sysA = bge0
Device@sysB = bge1
Address = "2001:db8:c18:2:214:4fff:fe96:102"
PrefixLen = 64
)
Proxy proxy_resource (
TargetResName = nic_resource
)
ip_resource requires proxy_resource
```

Exclusive IP Zone configuration for NIC agent

Following is the configuration example for Exclusive IP zone with NIC and IP resources. In the following sample, nic_value represents the base NIC name (for example, bge0) and zone_name is the name of the exclusive IP zone. (For more details about Zone resource configuration, refer to the Zone agent section.)

```
group grp1 (
SystemList = { sysA = 0 }
ContainerInfo@sysA = { Name = zone_name, Type = Zone,
Enabled = 1 }
AutoStartList = { sysA }
Administrators = { z_zone_res_sysA }
)
IP ip_res (
Device = nic_value
Address = "166.93.3.10"
NetMask = "255.255.255.0"
ExclusiveIPZone = 1
)
NIC nic_res (
Device = nic_value
NetworkHosts = { "166.93.3.1" }
ExclusiveIPZone = 1
)
Zone zone_res (
)
ip_res requires nic_res
ip_res requires zone_res
```

Note that whenever a fault is detected for a NIC resource configured in an exclusive IP zone, perform the following steps to clear the fault.

1. Repair the device configured with NIC resource. Verify that the device is healthy (check for cable connectivity, network connectivity, and so on).
2. If the state of the exclusive IP zone on the system on which the NIC was faulted is:
   - Running: No action is required, and the next NIC monitor cycle will clear the fault after detecting the healthy NIC device.
   - NOT running: Clear the fault on the NIC device by invoking 'clearNICFaultInZone' action entry point for the NIC resource as follows:
     # hares -action nic_res clearNICFaultInZone -sys sysA

Note: When a NIC resource is configured for an Exclusive IP zone, Veritas recommends to set the ToleranceLimit attribute to a non-zero value.

With a NIC resource configured for an Exclusive IP zone, the NIC resource is monitored inside the zone when the zone is functional. While the zone is shutting down if the NIC monitor program is invoked, the monitor may falsely report the NIC resource as offline. This may happen if some of the networking services are offline but the zone is not completely shut down. Such reports can be avoided if you override and set the ToleranceLimit value to a non-zero value.

Calculate the ToleranceLimit value as follows: Time taken by a zone to completely shut down must be less than or equal to NIC resource's MonitorInterval value + (MonitorInterval value x ToleranceLimit value).

For example, if a zone takes 90 seconds to shut down and the MonitorInterval for NIC agent is set to 60 seconds (default value), set the ToleranceLimit value to 1.

Debug log levels for NIC agent

The NIC agent uses the following debug log levels:

DBG_1, DBG_5

About the IPMultiNICB and MultiNICB agents

The IPMultiNICB and the MultiNICB agents can handle multiple NIC connections. Due to differences in the way that each platform handles its networking connections, these agents vary in design between platforms.

Checklist to ensure the proper operation of MultiNICB

For the MultiNICB agent to function properly, you must satisfy each item in the following list:

- Each interface must have a unique MAC address.
- A MultiNICB resource controls all the interfaces on one IP subnet.
- At boot time, you must configure and connect all the interfaces that are under the MultiNICB resource and give them base IP addresses.
- All base IP addresses for the MultiNICB resource must belong to the same subnet as the virtual IP address.
- Reserve the base IP addresses, which the agent uses to test the link status, for use by the agent. These IP addresses do not get failed over.
- The IgnoreLinkStatus attribute is set to 1 (default) when using trunked interfaces.
- If you specify the NetworkHosts attribute, then that host must be on the same subnet as the base IP addresses for the MultiNICB resource.
- Test IP addresses have "nofailover" and "deprecated" flags set at boot time.
- /etc/default/mpathd has TRACK_INTERFACES_ONLY_WITH_GROUPS=yes.
- If you are not using Solaris in.mpathd, all MultiNICB resources on the system have the UseMpathd attribute set to 0 (default). You cannot run in.mpathd on this system.
- If you are using Solaris in.mpathd, all MultiNICB resources on the system have the UseMpathd attribute set to 1.

IPMultiNICB agent

The IPMultiNICB agent works with the MultiNICB agent. The agent configures and manages virtual IP addresses (IP aliases) on an active network device that the MultiNICB resource specifies. When the MultiNICB agent reports a particular interface as failed, the IPMultiNICB agent moves the virtual IP address to the next active interface. You can use this agent for IP addresses on multiple-adapter systems.

If multiple service groups have IPMultiNICB resources associated with the same MultiNICB resource, only one group should have a MultiNICB resource. The other groups should have a proxy resource pointing to the MultiNICB resource.

For the MultiNICB and IPMultiNICB agents, VCS supports Oracle trunking.

The ContainerName and ContainerType attributes are deprecated.

This agent is zone-aware. The ContainerOpts resource type attribute for this type has a default value of 0 for RunInContainer and a default value of 1 for PassCInfo. Veritas recommends that you do not change these values.

Refer to the *Veritas InfoScale Virtualization Guide*.

The value of the MonitorInterval attribute for the MultiNICB type must be less than its value for the IPMultiNICB type. The IPMultiNICB agent relies on the MultiNICB agent to accurately report the state of the NICs. If the value of the MonitorInterval of the IPMultiNICB agent is less than the interval for the MultiNICB agent, then in some monitor cycles the potential exists for the MultiNICB agent to provide the IPMultiNICB agent stale information.

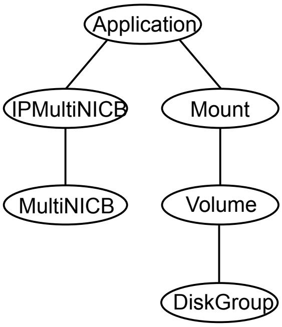
The default value for the MonitorInterval attribute for IPMultiNICB is 30 seconds, and the minimum and default value for MultiNICB is 10 seconds.

Dependencies for IPMultiNICB agent

IPMultiNICB resources depend on MultiNICB resources.

They can also depend on Zone resources.

Figure 3-3 Sample service group that includes an IPMultiNICB resource



Requirements for IPMultiNICB

The following conditions must exist for the IPMultiNICB agent to function correctly:

- The MultiNICB agent must be running to inform the IPMultiNICB agent of the available interfaces.
- One IPMultiNICB agent can control only one virtual IP address.

Agent functions for IPMultiNICB agent

Online

Finds a working interface with the appropriate interface alias or interface name, and configures the virtual IP address on it.

Offline

Removes the virtual IP address.

Clean

Removes the virtual IP address.

Monitor

If the virtual IP address is not configured as an alias on one of the working interfaces under a corresponding MultiNICB resource, monitor returns OFFLINE. If the current interface fails, the agent fails over the virtual IP address to the next available working interface that is within the MultiNICB resource on the same node. If no working interfaces are available then monitor returns OFFLINE.

State definitions for IPMultiNICB agent

The state definitions for this agent follow:

ONLINE

Indicates that the IP address is up on one of the working network interfaces of the MultiNICB resource. The IP address is specified in the Address attribute. The MultiNICB resource is specified in the BaseResName attribute.

OFFLINE

Indicates that the IP address is not up on any of the network interfaces of the MultiNICB resource. The IP address is specified in the Address attribute. The MultiNICB resource is specified in the BaseResName attribute.

UNKNOWN

Indicates that the agent cannot determine the status of the virtual IP address that is specified in the Address attribute.

FAULTED

Indicates that the virtual IP address could not be brought online, usually because all the interfacess configured in the MultiNICB resource have failed or the virtual IP address was removed out of VCS control.

Attributes for IPMultiNICB agent

Table 3-5 Required attributes

Required attribute	Description
Address	The logical IP address that the IPMultiNICB resource must handle. This IP address must be different than the base or test IP addresses in the MultiNICB resource. Type and dimension: string-scalar Example: "10.112.10.15"
BaseResName	Name of MultiNICB resource from which the IPMultiNICB resource gets a list of working interfaces. The logical IP address is placed on the physical interfaces according to the device number information. Type and dimension: string-scalar Example: "gnic_n"
One of the following attributes: ■ NetMask: Mandatory only if you configure an IPv4 address. ■ PrefixLen: Mandatory only if you configure an IPv6 address.	
NetMask	The netmask that is associated with the logical IP address. This attribute is required if you configure this resource for IPv4 protocol. Type and dimension: string-scalar Example: "255.255.255.0"
PrefixLen	This is the prefix for the IPv6 address represented as the CIDR value. When you use the IPv6 protocol, you must configure values for this attribute and the corresponding MultiNICB agent's Device attribute. Type-dimension: integer-scalar Range: 1 - 128 Example: 64

Table 3-6 Optional attributes

Optional attribute	Description
DeviceChoice	Indicates the preferred NIC where you want to bring the logical IP address online. Specify the device name or NIC alias as determined in the Device attribute of the MultiNICB resource. Type and dimension: string-scalar Default: 0 Examples: "qfe0" and "1"
RouteOptions	String to add or delete a route when configuring or unconfiguring an interface. Use the attribute only when configuring the local host as the default gateway. This string contains *"destination gateway metric route-add-modifiers*". No routes are added or deleted if this string is set to NULL. When DeleteRouteOptions attribute is not configured, RouteOptions attribute is used for adding and deleting the route. When DeleteRouteOptions attribute is configured, RouteOptions attribute is used to add route and DeleteRouteOptions attribute is used to delete route. Example: "default 166.98.16.103 0 -setsrc 166.98.16.10"
DeleteRouteOptions	String to delete a route when un-configuring an interface. When RouteOptions and DeleteRouteOptions attributes are configured, RouteOptions attribute is used to add route and DeleteRouteOptions attribute is used to delete route. When RouteOptions attribute is not configured, DeleteRouteOptions attribute is ignored. Example: "default 166.98.16.103"
IgnoreMultiNICBFailure	Set this value to ignore a MultiNICB resource failure when all configured interfaces fail. A value of 1 for this attribute causes the IPMultiNICB agent to ignore the failure that its underlying MultiNICB resource detects. A value of 0 for this attribute causes the IPMultiNICB agent to detect network failure. When the value of this attribute is 1, the value for the MultiNICB LinkTestRatio attribute cannot be 0. Type and dimension: integer-scalar Default: 0

Table 3-6 Optional attributes *(continued)*

Optional attribute	Description
Options	Options for the ifconfig command. Type and dimension: string-scalar Example: "failover"

Note: The value of the ToleranceLimit static attribute is 1. A value of 1 avoids spurious agent faults in the multi-pathing mode while Oracle's mpathd daemon migrates the IP address from one interface to another. Due to the change in the ToleranceLimit attribute, the value of the MonitorInterval static attribute is now 30 seconds. The 30-second value means that the agent tries to online the resource twice a minute. This value ensures that the overall fault detection time is still 60 seconds.

Resource type definition for IPMultiNICB agent

The resource definition for this agent on Solaris follows:

```
type IPMultiNICB (
static int ToleranceLimit = 1
static int MonitorInterval = 30
static int OnlineRetryLimit=1
static str ArgList[] = { BaseResName, Address, NetMask,
DeviceChoice, RouteOptions, DeleteRouteOptions, PrefixLen,
IgnoreMultiNICBFailure, "BaseResName:Protocol", Options }
static int ContainerOpts{} = { RunInContainer=0, PassCInfo=1 }
str BaseResName
str Address
str NetMask
str DeviceChoice = 0
str RouteOptions
str DeleteRouteOptions
str Options
int PrefixLen
int IgnoreMultiNICBFailure = 0
)
```

Manually migrating a logical IP address for IPMultiNICB agent

Use the haipswitch command to migrate the logical IP address from one interface to another.

This command shows the status of the interfaces for the specified MultiNICB resource:

```
# haipswitch -s MultiNICB_resname
```

In the following example, the command checks that both the *from* and *to* interfaces are associated with the specified MultiNICB resource. The command also checks if the *to* interface works. If the interface does not work, the command aborts the operation. It then removes the IP address on the *from* logical interface and configures the IP address on the *to* logical interface. It finally erases any previous failover information that is created by MultiNICB for this logical IP address.

```
# haipswitch MultiNICB_resname IPMultiNICB_resname ip_address
<netmask|prefix> from to [route_options]
```

Sample configurations for IPMultiNICB agent

The sample configurations for the IPMultiNICB and MultiNICB agent follows:

Other sample configurations for IPMultiNICB and MultiNICB

Refer to the sample configurations in the MultiNICB agent.

Debug log levels for IPMultiNICB agent

The IPMultiNICB agent uses the following debug log levels:

DBG_1, DBG_2, DBG_3, DBG_4, DBG_5

MultiNICB agent

The MultiNICB agent works with the IPMultiNICB agent. It allows IP addresses to fail over to multiple interfaces on the same system before VCS tries to fail over to another system. You can use the agent to make IP addresses on multiple-adapter systems highly available or to monitor them.

When you use the MultiNICB agent, you must configure the interfaces before putting them under the agent’s control. You must configure all the interfaces in a single MultiNICB resource with the base IP addresses that are in the same subnet.

If multiple service groups have IPMultiNICB resources associated with the same MultiNICB resource, only one group should have the MultiNICB resource. The other groups can have a proxy resource pointing to it.

For the MultiNICB and IPMultiNICB agents, VCS supports Oracle trunking.

For the MultiNICB and IPMultiNICB agents, VCS supports either IPv4 or IPv6.

The value of the MonitorInterval attribute for the MultiNICB type must be less than its value for the IPMultiNICB type. The IPMultiNICB agent relies on the MultiNICB agent to accurately report the state of the NICs. If the value of the MonitorInterval of the IPMultiNICB agent is less than the interval for the MultiNICB agent, then in some monitor cycles the potential exists for the MultiNICB agent to provide the IPMultiNICB agent stale information.

The default value for the MonitorInterval attribute for IPMultiNICB is 30 seconds, and the minimum and default value for MultiNICB is 10 seconds.

Base and Multi-pathing modes for MultiNICB agent

You can use the MultiNICB agent in one of two modes that follow:

- Base mode
- Multi-pathing mode

Oracle trunking for MultiNICB agent

You can configure MultiNICB for use with a single trunk head or multiple trunk heads. You need to set the value of the IgnoreLinkStatus attribute to 1. You must also ensure that all interfaces that belong to the same MultiNICB resource are in the same subnet.

The haping utility for MultiNICB agent

Use the haping utility (`/opt/VRTSvcs/bin/MultiNICB/haping`) to test each interface before you configure the MultiNICB resource. This utility takes the interface as an argument. You can use this utility to perform a link test, a broadcast ping, or to ping a specific remote host. Veritas recommends that the administrator perform a test ping with the remote host before adding it to the NetworkHosts parameter. Note that the remote host should be on the same network as the interface from which you are performing the test ping.

Link test only on interface bge0:

```
haping -l bge0
```

Ping a remote host 10.10.10.10 from interface bge0:

```
haping -g 10.10.10.10 bge0
```

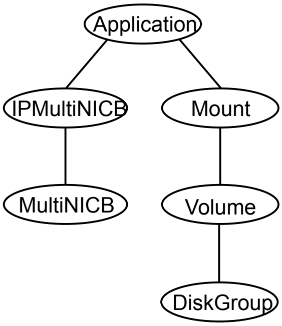
Ping a remote IPv6 host from interface bge0:

```
haping -g fe80::1 bge0
```

Dependencies for MultiNICB agent

The MultiNICB resource does not depend on any other resources.

Figure 3-4 Sample service group that includes a MultiNICB resource



Agent functions for MultiNICB agent

Open — Allocates an internal structure to store information about the resource.

Close — Frees the internal structure that is used to store information about the resource.

Monitor — Checks the status of each physical interface. Writes the status information to the export information file for IPMultiNICB resources to read it.

Performs a failover. Performs failback if the value of the Failback attribute is 1.

State definitions for MultiNICB agent

ONLINE — Indicates that one or more of the network interfaces listed in the Device attribute of the resource is in working condition.

UNKNOWN — Indicates that the MultiNICB resource is not configured correctly.

FAULTED	Indicates that all of the network interfaces listed in the Device attribute failed.

Attributes for MultiNICB agent

Table 3-7 Required attributes

Required attribute	Description
Device	List of NICs that you want under MultiNICB control. For IPv4, you must manually configure the test IP addresses on each NIC. This address must be the base IP address (the first address) on that NIC. For IPv6, VCS uses link local addresses as the test IP addresses. You do not need to configure the test IP address. List of NICs that you want under MultiNICB control, and the failover order of those NICs. The IPMultiNICB agent uses the NIC aliases to configure IP addresses. The IPMultiNICB agent uses these interface aliases to determine the order of the interface on which to bring the IP addresses online. When you use the IPv6 protocol, you must configure values for this attribute, the Protocol attribute, and the corresponding IPMultiNICB agent's PrefixLen attribute. Type and dimension: string-association Examples: Device = { "qfe0" = 0, "qfe1" = 1 }
Protocol	Required to use the IPv6 protocol. See Table 3-8 on page 121.
IPMPDevice (Solaris 11 only)	Set the value of the attribute to the name of the IPMP interface created for interfaces under MultiNICB resource control. The value of the attribute must be set to a valid IPMP interface name and the value of UseMpathd attribute must be set to 1. Example: "ipmp0" **Note:** Before you configure the attribute, you want to configure an IP multi-pathing (IPMP) interface for the interfaces under MultiNICB resource control. See "Configuring MultiNICB and IPMultiNICB agents on Solaris 11" on page 130.

Optional attributes for Base and Mpathd modes for MultiNICB agent

Table 3-8 Optional attributes for Base and Mpathd modes

Optional attribute	Description
DefaultRouter	This attribute is the IP address of the default router on the subnet. If you specify this value, the agent removes the default route when the resource goes offline. The agent adds the route back when the group returns online. You must specify this attribute if multiple IP subnets exist on one host. If you do not specify the value, the packets cannot be routed properly when the subnet corresponding to the first default route goes down. Type and dimension: string-scalar Default: 0.0.0.0 Example: "192.1.0.1"
GroupName	The GroupName attribute is the name of the IPMP group that you want to assign to the interfaces under the control of the agent. The name’s length should not exceed 31 bytes characters. If this attribute is not specified, the MultiNICB resource name is configured as IPMP group name. Type and dimension: string-scalar Example: "IPMPgrp1"
MpathdCommand	This value is the path to the mpathd executable. Use MpathdCommand to kill or restart mpathd. See the UseMpathd attribute for details. Type and dimension: string-scalar Default: /usr/lib/inet/in.mpathd

Table 3-8 Optional attributes for Base and Mpathd modes *(continued)*

Optional attribute	Description
UseMpathd	The legal values for this attribute are 0 and 1. All the MultiNICB resources on one system must have the same value for this attribute. If the value of the attribute is 0, in.mpathd is automatically killed on that system. For more information about mpathd, refer to the Oracle documentation. If the value of the attribute is 1, MultiNICB assumes that mpathd (in.mpathd) is running. This value restarts mpathd if it is not running already. Type and dimension: integer-scalar Default: 0
Protocol	Specifies the type of IP protocol (IPv4 or IPv6) that you want to use with the agent. When you use the IPv6 protocol, you must configure values for this attribute, the Device attribute, and the corresponding IPMultiNICB agent’s PrefixLen attribute. Type-dimension: string-scalar Default: IPv4 Example: IPv6

Optional attributes for Base mode for MultiNICB agent

Table 3-9 Optional attributes for Base mode

Optional attribute	Description
Failback	If the value of the attribute is 1, the virtual IP addresses are failed back to the original physical interface whenever possible. A value of 0 disables this behavior. Type and dimension: integer-scalar Default: 0

Table 3-9 Optional attributes for Base mode *(continued)*

Optional attribute	Description
IgnoreLinkStatus	If the value of the attribute is 1, the agent ignores the driver-reported interface status while testing the interfaces. If the value of the attribute is 0, the agent reports the interface status as DOWN if the driver-reported interface status indicates the DOWN state. Using interface status for link testing may considerably speed up failovers. When you use trunked interfaces, you must set the value of this attribute to 1. Otherwise set it to 0. Type and dimension: integer-scalar Default: 1

Table 3-9 Optional attributes for Base mode *(continued)*

Optional attribute	Description
LinkTestRatio	This attribute is the ratio of: - The total monitor cycles to - The monitor cycles in which the agent tests the interfaces by sending packets At all other times, the agent tests the link by checking the "link-status" as reported by the device driver. Checking the "link-status" is a faster way to check the interfaces, but only detects cable disconnection failures. If the value of the attribute is 1, packets are sent during every monitor cycle. If the value of the attribute is 0, packets are never sent during a monitor cycle. Do not set the value of this attribute to 0 when its corresponding IPMultiNICB resource's IgnoreMultiNICBFailure attribute has a value of 1. Value of this attribute has no effect when IgnoreLinkStatus attribute is set to 1. In such scenario, packets are sent during every monitor cycle to determine the state of an interface. Type and dimension: integer-scalar Default: 1 Example: "3" In this example, if monitor entry-point invoking is numbered as 1, 2, 3, 4, 5, 6, ..., the actual packet send test is done at 3, 6, etc. monitor agent functions. For LinkTestRatio=4, the packet send test is done at 4, 8, etc., monitor agent functions.

Table 3-9 Optional attributes for Base mode *(continued)*

Optional attribute	Description
NetworkHosts	List of host IP addresses on the IP subnet that are pinged to determine if the interfaces work. NetworkHosts only accepts IP addresses to avoid DNS lookup delays. The IP addresses must be directly present on the IP subnet of interfaces (the hosts must respond to ARP requests). If IP addresses are not provided, the hosts are automatically determined by sending a broadcast ping (unless the NoBroadcast attribute is set to 1). The first host to reply serves as the ping destination. Type and dimension: string-vector Example: { "2620:128:f0a2:9001::1", "10.209.72.1" }
NetworkTimeout	Timeout for ARP and ICMP packets in milliseconds. MultiNICB waits for response to ICMP and ARP packets only during this time period. Assign NetworkTimeout a value in the order of tens of milliseconds (given the ICMP and ARP destinations are required to be on the local network). Increasing this value increases the time for failover. Type and dimension: integer-scalar Default: 100
NoBroadcast	If the value of the attribute is 1, NoBroadcast prevents MultiNICB from sending broadcast ICMP packets. Note that MultiNICB can still send ARP requests. If NetworkHosts are not specified and NoBroadcast is set to 1, the MultiNICB agent cannot function properly. **Note:** Veritas does not recommend setting the value of NoBroadcast to 1. Type and dimension: integer-scalar Default: 0

Table 3-9 Optional attributes for Base mode *(continued)*

Optional attribute	Description
OfflineTestRepeatCount	Number of times the test is repeated if the interface status changes from UP to DOWN. For every repetition of the test, the next NetworkHost is selected in round-robin manner. At the end of this process, broadcast is performed if NoBroadcast is set to 0. A greater value prevents spurious changes, but also increases the response time. Type and dimension: integer-scalar Default: 3
OnlineTestRepeatCount	Number of times the test is repeated if the interface status changes from DOWN to UP. This test helps to avoid oscillations in the status of the interface. Type and dimension: integer-scalar Default: 3

Optional attributes for Multi-pathing mode for MultiNICB agent

Table 3-10 Optional attributes for Multi-pathing mode

Optional attribute	Description
ConfigCheck	If the value of the attribute is 1, the MultiNICB agent checks for: ■ All specified physical interfaces are in the same IP subnet and group, and have "DEPRECATED" and "NOFAILOVER" flags set on them. ■ No other physical interface has the same subnet as the specified interfaces. If the value of the attribute is 1, the MultiNICB agent also assigns all the specified interfaces in one multi-pathing group. Valid values for this attribute are 0 and 1. Type and dimension: integer-scalar Default: 1

Table 3-10 Optional attributes for Multi-pathing mode *(continued)*

Optional attribute	Description
MpathdRestart	If the value of the attribute is 1, MultiNICB tries to restart mpathd. Valid values for this attribute are 0 and 1. Type and dimension: integer-scalar Default: 1

Resource type definition for MultiNICB agent

The resource definition for this agent on Solaris follows:

```
type MultiNICB (
  static int MonitorInterval = 10
  static int OfflineMonitorInterval = 60
  static str Operations = None
  static str ArgList[] = { UseMpathd, MpathdCommand, ConfigCheck,
  MpathdRestart, Device, NetworkHosts, LinkTestRatio,
  IgnoreLinkStatus, NetworkTimeout, OnlineTestRepeatCount,
  OfflineTestRepeatCount, NoBroadcast, DefaultRouter, Failback,
  GroupName, Protocol, IPMPDevice }
  int UseMpathd
  str MpathdCommand = "/usr/lib/inet/in.mpathd"
  int ConfigCheck = 1
  int MpathdRestart = 1
  str Device{}
  str NetworkHosts[]
  int LinkTestRatio = 1
  int IgnoreLinkStatus = 1
  int NetworkTimeout = 100
  int OnlineTestRepeatCount = 3
  int OfflineTestRepeatCount = 3
  int NoBroadcast
  str DefaultRouter = "0.0.0.0"
  int Failback
  str GroupName
  str Protocol = IPv4
  str IPMPDevice
  )
```

Solaris operating modes: Base and Multi-Pathing for MultiNICB agent

The MultiNICB agent has two modes of operation, Base and Multi-Pathing, which you can set with the UseMpathd attribute.

Note: On Oracle Solaris 11, only Multi-Pathing mode is supported for MultiNICB agent.

Base mode for MultiNICB agent

The value of the UseMpathd attribute is 0 by default for this mode.

In Base mode, to monitor the interfaces that it controls, the agent:

- sends the packets to other hosts on the network for probe-based detection
- tests the link status of the interfaces for link-based detection
- checks that all the specified interfaces are in the same IP subnet and group, and have "NOFAILOVER" and "DEPRECATED" flags set on them
- checks that no other physical interface has the same subnet as the specified interfaces
- assigns all the specified interfaces in one multi-pathing group

The agent logs link failures and failovers when it uses either link- or probe-based detection.

If a NIC goes down, the MultiNICB agent notifies the IPMultiNICB agent. The IPMultiNICB agent fails over the virtual IP addresses to a different NIC on the same system. When the original NIC comes up, the agents fail back the virtual IP address if the Failback attribute for the corresponding MultiNICB resource is set to 1.

Each NIC must have its own unique and exclusive base IP address, which the MultiNICB agent uses as the test IP address.

The MultiNICB agent, in Base mode, uses the following criteria to determine if an interface works:

- Link-based detection of the interface status

  The interface driver reports the status of the link. Note that not all drivers support this feature. Set the value of IgnoreLinkStatus to 1 to disable this test.
- Probe-based detection using Internet Control Message Protocol (ICMP) echo

  Set the LinkTestRatio attribute to a value greater than 0 to send ICMP echo request packets to a specified network host. You specify the network hosts in the NetworkHosts attribute. You must assign test IP addresses to the interface for probe-based detection. The test IP address is needed to send the ICMP packets, which determines the link's status. If you set the value of the LinkTestRatio attribute to 0, you do not need to assign test IP addresses.

If you specify no hosts in the NetworkHosts attribute, the agent uses the ICMP broadcast when the value of the NoBroadcast attribute is 0. It caches the sender of the first reply for future use as a network host. While the agent sends and receives ICMP packets, the IP layer is completely bypassed.

You can assign addresses and still do only link-based detection by setting the values of the LinkTestRatio attribute to greater than 0 and the IgnoreLinkStatus attribute to 0.

You can skip link-based detection (link driver tests) and only do ICMP tests if:

- the value of the IgnoreLinkStatus attribute is 1, and
- the value of the LimitTestRation attribute is greater than 0, and
- the test IP addresses are assigned to the interface

The MultiNICB agent performs both link-based detection and probe-based detection if:

- the value of the LinkTestRatio attribute is greater than 0, and
- the value of the IgnoreLinkStatus attribute is 0, and
- the test IP addresses are assigned to the interface

The MultiNICB agent writes the status of each interface to an export information file, which other agents (like IPMultiNICB) or commands (like `haipswitch`) can read.

Failover and failback for MultiNICB agent

During an interface failure, the MultiNICB agent fails over all logical IP addresses to a working interface under the same resource. The agent remembers the first physical interface from which an IP address was failed over. This physical interface becomes the "original" interface for the particular logical IP address. When the original interface is repaired, the logical IP address fails back to it.

Multi-Pathing mode for MultiNICB agent

To activate this mode set the value of the UseMpathd attribute to 1. The MultiNICB agent, in Multi-Pathing mode, monitors Oracle’s IP Multi-Pathing daemon (mpathd). The MultiNICB agent specifically monitors the FAILED flag on physical interfaces and the mpathd process. See the man page: in.mpathd (1M) for more information on this daemon.

Oracle’s mpathd daemon monitors the interfaces that are part of the IPMP group. The daemon:

- sends the packets to other hosts on the network for probe-based detection as long as a test IP address is assigned to the network interface
- checks the link status of the interfaces for link-based detection as long as the interface supports the test for detection

The mpathd daemon can perform both link- and probe-based detection when test IP addresses are assigned to NIC interfaces.

The MultiNICB agent logs errors when the daemon is not running, or if a configuration path error exits. The mpathd daemon logs link failures and IP address failovers in the system log.

Configuring MultiNICB and IPMultiNICB agents on Solaris 11

On Solaris 11 you must configure IPMultiNICB and MultiNICB resources as follows:

1. Create an IPMP interface manually for the interfaces under MultiNICB control.

   For more information, refer to *Oracle Solaris Administration: Network interfaces and Network Virtualization Guide*.
2. Specify the IPMP interface name as the IPMPDevice attribute value of the MultiNICB resource.
3. Set the UseMpathd and ConfigCheck attribute values of the MultiNICB resource to 1 and 0 respectively.
4. Ensure that the IPMP interface and corresponding base interfaces are configured correctly and are up before enabling the MultiNICB resource.

   Note: If you configure IPv6 address in the IPMultiNICB agent, you must configure addrconf IPv6 address on the IPMP device. Use the following command to create the IPv6 addrconf.

   ```
   ipadm create-addr -T addrconf -p stateless=no addrobj
   ```

Trigger script for MultiNICB agent

MultiNICB monitor agent function calls a VCS trigger in case of an interface going up or down.

The agent passes the following arguments to the script:

- MultiNICB resource name

- The device whose status changed, for example:
  - qfe0
- The device's previous status (0 for down, 1 for up)
- The device's current status and monitor heartbeat

The agent also sends a notification (which may be received via SNMP or SMTP) to indicate that status of an interface changed. The notification is sent using "health of a cluster resource declined" and "health of a cluster resource improved" traps. These traps are mentioned in the *Cluster Server Administrator's Guide*. A sample mnicb_postchange trigger is provided with the agent. You can customize this sample script as needed or write one from scratch.

The sample script does the following:

- If interface changes status, it prints a message to the console, for example:

  ```
  MultiNICB: Interface qfe0 came up
  ```

- The script saves last IP address-to-interface name association. If any of the IP addresses have been moved, added, or removed, it prints out a message to the console, for example:

  ```
  MultiNICB: IP address 192.4.3.3 moved from interface

  qfe1:1 to interface qfe0:1
  ```

Sample configurations for MultiNICB agent

Interface configuration for MultiNICB agent on Solaris

Set the EPROM variable to assign unique MAC addresses to all ethernet interfaces on the host:

```
   # eeprom local-mac-address?=true
```

Reboot the system after setting the eprom variable to complete the address setup. The base IP addresses must be configured on the interfaces before the MultiNICB agent controls the interfaces. You can configure these addresses at system start up using /etc/hostname.*XXX* initialization files. Refer to the following examples for more information.

Setting up test IP addresses for Base Mode

These examples demonstrate setting up test IP addresses for your clustered systems. These IP addresses allow the agent determine if the NIC works. The agent determines that the NIC works if it receives responses for the ping packets that it sends to other nodes on the network. You do not need to perform the following steps for the floating IP addresses. The agent performs these steps.

In the file /etc/hostname.qfe0, add the following two lines:

```
north-qfe0 netmask + broadcast + deprecated -failover up \
        addif north netmask + broadcast + up
```

Where north-qfe0 is the test IP address that the agent uses to determine the state of the qfe0 network card.

In the file /etc/hostname.qfe4, add the following line:

```
north-qfe4 netmask + broadcast + deprecated -failover up
```

Where north-qfe4 is the test IP address that the agent uses to determine the state of the qfe4 network card.

In the example, north-qfe0 and north-qfe4 are the host names that correspond to test IP addresses. north is the host name that corresponds to the test IP address.

IPMultiNICB and MultiNICB configuration for MultiNICB agent

The following is an example IPMultiNICB and MultiNICB configuration:

```
cluster clus_north (
UserNames = { admin = "cDRpdxPmHpzS." }
Administrators = { admin }
CounterInterval = 5
)
system north (
)
system south (
)
group g11 (
SystemList = { north = 0, south = 1 }
AutoStartList = { north, south }
)
IPMultiNICB g11_i1 (
BaseResName = gnic_n
Address = "192.1.0.201"
NetMask = "255.255.0.0"
DeviceChoice = "1"
)
Proxy g11_p1 (
TargetResName = gnic_n
)
g11_i1 requires g11_p1
// A parallel group for the MultiNICB resource
group gnic (
SystemList = { north = 0, south = 1 }
AutoStartList = { north, south }
Parallel = 1
)
MultiNICB gnic_n (
Device @north = { qfe0 = 0, qfe4 = 1 }
Device @south = { qfe0 = 0, qfe4 = 1 }
NetworkHosts = { "192.1.0.1" }
)
Phantom gnic_p (
)
```

IPv6 configuration for MultiNICB agent

The following is a basic configuration for IPv6 with IPMultiNICB and MultiNICB resources for Oracle Solaris 11.

```
group mnicb_grp (
    SystemList = { sysA = 0, sysB = 1 }
    )

    IPMultiNICB ipmnicb (
        BaseResName = mnicb
        Address = "2300::18"
        PrefixLen = 64
               )

    MultiNICB mnicb (
        UseMpathd = 1
        ConfigCheck = 0
        Device = { net1 = 0, net2 = 1 }
        NetworkHosts = { "192.168.1.1", "2620:128:f0a2:9001::1"}
        IPMPDevice = ipmp0
        )
```

```
ipmnicb requires mnicb
```

Debug log levels for MultiNICB agent

The MultiNICB agent uses the following debug log levels:

DBG_1, DBG_2, DBG_3, DBG_4, DBG_5

DNS agent

The DNS agent updates and monitors the mapping for the following:

- The host name to IP address (A, AAAA, or PTR record)
- Alias to hostname or canonical name (CNAME)

The agent performs these tasks for a DNS zone when failing over nodes across subnets (a wide-area failover). Resource records (RR) can include different types: A, AAAA, CNAME, and PTR records.

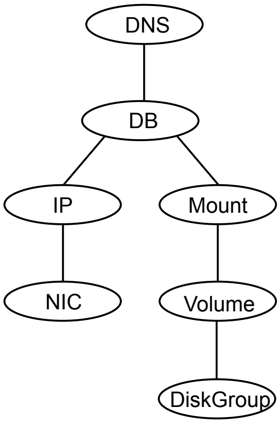
Use the DNS agent if the Resource Records need to be dynamically added and deleted from the DNS servers during failover. The agent updates the name server with the new resource record mappings while failing over and allows the clients to connect to the failed over instance of the application.

For important information about this agent, refer to Agent notes for DNS agent

Dependencies for DNS agent

No dependencies exist for the DNS resource.

Figure 3-5 Sample service group that includes a DNS resource



Agent functions for DNS agent

Online	Updates one or more name servers with the resource records. The agent updates the name servers defined in the StealthMasters attribute. If you have not configured this attribute then the agent obtains the name of the master server by sending an Start of Authority (SOA) query. This query retrieves the SOA record of the zone defined in the agent's Domain attribute. This SOA record contains the name of the master server. The agent creates PTR records for each RR of type A or AAAA if the value of the CreatePTR attribute is true. A prerequisite for this feature is that the same master or stealth server serves the forward (A or AAAA) and reverse zones. Finally the agent generates an Online lock file to indicate that the resource is online on the current system. **Note:** The DNS agent does not send any update for a resource record if it is already present on the name server.
Offline	Removes the Online lock file. If attribute OffDelRR is true, offline removes all records that the ResRecord keys define.

Monitor

Returns the ONLINE state if at least one name server reports all mappings that ResRecord defines. The name servers are the master or StealthMaster servers and all the servers for which an NS record for the zone exists.

The monitor entry point also sends periodic refresh requests to DNS server if the RefreshInterval attribute is set.

Clean

Removes the Online lock file, if it exists. If attribute OffDelRR is true, clean removes all records that the ResRecord keys define.

Open

Removes the Online lock file if the resource is reported online on another node inside the cluster to prevent concurrency violation. If the lock file exists, at least one name server has to report all the records that the ResRecord attribute defines. If all the name servers fail to report all the records, the agent function removes the Online lock file.

Action

Different action agent functions follow:

- keyfile.vfd
  This action entry point checks if the key file as specified in the TSIGKeyFile attribute exists either locally or on shared storage.
- dig.vfd
  This action entry point checks if dig and nsupdate binaries exist and are executable.
- master.vfd
  This action entry point checks if stealth masters are able to reply to SOA query for the configured domain.

State definitions for DNS agent

The state definitions for this agent follow:

ONLINE

Online lock file exists and at least one name server can return all configured resource records.

OFFLINE

At least one of the following is true:

- The online lock does not exist.
- None of the name servers can report all of the RRs' mappings.

UNKNOWN

Indicates that the DNS resource is not configured correctly. Can indicate that the resource record list contains an invalid value as a part of the record key or a record value of the ResRecord attribute.

Attributes for DNS agent

Table 3-11 Required attributes

Required attribute	Description
Domain	A string representing the DNS zone that the agent administers. The domain name can only contain alphanumeric symbols and the dash. Type and dimension: string-scalar Examples: ▪ Forward mapping: "demo.example.com" ▪ IPv4 reverse mapping: "2.168.192.in-addr.arpa"

Table 3-11 Required attributes *(continued)*

Required attribute	Description
ResRecord	

Table 3-11 Required attributes *(continued)*

Required attribute	Description
	ResRecord is an association of DNS resource record values. Each ResRecord attribute consists of two values: *DNS record key* = *DNS record data*. Note that the record key must be a unique value. If the resource record list contains any invalid value as a part of the record key or a record data of the ResRecord attribute, the resource reports an UNKNOWN state. Type and dimension: string-association Examples: ■ For forward mapping, where the zone is demo.example.com: - solX01 = "192.168.2.191" - ww2 = solX01 - solXip6 = "2007::1:2:3:abc" ■ For a multi-home DNS record, typically for one host with two network interfaces and different addresses, but the same DNS name. The A type ResRecord configuration should be as follows: solX02 = "192.168.2.102 10.87.13.22" A multi-home AAAA DNS record can be configured as follows: solX02 = "1234::5678 1234::AABB:CCDD" ■ For reverse IPv4 address mapping, where the zone is 2.168.192.in-addr.arpa: 191 = "solX01.demo.example.com" ■ For reverse IPv6 address mapping: ■ For 2620:128:f0a2:9001::/64 IPv6 Address prefix, the reverse lookupzone will be 1.0.0.9.2.a.0.f.8.2.1.0.0.2.6.2.ip6.arpa ■ If you want to create PTR record for 2620:128:f0a2:9001::102:108 address, you must mention 1020108 as key value in the ResRecord attribute. That is, ResRecord={1020108="solXip6.demo.example.com"} Use only partial host names. If you use a fully qualified domain name, append a period "." at the end of the name. For CNAME records, use: ■ ResRecord = { www = mydesktop } or ■ ResRecord = { www = "mydesktop.marketing.example.com."

Table 3-11 Required attributes *(continued)*

Required attribute	Description
	} Where the Domain attribute is "marketing.example.com"
ResRecord (Continued)	The agent uses case-insensitive pattern matching—and a combination of the Domain and ResRecord attribute values—to determine the resource record type. The RR types are as follows: ■ PTR: if the Domain attribute ends with .arpa ■ A: if the record data field is an IPv4 address (four sets of numbers, where a period separates each set. The following details the pattern it tries to match: [*1-223*].[*0-255*].[*0-255*].[*0-255*] Hexadecimal is not supported.) ■ AAAA: if the record data fields are in multiple sets of hexadecimal format, then this record is an IPv6 associated type AAAA record. ■ CNAME: for any other valid record data. **Note:** If a name in the ResRecord attribute does not comply with RFC 1035, then the agent logs a warning message to the engine log file. This ResRecord association is not used. As an exception to this, the DNS agent allows underscore character ("_") in hostnames. Make sure that the DNS server supports the underscore character before you configure any DNS resource records to have the underscore character in their hostnames.

Table 3-12 Optional attributes

Optional attribute	**Description**
TTL	This attribute (a non-zero integer) represents the Time To Live (TTL) value, in seconds, for the DNS entries in the zone that you want to update. A lower value means more hits on your DNS server, while a higher value means more time for your clients to learn about changes. The TTL may take the value 0, which indicates never caching the record, to a maximum of 2,147,483,647, which is over 68 years! The current best practice recommendation (RFC 1912) proposes a value greater than one day, and on RRs that do not change often, consider multi-week values. Type and dimension: integer-scalar Default: 86400 Example: 3600
StealthMasters	The list of primary master name servers in the domain. This attribute is optional since the first name server is retrieved from the zone's SOA (Start of Authority) record. If the primary master name server is a stealth server, define this attribute. A stealth server is a name server that is authoritative for a zone, but does not appear in that zone's SOA record. It is hidden to prevent direct attacks from the Internet. Type and dimension: string-vector Example: {"10.190.112.23", "2620:128:f0a2:9001::102:108"}
TSIGKeyFile	Required when you configure DNS for secure updates. Specifies the absolute path to the file containing the private TSIG (Transaction Signature) key. This attribute should be configured only when the DNS server configured is a Unix based DNS server. Type and dimension: string-scalar Example: /var/tsig/example.com.+157+00000.private

Table 3-12 Optional attributes *(continued)*

Optional attribute	Description
CreatePTR	Use the CreatePTR attribute to direct the online agent functions to create PTR records for each RR of type A or AAAA. You must set the value of this attribute to true (1) to create the records. Before you can use this attribute, make sure that the same master or stealth servers serve the forward (A or AAAA) and reverse zones. Type and dimension: boolean-scalar Default: 0 Example: 1
OffDelRR	Use the OffDelRR attribute to direct the offline and clean agent functions to remove all records that the ResRecord key defines. You must set the value of this attribute to 1 (true) to have the agent remove all the records. Type and dimension: boolean-scalar Default: 0 Example: 1
UseGSSAPI	Use the UseGSSAPI attribute if the DNS server that you have configured is a Windows DNS server and only if it accepts secure dynamic updates. **Note:** Do not set this attribute if the Windows DNS server accepts non-secure updates. If this attribute is set to 1, the agent uses the -g option with the nsupdate command. See "Agent notes for DNS agent" on page 144. for more information on requirements to use the DNS agent with the secure Windows DNS server. Type and dimension: boolean-scalar Default: 0 Example: 1

Table 3-12 Optional attributes *(continued)*

Optional attribute	Description
RefreshInterval	This attribute represents the time interval in seconds after which the DNS agent attempts to refresh the resource records (RRs) on the DNS servers. The default value of zero indicates that the DNS agent does not attempt to refresh the records on the DNS servers. The DNS agent writes the warning message to the logs if it is not able to refresh the DNS records. **Note:** The refresh request is sent in the next monitor cycle after the RefreshInterval period is reached. If the DNS agent is unable to refresh the DNS records, and the records are removed as a result of a scavenging operation or by the DNS administrator, the DNS resource will fault. Type and dimension: integer-scalar Default: 0 Example: 3600
CleanRRKeys	Use this attribute to direct the online agent function to clean up all the existing DNS records for the configured keys before adding new records. The default value (0) disables this behavior. **Note:** If multiple DNS resources are configured with the same key value in their ResRecord attribute, then do not set this attribute value to 1. Type and dimension: boolean-scalar Default: 0 Example: 1

Resource type definition for DNS agent

The resource definition for this agent on Solaris follows:

```
type DNS (
        static keylist SupportedActions = { "dig.vfd",
        "master.vfd", "keyfile.vfd" }
        static str ArgList[] = { Domain, TTL, TSIGKeyFile,
        StealthMasters, ResRecord, CreatePTR, OffDelRR,
        UseGSSAPI, RefreshInterval, CleanRRKeys }
        str Domain
        int TTL = 86400
        str TSIGKeyFile
        str StealthMasters[]
        str ResRecord{}
        boolean CreatePTR = 0
        boolean OffDelRR = 0
        boolean UseGSSAPI = 0
        int RefreshInterval = 0
        boolean CleanRRKeys = 0
)
```

Agent notes for DNS agent

The DNS agent has the following notes:

- About using the VCS DNS agent on UNIX with a secure Windows DNS server
- High availability fire drill for DNS agent
- Monitor scenarios for DNS agent
- Sample Web server configuration for DNS agent
- Secure DNS update for BIND 9 for DNS agent
- Setting up secure updates using TSIG keys for BIND 9 for DNS agent

About using the VCS DNS agent on UNIX with a secure Windows DNS server

This section describes the requirements for using the DNS agent with a secure Windows DNS server. Note that there are no special requirements for sending non-secure updates to a Windows DNS server.

Software requirement for DNS agent

For the secure updates on Windows DNS server to work, the VCS DNS agent on UNIX requires BIND version 9.7.2-P3 or later installed on all cluster nodes.

Configuration requirement for DNS agent

The VCS DNS agent on UNIX requires setting up Kerberos authentication with the Windows DNS server and configuring the domain and DNS server information in /etc/resolv.conf at the client node.

To set up the Kerberos authentication from the UNIX host to the Windows DNS server, configure the Kerberos configuration file (/etc/krb5.conf or /etc/ krb/krb5.conf) to use the Windows DNS server as Key Distribution Centre (KDC).

A sample Kerberos configuration file with domain privdns.sym and DNS server master.privdns.sym is as follows:

```
[libdefaults]
default_realm = PRIVDNS.SYM
dns_lookup_realm = true
dns_lookup_kdc = true
default_tkt_enctypes = des-cbc-md5
default_tgs_enctypes = des-cbc-md5
ticket_lifetime = 24h
renew_lifetime = 7d
forwardable = true
allow_weak_crypto = true
[realms]
PRIVDNS.SYM = {
kdc = master.privdns.sym:88
kpasswd_server = master.privdns.sym:464
admin_server = master.privdns.sym
}
[domain_realm]
.privdns.sym = PRIVDNS.SYM
privdns.sym = PRIVDNS.SYM
```

Note: The DNS agent does not support KDC and Domain Controller/DNS located on different servers.

Authenticate all the nodes on the cluster (on which the DNS agent is configured to run) with the Active directory. Use kinit on your user account and use klist to verify that you have a ticket to the configured realm principal. Refer to the man page of kinit for more information on obtaining Kerberos ticket granting tickets from KDC.

Note: The DNS agent requires a node to be authenticated with Kerberos all the time. Renew the obtained tickets periodically if your authentication method requires you to do so.

A sample run of kinit and klist for the above configuration with user vcsdns will look as follows:

```
# kinit vcsdns
Password for vcsdns@PRIVDNS.SYM:
# klist
Ticket cache: FILE:/tmp/krb5cc_0
```

```
Default principal: vcsdns@PRIVDNS.SYM
Valid starting Expires Service principal
12/14/09 16:17:37 12/15/09 02:19:09 krbtgt/PRIVDNS.SYM@PRIVDNS.SYM
renew until 12/21/09 16:17:37
```

If the environment variable KRB5CCNAME is set to some non-default location (default is `/tmp`), then VCS will not inherit it by default and will look for the Kerberos tickets in default location `/tmp`.

To resolve this issue, un-set the environment variable KRB5CCNAME and run the kinit command again. This will update the Kerberos tickets in default location (`/tmp`). Else, for a customized location (for example, `/cache/krb_ticket`) for Kerberos tickets, add an entry in `/opt/VRTSvcs/bin/vcsenv` file on each cluster node before VCS starts:

```
KRB5CCNAME="FILE:/cache/krb_ticket"
```

```
export KRB5CCNAME
```

Update `/etc/resolv.conf` on your client node to add information for the Windows DNS server and the configured domain.

High availability fire drill for DNS agent

The high availability fire drill detects discrepancies between the VCS configuration and the underlying infrastructure on a node; discrepancies that might prevent a service group from going online on a specific node.

For DNS resources, the high availability drill tests the following conditions:

- Checks if the key file as specified by the TSIGKeyFile attribute is available either locally or on shared storage.
- Checks if the dig and nsupdate binaries are available on the cluster node and are executable on that node.
- Checks if the stealth masters can respond to the SOA query made from the cluster node so as to ensure that there is no network issue that would prohibit the DNS update and query requests from reaching the stealth master server.

For more information about using the high availability fire drill see the *Cluster Server Administrator’s Guide*.

Monitor scenarios for DNS agent

Depending on the existence of the Online lock file and the defined Resource Records (RR), you get different status messages from the Monitor function.

Table 3-13 summarizes the monitor scenarios for the Online lock files.

Table 3-13 Monitor scenarios for the Online lock file

Online lock file exists	Expected RR mapping	Monitor returns
NO	N/A	OFFLINE
YES	NO	OFFLINE
YES	YES	ONLINE

Sample Web server configuration for DNS agent

Take the example of a Web server. A browser requests the URL http://www.example.com that maps to the canonical name server1.example.com. The browser retrieves the IP address for the web server by querying a domain name server. If the web server fails over from server one to server two (server2.example.com), the domain name servers need a new canonical name mapping for www.example.com. After the failover, the DNS resource updates this mapping of www.example.com to point to canonical name server2.example.com

Note: In this configuration, the Domain attribute should be configured with value "example.com"

Secure DNS update for BIND 9 for DNS agent

The DNS agent expects that the zone’s allow-update field contains the IP address for the hosts that can dynamically update the DNS records. This functionality is default for the DNS agent. Since a competent black hat can, however, spoof IP addresses, consider TSIG as an alternative.

TSIG (Transaction Signature) as specified in RFC 2845 is a shared key message authentication mechanism that is available in BIND DNS. A TSIG key provides the means to authenticate and verify the validity of exchanged DNS data. It uses a shared secret key between a resolver and either one or two servers to provide security.

Setting up secure updates using TSIG keys for BIND 9 for DNS agent

In the following example, the domain is example.com.

To use secure updates using TSIG keys, perform the following steps at the DNS server:

1 Run the `dnssec-keygen` command with the HMAC-MD5 option to generate a pair of files that contain the TSIG key:

```
# dnssec-keygen -a HMAC-MD5 -b 128 -n HOST example.com.
```

2 Open the example.com.+157+00000.key file. After you run the `cat` command, the contents of the file resembles:

```
# cat example.com.+157+00000.key
    example.com. IN KEY 512 3 157 +Cdjlkef9ZTSeixERZ433Q==
```

3 Copy the shared secret (the TSIG key), which looks like:

```
+Cdjlkef9ZTSeixERZ433Q==
```

4 Configure the DNS server to only allow TSIG updates using the generated key. Open the named.conf file and add these lines.

```
key example.com. {
    algorithm hmac-md5;
    secret "+Cdjlkef9ZTSeixERZ433Q==";
};
```

Where `+Cdjlkef9ZTSeixERZ433Q==` is the key.

5 In the named.conf file, edit the appropriate zone section and add the allow-updates sub-statement to reference the key:

```
allow-update { key example.com. ; } ;
```

6 Save and restart the named process.

7. Place the files containing the keys on each of the nodes that are listed in your group's SystemList. The DNS agent uses this key to update the name server.

   Copy both the private and public key files on to the node. A good location is in the /var/tsig/ directory.

8. Set the TSIGKeyFile attribute for the DNS resource to specify the file containing the private key.

   ```
   DNS www (
   Domain = "example.com"
   ResRecord = {www = north}
   TSIGKeyFile = "/var/tsig/example.com.+157+00000.private"
   )
   ```

Sample configurations for DNS agent

This section contains sample configurations for this agent.

Basic IPv6 configuration for DNS agent

This sample configuration provides basic configuration for IPv6 support. In the following configuration, *nic_value* represents the base NIC value for the platform

For example: e1000g0

```
group ipv6_group_dns (
    SystemList = { sysA = 0, sysB = 1 }
    )

    DNS ipv6group_dns_res (
        Critical = 0
        Domain = "example.com"
        TSIGKeyFile = "/var/tsig/Kipv6.vcscd.net.+157+18435.private"
        StealthMasters = { "2001:db8:c18:2:69c4:3251:bac1:6cbe" }
        ResRecord = {
            vcssysCv6 = "2001:db8:c18:2:214:4fff:fe96:8833",
            sysC = vcssysCv6 }
        )

    IP ipv6group_ip_res (
        Device @sysA = nic_value
        Device @sysB = nic_value
        Address = "2001:db8:c18:2:214:4fff:fe96:8833"
        PrefixLen = 64
```

```
        )

    NIC ipv6group_nic_res (
        Device @sysA = nic_value
        Device @sysB = nic_value
        NetworkHosts = { "2001:db8:c18:2:214:4fff:fea2:fd50" }


Protocol = IPv6


)

    ipv6group_dns_res requires ipv6group_ip_res
    ipv6group_ip_res requires ipv6group_nic_res
```

IPv6 CNAME sample configuration for DNS agent

The following sample configuration uses CNAME values.

```
group cname_group (
    SystemList = { sysA = 0, sysB = 1 }
    )

    DNS cname_group_dns_res (
        Domain = "example.com"
        StealthMasters = { "3ffe:556::1000:5761" }
        ResRecord @sysA = { www = server1 }
        ResRecord @sysB = { www = server2 }
        OffDelRR = 1
    )
```

IPv4 A sample configuration for DNS agent

The following sample configuration uses A values.

```
group forwardv4_group (
    SystemList = { sysA = 0, sysB = 1 }
    )

    DNS forward_group_v4_resource (
        Domain = "example.com"
        StealthMasters = { "3ffe:556::1000:5761" }
        ResRecord @sysA = { www = "10.200.56.240" }
        ResRecord @sysB = { www = "10.200.56.244" }
        OffDelRR = 1
        )
```

Debug log levels for DNS agent

The DNS agent uses the following debug log levels:

DBG_1, DBG_2, DBG_3, DBG_4, DBG_5

File share agents

This chapter includes the following topics:

About the file service agents

Use the file service agents to provide high availability for file share resources.

NFS agent

Starts and monitors the nfsd and mountd daemons required by all exported NFS file systems.

You should configure only a single NFS resource in a service group on a node. If you have more than one service group that uses the NFS resource, the other service groups must use a Proxy resource. The Proxy resource can point to the NFS resource in the first group. Duplicate NFS resources will cause a problem when the NFS resources are brought online concurrently—only the NFS resource started first will be successfully brought online, while the rest of the NFS resources may report online failure.

Note: VCS supports a maximum of one service group with NFS shares when the UseSMF attribute's value is 1.

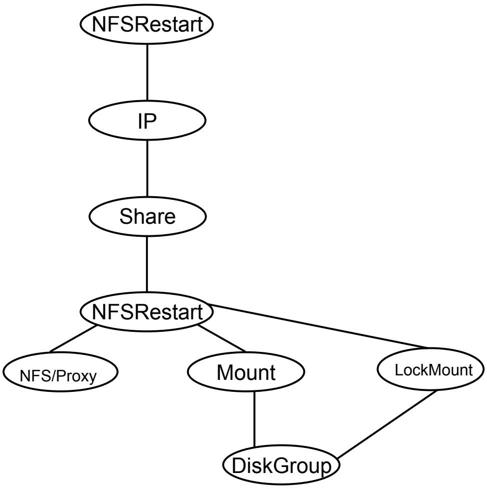
Note: VCS does not support the resource to use SMF if you set the UseSMF attribute to 0.

For important information about this agent,

Dependencies for NFS agent

For more information regarding NFS resource dependencies, refer to the *Cluster Server Administrator's Guide*.

Figure 4-1 Sample service group that includes an NFS resource



Agent functions for NFS agent

Online	Checks if nfsd and mountd daemons are running. If they are not running, the agent starts the daemons.

Monitor

Monitors versions 2, 3, and 4 of the nfsd daemons, and versions 1, 2, and 3 of the mountd daemons. Monitors TCP and UDP versions of the daemons by sending RPC (Remote Procedure Call) calls `clnt_create` and `clnt_call` to the RPC server. If the calls succeed, the resource is reported ONLINE.

Clean

Terminates and restarts the nfsd, mountd, and nfsmapid daemons.

Attr_changed

When the Protocol attribute is changed, this function dynamically restarts the NFS services if the services are not running as per the protocol specified in the Protocol attribute.

State definitions for NFS agent

ONLINE

Indicates that the NFS daemons are running in accordance with the supported protocols and versions.

OFFLINE

Indicates that the NFS daemons are not running in accordance with the supported protocols and versions.

FAULTED

Indicates that the NFS daemons are not running in accordance with the supported protocols and versions.

UNKNOWN

Unable to determine the status of the NFS daemons.

Attributes for NFS agent

Table 4-1 Optional attributes for Solaris

Optional attributes	Description
CleanRmtab	To clear the /etc/rmtab file before staring the mount daemon, set the value of CleanRmtab to 1. Type and dimension: boolean-scalar Default: 0

Table 4-1 Optional attributes for Solaris *(continued)*

Optional attributes	Description
LockFileTimeout	Specifies the time period in seconds after which the agent deletes the lock files. The agent maintains the files internally to synchronize the starting and stopping of NFS daemons between multiple service groups. Set this value to the total time needed for a service group to go offline or come online on a node. In situations where you have multiple service groups, set this value for the service group that takes the longest time. Type and dimension: integer-scalar Default: 180 Example: 240
Nservers	Specifies the number of concurrent NFS requests the server can handle. Type and dimension: integer-scalar Default: 16 Example: 24

Table 4-1 Optional attributes for Solaris *(continued)*

Optional attributes	Description

UseSMF	On Solaris 11, UseSMF must be set to 1.
	Note: On Solaris 11, when you use a SMF service to enable the NFS server, ensure that there is at least one shared directory present on each node to configure the agent. Use the following command to share a directory across reboots:
	`#share /`*`xyz`*
	On Solaris 11, to keep NFS daemons always running, do one of the following:
	■ On each node, set `application/auto_enable` to false for `nfs/status`, `nfs/server`, and `nfs/lockmgr` SMF service:
	`# svccfg -s nfs/status listprop -f application/auto_enable`
	`# svccfg -s nfs/server listprop -f application/auto_enable`
	`# svccfg -s nfs/nlockmgr listprop -f application/auto_enable`
	Or
	■ Ensure that there is at least one shared directory present on each node to configure the agent. Use the following command to share a directory across reboots:
	`# share /`*`xyz`*
	Type and dimension: boolean-scalar
	Default: 0

Table 4-1 Optional attributes for Solaris *(continued)*

Optional attributes	Description
Protocol	Specify the protocol to run the nfsd daemon. The following are the valid values: ▪ tcp ▪ udp ▪ all The agent uses this attribute to ensure that the NFS daemon is running using the specified protocol. **Note:** VCS overrides any change to the NFS protocol outside of VCS control, so you must not change the NFS protocol outside of VCS control. Type and dimension: string-scalar Default: all Example: tcp
MountdOptions	Options for the mountd daemon. For more information, see the *mountd* manual page . For example: `-v` In the above example, the agent executes the mountd daemon in verbose mode.

Resource type definition for NFS agent

```
type NFS (
    static int RestartLimit = 1
    static str ArgList[] = { UseSMF, Nservers, LockFileTimeout,
    CleanRmtab, MountdOptions, Protocol }
    static str Operations = OnOnly
    int Nservers = 16
    int LockFileTimeout = 180
    boolean UseSMF = 0
    boolean CleanRmtab = 0
    str MountdOptions
    str Protocol = all
)
```

Sample configurations for NFS agent

On each node in your cluster, you can find sample NFS, NFSRestart, and Share configurations in /etc/VRTSvcs/conf/sample_nfs/.

For more information regarding agent configuration, refer to the *Cluster Server Administrator's Guide*.

Debug log levels for NFS agent

The NFS agent uses the following debug log levels:

DBG_1, DBG_3, DBG_4, DBG_5

NFSRestart agent

The NFSRestart agent provides the following functionalities:

Manages essential NFS locking services, network status manager, and lock manager.

Manages NFS lock recovery service by recovering the NFS record locks after sudden server crash.

Prevents potential NFS ACK storms by terminating NFS server services before offline of NFS VIP to close all TCP connections with the NFS client.

If you have configured the NFSRestart agent for lock recovery, the NFSRestart agent starts the smsyncd daemon. The daemon copies the NFS locks from the local directory /var/statmon/sm to shared storage. The agent's online function copies the locks from shared storage to local directory /var/statmon/sm.

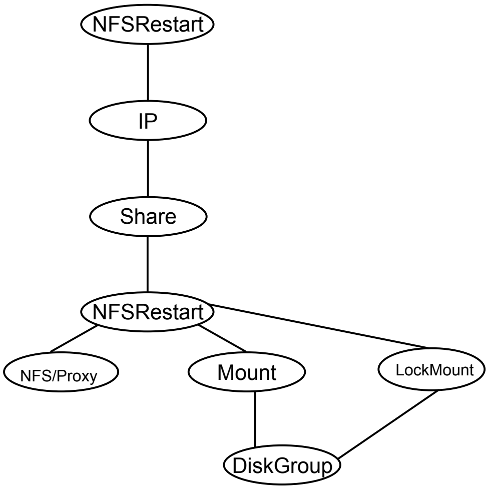
For important information about this agent, refer to Notes for NFSRestart agent

Dependencies for NFSRestart agent

For more information regarding NFSRestart resource dependencies, refer to the *Cluster Server Administrator's Guide*.

You must use two NFSRestart resources in a service group. Both the NFSRestart resources provide combined protection from potential corruption of NFS locks and potential NFS ACK storms. The lower NFSRestart resource must have its Lower attribute set to 1. The upper NFSRestart resource should be at the top of the resource dependency tree and the lower NFSRestart resource should be below the Share resource in the resource dependency tree. The NFSRestart resources and the Share resources must be inside the same service group.

Figure 4-2 Sample service group that includes an NFSRestart resource



Agent functions for NFSRestart agent

The agent functions for this agent follow:

Online

For the lower NFSRestart resource:

- If the value of the NFSLockFailover attribute is 1, the agent terminates statd and lockd.
- If the value of the NFSLockFailover attribute is 1 and if NFSv4 is configured, the agent copies the NFSv4 state data of clients from the shared storage to local path.

For the upper NFSRestart resource:

- If the value of the NFSLockFailover attribute is 1, the agent copies the NFS record locks from shared storage to the /var/statmon/sm directory.
- Starts the statd and lockd daemons.
- Starts the smsyncd daemon to copy the contents of the /var/statmon/sm directory to the shared storage (LocksPathName) at regular two-second intervals.
- Starts the smsyncd daemon to copy the contents of the /var/statmon/sm directory to the shared storage (LocksPathName) and NFSv4 state data from local path to shared storage at regular two-second intervals.

Monitor

For the lower NFSRestart resource:

- The monitor agent function does nothing.

For the upper NFSRestart resource:

- If the value of the NFSLockFailover attribute is 1, the agent monitors smsyncd daemon. It restarts the smsyncd daemon if it is not running.
- Monitors the statd and lockd daemons.

Offline

For the lower NFSRestart resource:

- Restarts all the NFS daemons that the upper NFSRestart resource stopped previously.

For the upper NFSRestart resource:

- Terminates the statd and lockd daemons to clear the lock state.
- Terminates the nfsd and mountd daemons to close the TCP/IP connections.
- Terminates the smsyncd daemon if the daemon is running.

Clean

For the lower NFSRestart resource:

- Restarts all the NFS daemons that the upper NFSRestart resource stopped previously.

For the upper NFSRestart resource:

- Terminates the statd and lockd daemons to clear the lock state.
- Terminates the nfsd and mountd daemons to close the TCP/IP connections.
- Terminates the smsyncd daemon if the daemon is running.

Action

- nfsconf.vfd
  Checks the runlevel information of the system service nfslock to confirm that the lock daemons do not come online automatically after reboot.
- lockdir.vfd
  Verifies that the NFS lock directory (which is specified by the LocksPathName attribute of NFSRestart) is on shared storage.

State definitions

ONLINE

Indicates that the daemons are running properly.

OFFLINE

Indicates that one or more daemons are not running.

UNKNOWN

Indicates the inability to determine the agent's status.

Attributes for NFSRestart agent

Table 4-2 Required attributes

Required attribute	Description
NFSRes	Name of the NFS resource. Do not set this to the name of the Proxy resource that points to the NFS resource. Type and dimension: string-scalar Example: "nfsres1"

Table 4-3 Optional attributes

Required attribute	Description
LocksPathName	The path name of the directory to store the NFS locks for all the shared file systems. You can use the pathname of one of the shared file systems for this value. Type and dimension: string-scalar Example: "/share1x"
NFSLockFailover	A flag that specifies whether the user wants NFS locks to be recovered after a failover Type and dimension: boolean-scalar Default: 0
LockServers	The maximum number of concurrent lockd threads for execution on a server. Type and dimension: integer-scalar Default: "20" Example: "25"
Lower	Defines the position of NFSRestart resource in the service group. The NFSRestart resource below the Share resource needs a value of 1. The NFSRestart resource on the top of the resource dependency tree has a Lower attribute value of 0. Type and dimension: integer-scalar Default: 0

Resource type definition for NFSRestart agent

```
type NFSRestart (
    static str ArgList[] = { LocksPathName, NFSLockFailover,
    LockServers, NFSRes, "NFSRes:Nservers",
    "NFSRes:LockFileTimeout","NFSRes:UseSMF", Lower, State,
    "NFSRes:MountdOptions", "NFSRes:Protocol" }
    static keylist SupportedActions = { "lockdir.vfd",
    "nfsconf.vfd" }
    str NFSRes
    str LocksPathName
    boolean NFSLockFailover = 0
    int LockServers = 20
    int Lower = 0
)
```

Notes for NFSRestart agent

The NFSRestart agent has the following notes:

- About high availability fire drill
- Mounting NFS export with the -vers options can cause lock failure
- Providing a fully qualified host name
- Service Management Facility

About high availability fire drill

The high availability fire drill detects discrepancies between the VCS configuration and the underlying infrastructure on a node; discrepancies that might prevent a service group from going online on a specific node.

For NFSRestart resources, the high availability drill performs the following, it:

- Checks the NFS configuration file to confirm that the NFS server does not come online automatically after reboot.
- Verifies that the NFS lock directory (which is specified by the LocksPathName attribute of NFSRestart) is on shared storage.

For more information about using the high availability fire drill see the *Cluster Server Administrator's Guide*.

Mounting NFS export with the -vers options can cause lock failure

For Solaris clients, the NFS lock recovery may fail to work if the NFS export is mounted with the "-vers=3" option.

Providing a fully qualified host name

You must provide a fully qualified host name, for example, nfsserver.example.edu, for the NFS server while mounting the file system on the NFS client. If you do not use a fully qualified host name, or if you use a virtual IP address (10.122.12.25) or partial host name (nfsserver), NFS lock recovery may fail.

If you want to use the virtual IP address or a partial host name, make the following changes to the service database (hosts) and the nsswitch.conf files:

```
/etc/hosts
```

To use the virtual IP address and partial host name for the NFS server, you need to add an entry to the /etc/hosts file. The virtual IP address and the partial host name should resolve to the fully qualified host name.

```
/etc/nsswitch.conf
```

You should also modify the hosts entry in this file so that upon resolving a name locally, the host does not first contact NIS/DNS, but instead immediately returns a successful status. Changing the nsswitch.conf file might affect other services running on the system.

For example:

```
hosts:  files [SUCCESS=return] dns nis
```

You have to make sure that the NFS client stores the same information for the NFS server as the client uses while mounting the file system. For example, if the NFS client mounts the file system using fully qualified domain names for the NFS server, then the /var/statmon/sm directory on the NFS client should also contain a fully qualified domain name of the NFS server after the acquisition of locks. Otherwise you need to stop and start the status daemon and lock daemon to clear the lock cache of the NFS client.

A time period exists where the virtual IP address is online but locking services are not registered on the server. Any NFS client trying to acquire a lock in this interval would fail and get ENOLCK error.

Every two seconds, the smsyncd daemon copies the list of clients that hold the locks on the shared filesystem in the service group. If the service group fails before smsyncd has a chance to copy the client list, the clients may not get a notification once the service group is brought up. This causes NFS lock recovery failure.

Service Management Facility

You need to enable the NFS attribute UseSMF to enable monitoring of NFS and Lock daemons through Service Management Facility (SMF) in Solaris.

SMF is the service framework for Solaris. SMF provides an infrastructure to automatically start and restart services. Previously, Unix start-up scripts and configuration files performed these functions.

SMF maintains the Service Configuration Repository, which stores persistent configuration information and runtime data for all the services. Thus, SMF now controls all NFS locking daemons (lockd, statd, etc.)

Sample configurations for NFSRestart agent

On each node in your cluster, you can find sample NFS, NFSRestart, and Share configurations in /etc/VRTSvcs/conf/sample_nfs/.

For more information regarding agent configuration, refer to the *Cluster Server Administrator's Guide*.

Basic agent configurations

For NFS lock recovery:

```
NFSRestart nfsrestart (
NFSRes = nfsres
LocksPathName="/shared_mnt/lockinfo"
NFSLockFailover = 1
Lower = 0
)
NFSRestart nfsrestart_L (
NFSRes = nfsres
LocksPathName="/shared_mnt/lockinfo"
NFSLockFailover = 1
Lower = 1
)
```

For no NFS lock recovery:

```
NFSRestart nfsrestart (
NFSRes = nfsres
)
```

```
NFSRestart nfsrestart_L (
NFSRes = nfsres
Lower = 1
)
```

Debug log levels for NFSRestart agent

The NFSRestart agent uses the following debug log levels:

DBG_1, DBG_3, DBG_4, DBG_5

Share agent

Shares, unshares, and monitors a single local resource for exporting an NFS file system to be mounted by remote systems.

Before you use this agent, verify that the files and directories to be exported are on shared disks.

For important information on this agent, refer to:

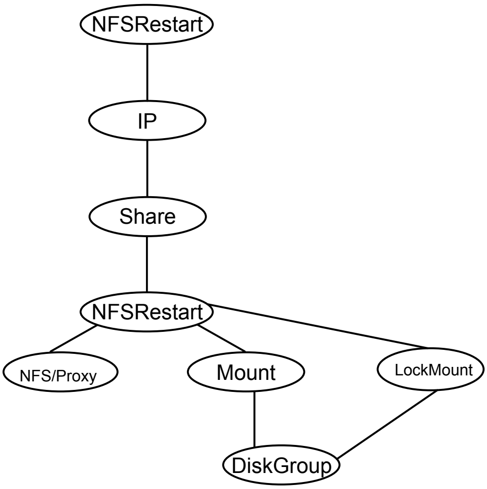
Notes for Share agent

Dependencies for Share agent

For more information regarding Share resource dependencies, refer to the *Cluster Server Administrator's Guide*.

Share resources depend on NFS. In an NFS service group, the IP family of resources depends on Share resources.

Figure 4-3 Sample service group that include a Share resource



Agent functions for Share agent

Online — Shares an NFS file system.

Offline — Unshares an NFS file system.

Monitor — Reads /etc/dfs/sharetab file and looks for an entry for the file system specified by PathName. If the entry exists, monitor returns ONLINE. If you specify the share options, the agent verifies whether the directory is shared with the specified options or not. If the directory is not shared with the specified options, a warning is displayed.

Clean — Terminates all ongoing resource actions and takes the resource offline, forcibly when necessary.

Action — direxists.vfd

Checks if the path specified by the PathName attribute exists on the cluster node. If the path name is not specified, it checks if a corresponding mount point is available to ensure that the path is on shared storage.

State definitions for Share agent

ONLINE — Indicates that specified directory is exported to the client.

OFFLINE — Indicates that the specified directory is not exported to the client.

UNKNOWN	Indicates that the agent could not determine the state of the resource or that the resource attributes are invalid.
FAULTED	Indicates that specified directory is unshared outside the control of VCS.

Attributes for Share agent

Table 4-4 Required attributes

Required attribute	Description
PathName	Pathname of the file system to be shared. Type and dimension: string-scalar Example: "/share1x"
NFSRes	This attribute has been deprecated.

Table 4-5 Optional attributes

Optional attribute	Description
Options	Options for the `share` command. Type and dimension: string-scalar Example: "-o rw"

Resource type definition for Share agent

```
type Share (
static keylist SupportedActions = { "direxists.vfd" }
static str ArgList[] = { PathName, Options, "NFSRes:State" }
str PathName
str Options
str NFSRes
)
```

Notes for Share agent

The following section contains notes on the Share agent.

- High availability fire drill

High availability fire drill

The high availability fire drill detects discrepancies between the VCS configuration and the underlying infrastructure on a node. These discrepancies might prevent a service group from going online on a specific node. For Share resources, the high availability fire drill checks if the path exists.

For more information about using the high availability fire drill see the *Cluster Server Administrator's Guide*.

On Solaris 11, the share options in the /etc/sharetab file may not match with the options provided in the share command, so a warning message is displayed in the agent log. To avoid this, it is recommended to use the share options in the /etc/sharetab file with Options attribute for the resource.

Sample configurations for Share agent

On each node in your cluster, you can find sample NFS, NFSRestart, and Share configurations in /etc/VRTSvcs/conf/sample_nfs/.

For more information regarding agent configuration, refer to the *Cluster Server Administrator's Guide*.

Debug log levels for Share agent

The Share agent uses the following debug log levels:

DBG_1, DBG_3, DBG_5

About the Samba agents

Samba is a suite of programs that allows a system running a UNIX or UNIX-like operating system to provide services using the Microsoft network protocol. Samba supports the following services:

- Filespace
- Printer
- WINS
- Domain Master

Configure these services in the Samba configuration file (smb.conf). Samba uses two processes: smbd and nmbd to provide these services.

VCS provides Samba failover using three agents: SambaServer, NetBios, and SambaShare.

The Samba agents

- The NetBios agent
- The SambaServer agent
- The SambaShare agent

Before using the Samba agents

- Verify that smbd and nmbd always run as daemons. Verify that they cannot be started using the meta-daemon inetd.
- Verify that Samba is configured properly and that the Samba configuration file is identical on all cluster systems. The user can replicate the file or store it on a shared disk accessible from all cluster systems.
- If configuring Samba as a WINS server or Domain Master, verify that the Samba lock directory is on the shared disk. This ensures that the WINS server database and Domain Master are created on the shared disk.

Supported versions for Samba agents

VCS Samba suite of agents support Samba version 3.0 and above. Please check your samba version using the following command:

```
# smbd -V
```

Notes for configuring the Samba agents

The following notes describe configuration considerations for the Samba agents.

Configuring multiple SambaServer resources

For configuring multiple SambaServer resources, configure the SocketAddress attribute with the unique value of the address where the respective samba daemon listens for connections. Configure the SambaServer resource as a parent resource of the IP resource. Configure this IP resource with the SocketAddress attribute value.

Configuring Samba for non-standard configuration files or non-standard lock directories

Configure the PidFile attribute if you use a non-standard configuration file for Samba or if the lock directory (the directory where Samba pid file resides) for Samba is different than the default location. Use the following command to check the standard locations for the Samba configuration file and the lock directory:

To check for the default value of the Samba configuration file

- Enter the following command:

```
# smbd -b | grep CONFIGFILE
```

To check for the default location of the Samba pidfile

- Enter the following command:

```
# smbd -b | grep PIDDIR
```

SambaServer agent

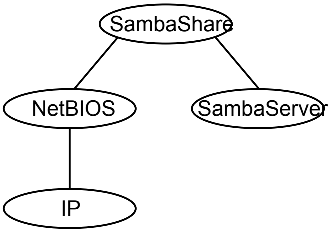
The SambaServer agent starts, stops, and monitors the smbd process as a daemon. Only one resource of this type is permitted. You can use the agent to make a smbd daemon highly available.

The smbd daemon provides Samba share services. The agent verifies that Samba is running by reading the pid of smbd daemon. The agent can perform in-depth monitoring by establishing a socket connection to Samba at ports where the daemon is listening and sending it a NetBIOS session request.

Dependencies for SambaServer agent

No dependencies exist for the SambaServer resource.

Figure 4-4 Sample service group that includes a SambaServer resource



Agent functions for SambaServer agent

Online	Starts the smbd daemon at specified or default ports.
Offline	Stops the smbd daemon.

Monitor	Verifies that the smbd daemon is running by reading its pid file. Does indepth monitoring periodically, if configured, by establishing a socket connection to Samba and sending it a NetBIOS session request.
Clean	Stops the smbd daemon forcefully if required.

State definitions for SambaServer agent

ONLINE	Indicates that the smbd daemon is running. If in-depth monitoring is configured, it indicates that a positive session response packet was received through a socket connection to the Samba server.
OFFLINE	Indicates that smbd is not running. If in-depth monitoring is enabled, it indicates that the agent could not establish a socket connection with the server, or that it received an incorrect response packet header, or the session response packet connection timed out.
UNKNOWN	Indicates that the agent could not determine the state of the resource.
FAULTED	Indicates that the smbd daemon has stopped unexpectedly or is not responding (if in-depth monitoring is enabled) outside of VCS control.

Attributes for SambaServer agent

Table 4-6 Required attributes

Required attribute	Description
ConfFile	Complete path of the configuration file that Samba uses. Type and dimension: string-scalar Example: "/etc/sfw/smb.conf"
LockDir	Lock directory of Samba. Samba stores the files smbd.pid, nmbd.pid, wins.dat (WINS database), and browse.dat (master browser database) in this directory. Type and dimension: string-scalar Example:"/var/samba/locks"

Table 4-6 Required attributes *(continued)*

Required attribute	Description

SambaTopDir	Parent path of Samba daemon and binaries.
	SambaServer agent uses SambaTopDir attribute value in an open entry point to determine the complete path of samba executables. If this attribute is configured after the resource is enabled, please disable and enable the resource again to bring this into effect as follows:
	`# hares -modify <res> Enabled 0`
	`# hares -modify <res> Enabled 1`
	Example:"/usr/sfw"

Table 4-7 Optional attributes

Optional attribute	Description
IndepthMonitorCyclePeriod	Number of monitor cycles after which the in-depth monitoring is performed. For example, the value 5 indicates that the agent monitors the resource in-depth every five monitor cycles. The value 0 indicates that the agent will not perform in-depth monitoring for the resource. Type and dimension: integer-scalar Default: 5
Ports	Ports where Samba accepts connections. To run Samba over NBT (NetBios over TCP/IP), set this attribute to 139. To run Samba directly over TCP/IP, set this attribute to 445. Type and dimension: integer-vector Default: 139, 445
ResponseTimeout	Number of seconds the agent waits to receive the session response packet after sending the session request packet. For example, the value 5 indicates that the agent waits for five seconds before receiving the session response packet. Configure this attribute if in-depth monitoring is enabled. Type and dimension: integer-scalar Default: 10

Table 4-7 Optional attributes *(continued)*

Optional attribute	Description
PidFile	The absolute path to the Samba daemon pid file. This file contains the process ID of the monitored smbd process. Configure this attribute if you are using a non-standard configuration file name or path. If this attribute is not configured for non-standard configuration file names, the agent checks the smbd-*ConfFile*.pid file for monitoring the resource. Type and dimension: string-scalar Example: "/var/samba/locks/smbd.pid"
SocketAddress	The IP address where the Samba daemon (smbd) listens for connections. Configure the SocketAddress attribute if you are configuring multiple SambaServer resources on a node. Type and Dimension: string-scalar Example: “10.128.10.14”, "2001::10"
Interfaces	List of network interfaces on which Samba handles browsing. Type and dimension: string-vector This attribute accepts a list of all strings supported by Samba, which can be in any of the following formats: ■ a network interface name This may include shell-like wildcards so eth* will match any interface starting with the substring "eth" ■ an IP address. In this case the netmask is determined from the list of interfaces obtained from the kernel ■ an IP/mask pair ■ a broadcast/mask pair **Note:** If you have configured Interfaces in a Samba Server Agent you cannot configure NetBIOS, because NetBIOS does not support IPv6 name resolution. For example: Interfaces = { "10.209.74.98", “eth*”, “eth0”, “10.209.74.98/16” }

Table 4-7 Optional attributes *(continued)*

Optional attribute	Description
BindInterfaceOnly	This parameter allows the Samba admin to limit what interfaces on a machine will serve SMB requests. Samba listens on the interfaces specified in the Interfaces attribute only if the BindInterfaceOnly attribute is set to 1. Type and dimension: boolean-scalar Default: 0 For example: BindInterfaceOnly = 1

Resource type definitions for SambaServer agent

```
type SambaServer (
static str ArgList[] = { ConfFile, SambaTopDir, LockDir, Ports,
IndepthMonitorCyclePeriod, ResponseTimeout, PidFile,
SocketAddress, Interfaces, BindInterfaceOnly }
str ConfFile
str LockDir
int Ports[] = { 139, 445 }
int IndepthMonitorCyclePeriod = 5
int ResponseTimeout = 10
str SambaTopDir
str PidFile
str SocketAddress
str Interfaces[]
boolean BindInterfaceOnly = 0
)
```

Sample configurations for SambaServer agent

The sample configurations for this agent follow:

```
SambaServer samba_server (
ConfFile = "/etc/sfw/smb.conf"
LockDir = "/var/samba/locks"
SambaTopDir = "/usr/sfw"
IndepthMonitorCyclePeriod = 3
ResponseTimeout = 15
Interfaces = { "2620:128:f0a2:9006:214:4fff:fefa:4367/64",
                "10.209.61.138","10.209.61.101" }
BindInterfaceOnly = 1
)
```

Debug log levels for SambaServer agent

The SambaServer agent uses the following debug log levels:

DBG_1, DBG_2, DBG_3, DBG_4, DBG_5

SambaShare agent

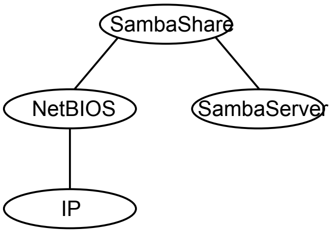
The SambaShare agent adds, removes, and monitors a share by modifying the specified Samba configuration file. You can use the agent to make a Samba Share highly available.

Each filespace service provided by Samba is a shared resource and is defined as a section in the Samba configuration file. The section name is the name of the shared resource and the section parameters define the share attributes.

Dependencies for SambaShare agent

SambaShare resources depend on the SambaServer, NetBios and Mount resources.

Figure 4-5 Sample service group for a SambaShare resource



Agent functions for SambaShare agent

Online	Edits the samba configuration file and adds the shares.
Offline	Removes the shares from the configuration file.
Monitor	Issues the command `smbclient` to check if the specified shares exist.

Clean — Terminates all ongoing connections with the particular samba share, removes its entry from the samba configuration file and reloads the configuration.

State definitions for SambaShare agent

ONLINE — Indicates that the share is available.

OFFLINE — Indicates that the share is not available.

FAULTED — Indicates that the share has become unavailable outside of VCS control.

UNKNOWN — Indicates that the agent could not determine the state of the resource.

Attributes for SambaShare agent

Table 4-8 Required attributes

Required attribute	Description
SambaServerRes	Name of the SambaServer resource. Type and dimension: string-scalar Example: "smb_res1"
ShareName	Name of the share resource as exported by samba. **Note:** This name can be different from the SambaShare resource name. Type and dimension: string-scalar Example: "share1"
ShareOptions	List of parameters for the share attributes. These parameters are specified as name=value pairs, with each pair separated by a semicolon (;). Type and dimension: string-scalar Example: "path=/shared; public=yes; writable=yes"

Table 4-9 Optional attributes

Optional attribute	Description
IPAddress	This attribute is reserved for internal use only.

Resource type definition for SambaShare agent

```
type SambaShare (
static str ArgList[] = { "SambaServerRes:ConfFile",
"SambaServerRes:SambaTopDir", "SambaServerRes:LockDir",
ShareName, ShareOptions, "SambaServerRes:Ports",
SambaServerRes, "SambaServerRes:PidFile",
"SambaServerRes:SocketAddress", "SambaServerRes:Interfaces",
"SambaServerRes:BindInterfaceOnly"  }
str SambaServerRes
str ShareName
str ShareOptions
)
```

Sample configuration for SambaShare agent

```
SambaShare Samba_SambaShare3 (
SambaServerRes = Samba_SambaServer
ShareName = smbshare3
ShareOptions = "path=/smbshare3; public=yes; writable=yes"
)
```

Debug log levels for SambaShare agent

The SambaShare agent uses the following debug log levels:

DBG_1, DBG_3, DBG_5

NetBios agent

The NetBios agent starts, stops, and monitors the nmbd daemon. Only one resource of this type is permitted. You can use the agent to make the nmbd daemon highly available.

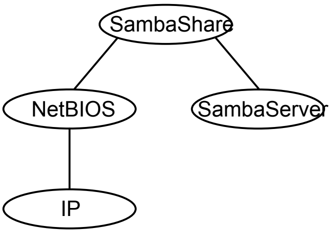
The agent sets, monitors, and resets the names and network interfaces by which the Samba server is known. The agent also sets, monitors and resets Samba to act as a WINS server or domain master or both.

Note: The nmbd broadcasts the NetBIOS name, or the name by which the Samba server is known in the network.

Dependencies for NetBios agent

The NetBios resource depends on the IP or IPMultiNICB resource if the virtual IP address configured in the IP or IPMultiNICB resource is being used in the Interfaces attribute of the NetBios resource.

Figure 4-6 Sample service group that includes a NetBIOS resource



Agent functions for NetBios agent

Online

Updates the Samba configuration with the NetBIOS name, all NetBIOS aliases and network interfaces, WINS support, and domain master options specified in the NetBIOS resource. Starts the nmbd daemon.

Offline

Removes the NetBIOS name, all NetBIOS aliases and network interfaces, WINS support, and domain master options specified in the NetBIOS resource from the Samba configuration file. Stops the nmbd daemon.

Monitor

Verifies that the Samba configuration contains the NetBIOS name, all NetBIOS aliases and network interfaces, WINS support, and domain master options specified in the NetBIOS resource. Also verifies that the nmbd daemon is running by reading its pid file.

Clean

Removes the NetBIOS name, all NetBIOS aliases and network interfaces, WINS support, and domain master options specified in the NetBIOS resource from the Samba configuration file. Stops the nmbd daemon, forcibly when necessary.

State definitions for NetBios agent

ONLINE

Indicates that the specified NetBIOS name and aliases are advertised and that Samba is handling requests for all specified network interfaces. Indicates that WINS and Domain support services are running, if configured.

OFFLINE

Indicates one or more of the following:

- NetBIOS name is not advertised.
- A NetBIOS alias is not advertised.
- Samba is not handling requests on any of the specified interfaces.
- If WINS support is configured, Samba is not providing WINS service.
- If domain support is set, Samba is not providing Domain Master service.

UNKNOWN

Indicates that the agent could not determine the state of the resource.

FAULTED

Indicates that the resource has become offline unexpectedly outside of VCS control.

Attributes for NetBios agent

Table 4-10 Required attributes

Required attribute	Description
NetBiosName	Name by which the Samba server is known in the network. Type and dimension: string-scalar Example: "samba_demon" **Note:** Samba has a limitation of 15 characters for NetBios names and aliases.
SambaServerRes	Name of the SambaServer resource. Type and dimension: string-scalar Example: "smb_res1"

Table 4-11 Optional attributes

Optional attribute	**Description**
Interfaces	List of network interfaces on which Samba handles browsing. Type and dimension: string-vector Example: "172.29.9.24/16" **Note:** If you have configured the SocketAddress attribute value for the corresponding SambaServer resource, then you must also configure the same value paired with the appropriate netmask in the list of interfaces.
NetBiosAliases	List of additional names by which the Samba server is known in the network. Type and dimension: string-vector Example: { host1_samba, myname } **Note:** Samba has a limitation of 15 characters for NetBios names and aliases.
WinsSupport	If set to 1, this flag causes the agent to configure Samba as a WINS server. Type and dimension: integer-scalar Default: 0
DomainMaster	If set to 1, the agent sets Samba as Domain Master. Note that there can be only one domain master in a domain. Type and dimension: integer-scalar Default: 0
PidFile	The absolute path to the NetBIOS daemon pid file. This file contains the process ID of the monitored nmbd process. Configure this attribute if you are using a nonstandard configuration file name or path. If this attribute is not configured for non-standard configuration file names, the agent checks for the nmbd-*ConfFile*.pid file for resource monitoring. Type and dimension: string-scalar Example: "/var/samba/locks/nmbd.pid"

Resource type definition for NetBios agent

```
type NetBios (
static str ArgList[] = { "SambaServerRes:ConfFile",
"SambaServerRes:SambaTopDir", "SambaServerRes:LockDir",
NetBiosName, NetBiosAliases, Interfaces, WinsSupport,
DomainMaster, "SambaServerRes:PidFile", SambaServerRes,
PidFile }
str SambaServerRes
str NetBiosName
str NetBiosAliases[]
str Interfaces[]
int WinsSupport
int DomainMaster
str PidFile
)
```

Sample configuration for NetBios agent

```
NetBios Samba_NetBios (
SambaServerRes = Samba_SambaServer
NetBiosName = samba_demon
NetBiosAliases = { asamba_demon, samba127 }
WinsSupport = 1
DomainMaster = 1
)
```

Debug log levels for NetBios agent

The NetBios agent uses the following debug log levels:

DBG_1, DBG_5

Chapter 5

Service and application agents

This chapter includes the following topics:

About the services and applications agents

Use service and application agents to provide high availability for application and process-related resources.

Apache HTTP server agent

The Apache HTTP server agent brings an Apache Server online, takes it offline, and monitors its processes. The Apache HTTP server agent consists of resource type declarations and agent scripts. You use the Apache HTTP server agent, in conjunction with other agents, to make an Apache HTTP server highly available.

This agent supports Apache HTTP server 2.0, 2.2, and 2.4. It also supports IBM HTTP Server 1.3, 2.0, and 7.x.

You can view the latest support information for Apache HTTP server at:
https://sort.veritas.com/agents

This agent can detect when an Apache HTTP server is brought down gracefully by an administrator. When Apache is brought down gracefully, the agent does not trigger a resource fault even though Apache is down.

This agent is IMF-aware and uses the AMF kernel driver for IMF notification.

IMF support is enabled by default. In VCS 6.1 and later, only PRON IMF monitoring is supported and the IMF Mode attribute value is set to 2.

For more information about IMF and intelligent resource monitoring, refer to the *Cluster Server Administrator’s Guide*.

Note: The Apache agent requires an IP resource for operation.

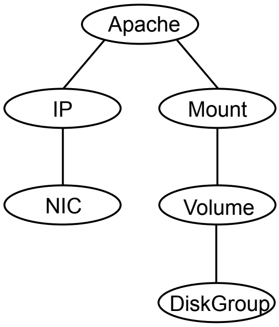
The agent performs detailed monitoring on the Apache resource. You can tune the frequency of detailed monitoring with the LevelTwoMonitorFreq attribute. By default, the agent monitors the Apache resource after every fifth monitor cycle to confirm the health of the resource.

See “Apache HTTP server notes” on page 190. for more information regarding this agent.

Dependencies

This type of resource depends on IP and Mount resources.

Figure 5-1 Sample service group for the Apache HTTP server agent



Agent functions

Online

To start the Apache HTTP server, the agent:

- Executes the httpdDir/httpd program with the appropriate arguments if the httpdDir program specifies the full path of the directory in which the httpd binary file is located.
- Alternatively, if the httpdDir attribute specifies the full path of the Apache HTTP server binary file, the binary file is executed with appropriate arguments.

When you specify a file with the EnvFile attribute, the file is sourced before the agent executes the Apache HTTP server commands.

Offline

To stop the Apache HTTP server, the agent:

- Executes the httpdDir/httpd program with the appropriate arguments, if httpdDir specifies the full path of the directory in which the httpd binary file is located.
- Alternatively, if the httpdDir attribute is used to specify the full path of the Apache HTTP server binary, the binary file is executed with appropriate arguments.
- Sends a TERM signal to the HTTP Server parent process (Apache).

When you specify a file with the EnvFile attribute, the file is sourced before the agent executes the Apache HTTP server commands.

Monitor

Monitors the state of the Apache server. First it checks for the processes, next it can perform an optional state check.

Clean

Removes the Apache HTTP server system resources that might remain after a server fault or after an unsuccessful attempt to online or offline. These resources include the parent httpd daemon and its child daemons.

Action

checkconffile.vfd

Checks for the existence of the Apache configuration file and the existence of the directory that contains the httpd binary that is used during start up.

For a local installation, if the config file or HttpdDir is not found, make sure that it exists on the failover node.

imf_init

Initializes the agent to interface with the AMF kernel driver. This function runs when the agent starts.

imf_getnotification

Gets notification about resource state changes during the online operation. This function runs after the agent initializes with the AMF kernel driver. The agent continuously waits for notification and takes action on the resource upon notification.

imf_register

Registers the resource entities for online monitoring with the AMF kernel driver. The Apache agent reports the resource as online when the parent Apache HTTP server process and at least one child HTTP server process is running. The Process ID of the parent Apache HTTP server process and one child process found on the system is registered with AMF.

For example, the function registers the PID of the process that requires online monitoring. This function runs for each resource after the resource goes into steady online state.

State definitions

ONLINE

Indicates that the Apache server is running.

OFFLINE

Indicates that the Apache server is not running.

Can also indicate that the administrator has stopped the HTTP server gracefully. Note that the agent uses the PidFile attribute for intentional offline detection.

UNKNOWN

Indicates that a problem exists with the configuration.

FAULTED

Indicates that the Apache server has stopped unexpectedly or is not responding (if in-depth monitoring is enabled) outside of VCS control.

Attributes

Table 5-1 Required attributes

Required attribute	Description
ConfigFile	Full path and file name of the main configuration file for the Apache server. Type and dimension: string-scalar Example: "/etc/httpd/conf/httpd.conf"

Table 5-1 Required attributes *(continued)*

Required attribute	Description
httpdDir	Full path of the Apache HTTP server binary file or full path of the directory in which the httpd binary file is located. Type and dimension: string-scalar Example: "/usr/apache2/bin"
PidFile	This attribute is required when you want to enable the detection of a graceful shutdown outside of VCS control. See Table 5-2 on page 186.
EnvFile	This attribute may be required when you use IBM HTTP Server. See Table 5-2 on page 186.

Table 5-2 Optional attributes

Optional attribute	Description
DirectiveAfter	A list of directives that httpd processes after reading the configuration file. Type and dimension: string-association Example: DirectiveAfter{} = { KeepAlive=On }
DirectiveBefore	A list of directives that httpd processes before it reads the configuration file. Type and dimension: string-association Example: DirectiveBefore{} = { User=nobody, Group=nobody }
User	Account name the agent uses to execute the httpd program. If you do not specify this value, the agent executes httpd as the root user. Type and dimension: string-scalar Example: "apache1"

Table 5-2 Optional attributes *(continued)*

Optional attribute	Description
EnableSSL	If this attribute is set to 1 (true) the online agent function will add support for SSL, by including the option `-DSSL` in the start command. For example: `/usr/sbin/httpd -f` *`path_to_httpd.conf`* `-k start -DSSL` Where path_to_httpd.conf file is the path to the httpd.conf file. If this attribute is set to 0 (false) the agent excludes the SSL support. Type and dimension: boolean-scalar Default: 0 Example: "1"
HostName	The virtual host name that is assigned to the Apache server instance. The host name is used in second-level monitoring for benchmarking the Apache HTTP server. You can use IPv4 or IPv6 addresses for the HostName attribute. **Note:** The HostName attribute is required only if you enable in-depth monitoring by setting the LevelTwoMonitorFreq attribute. Type and dimension: string-scalar Example: "web1.example.com"
Port	Port number where the Apache HTTP server instance listens. The port number is used in second-level monitoring for benchmarking the Apache HTTP server. Specify this attribute only if you have enabled in-depth monitoring by setting the LevelTwoMonitorFreq attribute. Type and dimension: integer-scalar Default: 80 Example: "80"

Table 5-2 Optional attributes *(continued)*

Optional attribute	Description
EnvFile	Full path and file name of the file that is sourced before executing Apache HTTP server commands. Specifying this attribute is optional. If EnvFile is specified, the shell for the user must be Bourne, Korn, or C shell. This attribute may be required when you use the IBM HTTP Server if the online action fails. For example: set the EnvFile to /usr/IBM/HTTPServer/bin/envvars. Type and dimension: string-scalar Example: "/apache/server1/bin/envvars"
PidFile	The PidFile attribute sets the file to which the server records the process ID of the daemon. The value of PidFile attribute must be the absolute path where the Apache instance records the PID. This attribute is required when you want the agent to detect the graceful shutdown of the Apache HTTP server. For the agent to detect the graceful shutdown of the Apache HTTP server, the value of the IntentionalOffline resource type attribute must be 1 (true). Type and dimension: string-scalar Example: /var/run/httpd.pid
SharedObjDir	Full path of the directory in which the Apache HTTP shared object files are located. Specifying this attribute is optional. It is used when the HTTP Server is compiled using the SHARED_CORE rule. If you specify this attribute, the directory is passed to the `-R` option when executing the httpd program. Refer to the httpd man pages for more information about the `-R` option. Type and dimension: boolean-scalar Example: "/apache/server1/libexec"

Table 5-2 Optional attributes *(continued)*

Optional attribute	Description
ResLogLevel	This attribute has been deprecated. Use the resource type attribute LogDbg to enable debug logs. Set LogDbg attribute to DBG_5 to enable debug logs for the Apache HTTP server agent. By default, setting the LogDbg attribute to DBG_5 enables debug logs for all Apache resources in the cluster. If debug logs must be enabled for a specific Apache resource, override the LogDbg attribute. For information on how to use the LogDbg attribute, refer to the *Cluster Server Administrator's Guide*.
LevelTwoMonitorFreq	Specifies the frequency at which the agent must perform second-level or detailed monitoring. You can also override the value of this attribute at the resource level. The value indicates the number of monitor cycles after which the agent will monitor Apache in detail. For example, the value 5 indicates that the agent will monitor Apache in detail after every five online monitor intervals. Type and dimension: integer-scalar Default: 0

Table 5-3 Resource type attribute

Optional attribute	Description
IntentionalOffline	For information on how to use the IntentionalOffline resource type attribute, refer to the *Cluster Server Administrator's Guide*.

Resource type definition

```
type Apache (
    static keylist SupportedActions = { "checkconffile.vfd" }
    static str ArgList[] = { ResLogLevel, State, IState, httpdDir,
    SharedObjDir, EnvFile, PidFile, HostName, Port, User,
    ConfigFile, EnableSSL, DirectiveAfter, DirectiveBefore }
    str ResLogLevel = INFO
    str httpdDir
    str SharedObjDir
    str EnvFile
    str PidFile
    str HostName
    int Port = 80
    str User
    str ConfigFile
    str DirectiveAfter{}
    str DirectiveBefore{}
    boolean EnableSSL
    static int IMF{} = { Mode = 2, MonitorFreq = 5, RegisterRetryLimit = 3 }
    static str IMFRegList[] = { ConfigFile, httpdDir }

    static int ContainerOpts{} = { RunInContainer=1, PassCInfo=1 }

    static boolean IntentionalOffline = 0
)
```

Apache HTTP server notes

The Apache HTTP server has the following notes:

- Tasks to perform before you use the Apache HTTP server agent
- About detecting application failure
- About bringing an Apache HTTP server online outside of VCS control
- About high Availability fire drill
- Using Apache agent with IMF
- Troubleshooting Apache service group issue

Tasks to perform before you use the Apache HTTP server agent

Before you use this agent, perform the following tasks:

- Install the Apache server on shared or local disks.
- Ensure that you are able to start the Apache HTTP server outside of VCS control, with the specified parameters in the Apache configuration file (for example: /etc/apache/httpd.conf). For more information on how to start the server:
  See "About bringing an Apache HTTP server online outside of VCS control" on page 191.
- Specify the location of the error log file in the Apache configuration file for your convenience (for example: ErrorLog /var/apache/logs/error_log).
- Verify that the floating IP has the same subnet as the cluster systems.

- If you use a port other than the default 80, assign an exclusive port for the Apache server.
- Verify that the Apache server configuration files are identical on all cluster systems.
- Verify that the Apache server does not autostart on system startup.
- Verify that `inetd` does not invoke the Apache server.
- The service group has disk and network resources to support the Apache server resource.
- Assign a virtual host name and port to the Apache server.
- Verify that you are able to start the Apache HTTP server outside of VCS control in non-interactive manner. For example, the `startup` command should not prompt for any password or any other interactive questions.
- Ensure that the directory to write Apache PID files is present persistently on the node, zone, or LDom for the Apache agent to detect online. The default directory (/var/run/apache2) gets created only when the service is enabled at least once using the `svcadm enable apache2` command and it is deleted during reboot.

About detecting application failure

The agent provides two methods to evaluate the state of an Apache HTTP server instance. The first state check is mandatory and the second is optional.

The first check determines the state of the Apache HTTP server. The check determines the state by searching for the existence of the parent httpd daemon. It also searches for at least one child httpd daemon. If the parent process and at least one child do not exist, VCS reports the resource as offline. If they do exist and if the agent attribute LevelTwoMonitorFreq is set, the Apache agent uses the Apache Benchmarking utility "ab" to perform detail monitoring. If the exit code of the "ab" utility is 0 and if the command output contains "Benchmarking HostName", the agent considers the server online, else the agent considers the server offline.

If the binary file `ab` is not found, Apache agent uses the `ab2` binary file for detail monitoring.

About bringing an Apache HTTP server online outside of VCS control

When you bring an Apache HTTP server online outside of VCS control, first source its environment file. Start the server with the `-f` option so the server knows which instance to start. You can then specify additional options (such as EnableSSL or SharedObjDir) that you want the server to use at start.

To start an Apache HTTP server outside of VCS control

1. Source the environment file if required.
2. Start the Apache HTTP server. You must use the `-f` option so that the agent can distinguish different instances of the server.

   ```
   httpdDir/httpd -f ConfigFile -k start
   ```

   In the above-mentioned command, replace *httpdDir* with /apache/v2.2/bin and *ConfigFile* with /apache/v2.2/conf/httpd.conf. When fully formed, the start example looks like:

   ```
   /apache/v2.2/bin/httpd -f /apache/v2.2/conf/httpd.conf -k start
   ```

3. Specify additional options such as EnableSSL or SharedObjDir that you want to use when you start server. When you add EnableSSL to the command, it resembles:

   ```
   httpdDir/httpd -f ConfigFile -k start -DSSL
   ```

   Note: You can specify the full path of a binary file without having httpd as part of httpdDir attribute.

   For example: `/usr/sbin/apache2 -f /etc/httpd/conf/httpd.conf -k start`

About high Availability fire drill

The high availability fire drill detects discrepancies between the VCS configuration and the underlying infrastructure on a node. These discrepancies might prevent a service group from going online on a specific node.

For Apache resources, when the Apache HTTP server is installed locally, the high availability fire drill checks for the validity of these attributes:

- ConfigFile
- httpdDir

For more information about using the high availability fire drill see the *Cluster Server Administrator’s Guide*.

Using Apache agent with IMF

The Apache agent only supports intelligent monitoring during the online operation. The agent registers the following two processes for Apache IMF:

- Process with parent PID init.
- Child process with the maximum elapsed time.

By default, the IMF Mode is set to 2. If the IMF Mode is set to 1 or 3, the offline registration with IMF fails repeatedly until RegisterRetryLimit is reached.

Troubleshooting Apache service group issue

When you restart a node, zone, or LDom, the directory containing the PID file gets emptied and hence the Apache PID file directory is not present on the system.

For example, if `/var/run/apache2` is missing, the PID file cannot be created and the start command `$HttpdDir/httpd -f` *`config_file`* `-k start` fails with the following error message:

```
No such file or directory:
could not create /var/run/apache2/httpd.pid
```

As the startup command fails outside of VCS control, the Apache agent fails to bring the Apache service group online.

You can resolve this issue in one of the following ways:

- Ensure that the directory to write Apache PID files is present persistently on the node, zone, or LDom.
  For example, in the `httpd.conf` file, change the PidFile attribute to the persistently available location: /apache/my_conf/http.pid.
- Write a preonline trigger script to create the required directory for the Apache service group.
  For example: `mkdir /var/run/apache2`
  Refer to the *Cluster Server Administrator's Guide* for more details on VCS triggers.

Sample configurations

Basic configuration for Solaris

The following is a basic configuration for the resource.

```
group ApacheG1(
SystemList = { host1 = 0, host2 = 1 }
)
Apache httpd_server (
httpdDir = "/apache/bin"
HostName = vcssol1
Port = 8888
User = root
ConfigFile = "/apache/conf/httpd.conf"
)
DiskGroup Apache_dg (
DiskGroup = apc1
)
IP Apache_ip (
Device = bge0
Address = "11.123.99.168"
NetMask = "255.255.254.0"
)
Mount Apache_mnt (
MountPoint = "/apache"
BlockDevice = "/dev/vx/dsk/apc1/apcvol1"
FSType = vxfs
FsckOpt = "-y"
)
Apache_mnt requires Apache_dg
httpd_server requires Apache_mnt
httpd_server requires Apache_ip
```

Basic IPv6 configuration

The following is a basic IPv6 configuration for the resource.

```
group ipv6group (
    SystemList = { sysA = 0, sysB = 1 }
    )

    Apache ipv6group_apache_res (
        HostName = "fd4b:454e:205a:110:211:25ff:fe7e:118"
        PidFile = "/myapache/apache/logs/httpd.pid"
        httpdDir = "/myapache/apache/bin"
        ConfigFile = "/myapache/apache/conf/httpd.conf"
        IntentionalOffline = 1
        )

    DiskGroup ipv6group_dg_res (
        DiskGroup = dg01
        )

    IP ipv6group_ip_res (
        Device = bge0

        Address = "fd4b:454e:205a:110:211:25ff:fe7e:118"
        PrefixLen = 64
        )

    Mount ipv6group_mnt_res (
        MountOpt = rw
        FsckOpt = "-n"
        BlockDevice = "/dev/vx/dsk/dg01/vol01"
        MountPoint = "/myapache/apache"
        FSType = vxfs
        )

    NIC ipv6group_nic_res (

Device = bge0


)
Volume ipv6group_vol_res (
        Volume = vol01
        DiskGroup = dg01
        )

    ipv6group_apache_res requires ipv6group_mnt_res
    ipv6group_apache_res requires ipv6group_ip_res
    ipv6group_mnt_res requires ipv6group_vol_res
    ipv6group_vol_res requires ipv6group_dg_res
    ipv6group_ip_res requires ipv6group_nic_res
```

Sample output of the amfstat command

The following is a sample output of the `amfstat` command:

```
IMFD
=====
RID     PID
7       7929886

Registered Reapers (3):
=======================
RID PID       MONITOR TRIGG   REAPER
29  13041840  1       0       VCSMountAgent
30  9175060   2       0       Apache
```

```
31  12189854  1       0       DiskGroup

Process ONLINE Monitors (2):
============================
RID   R_RID   PID       GROUP
34    30      6488150   httpd_server
35    30      8847606   httpd_server

Mount ONLINE Monitors (1):
==========================
RID R_RID FSTYPE DEVICE          MOUNTPOINT
33  29    vxfs   /dev/vx/dsk/Apache_Conf/apache_vol /Apache
GROUP   CONTAINER
Apache_mnt none

DG online Monitors (1):
==========================
RID    R_RID   GROUP           DGName
32     31      Apache_dg       Apache_Conf
```

Debug log level

The Apache agent uses the following debug log level:

DBG_5

Application agent

The Application agent brings applications online, takes them offline, and monitors their status. Use it to specify different executables for the online, offline, and monitor routines for different programs. The executables can be on local storage or shared storage. You can use this agent to provide high availability for applications that do not have bundled, enterprise, or custom agents.

An application runs in the default context of root. Specify the user name to run an application in a user context.

You can monitor the application in the following ways:

- Use the monitor program
- Specify a list of processes
- Specify a list of process ID files

- Any combination of the above

This agent is zone-aware. The ContainerOpts resource type attribute for this type has a default value of 1 for RunInContainer and a default value of 0 for PassCInfo. Veritas recommends that you do not change these values.

Prevention Of Concurrency Violation (ProPCV) can be enabled to prevent an online resource on a node from coming online on another node, outside of VCS control, in the same cluster. In that, ProPCV prevents the execution of StartProgram and processes that are configured in MonitorProcesses on the offline node. This action prevents data corruption of resources and detects concurrency violation at an early stage. The attribute can only be set for a local failover type group. To enable this feature you need to set the ProPCV attribute value to 1. For more information about ProPCV, refer to the *Cluster Server Administrator’s Guide.*

IMF awareness

The Application agent is IMF-aware and uses asynchronous monitoring framework (AMF) kernel driver for IMF notification. For more information about IMF and intelligent resource monitoring, refer to the *Cluster Server Administrator’s Guide*.

For more information about IMF-related Application agent functions, see Agent functions.

High availability fire drill for Application agent

The high availability fire drill detects discrepancies between the VCS configuration and the underlying infrastructure on a node. These discrepancies might prevent a service group from going online on a specific node. For Application resources, the high availability fire drill checks for:

- The availability of the specified program and execution permissions for the specified program (program.vfd)
- The existence of the specified user on the host (user.vfd)
- The existence of the same binary on all nodes (cksum.vfd)

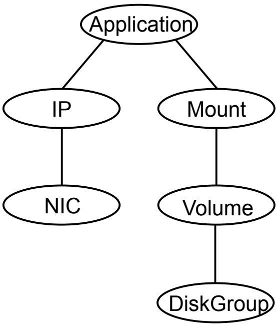
For more information, refer to the *Cluster Server Administrator’s Guide*.

Dependencies for Application agent

No fixed dependency exists for Application agent.

Depending on how you plan to use it, an Application type of resource can depend on IP and Mount resources. Alternatively, instead of the IP resource you can also use the IPMultiNICB resource.

Figure 5-2 Sample service group that includes an Application resource



Agent functions

Online

Runs the executable file with the parameters that are specified in the StartProgram attribute in the context of the specified user.

To bring the resource online, the agent function performs the command:

```
su [-] user -c executable_to_online_resource
```

When DelayAfterOnline attribute is set, the monitor function is invoked after completion of online function, and after the number of seconds specified in DelayAfterOnline attribute have elapsed.

Offline

Runs the executable file with the parameters that are specified in the StopProgram attribute in the context of the specified user.

To take the resource offline, the agent function performs the command:

```
su [-] user -c executable_to_offline_resource
```

When DelayAfterOffline attribute is set, the monitor function is invoked after completion of offline function and after the number of seconds specified in DelayAfterOffline attribute have elapsed.

Monitor

How the agent performs this function depends on the attributes that you specify:

- If you specify the MonitorProgram attribute, the agent executes the user-defined monitor program in the user-specified context.
- If you specify the PidFiles attribute, the routine verifies that the process ID that is found in each listed file is running.
- If you specify the MonitorProcesses attribute, the routine verifies that each listed process is running in the context of the user that you specify.

You can use any combination of the MonitorProgram, PidFiles, or MonitorProcesses attributes to monitor the application.

If this function determines that one or more of the processes that are specified in either PidFiles or MonitorProcesses are not running, it returns OFFLINE. If the process terminates ungracefully, the monitor returns OFFLINE, and a failover occurs.

To monitor the resource, the agent function performs the command:

```
su [-] user -c executable_to_monitor_resource
```

imf_init

Initializes the agent to interface with the asynchronous monitoring framework (AMF) kernel driver. This function runs when the agent starts up.

imf_getnotification

Gets notification about resource state changes. This function runs after the agent initializes with the AMF kernel driver. The agent continuously waits for notification and takes action on the resource upon notification.

imf_register

Registers the resource entities, which the agent must monitor, with the AMF kernel driver. For example, the function registers the PID for online monitoring of a process. This function runs for each resource after the resource goes into steady state (online or offline). The Application agent uses IMF for the processes configured with PidFiles and the MonitorProcesses attribute.

Clean

Terminates processes specified in PidFiles or MonitorProcesses. Ensures that only those processes (that are specified in the MonitorProcesses attribute) running with the user ID specified in the User attribute are killed. If the CleanProgram is defined, the agent executes the CleanProgram.

To forcefully stop the resource, the agent function performs the command:

```
su [-] user -c executable_to_clean_resource
```

Note that the agent uses the `su -` option only when the attribute UseSUDash is enabled (1). The UseSUDash attribute is disabled (0) by default.

Action

The various functions of the action entry point are as follows:

- program.vfd
  Checks the availability of the specified program and the execution permissions for the specified program.
- user.vfd
  Checks the existence of the specified user on the host.
- cksum.vfd
  Checks the existence of the same binary on all nodes.
- propcv
  [For internal use only] Invokes the AMF call with arguments to decide whether to allow or prevent processes from starting for an application resource, outside the VCS control, in the cluster. The StartProgram and the processes configured under MonitorProcesses, registered with AMF for offline monitoring, are prevented from starting on the offline node. This helps prevent concurrency violation at an early stage.
- getcksum
  Returns the checksum of the specified program

State definitions for Application agent

ONLINE

Indicates that all processes that are specified in the PidFiles and the MonitorProcesses attribute are running and that the MonitorProgram returns ONLINE.

OFFLINE

Indicates that at least one process that is specified in the PidFiles attribute or MonitorProcesses is not running, or that the MonitorProgram returns OFFLINE.

UNKNOWN	Indicates an indeterminable application state or invalid configuration or that the required attributes have not been configured.
FAULTED	Indicates that the process has terminated unexpectedly or MonitorProgram returns OFFLINE unexpectedly.

Attributes for Application agent

Table 5-4 Required attributes for Solaris

Required attribute	Description
StartProgram	The executable, which starts the application. Specify the complete path of the executable. Applicable command line arguments follow the name of the executable and have spaces separating them. This executable can be on local storage or shared storage. For example, if the attribute for StartProgram is /usr/sbin/vxnotify -g dg00 -m >> /var/log/vxnotify.log (and vxnotify is blocking command) set it like: /usr/sbin/vxnotify -g dg00 -m >> /var/log/vxnotify.log & For applications running in Solaris zones, use the path as seen from the non-global zone. **Note:** The agent logs the return value of the StartProgram executable. The agent does not treat a non-zero return value as failure of execution and brings the resource online. **Note:** Do not use the opening and closing ({ }) brace symbols in this string. **Note:** In the script, specify a return value that is between 0 and 255. Type and dimension: string-scalar Example: "/usr/sbin/sample_app start"

Table 5-4 Required attributes for Solaris *(continued)*

Required attribute	Description
StopProgram	The executable, which stops the application. Specify the complete path of the executable. Applicable command line arguments follow the name of the executable and have spaces separating them. This executable can be on local storage or shared storage. For applications running in Solaris zones, use the path as seen from the non-global zone. **Note:** The agent logs the return value of the StopProgram executable. The agent does not treat a non-zero return value as failure of execution and takes the resource offline. **Note:** Do not use the opening and closing ({ }) brace symbols in this string. **Note:** In the script, specify a return value that is between 0 and 255. Type and dimension: string-scalar Example: "/usr/sbin/sample_app stop"
At least one of the following attributes: ■ MonitorProcesses ■ MonitorProgram ■ PidFiles	See Table 5-5 on page 203.

Table 5-5 Optional attributes for Solaris

Optional attribute	Description
CleanProgram	The executable, which forcibly stops the application. Specify the complete path of the executable. Applicable command line arguments follow the name of the executable and have spaces separating them. This executable can be on local storage or shared storage. For applications running in Solaris zones, use the path as seen from the non-global zone. **Note:** Veritas recommends to have the CleanProgram on the local storage so that in case of loss of storage connectivity VCS can take appropriate action to stop the application. **Note:** If the CleanProgram executable returns a non-zero value, the agent treats it as a clean failure and the resource does not fault. Type and dimension: string-scalar Example: "/usr/sbin/sample_app force stop"
MonitorProcesses	A list of processes that you want to be monitored and cleaned. Each process name is the name of an executable. Provide the full path name of the executable if the agent uses that path to start the executable. The process name must be the full command line argument that the following command displays for the process: ■ Solaris 11: `/usr/bin/ps ww` *`pid`* Type and dimension: string-vector Example: "/app/sample_bin" or "/usr/bin/bash /app/sample_process.sh"

Table 5-5 Optional attributes for Solaris *(continued)*

Optional attribute	Description
MonitorProgram	The executable, which monitors the application. Specify the complete path of the executable. Applicable command line arguments follow the name of the executable and have spaces separating them. This executable can be on local storage or shared storage. For applications running in Solaris zones, use the path as seen from the non-global zone. MonitorProgram can return the following VCSAgResState values: OFFLINE value is 100 or 1; ONLINE values range from 101 to 110 or 0 (depending on the confidence level); 110 equals confidence level of 100%. Any other value = UNKNOWN. **Note:** Do not use the opening and closing ({ }) brace symbols in this string. If MonitorProgram is configured and not available, then resource state will be: ■ OFFLINE if the resource was in OFFLINE state and not waiting for any action ■ UNKNOWN if the resource was in any other state or waiting for some action. Type and dimension: string-scalar Example: "/usr/sbin/sample_app_monitor all"
PidFiles	A list of PID (process ID) files that contain the PID of the processes that you want monitored and cleaned. These are application generated files. Each PID file contains one monitored PID. Specify the complete path of each PID file in the list. For applications running in Solaris zones, use the path as seen from the non-global zone. The process ID can change when the process restarts. If the application takes time to update the PID file, the agent’s monitor function may return an incorrect result. If incorrect results occur, increase the ToleranceLimit in the resource definition. Type and dimension: string-vector Example: "/var/lock/samba/smbd.pid"

Table 5-5 Optional attributes for Solaris *(continued)*

Optional attribute	Description
User	The user name for running StartProgram, StopProgram, MonitorProgram, and CleanProgram. The processes that are specified in the MonitorProcesses list must run in the context of the specified user. Monitor checks the processes to make sure they run in this context. **Note:** If the configured user does not exist, the resource state will be UNKNOWN. Type and dimension: string-scalar Default: root Example: user1
EnvFile	The environment file that should get sourced before running any of the StartProgram, StopProgram, MonitorProgram or CleanProgram. **Note:** Please make sure that the EnvFile adheres the default shell syntax of the configured use. Type and dimension: string-scalar Default: "" Example: /home/username/envfile
UseSUDash	When the value of this attribute is 0, the agent performs an `su user` command before it executes the StartProgram, the StopProgram, the MonitorProgram, or the CleanProgram agent functions. When the value of this attribute is 1, the agent performs an `su - user` command before it executes the StartProgram, the StopProgram, the MonitorProgram or the CleanProgram agent functions. Type and dimension: boolean-scalar Default: 0 Example: 1
DelayAfterOnline	Specifies the number of seconds that elapse after the Online entry point is complete and before the next monitor cycle is invoked. Type and dimension: integer-scalar Default: 0

Table 5-5 Optional attributes for Solaris *(continued)*

Optional attribute	Description
DelayAfterOffline	Specifies the number of seconds that elapse after the Offline entry point is complete and before the next monitor cycle is invoked. Type and dimension: integer-scalar Default: 0

Resource type definition for Application agent

```
type Application (
   static keylist RegList = { MonitorProcesses, User }
   static int IMF{} = { Mode=3, MonitorFreq=1, RegisterRetryLimit=3 }
   static str IMFRegList[] = { MonitorProcesses, User, PidFiles,
   MonitorProgram, StartProgram, LevelTwoMonitorFreq }
   static keylist SupportedActions = { "program.vfd", "user.vfd",
   "cksum.vfd", getcksum, propcv }
   static int LevelTwoMonitorFreq = 1
   static str ArgList[] = { State, IState, User, StartProgram,
   StopProgram, CleanProgram, MonitorProgram, PidFiles, MonitorProcesses,
   EnvFile, UseSUDash, DelayAfterOnline, DelayAfterOffline }
   static int ContainerOpts{} = { RunInContainer=1, PassCInfo=0 }
   str User = root
   str StartProgram
   str StopProgram
   str CleanProgram
   str MonitorProgram
   str PidFiles[]
   str MonitorProcesses[]
   str EnvFile
   int DelayAfterOnline
   int DelayAfterOffline
   boolean UseSUDash = 0
)
```

Notes for Application agent

Using Application agent with IMF

Intelligent monitoring is supported for the Application agent only under specific configurations. The complete list of such configurations is provided in the following table:

Table 5-6

MonitorProgram	MonitorProcesses	PidFiles	IMF Monitoring Mode
Not Configured	Not Configured	Not Configured	Not Applicable
Not Configured	Not Configured	Configured	Online, Offline
Not Configured	Configured	Not Configured	Online, Offline
Not Configured	Configured	Configured	Online, Offline
Configured	Not Configured	Not Configured	Offline Only
Configured	Not Configured	Configured	Online, Offline
Configured	Configured	Not Configured	Online, Offline
Configured	Configured	Configured	Online, Offline

Note: When you do not configure MonitorProcesses, IMF monitors only the StartProgram on the offline node. Hence, the MonitorFreq of IMF attribute must be set to 1 so that IMF monitors the resource on the offline node every monitor cycle.

When multiple processes are configured under the MonitorProcesses attribute and only some of them are running, offline registration with IMF fails repeatedly until RegisterRetryLimit is reached. In such a scenario, IMF cannot determine when the resource goes ONLINE and the agent monitors the resource in the traditional way.

Level two monitoring through MonitorProgram

MonitorProgram can be executed as a second level monitor whereas PidFiles/MonitorProcesses are monitored as first level monitor. To enable level two monitoring for the Application agent, the LevelTwoMonitorFreq attribute of Application type has to be set to a value greater than zero. When configured, the MonitorProgram is executed in monitoring cycles at intervals specified in LevelTwoMonitorFreq attribute.

For example, if j is the value of the MonitorFreq key of the IMF attribute and k is the value of the LevelTwoMonitorFreq attribute, and if the resource is in online state, then traditional monitors for PidFiles/MonitorProcesses run in every j-th monitor cycle and MonitorProgram runs in every k-th monitor cycle.

When MonitorProgram runs as a second level monitor by setting the LevelTwoMonitorFreq value, the limitation of Application agent to leverage IMF for monitoring PidFiles/MonitorProcesses when resource is in online state is overcome. The processes configured in PidFiles/MonitorProcesses are then registered for IMF monitoring.

If the LevelTwoMonitorFreq attribute is set to zero and when MonitorProgram is configured, then none of the processes specified in PidFiles/MonitorProcesses are registered with IMF for monitoring when the resource is online. In this case, MonitorProgram and the checks for PidFiles and MonitorProcesses execute in every monitor cycle.

LevelTwoMonitorFreq is a type-level attribute. The default value for the LevelTwoMonitorFreq attribute is one (1) so by default MonitorProgram runs as a second level monitor in every monitor cycle. Any changes to this attribute at the Application type level changes the behavior for all application resources.

To modify the LevelTwoMonitorFreq value at type level to a non-default value (for example, 3), execute the following command:

```
# hatype -modify Application LevelTwoMonitorFreq 3
```

If you want to change the LevelTwoMonitorFreq value for selected resources, execute the following commands for each resource in the following sequence. Note that the LevelTwoMonitorFreq value used in the command is only an example.

```
# hares -override app_res_name LevelTwoMonitorFreq
```

```
# hares -modify app_res_name LevelTwoMonitorFreq 3
```

The preceding commands override the LevelTwoMonitorFreq attribute at resource level and modify the value of the attribute for a particular resource.

Using Application agent with ProPCV

ProPCV functionality prevents the StartProgram and binary-based processes that are configured under MonitorProcesses from executing on the offline node. This action detects concurrency violation at an early stage in the cycle. However, ProPCV does not prevent script-based processes that are configured under MonitorProcesses from executing on the offline node. Considerations for ProPCV to function:

- You must run the StartProgram with the same order of arguments as configured in the StartProgram attribute. If you change the order of arguments, ProPCV does not prevent the execution of StartProgram. This causes delay in detecting concurrency violation.

For example, a single command can be run in multiple ways:

```
/usr/bin/tar -c -f a.tar

/usr/bin/tar -f a.tar -c
```

So, ProPCV does not function if you run the command in a way that is not configured in the StartProgram attribute.

- You must start the StartProgram by using the commands or the way specified in StartProgram attribute. But if you use another way or command to start the program that is not specified in the attribute, ProPCV does not prevent the startup of the program. This causes delay in detecting concurrency violation.
- If StartProgram is a script, the script must have the interpreter path as the first line and start with #!.

  For example, a shell script should start with “#!/usr/bin/sh”.
- If the StartProgram is a script, do not change the interpreter path in the script file after the StartProgram is registered for offline monitoring. Else, ProPCV may not function for the StartProgram.
- You must not append the StartProgram attribute with the special character **&**. For example, '/app/start.sh &'.

Requirement for programs

The programs specified in StartProgram, StopProgram, MonitorProgram, CleanProgram should not continuously write to STDOUT or STDERR. If required, please redirect STDOUT and STDERR to some file.

Requirement for default profile

The default profile of configured user should not have any blocking command such as `bash` or any other command such as `exec` that changes the behavior of the shell. This may lead to unexpected behavior.

Support for cloned Application agent

The Application agent is used to make applications highly available when an appropriate ISV agent is not available. To make multiple different applications highly available using a cluster, you must create a service group for each application. InfoScale lets you clone the Application agent so that you can configure a different service group for each application. You must then assign the appropriate operator permissions for each service group for it to function as expected.

Note: A cloned Application agent is also IMF-aware.

To clone agent

1 Stop the cluster.

```
# hastop -all -force
```

2 On each node, copy the Application agent directory, and rename the agent as follows:

```
# cd /opt/VRTSvcs/bin

# cp -r Application newAppName

# cd newAppName

# mv ApplicationAgent newAppNameAgent
```

3. On any one cluster node, navigate to the following directory:

   ```
   # cd /etc/VRTSvcs/conf/config
   ```

   Create a *newAppName*Agent.cf file in this directory with following content:

   ```
   type newAppName (
   static int IMF{} = { Mode=3, MonitorFreq=1, RegisterRetryLimit=3 }
   static str IMFRegList[] = { MonitorProcesses, User, PidFiles,
     MonitorProgram, StartProgram, LevelTwoMonitorFreq }
   static keylist SupportedActions = { "program.vfd", "user.vfd",
     "cksum.vfd", getcksum, propcv }
   static int LevelTwoMonitorFreq = 1
   static str ArgList[] = { User, StartProgram, StopProgram,
     CleanProgram, MonitorProgram, PidFiles, MonitorProcesses,
     EnvFile, UseSUDash, State, IState, StartOnly }
   static int ContainerOpts{} = { RunInContainer=1, PassCInfo=0 }
   str User = root
   str StartProgram
   str StopProgram
   str CleanProgram
   str MonitorProgram
   str PidFiles[]
   str MonitorProcesses[]
   str EnvFile
   boolean UseSUDash = 0
   boolean StartOnly = 0
   )
   ```

   Include the *newAppName*Agent.cf file in main.cf.

   Then, start the cluster.

   ```
   # hastart
   ```

4. Start the cluster on all the other nodes to propagate the addition of cloned new application agent.

Sample cloned Application agent configuration

The following sample includes an application (app1) and a cloned application (my_app1).

```
Application app1 (
    StartProgram = "/opt/app1/start"
    StopProgram = "/opt/app1/stop"
    CleanProgram = "/opt/app1/stop"
```

```
    MonitorProgram = "/opt/app1/monitor"
    PidFiles = { "/tmp/app1.pid" }
    )

MyApplication my_app1 (
    StartProgram = "/opt/my_app1/start"
    StopProgram = "/opt/my_app1/stop"
    CleanProgram = "/opt/my_app1/stop"
    MonitorProgram = "/opt/my_app1/monitor"
    PidFiles = { "/tmp/my_app1.pid" }
    )
```

Sample configurations for Application agent

The sample configurations for this agent follow:

Configuration 1 for Application agent

In this example, you configure the executable sample_app as StartProgram and StopProgram, with start and stop specified as command line arguments respectively. Configure the agent to monitor two processes: a process that the app.pid specifies and the process sample_app.

```
Application samba_app (
User = "root"
StartProgram = "/usr/sbin/sample_app start"
StopProgram = "/usr/sbin/sample_app stop"
PidFiles = { "/var/lock/sample_app/app.pid" }
MonitorProcesses = { "sample_app" }
)
```

Configuration 2 for Application agent

In this example, since no user is specified, it uses the root user. The executable sample_app starts and stops the application using start and stop as the command line arguments. The executable sample_app_monitor monitors the application and uses all as its command line argument. The agent also monitors the sample_app1 and sample_app2 processes.

```
Application samba_app2 (
StartProgram = "/usr/sbin/sample_app start"
StopProgram = "/usr/sbin/sample_app stop"
CleanProgram = "/usr/sbin/sample_app force stop"
MonitorProgram = "/usr/local/bin/sample_app_monitor all"
MonitorProcesses = { "sample_app1", "sample_app2" }
)
```

Configuration 3 for Application agent

In this example, configure a resource in a non-global zone: zone1. The ZonePath of zone1 is /zone1/root. Configure the executable samba as StartProgram and StopProgram, with start and stop specified as command line arguments respectively. Configure the agent to monitor two processes: a process that the smbd.pid specifies and the process nmbd.

```
Application samba_app (
StartProgram = "/usr/sbin/samba start"
StopProgram = "/usr/sbin/samba stop"
PidFiles = { "/var/lock/samba/smbd.pid" }
MonitorProcesses = { "nmbd" }
)
```

Debug log levels for Application agent

The Application agent uses the following debug log levels:

DBG_1, DBG_2, DBG_3, DBG_4, DBG_5

CoordPoint agent

Use the Coordination Point (CoordPoint) agent to monitor the registrations on the different coordination points on each node.

In addition, the CoordPoint agent monitors changes to the Coordinator Disk Group constitution, such as when a disk is accidently deleted from or added to the Coordinator Disk Group or if the VxVM private region of a disk is corrupted.

The agent also performs detailed monitoring on the CoordPoint resource. You can tune the frequency of the detailed monitoring with the LevelTwoMonitorFreq attribute. For example, if you set this attribute to 5, the agent monitors the Coordinator Disk Group constitution in every fifth monitor cycle.

The CoordPoint agent is a monitor-only agent that runs on each node within the client cluster. It can monitor Coordination Point (CP) servers and SCSI-3 disks.

Coordination Point server as a coordination point

When you have configured a CP server as a coordination point, the CoordPoint agent performs the following tasks:

- Confirms that the client cluster can communicate with the CP server coordination point.
- Validates the node registrations in the CP server database using the `cpsadm` command.

SCSI-3 based disk as a coordination point

In case the coordination point is a SCSI-3 based disk, the CoordPoint agent uses the `vxfenadm` command to confirm that the registered keys on the disk are intact. The Monitor agent function contains the monitoring functionality for SCSI-3 disks and CP servers.

If the agent detects an anomaly, the agent reports it to you so you can repair the coordination point. You may have to perform an online coordinator point replacement procedure if the problem is isolated to the keys registered or you can repair the coordination points automatically (if some keys are missing on one or more coordination points) by configuring the `ActionOnCoordPointFault` attribute. For more information on using the attribute, refer to the Attributes section.

Note: The CoordPoint agent that runs on a given client cluster node monitors the keys for coordination points visible to that node alone.

Note: [Solaris SPARC only] The `ActionOnCoordPointFault` attribute is not supported in Solaris 11 x86 platform.

For important information about this agent, refer to:

See “Notes for the CoordPoint agent” on page 218.

Dependencies

No dependencies exist for the CoordPoint resource.

Agent functions

Monitor

Enables the CoordPoint agent to validate the node registrations in the coordination points and confirms that the coordination points are accessible. In addition, enables the agent to monitor disks in the Coordinator Disk Group. Specifically, if a disk is deleted from or added to the disk group or the VxVM private region of a disk is corrupted.

CoordPoint resources are persistent, which means that they cannot be brought online or taken offline. They can only monitor the coordination point registrations.

The CoordPoint agent also performs I/O fencing reporting activities.

See “CoordPoint agent I/O fencing reporting activities” on page 218.

State definitions

ONLINE

Indicates that the CoordPoint resource is working.

UNKNOWN

Indicates the agent cannot determine the coordination points resource's state. This state may be due to an incorrect configuration.

FAULTED

Indicates that CoordPoint resource is reported for one or more of the following conditions:

- The number of coordination points with missing keys (or registrations) has exceeded the value of the FaultTolerance attribute.
- The number of unreachable coordination points has exceeded the value of the FaultTolerance attribute.
- Coordinator disks are deleted from or added to the Coordinator Disk Group.
- Public character path of a disk and the device path that corresponds to the device number of that disk in the kernel driver do not match.

Attributes

Table 5-7 Required attribute

Required attribute	Description
FaultTolerance	The FaultTolerance attribute determines when the CoordPoint agent declares that the registrations on the coordination points are missing or connectivity between the nodes and the coordination points is lost. If the number of coordination points with missing keys (or registrations) and or the number of unreachable coordination points exceeds the value of the FaultTolerance attribute, then the agent reports FAULTED. Set the value of this attribute depending on your own configuration requirements. For example, if the FaultTolerance value is set to 1, then the CoordPoint agent reports FAULTED if it sees 2 or more number of coordinator points with missing keys (or registrations) and or the number of unreachable coordination points. Change the value of the FaultTolerance attribute either before the CoordPoint agent starts to monitor or while the CoordPoint agent is monitoring. If the attribute is set while the CoordPoint agent is monitoring, then the CoordPoint agent reads the new value in the next monitor cycle. To view the current FaultTolerance value, enter the following command: # hares -display coordpoint -attribute FaultTolerance Type and dimension: integer-scalar Default: "0"

Table 5-8 Optional attribute

Optional attribute	Description
ActionOnCoordPointFault	[Solaris SPARC only] This attribute determines whether lost registration keys (if any) on any coordination point can be automatically replaced. It also determines whether to take corrective action if the public character path of a coordinator disk does not match with the device path of that coordinator disk in the kernel driver. By default, the attribute is disabled. To enable the attribute set its value to `RefreshRegistrations`. If the refresh procedure fails twice consecutively, it is not attempted again on that node. You can re-enable the refresh procedure on that node by executing the hares command. # `/opt/VRTS/bin/hares -action coordpoint enable_refresh -sys <sys_name>` Type and dimension: string-scalar Default: None To enable the attribute: Set its value to `RefreshRegistrations`

Resource type definition

The ActionOnCoordPointFault is not supported on Solaris 11 x86 platform.

```
type CoordPoint (
    static keylist SupportedActions = { enable_refresh }
    static int InfoInterval = 300
    static int OfflineMonitorInterval = 60
    static str ArgList[] = { FaultTolerance,
    ActionOnCoordPointFault, RefreshLockHolderSysName }
    static str Operations = None
    int FaultTolerance
    str ActionOnCoordPointFault = None
    str RefreshLockHolderSysName = None
)
```

Notes for the CoordPoint agent

The notes are as follows:

CoordPoint agent I/O fencing reporting activities

The CoordPoint agent also performs the following I/O fencing reporting activities:

- Checks to determine if I/O fencing is running.
  If I/O fencing is not running, then the CoordPoint agent reports failure.
- Checks the mode of fencing operation. I/O fencing can operate in one of the following three modes:
  - SCSI-3 mode: If I/O fencing runs in SCSI-3 mode, then the CoordPoint agent continues to monitor.
  - Customized mode: If I/O fencing runs in Customized Fencing mode, then the CoordPoint agent continues to monitor.
  - Disabled mode: If I/O fencing runs in disabled mode, no action is required. The CoordPoint agent returns success.
  - Majority mode: If I/O fencing runs in majority mode, no action is required. The CoordPoint agent returns success.

AutoStartList attribute

AutoStartList is a service group attribute that needs to be populated with a system list. The VCS engine brings up the specified service group on the nodes in the list.

AutoStartList is not a required attribute for the service group that contains the CoordPoint resource. The CoordPoint resource is a persistent resource and when a service group is configured with this type of resource, it cannot be brought online.

Specifying the AutoStartList with a system list does not change the behavior of the service group.

Detailed monitoring for the Coordpoint resource

The agent fetches disk names and unique identifiers from the kernel driver for I/O fencing. It checks for disks that are no longer part of the Coordinator Disk Group and also checks for any newly added disks to the disk group. It also compares the public character path of the disks with the device path stored in the kernel driver. The agent faults the resource when any of the checks fail.

The ActionOnCoordPointFault attribute set to RefreshRegistrations

Note: Applicable only for Solaris SPARC

The ActionOnCoordPoint attribute has an impact on the resource state. If the refresh procedure completes successfully, then you might see the resource temporarily going into FAULTED state. In the next monitor cycle, the resource comes back to ONLINE state without any external trigger.

Sample configuration

In this example, the coordination point agent type resource is configured with the value of the FaultTolerance attribute set to 0. At this value setting, the CoordPoint agent reports FAULTED, when the agent determines that at least one coordination point has keys (or registrations) missing and or one coordination point is not reachable.

The following is an example service group (vxfen) extracted from a main.cf file:

```
group vxfen (
    SystemList = { sysA = 0, sysB = 1 }
    AutoFailOver = 0
    Parallel = 1
    AutoStartList = { sysA, sysB }
    )
        CoordPoint coordpoint (
            FaultTolerance=0
            LevelTwoMonitorFreq = 5
            )
    // resource dependency tree
    //
    //      group vxfen
    //      {
    //      CoordPoint coordpoint
    //      }
```

Debug log levels

The CoordPoint agent uses the following debug log levels:

DBG_10

Process agent

The Process agent starts, stops, and monitors a process that you specify. You can use the agent to make a process highly available.

This agent is zone-aware. The ContainerOpts resource type attribute for this type has a default value of 1 for RunInContainer and a default value of 0 for PassCInfo. Veritas recommends that you do not change these values.

For more information on ContainerOpts attribute refer to the *Veritas InfoScale Virtualization Guide*.

IMF awareness

The Process agent is IMF-aware and uses Asynchronous Monitoring Framework (AMF) kernel driver for IMF notification.

For more information about IMF and intelligent resource monitoring, refer to the *Cluster Server Administrator's Guide*.

For more information about IMF-related Process agent functions, see Agent functions for Process agent.

High availability fire drill for Process agent

The high availability fire drill detects discrepancies between the VCS configuration and the underlying infrastructure on a node; discrepancies that might prevent a service group from going online on a specific node.

For Process resources, the high availability fire drill checks for:

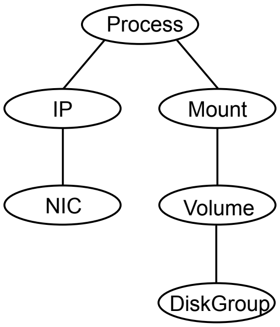
- The existence of a binary executable for the specified process (program.vfd)
- The existence of the same binary on all nodes (program.vfd)

For more information refer to the *Cluster Server Administrator's Guide*.

Dependencies for Process agent

Depending on the context, this type of resource can depend on IP, IPMultiNICB, Zone, or Mount resources.

Figure 5-3 Sample service group for a Process resource



Agent functions for Process agent

Online	Starts the process with optional arguments.
Offline	Terminates the process with a `SIGTERM`. If the process does not terminate, a `SIGKILL` is sent.
Monitor	Checks to see if the process is running by scanning the process table for the name of the executable pathname and argument list.
Clean	Terminates all ongoing resource actions and takes the resource offline, forcibly when necessary.
imf_init	Initializes the agent to interface with the asynchronous monitoring framework (AMF) kernel driver. This function runs when the agent starts up.
imf_getnotification	Gets notification about resource state changes. This function runs after the agent initializes with the AMF kernel driver. The agent continuously waits for notification and takes action on the resource upon notification.
imf_register	Registers the resource entities, which the agent must monitor, with the AMF kernel driver. For example, the function registers the PID for online monitoring of a process. This function runs for each resource after the resource goes into steady state (online or offline).

State definitions for Process agent

ONLINE	Indicates that the specified process is running. The agent only reports the process as online if the value configured for PathName attribute exactly matches the process listing from the ps output along with the arguments.
OFFLINE	Indicates that the specified process is not running.
FAULTED	Indicates that the process has terminated unexpectedly.
UNKNOWN	Indicates that the agent can not determine the state of the process.

Attributes for Process agent

Table 5-9 Required attribute

Required attribute	Description
PathName	Absolute path to access an executable program. This path includes the program name. If a script controls the process, the PathName defines the complete path to the shell. This attribute must not exceed 80 characters. Type and dimension: string-scalar Example: "/usr/lib/sendmail"

Table 5-10 Optional attributes

Optional attribute	Description
Arguments	Passes arguments to the process. If a script controls the process, the script is passed as an argument. Separate multiple arguments with a single space. A string cannot accommodate more than one space between arguments, nor allow for leading or trailing whitespace characters. This attribute must not exceed 80 characters. Type and dimension: string-scalar Example: "-bd -q15m"

Note: For a process resource, the combination of PathName and Arguments attributes configured must be exactly the same as that is displayed in the output of the `ps` command.

Resource type definition for Process agent

```
type Process (
static keylist SupportedActions = { "program.vfd", getcksum }
static str ArgList[] = { PathName, Arguments }
static int ContainerOpts{} = { RunInContainer=1, PassCInfo=0 }
str PathName
str Arguments
)
```

Usage notes for Process agent

The Process agent has the following notes:

- Prerequisites for processes

Prerequisites for processes

- The processes specified in the PathName attribute must not continuously write to STDOUT or STDERR. If required, redirect STDOUT and STDERR to some file.
- The process must not modify its arguments. If the process modifies its arguments, the Process agent will not be able to monitor the process.

Sample configurations for Process agent

Configuration 1 for Process agent

Configuration 1 for Solaris follows:

```
Process usr_lib_sendmail (
    PathName = "/usr/lib/sendmail"
    Arguments = "-bd -q15m"
    )
```

Configuration 2 for Process agent

Configuration 2 follows:

```
include "types.cf"
cluster ProcessCluster (
.
.
.
group ProcessGroup (
SystemList = { sysa = 0, sysb = 1 }
    AutoStartList = { sysa }
)
Process Process1 (
PathName = "/usr/local/bin/myprog"
Arguments = "arg1 arg2"
)
Process Process2 (
PathName = "/bin/csh"
Arguments = "/tmp/funscript/myscript"
)
// resource dependency tree
//
// group ProcessGroup
// {
// Process Process1
// Process Process2
// }
```

Debug log levels for Process agent

The Process agent uses the following debug log levels:

DBG_1, DBG_4, DBG_5

ProcessOnOnly agent

The ProcessOnOnly agent starts and monitors a process that you specify. You can use the agent to make a process highly available or to monitor it. This resource’s Operation value is OnOnly.

This agent is zone-aware. The ContainerOpts resource type attribute for this type has a default value of 1 for RunInContainer and a default value of 0 for PassCInfo. Veritas recommends that you do not change these values.

Refer to the *Veritas InfoScale Virtualization Guide*.

VCS uses this agent internally to monitor security processes in a secure cluster.

Dependencies

No child dependencies exist for this resource.

Agent functions

Online — Starts the process with optional arguments.

Monitor — Checks to see if the process is alive by scanning the process table for the name of the executable pathname and argument list.

Clean — Terminates all ongoing resource actions and takes the resource offline, forcibly when necessary.

State definitions

ONLINE — Indicates that the specified process is running.

The agent only reports the process as ONLINE if the value configured for PathName attribute exactly matches the process listing from the ps output along with the arguments.

FAULTED — Indicates that the process has unexpectedly terminated.

UNKNOWN — Indicates that the agent can not determine the state of the process.

Attributes

Table 5-11 Required attributes

Required attribute	Description
PathName	Defines complete pathname to access an executable program. This path includes the program name. If a process is controlled by a script, the PathName defines the complete path to the shell. Pathname must not exceed 80 characters. The value configured for this attribute needs to match the process listing from the ps output for the agent to display as ONLINE. Type and dimension: string-scalar Example: "/usr/lib/nfs/nfsd"

Table 5-12 Optional attributes

Optional attribute	Description
Arguments	Passes arguments to the process. If a process is controlled by a script, the script is passed as an argument. Multiple arguments must be separated by a single space. A string cannot accommodate more than one space between arguments, nor allow for leading or trailing whitespace characters. Arguments must not exceed 80 characters (total). Type and dimension: string-scalar Example: "- a 8"
IgnoreArgs	A flag that indicates whether monitor ignores the argument list. ■ If the value is 0, it checks the process pathname and argument list. ■ If the value is 1, it only checks for the executable pathname and ignores the rest of the argument list. Type and dimension: boolean-scalar Default: 0

Resource type definition

```
type ProcessOnOnly (
    static str ArgList[] = { IgnoreArgs, PathName, Arguments }
    static int ContainerOpts{} = { RunInContainer=1, PassCInfo=0 }
    static str Operations = OnOnly
    boolean IgnoreArgs = 0
    str PathName
    str Arguments
)
```

ProcessOnOnly agent usage notes

The ProcessOnOnly agent has the following notes:

- Requirement for programs

Requirement for programs

The programs specified in PathName should not continuously write to STDOUT or STDERR. If required, please redirect STDOUT and STDERR to some other file.

Sample configurations

```
group testgrp (
SystemList = { sysA = 0, sysB = 1 }
Parallel = 1
AutoStartList = { sysA, sysB }
OnlineRetryLimit = 3
OnlineRetryInterval = 120
)
Phantom phantom_test (
)
ProcessOnOnly testres (
IgnoreArgs = 1
PathName = "/testApp/testproc"
)
```

Debug log levels

The ProcessOnOnly agent uses the following debug log levels:

DBG_1

Zone agent

The Zone agent brings online, takes offline, monitors, and cleans Solaris zones. You can use the agent to make zones highly available and to monitor them.

The ContainerOpts resource type attribute for this type has a default value of 0 for RunInContainer and a default value of 1 for PassCInfo. Veritas recommends that you do not change the values for these keys.

On Oracle Solaris 11, Solaris native zone is supported.

Before you bring a zone under VCS control, ensure that the autoboot attribute of the zone is set to false. If the attribute is set to true, the zone is automatically brought up at the time of starting the system.

IMF awareness

The Zone agent is IMF-aware and uses Asynchronous Monitoring Framework (AMF) kernel driver for IMF notification. For more information about IMF and intelligent resource monitoring, refer to the *Cluster Server Administrator’s Guide*.

Note: When zone goes into maintenance state, AMF does not detect the state. The change in state is detected by Zone monitor in the next cycle.

For more information, refer to the *Veritas InfoScale Virtualization Guide*.

For more information about IMF-related Zone agent functions, see Agent functions.

For more information about using the Zone agent with IMF, see Using the Zone agent with IMF.

Dependencies

Typically no dependencies are required for the Zone resource, however if the zone root is on shared storage the resource may require the Mount and DiskGroup resources.

Figure 5-4 Sample service group that includes a Zone resource when the zone root is on shared storage with a loopback file system. The loopback file system is mounted inside the zone using the zonecfg command

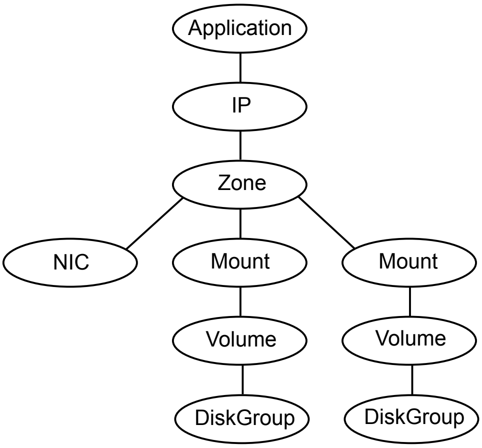


Figure 5-5 Sample service group that includes a Zone resource for the zone root on shared storage with a loopback file system when VCS manages the loopback file system as a Mount resource

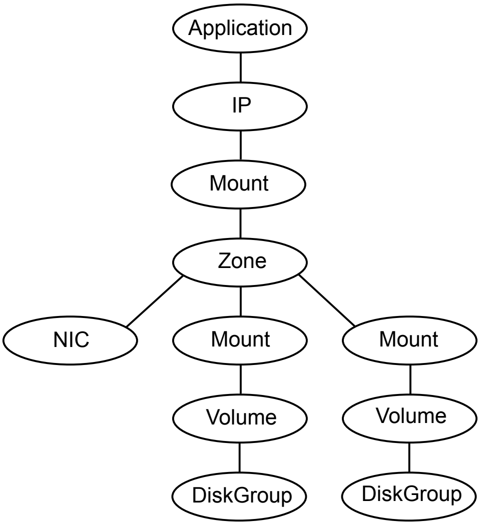


Figure 5-6 Sample service group that includes a Zone resource with the zone root on shared storage a direct mounted file system

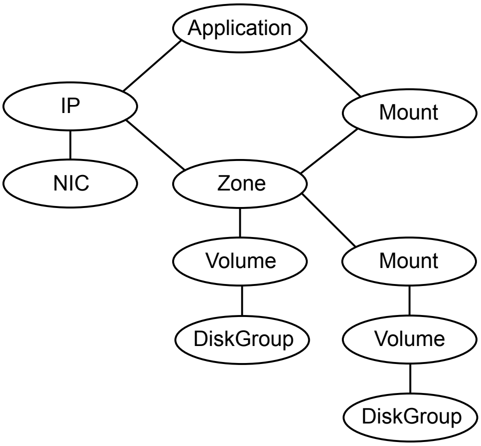
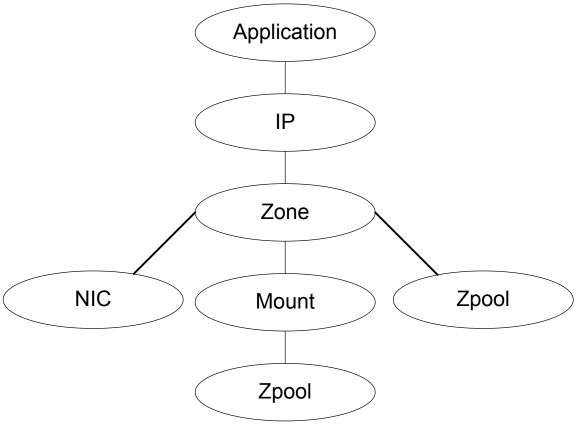


Figure 5-7 Sample service group that includes a Zone resource with the zone root on ZFS shared storage monitored by Zpool resource



Agent functions

Online

Brings a non-global zone up and running.

Offline

Takes a non-global zone down.

Monitor

Checks if the specified non-global zone is up and running.

If IMF is enabled for the Zone agent, the resource is monitored asynchronously and any change in the resource state is immediately sent to VCS for appropriate action.

imf_init

Initializes the agent to interface with the asynchronous monitoring framework (AMF) kernel driver. This function runs when the agent starts up.

imf_getnotification

Waits for notification about resource state changes. This function runs after the agent initializes with the AMF kernel driver. The agent continuously waits for notification and takes action on the resource upon notification.

imf_register

Registers the resource entities, which the agent must monitor, with the AMF kernel driver. This function runs for each resource after the resource goes into steady state (online or offline)

Clean

A more forceful method for halting a non-global zone.

Action	zoneconfigsync: This action entry point is used by the hazoneconfigsync utility. It synchronizes the XML file containing the configuration of the non-global zone configured in the Zone resource to all other nodes in the cluster. If the service group is a parallel service group or if the zone name is different on cluster nodes, this utility does not synchronize the configuration.

Attributes

Table 5-13 Optional attributes for Solaris

Optional attribute	Description
Pool	This is the resource pool name that is associated with the zone. Type and dimension: string-scalar
BootState	The value for the milestone service. Acceptable values follow: ■ single-user ■ multi-user ■ multi-user-server **Note:** Veritas recommends that you use the multi-user-server value for the BootState attribute. Type and dimension: string-scalar Default: multi-user
ShutdownGracePeriod	Specifies the interval in seconds from the Offline action to the execution of the shutdown within the zone. Type and dimension: integer-scalar Default: 0 Example: "10"
RunFsck	If the value of this attribute is 1, the Zone agent checks file system consistency for VxFS file systems. It uses the `fsck -y` command on all VxFS file systems that are defined in the zone's xml file. This file is located in /etc/zones. Adjust the Zone agent default OnlineTimeout value so that the Zone agent has sufficient time to run the `fsck` command before it brings the zone online. Type and dimension: boolean-scalar Default: 0 Example: 1

Table 5-13 Optional attributes for Solaris *(continued)*

Optional attribute	Description
DetachZonePath	If disabled, the Zone agent skips detaching the Zone root during zone resource offline and clean. DetachZonePath is enabled (1) by default. Type and dimension: boolean-scalar Default: 1 Example: 0
ForceAttach	If disabled, the Zone agent attaches the ZonePath without the -F option during zone resource online. ForceAttach is enabled (1) by default. On Solaris 11, the ForceAttach attribute must be set to 0 for zones created on shared storage. Type and dimension: boolean-scalar Default: 1 Example: 0
DeleteVCSZoneUser	If enabled on a non-secure cluster Zone agent deletes the VCS Zone user created for password-less communication between local-zone and global zone, during offline and clean entry points. DeleteVCSZoneUser is disabled by default. Type and dimension: boolean-scalar Default: 0
AttachOptions	Specifies the option that the agent uses while attaching the zone. In Solaris 11.3, if a zone is migrated from one global zone to another, the active Boot Environment (BE) in the zone is different from the one that was active before the zone failed over. Therefore, for a zone failover to work seamlessly, use `-x attach-last-booted-zbe -x deny-zbe-clone`. Type and dimension: string-scalar Example : `-x attach-last-booted-zbe -x deny-zbe-clone` **Note:** If you use the command line, add the % sign to escape '-'. Example: `hares -modify Zoneres AttachOptions "%-x attach-last-booted-zbe -x deny-zbe-clone"`

Configuring the Zone agent for DR in a Global Cluster environment

For information about configuring the Zone agent for DR in a Global Cluster environment, refer to the *Veritas InfoScale Virtualization Guide*.

Resource type definition

```
type Zone (
static str ArgList[] = { Pool, BootState,
ShutdownGracePeriod, RunFsck, DetachZonePath, ForceAttach,
DeleteVCSZoneUser, DROpts }
static keylist SupportedActions = { zoneconfigsync }
static boolean AEPTimeout = 1
static str IMFRegList[] = { BootState }
static str IMF{} = { Mode = 3, MonitorFreq = 5,
RegisterRetryLimit = 3 }
static int ContainerOpts{} = { RunInContainer=0, PassCInfo=1 }
str Pool
str BootState = multi-user
int ShutdownGracePeriod
boolean RunFsck = 0
boolean DetachZonePath = 1
boolean ForceAttach = 1
boolean DeleteVCSZoneUser = 0

)
```

Zone agent notes

The Zone agent has the following notes:

- See “Using the Zone agent with IMF” on page 233.

Using the Zone agent with IMF

If you use IMF for intelligent resource monitoring, review the following recommendations.

- Set the value of the MonitorFreq key to a high value to ensure that the agent does not run the traditional monitor function frequently.
- Monitor the health of the storage, on which the zone root is created, using one of the storage agents such as Mount, Zpool, or Volume. The Zone agent should have a dependency on the storage agent as depicted in Dependencies.

- The Zpool agent is not IMF aware. Hence, if the Zpool agent is used to monitor the zone root mount point, then no instantaneous notification will be received in case the zone root mount point is un-mounted externally. Still, traditional monitoring of Zpool will detect failure of the mount point.
- When you run the `svcadm` command to update the state of a service, outside of VCS control, IMF notifies the Zone agent. If IMF falsely notifies about an update to the state of the service group, the state of the resource does not change. Thereafter, the Zone agent reregisters events with IMF.

  Note: With the zone agent registered to IMF for Online or Offline monitoring, if the IMF notification to the Zone agent is false, agent framework schedules the zone monitor. Zone monitor verifies and confirms that there is no change in the state of the resource. Thereafter, Zone agent reregisters the events with IMF.

- Zone agent registered to IMF for Directory Online event
  The Directory Online event monitors the Zone root directory. If the parent directory of the Zone root directory is deleted or moved to another location, AMF does not provide notification to the Zone agent. In the next cycle of the zone monitor, it detects the change and reports the state of the resource as offline.

Debug log levels

The Zone agent uses the following debug log levels:

DBG_1, DBG_2, DBG_3, DBG_4, DBG_5

LDom agent

The LDom agent brings the logical domain (LDom) online, takes them offline, and monitors them in Oracle VM server for SPARC environment. You can use this agent to monitor LDoms and to make them highly available.

The LDom agent also performs VCS-initiated live migration of the logical domain from one cluster node to another cluster node. If the logical domain is provided with storage and network services from more than one I/O domain, VCS supports LDoms to remain online and functional even when the primary domain is rebooted or shut down for planned maintenance. The LDom agent supports dynamic selection of target node for a virtual machine service group (AdaptiveHA).

For more information about AdaptiveHA, refer to the *Cluster Server Administrator's Guide*.

For detailed information about support for logical domains, refer to the *Veritas InfoScale Virtualization Guide*.

Configuring primary and logical domain dependencies and failure policy

For all the logical domains that are configured in the cluster, the agent performs the following commands to set:

- The dependency between the primary and logical domains.

  ```
  # ldm set-domain master=primary guestldom
  ```

- The failure-policy of the primary domain to stop.

  ```
  # ldm set-domain failure-policy=stop primary
  ```

You can change this behavior by using the DomainFailurePolicy attribute.

See “Notes for the DomainFailurePolicy attribute” on page 248. for more information.

IMF awareness

The LDom agent is Intelligent Monitoring Framework (IMF)-aware and uses Asynchronous Monitoring Framework (AMF) kernel driver for IMF notification. For more information about IMF and intelligent resource monitoring, refer to the *Cluster Server Administrator’s Guide*.

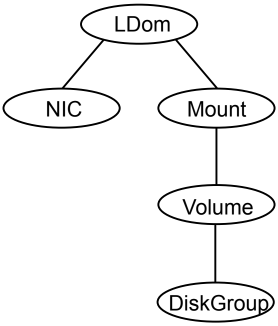
For more information about IMF-related functions, see Agent functions.

For more information about using the LDom agent with IMF, see Using the LDom agent with IMF.

Dependencies

The LDom resource depends on the NIC resource. It also depends on a storage resource, such as Mount, Volume, DiskGroup, Zpool, or Disk.

Figure 5-8 Sample service group for an LDom resource that monitors an image file



Network resources

Use the NIC agent to monitor the network adapter for the LDom, whether it is virtual or physical.

Storage resources

- Veritas Volume Manager (VxVM) exposed volumes
  Use the Volume and DiskGroup agents to monitor a VxVM volume.
- ZFS volumes
  Use the Zpool agent to monitor a ZFS volume.
- Disk or LUN
  Use Disk agent to monitor Disk or LUN
- Image file
  - Image file in a volume that is managed by Veritas Volume Manager (VxVM)
    Use the Mount, Volume, and DiskGroup agents to monitor the file system containing the image file.
  - Image file in an NFS share
    Use the Mount agent to monitor the file system containing the image file.
  - Image file in a ZFS volume
    Use the Mount and Zpool agents to monitor the file system containing the image file.
  - Image file in a partition of a physical disk
    Use the Mount and Disk agents to monitor the file system containing the image file.

Agent functions

Online

Starts the LDom.

Offline

Stops the LDom.

Monitor

Monitors the status of the LDom.

If IMF is enabled for the LDom agent, the resource is monitored asynchronously and any change in the LDom state is immediately sent to the LDom agent for appropriate action.

Clean

Stops the LDom forcefully.

Migrate

Migrates logical domains from a node to the target cluster node.

Action

vmmigrate: Checks whether an LDom resource is able to migrate to another node using the `hagrp -migrate` command.

Command usage:

```
# hares -action ldmres vmmigrate -actionargs
target_system_name -sys source_system_name
```

Meter

Measures the VCPU and Memory requirement of the LDom based on the keys in the Meters attribute. If the NumCPU and Memory attributes are configured, the function reads the VCPU and Memory from these attributes respectively. If these attributes are not configured, the function reads the VCPU and Memory from the LDom configuration available on the system. If the LDom configuration is not available on the system, the function reads the VCPU and Memory from the LDom configuration file configured in the CfgFile attribute.

imf_init

Initializes the agent to interface with IMF. The function runs when the agent starts up.

imf_register

Registers the resource entities, which the agent must monitor using IMF. The function runs for each resource after the resource goes into a steady state, either online or offline.

Currently, IMF awareness for LDom is been added for following events:

- start domain
- stop domain
- stop guest
- migration domain

imf_getnotification

Gets notification about resource state changes. This function runs after the agent initializes with the AMF kernel driver. The agent continuously waits for notification and takes action on the resource upon notification.

attr_changed

When attributes such as Memory, NumCPU, or DomainFailurePolicy is changed, this function dynamically updates the Memory, CPU, or the master domain of the LDom together with the failure policy of the master domain respectively. It also reports an error if the value of the OnlineStableTime attribute is set to more than the value of the LDom MonitorInterval attribute.

State definitions

ONLINE

Indicates that the LDom is up and running.

OFFLINE

Indicates that the LDom is down.

FAULTED

Indicates that the LDom is down when the VCS engine expects it to be up and running.

If the MonitorCPU attribute is set to true (1), CPU usage of either 0% or 100% is interpreted as a fault.

UNKNOWN

Indicates the agent cannot determine the LDom's state. A configuration problem likely exists in the resource or the LDom.

Attributes

Table 5-14 summarizes the required attributes for the LDom agent.

Table 5-14 Required attributes for Solaris

Required attribute	Description
LDomName	The name of the LDom that you want to monitor. Type-dimension: string-scalar Default: n/a Example: "ldom1"

Table 5-15 summarizes the optional attributes for the LDom agent.

Table 5-15 Optional attributes for Solaris

Optional attribute	Description
CfgFile	The absolute path of the XML file that contains the LDom configuration. To create the configuration file for an LDom, run the following command: `$ ldm list-constraints -x ldom_name > ldom_name.xml` If this attribute is configured, LDom agent uses this file to create an LDom only when LDom configuration is not found on a node. Make sure that default services like virtual disk server (vds), virtual switch service (vsw), and virtual console concentrator service (vcc) are already created on all the nodes to use the xml file. If domain migration is planned for the LDom, then this attribute must be configured. Refer to the *ldm(1M) manual page* for information on this file. The configuration file must be present locally on all of the systems or on a shared disk where it is accessible by all of the systems. Type-dimension: string-scalar Default: n/a Example: "/root/ldom-cfg/ldom1.xml"
MonitorCPU	Specifies whether the LDom agent monitors the CPU usage of the LDom. If the CPU usage of all of the VCPUs attached to the LDom is equal to either 0% or 100%, then the resource is declared FAULTED. For an LDom with one VCPU, set this attribute to 0. This setting is to work around an LDom limitation where an LDom with one VCPU always reports 100% CPU usage. Type-dimension: boolean-scalar Default: 0

Table 5-15 Optional attributes for Solaris *(continued)*

Optional attribute	Description
NumCPU	The number of virtual CPUs that you want to attach to the LDom when it is online. If you set this attribute to a positive value, the agent detaches all of the virtual CPUs when the service group goes offline. Do not reset this value to 0 after setting it to 1. If the agent's monitor entry point finds that the CPUs available to the LDom is less than the value of the NumCPU attribute, it logs a warning about the mismatch. It also sets ConfidenceMsg and ConfidenceLevel accordingly. This attribute can be modified while LDom is online to dynamically reconfigure virtual CPUs assigned to the LDom. Type-dimension: integer-scalar Default: 0 Example: 16
Memory	The amount of memory to be assigned to an LDom. The expected value of this attribute is a positive integer followed by the unit G, M, or K, where G stands for gigabytes, M stands for megabytes, and K stands for kilobytes. The memory specified here is governed by the --auto-adj option of the ldm set-memory command which enforces it to 256 megabyte alignment. The success of the ldm set-memory command depends on the availability of the memory pool that is free. If the agent's monitor entry point finds that memory available to the LDom is less than the value of the Memory attribute, it logs a warning about the mismatch. It also sets ConfidenceMsg and ConfidenceLevel accordingly. This attribute can be modified while LDom is online to dynamically reconfigure memory assigned to the LDom. Type-dimension: string-scalar Default: Null Example 1: 4G Example 2: 2048M

Table 5-15 Optional attributes for Solaris *(continued)*

Optional attribute	Description
ConfigureNetwork	Specifies if the LDom agent configures the network-boot-arguments PROM variable of the logical domain. If the value of the attribute is set to 1, then the LDom agent updates the network-boot-arguments PROM variable using the following attributes: IPAddress, Netmask, Gateway, and DNS. Type and dimension: integer-scalar Default: ""
IPAddress	During the online operation, the agent sets the value of this attribute to the host-ip key of the network-boot-arguments PROM variable of the LDom. Type and dimension: string-scalar Default: ""
Netmask	During the online operation, the agent sets the value of this attribute to the subnet-mask key of the network-boot-arguments PROM variable of the LDom. Type and dimension: string-scalar Default: ""
Gateway	During the online operation, the agent sets the value of this attribute to the router-ip key of the network-boot-arguments PROM variable of the LDom. Type and dimension: string-scalar Default: ""
DNS	During the online operation, the agent sets the value of this attribute to the name-servers key of the network-boot-arguments PROM variable of the LDom. Type and dimension: string-scalar Default: ""

Table 5-15 Optional attributes for Solaris *(continued)*

Optional attribute	Description
RemoveLDomConfigForMigration	If enabled, the agent removes the LDom configuration from the system during offline and clean operations provided CfgFile attribute is configured. You need to enable this attribute if domain migration is planned for the logical domain. This is because domain migration cannot be performed if logical domain configuration is present on the target node for migration. Type and dimension: boolean-scalar Default: 0 Example: 1

Table 5-15 Optional attributes for Solaris *(continued)*

Optional attribute	Description
DomainFailurePolicy	Specifies the list of master domains and the failure policies set for master domains that govern the behavior of VCS-managed logical domains. The key of the attribute is the name of the master domain and the value for the key of the attribute is the failure policy enacted by the master domain on the logical domain. The failure policy is enacted by the master domain in the event of loss of the master domain. The possible values for the failure policy are ignore, stop, reset, and panic. For more information on the failure policies of the master domain, refer to the *ldm(1M) manual page*. By default, the DomainFailurePolicy attribute is set with master domain as ‘primary’ and failure policy as ‘stop’. The DomainFailurePolicy attribute need to be set to a non-default value only when the logical domain is provided with storage and network services from more than one I/O domain and the logical domain need to be made available even when the primary domain is rebooted or shut down for maintenance. It is highly recommended to set the same failure policy for a master domain for all the LDom resources (logical domains) configured on a physical system. If the attribute is set to different failure policies for the same master domain for different LDom resources, then the agent ensures that the failure policy with the highest priority is set on the master domain. For more information on failure policy priority, see Notes for the DomainFailurePolicy attribute. You can change the attribute value dynamically when the LDom resource is online. The LDom agent ensures that the changed failure policy is dynamically set on the master domain in the order of priority, without the need to take down the LDom resource. For more information on the behavior of this attribute, see the Notes for the DomainFailurePolicy attribute section.

Table 5-15 Optional attributes for Solaris *(continued)*

Optional attribute	Description
DomainFailurePolicy (Continued)	Type-dimension: string-association Default: {primary = “stop”} Example: {primary= ignore, secondary = “stop”}. In this example, primary and secondary are the master domains from where storage and network services are exported to the logical domain.
UserName	Name of the user authorized to perform migration of the logical domain on to the host from another host. Type and dimension: string-scalar Default: root Example: lmuser

Password	Specifies encrypted password for the user configured using the UserName attribute. Use one of the following commands to encrypt the password:
	`vcsencrypt -agent`
	Or
	`vcsencrypt -agent -secinfo`
	Type and dimension: string-scalar
	Default: None
	Example: password

MigrateTimeout	The maximum time for the migrate entry point to complete migration of the logical domain after which it is terminated. The value is in seconds. Type and dimension: integer-scalar Default: 600 Example: 900
MigrateWaitLimit	Number of monitor intervals to wait after completing the migrate procedure and before the resource (logical domain) migrates. Type and dimension: integer-scalar Default: 2 Example: 4

Table 5-15 Optional attributes for Solaris *(continued)*

Optional attribute	Description
RereqPGR	If the RereqPGR attribute is set to 1 when you live migrate the logical domain using the `hagrp -migrate` command, the LDom agent runs the `vxdmpadm pgrrereg` command inside the logical domain. This command re-registers the Persistent Group Reservation (PGR) keys after the migration is complete. To execute the `vxdmpadm` command, you must enable password-less SSH communication for the root user between the control domain and the logical domain. You must set the RereqPGR attribute to 1 only when all the following parameters are met: ■ There are clusters at two levels. At one level, there is a cluster between logical domains; and at the other level, there is a cluster between the control domains on the associated physical servers. ■ The cluster between the logical domains is configured with disk-based IO fencing. ■ The cluster between the control domains manages the logical domains with the LDom resource. ■ For logical domains forming the cluster, the logical domain migration is done using the `hagrp -migrate` command. Type and dimension: boolean-scalar Default: 0 Example: 0

Meter	This attribute defines the meters based on which the failover decision is taken for the service group containing the LDom resource. The keys of this attribute must be subset of intersection of HostMeters (cluster attribute) and AvailableMeters (type level). You cannot override this attribute at the resource level and you cannot modify it at run time.
	Type and dimension: string-association
	Default: { SCPU, SMem }
	Example: { SCPU }

Table 5-15 Optional attributes for Solaris *(continued)*

Optional attribute	Description
MeterControl	At every ForecastCycle, an extra flag is passed to the meter entry point to perform the forecasting. This flag is currently not used by the LDom agent. Type and dimension: string-association Default: { MeterInterval=600, ForecastCycle=0 }
MeterTimeout	The maximum time for the meter entry point to complete. The value is in seconds. Type and dimension: integer-scalar Default: 300 Example: 900
OnlineStableTime	The expected boot time of the LDom guest in seconds. This attribute is only applicable when the resource is under IMF monitoring. If the LDom agent is under IMF monitoring and if it receives an user-initiated start domain event, the agent waits for the number of seconds mentioned in the OnlineStableTime attribute and then schedules the monitor cycle. Type and dimension: integer-scalar Default: 20

Table 5-16 Internal attributes

Optional attribute	Description
IntentionalOffline	This attribute is required for performing domain migration. The value of this attribute must be set to 1 when performing domain migration. For information on how to use the IntentionalOffline resource type attribute, refer to the *Cluster Server Administrator's Guide.* Default: 1

Table 5-16 Internal attributes *(continued)*

Optional attribute	Description
ResyncVMCfg	The ResyncVMCfg attribute is set by the havmconfigsync utility. If this attribute is set, the agent redefines the virtual machine configuration if it already exists using the CFgFile attribute. **Note:** You must not set the ResyncVMCfg attribute manually.
AvailableMeters	This attribute defines the meters that the agent supports. You cannot override this attribute at the resource level and you should not edit this attribute. Type and dimension: string-association Default: { SCPU="", SMem="" }

Resource type definition

```
type LDom (
    static keylist RegList = { NumCPU, Memory, DomainFailurePolicy,
     OnlineStableTime }
    static str MeterRegList[] = { LDomName, CfgFile, NumCPU, Memory }
    static str AvailableMeters{} = { SCPU="", SMem="" }
    static str IMFRegList[] = { LDomName }
    static keylist Meters = { SCPU, SMem }
    static boolean IntentionalOffline = 1
    static int IMF{} = { Mode=3, MonitorFreq=1, RegisterRetryLimit=5 }
    static int MigrateTimeout = 600
    static int MeterTimeout = 300
    static int MigrateWaitLimit = 2
    static int MeterRetryLimit = 10
    static keylist SupportedOperations = { migrate, meter }
    static keylist SupportedActions = { "vmconfigsync", "vmmigrate" }
    static int MeterControl{} = { MeterInterval=600, ForecastCycle=0 }
    static str ArgList[] = { State, IState, LDomName, CfgFile, MonitorCPU,
     NumCPU, ConfigureNetwork, IPAddress, Netmask, Gateway, DNS, Memory,
     CEInfo, RemoveLDomConfigForMigration, ResyncVMCfg, DomainFailurePolicy,
     Password, UserName, ReregPGR, OnlineStableTime }
    str CEInfo{} = { Enabled=0, CESystem=NONE, FaultOnHBLoss=1 }
    str LDomName
    int ConfigureNetwork
    str IPAddress
    str Netmask
    str Gateway
    str DNS
    str CfgFile
    int NumCPU
    str Memory
    str DomainFailurePolicy{} = { primary="stop" }
    boolean MonitorCPU = 0
    static boolean AEPTimeout = 1
    boolean RemoveLDomConfigForMigration = 0
    boolean ResyncVMCfg = 0
    boolean ReregPGR = 0
    str Password
    str UserName = root
    int OnlineStableTime = 20
)
```

LDom agent notes

The LDom agent has the following notes:

- About the auto-boot? variable
- Notes for the DomainFailurePolicy attribute
- Using VCS to migrate a logical domain
- Configuring primary and logical domain dependencies and failure policy
- Using the LDom agent with IMF

About the auto-boot? variable

Caution: Do not change the value of the auto-boot? variable to true.

Veritas recommends that you do not change the value of the auto-boot? variable to true. Changing the value to true can lead to data corruption in a scenario where the logical domain shares a disk in a cluster and the domain starts on more than one node.

Notes for the DomainFailurePolicy attribute

When the DomainFailurePolicy attribute is set, the LDom agent sets the master domain of the logical domain with the key of the attribute and the value of the key as the failure policy of the master domain.

The LDom agent uses the following command to set the master for the logical domain:

```
# ldm set-domain master=master-domain guestldom
```

The LDom agent uses the following command to set the failure policy for the master domain:

```
# ldm set-domain failure-policy=failure-policy master-domain
```

As the DomainFailurePolicy attribute is available at the resource level, you can set the failure policy of the master domain to different values for different LDom resource knowingly or unknowingly. However, at any given point, the LDom agent can set only one failure policy for the master domain. In a cluster with multiple LDom resources, different values for the failure policy of the master domain can create a conflict. To avoid such a conflict, the LDom agent uses internal priority while setting the failure policy of the master domain.

The internal priority is as follows:

- panic: highest
- reset: high
- stop: low
- ignore: lowest

If the failure policy of the master domain is set to a lower priority on the system than the one set in the LDom resource for the DomainFailurePolicy attribute, then the failure policy of the master domain is changed to the value in the attribute.

If the failure policy of the master domain is set to a higher priority on the system than the one set in the LDom resource for the DomainFailurePolicy attribute, then the failure policy of the master domain will not be changed to the value in the attribute. The LDom agent logs a message to indicate the conflict for the first time.

If the failure policy of the master domain is set to ignore, then the LDom agent does not add the master domain to the master list of the logical domain. If the master domain is already part of the masters list of the logical domain, the LDom agent removes the master domain from the masters list.

Note: Veritas does not recommend to set the failure policy of any of the master domains to panic.

Example 1

Failure policy of the master domain (primary) is set to ignore on the system and the DomainFailurePolicy attribute for the LDom resource is changed to { primary =

“stop” }. To check whether the failure policy of the master domain (primary) is set to ignore on the system, enter the following command:

```
# ldm list-bindings primary | grep failure-policy
```

In this example, as the internal priority of the LDom agent is assigned to stop and as it is higher than ignore, the failure policy of the primary domain is changed to stop. The LDom agent uses the following command to change the failure policy of the primary domain to stop:

```
# ldm set-domain failure-policy=stop primary
```

Example 2

Failure policy of the master domain (primary) is set to panic on the system and the DomainFailurePolicy attribute for the LDom resource is changed to { primary = “stop” }. To check whether the failure policy of the master domain (primary) is set to panic on the system, enter the following command:

```
# ldm list-bindings primary | grep failure-policy
```

In this example, as the internal priority of the LDom agent is assigned to stop and as it is lower than panic, the failure policy of the primary domain is retained as panic.

If the failure policy of a master domain need to be set to a value of lower priority than the value currently set on the system, you must manually execute the `ldm` command. The LDom agent uses the following command to change the failure policy of the primary domain to stop from reset or panic:

```
# ldm set-domain failure-policy=stop primary
```

Example 3

If the value of the failure policy of the master domain is specified as ignore in the DomainFailurePolicy attribute, then the master domain is excluded from the masters list of the logical domain by the LDom agent.

If the masters list of a logical domain contains primary and secondary and if the DomainFailurePolicy attribute of the LDom resource for the logical domain is changed to {primary = ignore, secondary = “stop” }, then the primary domain is removed from the masters list of the logical domain.

Before you change the DomainFailurePolicy attribute, you can enter the following command to check whether the masters list of a logical domain contains primary and secondary:

```
# ldm list-bindings guestldom | grep master
```

The following output shows that the logical domain contains both primary and secondary:

```
master=primary, secondary
```

After you change the DomainFailurePolicy attribute, you can enter the following command to check whether the primary domain is removed from the masters list of the logical domain.

```
# ldm list-bindings guestldom | grep master
```

The following output shows that the primary domain is removed from the masters list:

```
master= secondary
```

For use case scenarios and for VCS configuration where the DomainFailurePolicy attribute must be set, refer to the *Veritas InfoScale Virtualization Guide*.

Using VCS to migrate a logical domain

You can migrate a logical domain using the `hagrp –migrate` command.

For example:

```
#hagrp –migrate service_group_name -to target_system_name
```

For more information, refer to the *Veritas InfoScale Virtualization Guide* and *Cluster Server Administrator’s Guide*.

Configuring the LDom agent for DR in a Global Cluster environment

For information about configuring the LDom agent for DR in a Global Cluster environment, refer to the *Veritas InfoScale Virtualization Guide* and *Veritas InfoScale Disaster Recovery Implementation Guide*.

Using the LDom agent with IMF

If you use IMF for intelligent resource monitoring, you can either set the MonitorFreq key to 0 or a higher value. Setting the MonitorFreq key to a higher value ensures that the agent does not run the monitor function frequently. Setting the MonitorFreq key to 0 disables the traditional monitoring while IMF monitoring is in progress. Traditional monitoring is done after receiving the notification for a resource. However, if the LDom is configured with reservation and value of the MonitorReservation attribute is set to 1, then set the MonitorFreq key value to the frequency at which you want the agent to run the monitor function, to verify the reservation on the LDom.

Sample configuration 1

The following is a minimal LDom configuration:

```
LDom ldom1 (
     LDomName = "ldom1"
)
```

Sample configuration 2

```
group ldom_sg (
        SystemList = { sys1 = 0, sys2 = 1 }
        )

        LDom ldmres (
                LDomName = ldg1
                )

        DiskGroup dgres (
                DiskGroup  = dg1
                )

        Volume volres (
                Volume  = vol1
                DiskGroup  = dg1
                )
        NIC nicres (
                Device  = nxge3
                )
 ldmres requires volres
 ldmres requires nicres
 volres requires dgres
```

Configuration to support user-initiated LDom migration

```
LDom ldom1 (
                LDomName = guest
                CfgFile = "/guest.xml"
                RemoveLDomConfigForMigration=1
                )
```

Configuration for VCS-initiated migration

```
group sg1 (
     SystemList = { sys1 = 0, sys2 = 1 }
     )
     LDom ldmres1 (
     LDomName = ldm1
     CfgFile = "/etc/ldm1.xml"
     NumCPU = 16
     Memory = 8G
     Password = aabbccddee
     UserName = user1
     RemoveLDomConfigForMigration=1

)
```

Sample configuration (Dynamic virtual machine service group failover)

```
cluster vcs (
    HostMeters = {CPU, Mem, Swap, SCPU, SMem}
    MeterWeight = { CPU = 10, Mem = 5, Swap = 1, SCPU = 10, SMem = 10}
    )

group ldom-sg (
        SystemList = { Sys1 = 0, Sys2 = 1 }
 )

        LDom ldmres (
                LDomName = ldm1
                CfgFile = "/ldm1.xml"
                NumCPU=8
                Memory=8G
         )
```

Debug log levels

The LDom agent uses the following debug log levels:

DBG_1, DBG_2, DBG_3, DBG_4, DBG_5

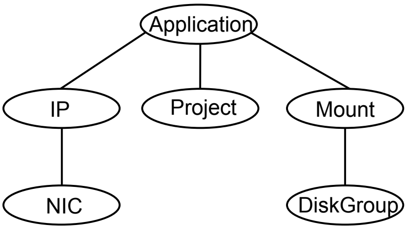
Project agent

The Project agent adds, deletes, and monitors Solaris projects. You can use the agent to make projects highly available or to monitor them.

Dependencies

Figure 5-9 shows a sample service group that includes a Project resource.

Figure 5-9 Sample service group that includes a Project resource



Agent functions

Online
Creates a Solaris project, if one is not present. Modifies a Solaris project, if one present.

Offline
Deletes a Solaris project if the value of the OfflineDelProject attribute is 1. If the value of the OfflineDelProject attribute is 0, then the Solaris project is not deleted from the /etc/project file.

Monitor
Checks if the specified project exists in /etc/project file.

Clean
Deletes a Solaris project if the value of the OfflineDelProject attribute is 1. If the value of the OfflineDelProject attribute is 0, then the Solaris project is not deleted from the /etc/project file.

Attributes

Table 5-17 Required attributes

Required attribute	Description
OfflineDelProject	Use the OfflineDelProject attribute to tell the Project agent to remove a project entry from the /etc/project file when the agent invokes the offline or clean agent functions. The OfflineDelProject's default value is 1, which instructs the Project agent to remove the project's entry from the /etc/project file. Set the value of the OfflineDelProject's attribute to 0, if you do not want the project to be deleted from the /etc/project file. Default: 1

Table 5-18 Optional attributes

Optional attribute	Description
User	Comma separated list of existing users that are part of the project. Type and dimension: string-scalar

Resource type definition

```
type Project (
static str ArgList[] = { User, OfflineDelProject }
static int ContainerOpts{} = { RunInContainer=0, PassCInfo=1 }
static int FaultPropagation = 0
str User
boolean OfflineDelProject = 1
)
```

Sample configuration

```
include "types.cf"
include "OracleTypes.cf"
cluster vcs (
)
group grp_xrm (
SystemList = { sysa = 0 }
ContainerInfo @sysa = { Name = ora_project, Type = XRM,
Enabled = 1 }
)
Oracle ora_res (
Sid = oradb
Owner = oracle
Home = "/oraHome/app"
)
Project proj_res (
User = oracle
OfflineDelProject = 0
)
ora_res requires proj_res
// resource dependency tree
//
// group grp_xrm
// {
```

```
// Oracle ora_res
// {
// Project proj_res
// }
// }
```

Debug log levels

The Project agent uses the following debug log levels:

DBG_1

AlternateIO agent

The AlternateIO agent monitors VCS storage and network service groups that in turn monitor redundant I/O services exported from the control domain and alternate I/O domain to a guest logical domain.

The AlternateIO agent functions when:

- Logical domain is managed by in Oracle VM Server for SPARC environment.
- Alternate I/O domains are configured to provide redundant storage and network services to the guest Logical Domain

The AlternateIO agent provides consolidated status of storage and network connectivity to a guest logical domain from multiple I/O domains.

For detailed information on support for logical domains with alternate I/O domains, refer to the *Veritas InfoScale Virtualization Guide*.

Dependencies for AlternateIO agent

AlternateIO resource does not directly depend on any other VCS resources. However, the state of AlternateIO resource depends upon the state of storage or network resources that are part of the service groups monitored bythe AlternateIO resource.

Figure 5-10 shows a sample service group configuration with an AlternateIO service group

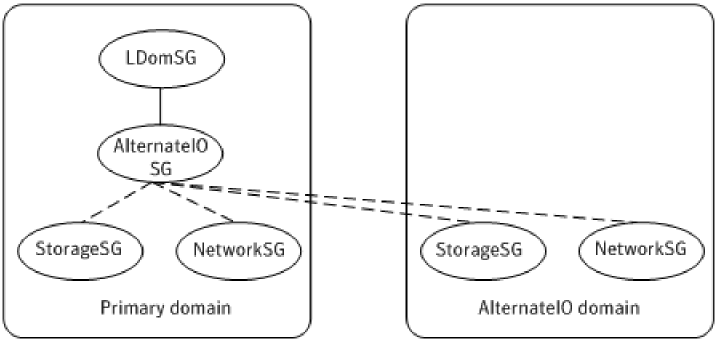
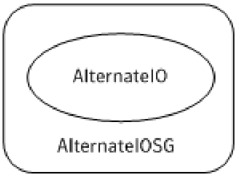


Figure 5-11 shows an AlternateIO service group



Agent functions for AlternateIO agent

Online — Brings the storage and network service groups online if value in the key-value pair is set to 1.

Offline — Takes the storage and network service groups offline if value in the key-value pair is set to 1.

Monitor — Monitors the state of storage and network service groups. The resource shows online if all the storage and network service groups are online atleast on one of the domains - primary domain or alternate I/O domain.

The resource shows offline even if one of the storage or network service group is offline on the primary domain and the alternate I/O domain.

Clean — No function performed.

State definitions for AlternateIO agent

ONLINE — Reports an ONLINE state when all the storage service groups and network service groups are online atleast on one of the domains - on either the primary or alternate I/O domain.

OFFLINE	Reports an OFFLINE state even if one of the storage service group or network service group is offline on the primary domain and the alternate I/O domain.
UNKNOWN	Reports an UNKNOWN state for the service groups given in StorageSG and NetworkSG attributes in the following conditions: ■ No service group is defined ■ No service group has the SystemList attribute defined ■ A service group is frozen temporarily or persistently

Attributes for AlternateIO agent

Table 5-19 Required Attributes

Required Attributes	Description
StorageSG	A key-value pair with service groups that cater to the storage requirements of the guest domain managed by VCS as keys and option to online and offline these service groups as values. Set the value to 1 to online the service groups during online operation and offline the service group during offline operation. If the service group contains resources like zpool that need to be onlined during online operation of AlternateIO resource, and offlined during online operation of AlternateIO resource set the value to 1. If the service group contains resources like disks or CVM shared disk group that need not be to be onlined during online operation of AlternateIO resource or offlined during online operation of AlternateIO resource, set the value to 0. Type and dimension: string-vector Default: n/a Example: StorageSG = { sg_storage_ldom1 = 1 }
NetworkSG	A key-value pair with service groups that cater to the network requirements of the guest domain managed by VCS as keys and option to online or offline these service groups as values. Set the value to 1 to online the service groups during online operation and offline the service group during offline operation. In general, for service group containing resources to monitor network resources set value to zero. Type and dimension: string-vector Default: n/a Example: NetworkSG = { sg_network_ldom1 = 0 }

Resource type definition for AlternateIO agent

```
type AlternateIO (
        static str AgentFile = "bin/Script51Agent"
        static str ArgList[] = { StorageSG, NetworkSG  }
        str StorageSG{}
        str NetworkSG{}

)
```

Sample configurations for AlternateIO agent

A main.cf example that shows the AlternateIO agent configuration.

```
include "types.cf"
cluster aioclus (
 UserNames = { admin = xxxxxxxxxxx }
 Administrators = { admin }
 HacliUserLevel = COMMANDROOT
 )
system primary1 (
 )
system alternate1 (
 )
system primary2  (
 )
system alternate2 (
 )
group aiosg (
 SystemList = { primary1 = 0, primary2  = 1 }
 Parallel = 1
 )
 AlternateIO aiores1 (
  StorageSG @primary1 = { primary1-strsg = 0 }
  StorageSG @primary2 = { primary2-strsg = 0}
  NetworkSG @primary1 = { primary1-nwsg = 0}
  NetworkSG @primary2 = { primary2-nwsg = 0}
  )

 // resource dependency tree
 //
 // group aiosg
 // {
 // AlternateIO aiores1
 // }

group ldmsg (
 SystemList = { primary1 = 0, primary2  = 1 }
 )

 LDom ldmres (
  LDomName = ldg1
  )

 requires group aiosg online local hard
 // resource dependency tree
 //
 // group ldmsg
 // {
 // LDom ldmres
 // }


group primary1-nwsg (
 SystemList = { primary1 = 0, alternate1 = 1 }
 Parallel = 1
 AutoStartList = { primary1, alternate1 }
 )
 NIC nicres (
  Device @primary1 = nxge3
  Device @alternate1 = nxge1
  )
 Phantom ph1 (
  )
 // resource dependency tree
 //
 // group primary1-nwsg
 // {
 // Phantom ph1
 //     {
 //     NIC nicres
 //     }
 // }
```

```
group primary1-strsg (
 SystemList = { primary1 = 0, alternate1 = 1 }
 Parallel = 1
 AutoStartList = { primary1, alternate1 }
 )
 Disk disk1 (
  Partition @primary1 = "/dev/rdsk/c3t50060E8000C46C50d2s2"
  Partition @alternate1 = "/dev/rdsk/c1t50060E8000C46C50d2s2"
  )

 Phantom ph2 (
  )
 // resource dependency tree
 //
 // group primary1-strsg
 // {
 // Phantom ph2
 //     {
 //     Disk disk1
 //     }
 // }
group primary2-nwsg (
 SystemList = { primary2  = 0, alternate2 = 1 }
 Parallel = 1
 AutoStartList = { primary2, alternate2 }
 )
 NIC nicres1 (
  Device @primary2  = nxge3
  Device @alternate2 = nxge1
  )
 Phantom ph3 (
  )
 // resource dependency tree
 //
 // group primary2-nwsg
 // {
 // Phantom ph3
 //     {
 //     NIC nicres1
 //     }
 // }
group primary2-strsg (
 SystemList = { primary2  = 0, alternate2 = 1 }
 Parallel = 1
 AutoStartList = { primary2, alternate2 }
 )
 Disk disk2 (
Partition @primary2  = "/dev/rdsk/c3t50060E8000C46C50d2s2"
  Partition @alternate2 = "/dev/rdsk/c1t50060E8000C46C50d2s2"
  )
 Phantom ph4 (
  )
 // resource dependency tree
 //
 // group primary2-strsg
 // {
 // Phantom ph4
 //     {
 //     Disk disk2
 //     }
 // }
```

Debug log levels for AlternateIO agent

The AlternateIO agent uses following debug log levels: DBG_1, and DBG_3

Chapter 6

Infrastructure and support agents

This chapter includes the following topics:

About the infrastructure and support agents

Use the infrastructure and support agents to monitor components and VCS objects.

NotifierMngr agent

Starts, stops, and monitors a notifier process, making it highly available. The notifier process manages the reception of messages from VCS and the delivery of those messages to SNMP consoles and SMTP servers. However, in dual stack mode, the NotifierMngr agent can communicate with the SNMP server and SMTP server only if the servers have both IPv4 and IPv6 IPs enabled on it. This ensures that the clients having any type of IP, that is, pure IPv4, pure IPv6, or both can easily communicate with the servers.

Refer to the *Admin Guide* for a description of types of events that generate notification. See the `notifier`(1) manual page to configure notification from the command line.

You cannot dynamically change the attributes of the NotifierMngr agent using the `hares -modify` command. Changes made using this command are only effective after restarting the notifier.

Dependency

The NotifierMngr resource can depend on the NIC resource.

Agent functions

Online
Starts the notifier process with its required arguments.

Offline
VCS sends a `SIGABORT`. If the process does not exit within one second, VCS sends a `SIGKILL`.

Monitor
Monitors the notifier process.

Clean
Sends `SIGKILL`.

State definitions

ONLINE
Indicates that the Notifier process is running.

OFFLINE
Indicates that the Notifier process is not running.

UNKNOWN
Indicates that the user did not specify the required attribute for the resource.

Attributes

Table 6-1 Required attributes for Solaris

Required attribute	Description
SnmpConsoles	Specifies the machine names of the SNMP managers and the severity level of the messages to be delivered. The severity levels of messages are Information, Warning, Error, and SevereError. Specifying a given severity level for messages generates delivery of all messages of equal or higher severity. **Note:** SnmpConsoles is a required attribute if SmtpServer is not specified; otherwise, SnmpConsoles is an optional attribute. Specify both SnmpConsoles and SmtpServer if desired. Type and dimension: string-association Example: "172.29.10.89" = Error, "172.29.10.56" = Information
SmtpServer	Specifies the machine name of the SMTP server. **Note:** SmtpServer is a required attribute if SnmpConsoles is not specified; otherwise, SmtpServer is an optional attribute. You can specify both SmtpServer and SnmpConsoles if desired. Type and dimension: string-scalar Example: "smtp.example.com"

Table 6-2 Optional attributes for Solaris

Optional attribute	Description
EngineListeningPort	Change this attribute if the VCS engine is listening on a port other than its default port. Type and dimension: integer-scalar Default: 14141
MessagesQueue	Size of the VCS engine’s message queue. Minimum value is 30. Type and dimension: integer-scalar Default: 30

Table 6-2 Optional attributes for Solaris *(continued)*

Optional attribute	Description
NotifierListeningPort	Any valid, unused TCP/IP port number. Type and dimension: integer-scalar Default: 14144
NotifierSourceIP	If this attribute is populated, all the notifications sent from the notifier (SMTP and SNMP) will be sent from the interface having this IP address. **Note:** Make sure that the Source IP given in this attribute is present in the /etc/hosts file or is DNS-resolvable. Type and dimension: string-scalar Example: "10.209.77.111"
SmtpFromPath	Set to a valid email address, if you want the notifier to use a custom email address in the FROM: field. Type and dimension: string-scalar Example: "usera@example.com"

SmtpRecipients	Specifies the email address where SMTP sends information and the severity level of the messages. The severity levels of messages are Information, Warning, Error, and SevereError. Specifying a given severity level for messages indicates that all messages of equal or higher severity are received.
	Note: SmtpRecipients is a required attribute if you specify SmtpServer.
	Type and dimension: string-association
	Example:
	`"james@example.com" = SevereError,`
	`"admin@example.com" = Warning`

SmtpReturnPath	Set to a valid email address, if you want the notifier to use a custom email address in the Return-Path: <> field. If the mail server specified in SmtpServer does not support SMTP VRFY command, then you need to set the SmtpVrfyOff to `1` in order for the SmtpReturnPath value to take effect. Type and dimension: string-scalar Example: "usera@example.com"

Table 6-2 Optional attributes for Solaris *(continued)*

Optional attribute	Description
SmtpServerTimeout	This attribute represents the time in seconds notifier waits for a response from the mail server for the SMTP commands it has sent to the mail server. This value can be increased if you notice that the mail server is taking a longer duration to reply back to the SMTP commands sent by notifier. Type and dimension: integer-scalar Default: 10
SmtpServerVrfyOff	Set this value to 1 if your mail server does not support SMTP VRFY command. If you set this value to `1`, the notifier does not send a SMTP VRFY request to the mail server specified in SmtpServer attribute while sending emails. Type and dimension: boolean-scalar Default: 0
SnmpCommunity	Specifies the community ID for the SNMP manager. Type and dimension: string-scalar Default: public
SnmpdTrapPort	Port on the SNMP console machine where SNMP traps are sent. If you specify more than one SNMP console, all consoles use this value. Type and dimension: integer-scalar Default: 162
MessageExpiryInterval	Time in seconds after which the messages expire. If the VCS engine is unable to send a message to the notifier within the message expiry interval, it deletes the message from the VCS engine's message queue. Minimum value: 3600 Type and dimension: integer-scalar Default: 3600

Resource type definition

```
type NotifierMngr (
static int RestartLimit = 3
static str ArgList[] = { EngineListeningPort, MessagesQueue,
MessageExpiryInterval, NotifierListeningPort, NotifierSourceIP,
SnmpdTrapPort, SnmpCommunity, SnmpConsoles, SmtpServer,
SmtpServerVrfyOff, SmtpServerTimeout, SmtpReturnPath,
SmtpFromPath, SmtpRecipients }
int EngineListeningPort = 14141
int MessagesQueue = 30
int MessageExpiryInterval = 3600
int NotifierListeningPort = 14144
str NotifierSourceIP
int SnmpdTrapPort = 162
str SnmpCommunity = public
str SnmpConsoles{}
str SmtpServer
boolean SmtpServerVrfyOff = 0
int SmtpServerTimeout = 10
str SmtpReturnPath
str SmtpFromPath
str SmtpRecipients{}
)
```

Sample configuration

In the following configuration, the NotifierMngr agent is configured to run with two resource groups: NicGrp and Grp1. NicGrp contains the NIC resource and a Phantom resource that enables VCS to determine the online and offline status of the group. See the Phantom agent for more information on verifying the status of groups that only contain OnOnly or Persistent resources such as the NIC resource. You must enable NicGrp to run as a parallel group on both systems.

Grp1 contains the NotifierMngr resource (ntfr) and a Proxy resource (nicproxy), configured for the NIC resource in the first group.

In this example, NotifierMngr has a dependency on the Proxy resource.

Note: Only one instance of the notifier process can run in a cluster. The process cannot run in a parallel group.

The NotifierMngr resource sets up notification for all events to the SNMP console `SNMPServerName`. In this example, only messages of SevereError level are sent to the SMTP server (`smtp.example.com`), and the recipient (`vcsadmin@example.com`).

Configuration

```
system north

system south

group NicGrp (
     SystemList = { north = 0, south = 1 }
     AutoStartList = { north }
     Parallel = 1
     )

     Phantom my_phantom (

     )

     NIC NicGrp_e1000g0 (
         Device = e1000g0
         )

group Grp1 (
     SystemList = { north = 0, south = 1 }
     AutoStartList = { north }
     )

     Proxy nicproxy(
     TargetResName = "NicGrp_en0"
     )

     NotifierMngr ntfr (
         SnmpConsoles = { "SNMPServerName" = Information }
         SmtpServer = "smtp.example.com"
         SmtpRecipients = { "vcsadmin@example.com" = SevereError }
     )

ntfr requires nicproxy

// resource dependency tree
//
// group Grp1
// {
// NotifierMngr ntfr
// {
```

```
// Proxy nicproxy
// }
// }
```

Debug log levels

The NotifierMngr agent uses the following debug log levels:

DBG_1, DBG_2, DBG_3, DBG_5

Proxy agent

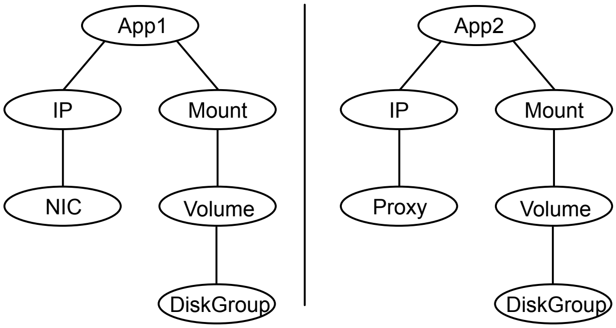
The Proxy agent mirrors the state of another resource on a local or remote system. It provides a means to specify and modify one resource and have its state reflected by its proxies. You can use the agent when you need to replicate the status of a resource.

A Proxy resource can only point to None or OnOnly type of resources, and can reside either in a failover or a parallel group. A target resource and its proxy cannot be in the same group.

Dependencies

No dependencies exist for the Proxy resource.

Figure 6-1 Sample service group that includes a Proxy resource



Agent functions

Monitor	Determines status based on the target resource status.

Attributes

Table 6-3 Required attribute

Required attribute	Description
TargetResName	Name of the target resource that the Proxy resource mirrors. The target resource must be in a different resource group than the Proxy resource. Type and dimension: string-scalar Example: "nic1"

Table 6-4 Optional attribute

Optional attribute	Description
TargetSysName	Mirrors the status of the TargetResName attribute on systems that the TargetSysName variable specifies. If this attribute is not specified, the Proxy resource assumes the system is local. Type and dimension: string-scalar Example: "sysa"

Resource type definition

```
type Proxy (
    static str ArgList[] = { TargetResName, TargetSysName,
    "TargetResName:Probed", "TargetResName:State" }
    static int OfflineMonitorInterval = 60
    static str Operations = None
    str TargetResName
    str TargetSysName
)
```

Sample configurations

Configuration 1

```
Proxy proxy1 (
    TargetResName = "nic1"
)
```

Configuration 2

The proxy resource mirrors the state of the resource nic2 on sysa.

```
Proxy proxy1(
    TargetResName = "nic2"
    TargetSysName = "sysa"
)
```

Configuration 3

The proxy resource mirrors the state of the resource mnic on the local system; the target resource is in grp1 and the proxy in grp2. A target resource and its proxy cannot be in the same group.

```
group grp1 (
SystemList = { sysa = 0, sysb = 1 }
AutoStartList = { sysa }
)
(
Device@sysa = { nxge0 = "166.98.16.103",qfe3 = "166.98.16.103"
}
Device@sysb = { nxge0 = "166.98.16.104",qfe3 = "166.98.16.104"
}
NetMask = "255.255.255.0"
ArpDelay = 5
Options = "failover"
)
(
Address = "166.98.16.78"
NetMask = "255.255.255.0"
MultiNICResName = mnic
Options = "failover"
)
ip1 requires mnic
group grp2 (
SystemList = { sysa = 0, sysb = 1 }
AutoStartList = { sysa }
)
(
Address = "166.98.16.79"
NetMask = "255.255.255.0"
MultiNICResName = mnic
Options = "mtu 1500"
)
Proxy proxy (
TargetResName = mnic
)
ip2 requires proxy
```

Debug log levels

The Proxy agent uses the following debug log levels:

DBG_1, DBG_2

Phantom agent

The agent enables VCS to determine the status of parallel service groups that do not include OnOff resources, which are resources that VCS can start and stop. Without the "dummy" resource provided by this agent, VCS cannot assess the status of groups that only contain None (Persistent) and OnOnly resources because the state of these resources is not considered in the process of determining whether a group is online. Refer to the *VCS Administrator's Guide* for information on categories of service groups and resources.

Do not use the Phantom resource in failover service groups.

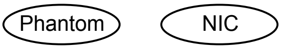
Also, the Phantom resource should not be used in service groups that don't contain any resources.

Note: Do not attempt manual online or offline operations on the Phantom resource at the resource level. Do not use `hares` commands on the Phantom resource at the resource level. Unpredictable behavior results when you try a manual online or offline procedure or an `hares` command on a Phantom resource. You can perform commands on the service group that contains the Phantom resource.

Dependencies

No dependencies exist for the Phantom resource.

Figure 6-2 Sample service group that includes a Phantom resource



Agent functions

Monitor	Determines status based on the status of the service group.

Resource type definition

```
type Phantom (
)
```

Sample configurations

Configuration 1

```
Phantom boo (
)
```

Configuration 2

The following example shows a complete main.cf, in which the FileNone resource and the Phantom resource are in the same group.

```
include "types.cf"
cluster PhantomCluster
system sysa (
)
system sysb (
)
group phantomgroup (
SystemList = { sysa = 0, sysb = 1 }
AutoStartList = { sysa }
Parallel = 1
)
FileNone my_file_none (
PathName = "/tmp/file_none"
)
Phantom my_phantom (
)
// resource dependency tree
//
// group maingroup
// {
// Phantom my_Phantom
// FileNone my_file_none
// }
```

RemoteGroup agent

The RemoteGroup agent establishes dependencies between applications that are configured on different VCS clusters. For example, you configure an Apache resource in a local cluster, and a MySQL resource in a remote cluster. In this example, the Apache resource depends on the MySQL resource. You can use the RemoteGroup agent to establish this dependency between these two resources.

With the RemoteGroup agent, you can monitor or manage a service group that exists in a remote cluster.

Some points about configuring the RemoteGroup resource follow:

- For each remote service group that you want to monitor or manage, you must configure a corresponding RemoteGroup resource in the local cluster.
- Multiple RemoteGroup resources in a local cluster can manage corresponding multiple remote service groups in different remote clusters.
- You can include the RemoteGroup resource in any kind of resource or service group dependency tree.
- A combination of the state of the local service group and the state of the remote service group determines the state of the RemoteGroup resource.

Veritas supports the RemoteGroup agent when:

- When it points to a global group

  The RemoteGroup agent must then map the state of the global group in the local cluster.
- When it is configured inside a local parallel service group

  The RemoteGroup resources on all cluster nodes monitor the same remote service group unless its attributes are localized.
- When it is configured inside a local failover service group

For more information on the functionality of this agent refer to the *Cluster Server Administrator's Guide*.

Dependency

As a best practice, establish a RemoteGroup resource dependency on a NIC resource. Veritas recommends that the RemoteGroup resource not be by itself in a service group.

Agent functions

Online

Brings the remote service group online. For more information:

See Table 6-5 on page 277.

Offline

Takes the remote service group offline. For more information:

See Table 6-5 on page 277.

Monitor

Monitors the state of the remote service group.

The true state of the remote service group is monitored only on the online node in the local cluster. For more information:

See Table 6-5 on page 277.

Clean

If the RemoteGroup resource faults, the Clean function takes the remote service group offline. For more information:

See Table 6-5 on page 277.

State definitions

ONLINE

Indicates that the remote service group is in an ONLINE state.

If the ReturnIntOffline attribute is not set to RemotePartial, then the remote service group is either in an ONLINE or PARTIAL state.

OFFLINE

Indicates that the remote service group is in an OFFLINE or FAULTED state. The true state of the remote service group is monitored only on the online node in the local cluster.

The RemoteGroup resource returns intentional offline if the attribute ReturnIntOffline is set to an appropriate value.

FAULTED

Indicates that the RemoteGroup resource has unexpectedly gone offline.

UNKNOWN

Indicates that a problem exists either with the configuration or the ability of the RemoteGroup resource to determine the state of the remote service group.

Attributes

Table 6-5 Required attributes

Required attribute	Description
IpAddress	The IP address or DNS name of a node in the remote cluster. The IP address can be either physical or virtual. When configuring a virtual IP address of a remote cluster, do not configure the IP resource as a part of the remote service group. Type and dimension: string-scalar Examples: "www.example.com" or "11.183.12.214"
Port	This is a required attribute when the remote cluster listens on a port other than the default value of 14141. See Table 6-6 on page 280.
GroupName	The name of the service group on the remote cluster that you want the RemoteGroup agent to monitor or manage. Type and dimension: string-scalar Example: "DBGrp"
VCSSysName	You must set this attribute to either the VCS system name or the ANY value. ■ ANY The RemoteGroup resource goes online if the remote service group is online on any node in the remote cluster. ■ *VCSSysName* Use the name of a VCS system in a remote cluster where you want the remote service group to be online when the RemoteGroup resource goes online. Use this to establish a one-to-one mapping between the nodes of the local and remote clusters. Type and dimension: string-scalar Example: "vcssys1" or "ANY"

Table 6-5 Required attributes *(continued)*

Required attribute	Description
ControlMode	Select only one of these values to determine the mode of operation of the RemoteGroup resource: MonitorOnly, OnlineOnly, or OnOff. ■ OnOff The RemoteGroup resource brings the remote service group online or takes it offline. When you set the VCSSysName attribute to ANY, the SysList attribute of the remote service group determines the node where the remote service group onlines. ■ MonitorOnly The RemoteGroup resource only monitors the state of the remote service group. The RemoteGroup resource cannot online or offline the remote service group. Make sure that you bring the remote service group online before you online the RemoteGroup resource. ■ OnlineOnly The RemoteGroup resource only brings the remote service group online. The RemoteGroup resource cannot take the remote service group offline. When you set the VCSSysName attribute to ANY, the SysList attribute of the remote service group determines the node where the remote service group onlines. Type and dimension: string-scalar

Table 6-5 Required attributes *(continued)*

Required attribute	Description
Username	This is the login user name for the remote cluster. When you set the ControlMode attribute to OnOff or OnlineOnly, the Username must have administrative privileges for the remote service group that you specify in the GroupName attribute. When you use the RemoteGroup Wizard to enter your username data, you need to enter your username and the domain name in separate fields. For a cluster that has the Veritas Product Authentication Service, you do not need to enter the domain name. For a secure remote cluster: ■ Local Unix user user@nodename—where the nodename is the name of the node that is specified in the IpAddress attribute. Do not set the DomainType attribute. ■ NIS or NIS+ user user@domainName—where domainName is the name of the NIS or NIS+ domain for the user. You must set the value of the DomainType attribute to either to nis or nisplus. Type and dimension: string-scalar Example: ■ For a cluster without the Veritas Product Authentication Service: "johnsmith" ■ For a secure remote cluster: "foobar@example.com"
Password	This is the password that corresponds to the user that you specify in the Username attribute. You must encrypt the password by using the `vcsencrypt` utility. For details, see the *Cluster Server Administrator's Guide*. **Note:** Do not use the vcsencrypt utility when entering passwords from a configuration wizard or the Cluster Manager (Java Console). Type and dimension: string-scalar

Table 6-6 Optional attributes

Optional attribute	Description
DomainType	For a secure remote cluster only, enter the domain type information for the specified user. For users who have the domain type unixpwd, you do not have to set this attribute. Type: string-scalar Example: "nis", "nisplus"
BrokerIp	For a secure remote cluster only. If you need the RemoteGroup agent to communicate to a specific authentication broker, set the value of this attribute to the broker's IP address. Type: string-scalar Example: "128.11.295.51"
Port	The port where the remote engine listens for requests. This is an optional attribute, unless the remote cluster listens on a port other than the default value of 14141. Type and dimension: integer-scalar Default: 14141
OfflineWaitTime	The maximum expected time in seconds that the remote service group may take to offline. VCS calls the clean function for the RemoteGroup resource if the remote service group takes a longer time to offline than the time that you have specified for this attribute. Type and dimension: integer-scalar Default: 0

Table 6-6 Optional attributes *(continued)*

Optional attribute	Description
ReturnIntOffline	Select one of the following values for RemoteGroup to return IntentionalOffline: ■ RemotePartial—Indicates that the RemoteGroup resource returns an IntentionalOffline if the remote service group is in an ONLINE|PARTIAL state. ■ RemoteOffline—Indicates that the RemoteGroup resource returns an IntentionalOffline if the remote service group is in an OFFLINE state. ■ RemoteFaulted—Indicates that the RemoteGroup resource returns an IntentionalOffline if the remote service group is OFFLINE|FAULTED. You can use these values in combinations with each other. You must set the IntentionalOffline attribute of the RemoteGroup resource type to 1 for this attribute to work properly. For more information about this attribute, see the *Cluster Server Administrator's Guide*. Type and dimension: string-vector Default: ""
OfflineMonitoringN ode	Defines the cluster node that performs the offline monitoring of the remote service group. This is an internal attribute. Do not modify.

Table 6-7 Type-level attributes

Type level attributes	Description
OnlineRetryLimit OnlineWaitLimit	In case of remote service groups that take a longer time to Online, Veritas recommends that you modify the default OnlineWaitLimit and OnlineRetryLimit attributes. See the *Cluster Server Administrator's Guide* for more information about these attributes.
ToleranceLimit MonitorInterval	If you expect the RemoteGroup agent to support sudden offline of the remote service group, modify the ToleranceLimit attribute. See the *Cluster Server Administrator's Guide* for more information about these attributes.

Table 6-7 Type-level attributes *(continued)*

Type level attributes	Description
ExternalStateChange	If you want the local service group to go online or offline when the RemoteGroup resource goes online or offline outside VCS control, set the attribute ExternalStateChange appropriately. See the *Cluster Server Administrator's Guide* for more information about these attributes.

Resource type definition

```
type RemoteGroup (
static int OnlineRetryLimit = 2
static int ToleranceLimit = 1
static boolean IntentionalOffline = 1
static str ArgList[] = { IpAddress, Port, Username, Password,
GroupName, VCSSysName, ControlMode, OfflineWaitTime,
DomainType, BrokerIp, ReturnIntOffline }
str IpAddress
int Port = 14141
str Username
str Password
str GroupName
str VCSSysName
str ControlMode
int OfflineWaitTime
str DomainType
str BrokerIp
str ReturnIntOffline[] = {}
temp str OfflineMonitoringNode
)
```

Debug log levels

The RemoteGroup agent uses the following debug log levels:

DBG_1

Chapter 7

Testing agents

This chapter includes the following topics:

About the testing agents

Use the testing agents to provide high availability for program support resources. These resources are useful for testing service groups.

ElifNone agent

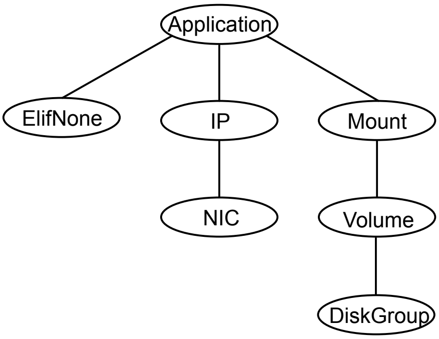
The ElifNone agent monitors a file. It checks for the file’s absence.

You can use the ElifNone agent to test service group behavior. You can also use it as an impostor resource, where it takes the place of a resource for testing.

Dependencies for ElifNone agent

No dependencies exist for the ElifNone resource.

Figure 7-1 Sample service group that includes an ElifNone resource



Agent function for ElifNone agent

Monitor

Checks for the specified file. If it exists, the resource faults. If it does not exist, the agent reports the resource as ONLINE.

State definitions for ElifNone agent

ONLINE

Indicates that the file specified in the PathName attribute does not exist.

FAULTED

Indicates that the file specified in the PathName attribute exists.

UNKNOWN

Indicates that the value of the PathName attribute does not contain a file name.

Attributes for ElifNone agent

Table 7-1 Required attribute

Required attribute	Description
PathName	Specifies the complete pathname. Starts with a slash (/) preceding the file name. Type and dimension: string-scalar Example: "/tmp/file01"

Resource type definition for ElifNone agent

```
type ElifNone (
    static str ArgList[] = { PathName }
    static int OfflineMonitorInterval = 60
    static str Operations = None
    str PathName
)
```

Sample configuration for ElifNone agent

```
ElifNone tmp_file01 (
    PathName = "/tmp/file01"
)
```

Debug log levels for ElifNone agent

The ElifNone agent uses the following debug log levels:

DBG_4, DBG_5

FileNone agent

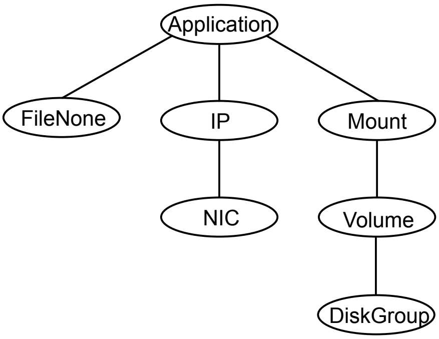
Monitors a file, checks for the file’s existence.

You can use the FileNone agent to test service group behavior. You can also use it as an "impostor" resource, where it takes the place of a resource for testing.

Dependencies for FileNone agent

No dependencies exist for the FileNone resource.

Figure 7-2 Sample service group that includes an FileNone resource



Agent functions for FileNone agent

Monitor — Checks for the specified file. If it exists, the agent reports the resource as ONLINE. If it does not exist, the resource faults.

State definitions for FileNone agent

ONLINE — Indicates that the file specified in the PathName attribute exists.

FAULTED — Indicates that the file specified in the PathName attribute does not exist.

UNKNOWN — Indicates that the value of the PathName attribute does not contain a file name.

Attribute for FileNone agent

Table 7-2 Required attribute

Required attribute	Description
PathName	Specifies the complete pathname. Starts with a slash (/) preceding the file name. Type and dimension: string-scalar Example: "/tmp/file01"

Resource type definition for FileNone agent

```
type FileNone (
    static str ArgList[] = { PathName }
    static int OfflineMonitorInterval = 60
    static str Operations = None
    str PathName
)
```

Sample configuration for FileNone agent

```
FileNone tmp_file01 (
    PathName = "/tmp/file01"
)
```

Debug log levels for FileNone agent

The FileNone agent uses the following debug log levels:

DBG_4, DBG_5

FileOnOff agent

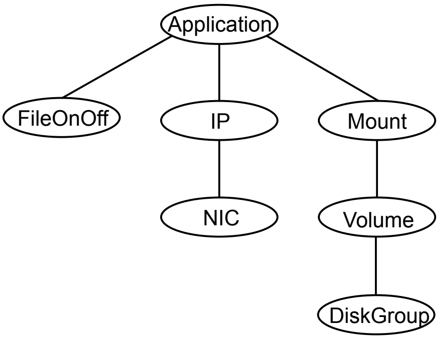
The FileOnOff agent creates, removes, and monitors a file.

You can use the FileNone agent to test service group behavior. You can also use it as an "impostor" resource, where it takes the place of a resource for testing.

Dependencies for FileOnOff agent

No dependencies exist for the FileOnOff resource.

Figure 7-3 Sample service group that includes a FileOnOff resource



Agent functions for FileOnOff agent

Online	Creates an empty file with the specified name if the file does not already exist.
Offline	Removes the specified file.
Monitor	Checks for the specified file. If it exists, the agent reports as ONLINE. If it does not exist, the agent reports as OFFLINE.
Clean	Removes the specified file forcibly when necessary.

State definitions for FileOnOff agent

ONLINE	Indicates that the file specified in the PathName attribute exists.
OFFLINE	Indicates that the file specified in the PathName attribute does not exist.
FAULTED	Indicates that the file specified in the PathName attribute has been removed out of VCS control.
UNKNOWN	Indicates that the value of the PathName attribute does not contain a file name.

Attribute for FileOnOff agent

Table 7-3 Required attribute

Required attribute	Description
PathName	Specifies the complete pathname. Starts with a slash (/) preceding the file name. Type and dimension: string-scalar Example: "/tmp/file01"

Resource type definition for FileOnOff agent

```
type FileOnOff (
    static str ArgList[] = { PathName }
    str PathName
)
```

Sample configuration for FileOnOff agent

```
FileOnOff tmp_file01 (
    PathName = "/tmp/file01"
)
```

Debug log levels for FileOnOff agent

The FileOnOff agent uses the following debug log levels:

DBG_4, DBG_5

FileOnOnly agent

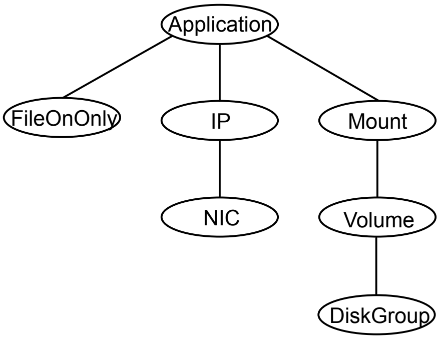
The FileOnOnly agent creates and monitors a file.

You can use the FileNone agent to test service group behavior. You can also use it as an "impostor" resource, where it takes the place of a resource for testing.

Dependencies for FileOnOnly agent

No dependencies exist for the FileOnOnly resource.

Figure 7-4 Sample service group that includes a FileOnOnly resource



Agent functions for FileOnOnly agent

Online	Creates an empty file with the specified name, unless one already exists.
Monitor	Checks for the specified file. If it exists, the agent reports as ONLINE. If it does not exist, the resource faults.

State definitions for FileOnOnly agent

The state definitions for this agent follow:

ONLINE	Indicates that the file specified in the PathName attribute exists.
OFFLINE	Indicates that the file specified in the PathName attribute does not exist and VCS has not attempted to bring the resource online.
FAULTED	Indicates that the file specified in the PathName attribute has been removed out of VCS control.
UNKNOWN	Indicates that the value of the PathName attribute does not contain a file name.

Attribute for FileOnOnly agent

Table 7-4 Required attributes

Required attribute	Description
PathName	Specifies the complete pathname. Starts with a slash (/) preceding the file name. Type and dimension: string-scalar Example: "/tmp/file02"

Resource type definition for FileOnOnly agent

```
type FileOnOnly (
    static str ArgList[] = { PathName }
    static str Operations = OnOnly
    str PathName
)
```

Sample configuration for FileOnOnly agent

```
FileOnOnly tmp_file02 (
PathName = "/tmp/file02"
)
```

Debug log levels for FileOnOnly agent

The FileOnOnly agent uses the following debug log levels:

DBG_4, DBG_5

Replication agents

This chapter includes the following topics:

About the replication agents

Use the replication agents to provide high availability for VVR resources.

Refer to the *Veritas InfoScale Replication Administrator's Guide* for information on configuring the Replication agents for high availability.

RVG agent

Brings the RVG online, monitors read and write access to the RVG, and takes the RVG offline. This is a failover resource. The RVG agent enables replication between clusters. It manages the Primary VVR node in one cluster and the Secondary VVR node in another cluster. Each node can be failed over in its respective cluster. In this way, replication is made highly available.

The RVG agent manages the state of the RVG during local failovers. The RVGPrimary agent manages the role of the RVG during a wide area failover.

Using a VCS global cluster enables you to fail over the Primary role from a Primary VVR node to a Secondary VVR node.

The RVG agent includes the following key features:

- Removes potential single points of failure by enabling Primary and Secondary VVR nodes to be clustered.
- Enables you to bring a service group online to start VCS-managed applications that use VVR.
- Continues replication after a node in a cluster fails without losing updates.
- Ensures that VVR can be added to any VCS cluster by including the RVG resource type definitions.

An example configuration file for this agent that can be used as a guide when creating your configuration is located at:

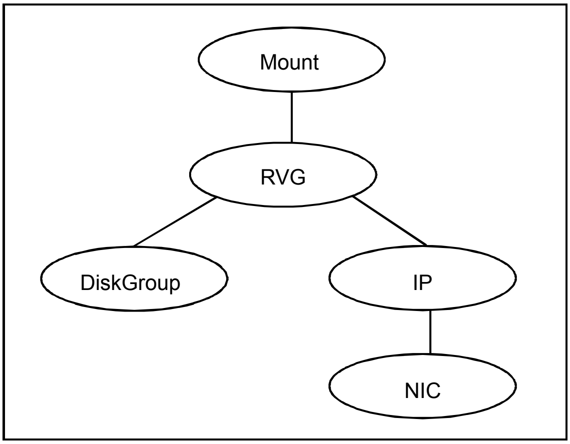
/etc/VRTSvcs/conf/sample_vvr/RVG

Dependencies

The RVG resource represents the RVG (Replicated Volume Group) in the RDS (Replicated Data Set). The RVG resource is dependent on the DiskGroup resource. The RVG resource is also dependent on the IP resources that it uses for replication.

Refer to the *Cluster Server Administrator's Guide* for more information on dependencies.

Figure 8-1 Sample service group for an RVG resource



Agent functions

The RVG agent has the following agent functions:

Online — Verifies whether the DiskGroup agent has recovered the RVG. If not, recovers and starts the data volumes and the Storage Replicator Log (SRL), recovers the RVG, recovers all RLINKs in the RVG, and then starts the RVG.

Offline — Stops the RVG.

Monitor — Monitors the state of the RVG using the `vxprint` command.

The RVG resource monitors an RVG for local access only. It does not monitor replication.

Clean — Stops the RVG.

Info — The info entry point displays information about the replication status of a RDS.

State definitions

The RVG agent has the following state definitions:

ONLINE — Indicates that the RVG is in `ENABLED`/`ACTIVE` state.

OFFLINE — Indicates that the RVG is in `DISABLED`/`CLEAN` state.

FAULTED — The RVG resource fails if the RVG is not in the ENABLED/ACTIVE state.

Attributes

Table 8-1 Required attributes

Attribute	Description
RVG	The name of the RVG being monitored. Type and dimension: string-scalar Example: "hr_rvg"
DiskGroup	The disk group that this RVG is associated with. Type and dimension: string-scalar Example: "hrbg"

Table 8-1 Required attributes *(continued)*

Attribute	Description
StorageDG	The name of the bunker disk group. Type and dimension: string-scalar Example: "hr_bdg"
StorageRVG	The name of the bunker RVG. Type and dimension: string-scalar Example: "hr_brvg"
StorageHostIds	A space-separated list of the hostids of each node in the bunker cluster. Type and dimension: string-keylist Example: "bunker_host"

Table 8-2 Optional attribute

Attribute	Description
NumThreads	Number of threads used within the agent process for managing resources. This number does not include threads used for other internal purposes. Do not modify this attribute for this agent. Setting this attribute to a higher value may result in agent function timeouts due to serialization of underlying commands. Default: 1

Resource type definitions

The RVG agent resource type definition follows.

```
type RVG (
    static int NumThreads = 1
    static str ArgList[] = { RVG, DiskGroup }
    str RVG
    str DiskGroup
    str StorageRVG
    str StorageDG
    str StorageHostIds
)
```

Sample configurations

```
RVG rvg (
       RVG = ApplicationRVG
       DiskGroup = vvrdg
       StorageRVG = ApplicationRVG
       StorageDG = vvrdg
       StorageHostIds = "bunker_host"
       )
```

RVGPrimary agent

The RVGPrimary agent enables migration and takeover of a VVR Replicated Volume Group (RVG) in a VCS environment. Bringing a resource of type RVGPrimary online causes the RVG on the local host to become a primary.

The agent is useful when hosts in both the primary and secondary side are clustered, in particular a VCS replicated data cluster or a VCS global cluster, to completely automate the availability of writable replicated disks to a VCS-managed application.

The RVGPrimary agent includes the following features:

- Removes the manual steps of migrating a VVR primary and secondary roles when failing over applications across a wide area.
- Minimizes the need for resynchronizing replicated volumes by attempting a migration before attempting a hard takeover.
- Waits for the two sides of a replicated data set to become completely synchronized before migrating roles.
- Supports an automatic fast failback resynchronization of a downed primary if it later returns after a takeover.
- Allows you to distinguish the Primary site after network failure or disaster
- Supports the ability to choose the Primary site after a site failure or network disruption is corrected.
- After a successful migration or takeover of a Secondary RVG, the RVGPrimary agent ensures to automatically start the replication from the new Primary to any additional Secondary(s) that exists in the RDS.
- Before a takeover, the RVGPrimary agent synchronizes the Secondary site with any bunker associated with the Primary site, when the Primary site is not available.

Refer to the *Veritas InfoScale Replication Administrator's Guide* for information on configuring the Replication agents for high availability.

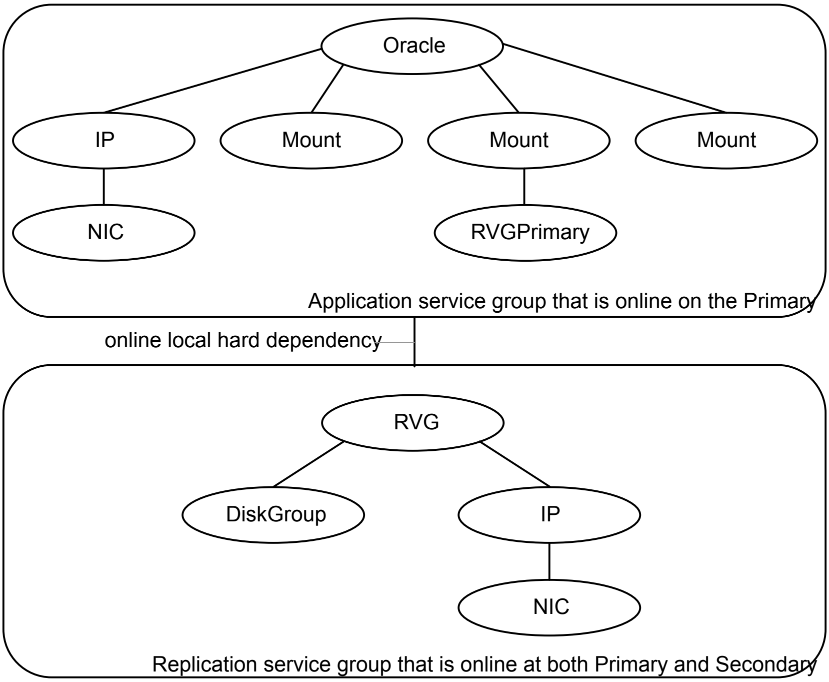
A sample configuration file for this agent that you can use as a guide to create the configuration is located at /etc/VRTSvcs/conf/sample_vvr/RVGPrimary.

Dependencies

You usually use the RVGPrimary agent in conjunction with the RVG agent in two groups with an online local hard group dependency. The parent group contains the resources that manage the actual application and file systems and as the RVGPrimary resource. The child group contains the resources managing the storage infrastructure, which include the RVG and DiskGroup type resources.

Refer to the *Veritas InfoScale Replication Administrator's Guide* for information about the setup of a VVR environment using the RVGPrimary agent.

Figure 8-2 Sample service group for an RVGPrimary resource


Oracle
IP
Mount
Mount
Mount
NIC
RVGPrimary
Application service group that is online on the Primary
online local hard dependency
RVG
DiskGroup
IP
NIC
Replication service group that is online at both Primary and Secondary

Agent functions

The RVGPrimary agent has the following agent functions:

Online

Determines the current role of the RVG. If the role is Secondary it attempts a migration. It waits for any outstanding writes from the original Primary. If the original Primary is down, it attempts a takeover. You can configure the RVGPrimary agent so that, before a takeover, the agent synchronizes the Secondary site with any bunker associated with the Primary site, when the Primary site is not available. If the RVG is a Primary, it performs no actions and goes online.

Offline

Performs no actions.

Monitor

Performs no actions. The RVG agents monitors the actual RVG.

Clean

Performs no actions.

fbsync

This is an action entry point.

It resynchronizes the original Primary with the new Primary that has taken over with fast-failback, after the original Primary had become unavailable. This needs to be executed when the original Primary becomes available and starts acting as a Secondary.

ElectPrimary

This is an action entry point.

It can be executed to retain the specified RVG as the Primary in a Primary-Primary configuration.

For more details, refer to the *Veritas InfoScale Replication Administrator's Guide*.

State definitions

The RVGPrimary agent has the following state definitions:

ONLLINE

Indicates that the role of the RVG is Primary.

FAULTED

The RVG agents monitors the actual RVG. Accidental migration of a VVR Primary outside of VCS causes other resources to fault immediately, such as Mount. No special monitoring by this agent is necessary.

Attributes

Table 8-3 Required attributes

Attribute	Description
RvgResourceName	The name of the RVG resource type that this agent promotes. The name RVG resource type which has been configured using the RVG agent. Type and dimension: string-scalar
AutoTakeover	A flag to indicate whether the agent should perform a takeover on online if the original Primary is down. AutoTakeover and AutoResync are mutually exclusive attributes. When AutoTakeover=0, the primary-elect feature is not applicable; therefore, it is not supported. Type and dimension: integer-scalar

Table 8-3 Required attributes *(continued)*

Attribute	Description
AutoResync	Indicates whether the agent should attempt to automatically perform a fast-failback resynchronization of the original Primary after a takeover and after the original Primary returns. You can use the following values for this attribute: ■ 0–instructs the agent to not attempt to perform a fast-failback resynchronization of the original Primary after a takeover and after the original Primary returns. ■ 1–instructs the agent to attempt to automatically perform a fast-failback resynchronization of the original Primary after a takeover and after the original Primary returns. ■ 2–instructs the agent to use the primary-elect feature. The agent does not attempt to perform a fast-failback resynchronization of the original Primary after a takeover and after the original Primary returns. The RVGPrimary agent also creates space-optimized snapshots for all the data volumes in the RVG resource. If you set the AutoResync attribute to 2 (to enable the primary-elect feature) the value of the BunkerSyncTimeOut attribute must be zero to disable the automated bunker replay feature. You cannot use the automated bunker replay feature and the primary-elect feature in the same environment. AutoTakeover and AutoResync are mutually exclusive attributes. When AutoTakeover=0, the primary-elect feature is not applicable; therefore, it is not supported. Type and dimension: integer-scalar

Table 8-3 Required attributes *(continued)*

Attribute	Description
BunkerSyncTimeOut	The value for the BunkerSyncTimeOut attribute determines if you want the bunker to perform a replay or not. You set the value in seconds for the time that you want to allot for the replay. Use one of the following values for the BunkerSyncTimeOut attribute: ■ If you do not use a value for this attribute (the default null value), the RVGPrimary agent considers it an infinite timeout value. The agent replays all the writes on the Bunker Replicator Log to the Secondary. Only after the agent sends all the writes, VCS performs the takeover on the Secondary. ■ If you set the value for this attribute to 0, you disable bunker replay for the agent. The RVGPrimary agent immediately performs a takeover on the Secondary. The agent does not send pending writes from the Bunker to the Secondary. ■ If you set the value to a number of seconds, then the RVGPrimary agent sends writes for that amount of time to the Secondary. After the agent meets the time limit, it performs the takeover on the Secondary. The bunker replay time in this case is equal to the value in seconds. You can set this value dynamically. The RVGPrimary agent's OnlineTimeout and OnlineRetryLimit attribute values determine the available time for an RVGPrimary resource to complete its online operation. Use the following formula to get the Time Available for Online to Complete (TAOC): TAOC = (OnlineTimeout + (OnlineRetryLimit * OnlineTimeout))

Table 8-3 Required attributes *(continued)*

Attribute	Description
BunkerSyncTimeOut (cont.)	When you set the BunkerSyncTimeOut value in seconds, the value of TAOC for the RVGPrimary agent should be greater than the desired BunkerSyncTimeOut value. Using a TAOC value that is greater than BunkerSyncTimeOut value ensures that the bunker replay and the RVG takeover can complete in the allotted time for that particular online operation. If the TAOC is smaller than BunkerSyncTimeOut value and the bunker replay does not complete within the allotted time for the online process, the resource faults. If the resource faults, clear the fault. Try the online operation again if the resource has not failed over to other cluster node in the configuration. If you increase the value of the BunkerSyncTimeOut attribute, you need to increase the value of the OnlineTimeout or OnlineRetryLimit attribute so that TAOC remain greater than changed value. This is to ensure to have bunker replay completed within allotted time for online. If the value of the AutoResync attribute is 2, you must set the value of the BunkerSyncTimeOut attribute to 0 (to disable automated bunker replay). Type and dimension: string-scalar Default value: ""

Table 8-4 Optional attributes

Attribute	Description
NumThreads	Number of threads used within the agent process for managing resources. This number does not include threads used for other internal purposes. Do not modify this attribute for this agent. Setting this attribute to a higher value may result in agent function timeouts due to serialization of underlying commands. Default: 1

Table 8-4 Optional attributes *(continued)*

Attribute	Description

ResyncType	Allows you to choose between automatic synchronization and difference-based synchronization.
	By default, difference-based synchronization is enabled for resynchronization.
	To enable automatic synchronization, run the following commands:
	`# haconf -makerw`
	`# hares -modify` *`RVGPrimary_resource_name`* `ResyncType 1`
	`# haconf -dump -makero`
	`# hares -value` *`RVGPrimary_resource_name`* `ResyncType`
	To track automatic synchronization progress, run the following commands:
	`# vxrlink -g` *`dg_name`* `-i` *`time_interval`* `status` *`rlk_name`*
	`# vradmin -g` *`dg_name`* `repstatus` *`rvg_name`*
	Type and dimension: integer-scalar
	Default: 0

Table 8-5 Internal attribute

Attribute	Description
BunkerSyncElapsedTime	For internal use only, do not modify. This value in seconds signifies the amount of time that a Secondary RVG has waited for synchronization from the bunker host to complete. Type and dimension: integer-scalar

Note: Default settings of AutoTakeover=1 and AutoResync=0 cause the first failover to succeed when the original Primary goes down, and upon the original Primary's return, the RDS (Replicated Data Set) has a Primary-Primary configuration error. Set the default value of the AutoResync attribute of the RVGPrimary and RVGSharedPri agents to 1 if you want the agent to automatically attempt a fast-failback resynchronization of the original Primary after a takeover and after the original Primary returns. This prevents the Primary-Primary configuration error.

Resource type definitions

The RVGPrimary resource type definition follows.

```
type RVGPrimary (
    static keylist SupportedActions = { fbsync, electprimary }
    static int NumThreads = 1
    static int OnlineRetryLimit = 1
    static str ArgList[] = { RvgResourceName, "RvgResourceName:RVG",
    "RvgResourceName:DiskGroup", AutoTakeover, AutoResync,
    BunkerSyncTimeOut, BunkerSyncElapsedTime }
    str RvgResourceName
    int AutoTakeover = 1
    int AutoResync = 0
    int ResyncType = 0
    str BunkerSyncTimeOut
    int BunkerSyncElapsedTime = 0
)
```

Sample configurations

```
 RVGPrimary rvg-pri (
   RvgResourceName = rvgRes
  )
```

RVGSnapshot

For a fire drill, creates and destroys a transactionally consistent space-optimized snapshot of all volumes in a VVR secondary replicated data set. The RVGSnapshot agent takes space-optimized snapshots on a secondary RVG. These snapshots can be mounted and written to without affecting the actual replicated data, which means that the space-optimized snapshot can be an effective tool for scheduling a “fire drill” to confirm that a wide-area failover is possible. By combining this agent with the VCS Mount agent, the CFSMount agent, and VCS agents that manage the application being replicated, you can create a special fire drill service group. You can bring this service group online and take it offline at regularly scheduled intervals to verify that the disaster recovery environment is robust.

In addition to the agent itself, a text-based wizard /opt/VRTSvcs/bin/fdsetup that prepares the VVR and VCS infrastructure for a fire drill and a script /opt/VRTSvcs/bin/fdsched that runs the fire drill and consolidates the results are also included.

Complete details are in the *Cluster Server Administrator's Guide*.

The RVGSnapshot agent includes the following key features:

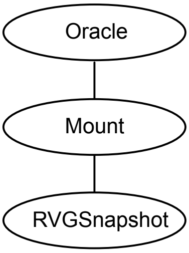
- Automates the process of creating a space-optimized snapshot on a VVR secondary that can be mounted to simulate a wide-area failover without affecting the production application.
- Includes a wizard to effectively set up and schedule fire drills that are completely managed by VCS.

Note: The RVGSnapshot agent does not support Volume Sets.

Dependencies

The RVGSnapshot agent depends on these resources.

Figure 8-3 Sample service group for an RVGSnapshot resource



Agent functions

The RVGSnapshot agent has the following agent functions:

Online	Creates a transactionally consistent snapshot of all volumes in the RVG.
Offline	Destroys the snapshot.
Monitor	No operation; failure of the snapshot will be indicated by the failure of the Mount resource of any file systems mounted on it.
Clean	Cleans up any failed snapshot creation or deletion.

State definitions

The RVGSnapshot agent has the following state definitions:

ONLINE	Indicates that a snapshot was created.
OFFLINE	Indicates that a snapshot was destroyed.

FAULTED	The RVGSnapshot resource faults on timeout if a snapshot creation did not succeed during an online.

Attributes

Table 8-6 Required attributes

Attribute	Description
RvgResourceName	The name of the VCS RVG-type resource that manages the RVG that will be snapshot by this agent. Type and dimension: string-scalar
CacheObj	Name of the cache object that is required for a space-optimized snapshot; the fdsetup wizard will create one if it does not exist Type and dimension: string-scalar
Prefix	Token put before the name of the actual volume when creating the snapshotted volumes. Type and dimension: string-scalar

Table 8-7 Optional attributes

Attribute	Description
DestroyOnOffline	A flag to indicate whether to destroy the snapshot upon taking the resources offline. For a fire drill, the snapshot should be deleted to reduce any performance impact of leaving the snapshot for a long period of time; however, if there is interest in keeping the data, then this value should be set to 0. The default is 1 (true). Type and dimension: integer-scalar Default: 1
FDFile	The fire drill schedule updates this attribute with the system name and the path to a file containing the output of the last complete fire drill for the group containing an RVGSnapshot resource. Type and dimension: string-scalar

Table 8-7 Optional attributes *(continued)*

Attribute	Description
NumThreads	Number of threads used within the agent process for managing resources. This number does not include threads used for other internal purposes. Do not modify this attribute for this agent. Setting this attribute to a higher value may result in agent function timeouts due to serialization of underlying commands. Default: 1

Resource type definitions

The resource type definition for the RVGSnapshot agent follows.

```
type RVGSnapshot (
   static keylist RegList = { Prefix }
   static int NumThreads = 1
   static str ArgList[] = { RvgResourceName, CacheObj, Prefix,
   DestroyOnOffline }
   str RvgResourceName
   str CacheObj
   str Prefix
   boolean DestroyOnOffline = 1
   temp str FDFile
   temp str VCSResLock
)
```

Sample configurations

```
RVGSnapshot rvg-sos (
       RvgResourceName = ApplicationRVG
       CacheObj = cacheobj
        Prefix = snap
        )
```

RVGShared agent

Monitors the RVG in a shared environment. This is a parallel resource. The RVGShared agent enables you to configure parallel applications to use an RVG in a cluster. The RVGShared agent monitors the RVG in a shared disk group environment. The RVGShared agent must be configured as a parallel group in VCS. Typically, the RVGShared resource is online or offline at the same time on all the nodes in the VCS cluster. An example configuration file for this agent that can be used as a guide when creating your configuration is located at /etc/VRTSvcs/conf/sample_vvr/RVGLogowner.

Dependencies

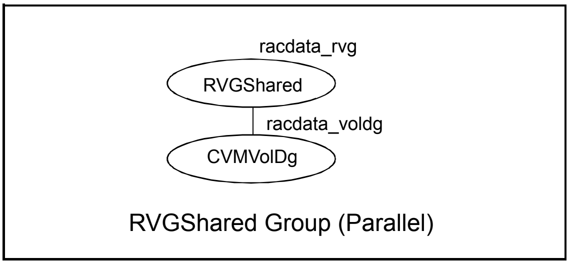
The RVGShared resource represents the RVG of the RDS. The RVGShared resource is dependent on the CVMVolDg resource.

The RVGShared resource must be configured in a parallel group.

Refer to the *Veritas InfoScale Replication Administrator's Guide* for information on configuring parallel applications for highly availability.

Refer to the *Cluster Server Administrator's Guide* for more information on dependencies.

Figure 8-4 Sample service group for an RVGShared resource



Note: Do not add any volumes that are part of the RVG in the CVMVolume attribute of the CVMVolDg resource. The volumes in the RVG are managed by the RVGShared resource.

Agent functions

The RVGShared agent has the following agent functions:

Online — Verifies whether the RVG is started. If the RVG is not started, recovers and starts the RVG.

Offline — No action.

Monitor — Displays the state as `ONLINE` if the RVG is started. Displays the state as `OFFLINE` if the RVG is not started.

Clean — No action.

Info — The info entry point displays information about the replication status of a RDS.

State definitions

The RVGShared agent has the following state definitions:

ONLINE — Indicates that the RVG is in the `ENABLED`/`ACTIVE` state.

OFFLINE — Indicates that the RVG is not in the `ENABLED`/`ACTIVE` state or that the administrator has invoked the offline entry point.

Attributes

Table 8-8 Required attributes

Attribute	Description
RVG	The name of the RVG being monitored. Type and dimension: string-scalar
DiskGroup	The shared-disk group with which this RVG is associated. Type and dimension: string-scalar

Table 8-9

Attribute	Description
NumThreads	Number of threads used within the agent process for managing resources. This number does not include threads used for other internal purposes. Do not modify this attribute for this agent. Setting this attribute to a higher value may result in agent function timeouts due to serialization of underlying commands. Default: 1

Resource type definitions

The RVGShared resource type definition follows.

```
type RVGShared (
    static int NumThreads = 1
    static str ArgList[] = { RVG, DiskGroup }
    str RVG
    str DiskGroup
)
```

Sample configurations

```
RVGShared racdata_rvg (
RVG = rac1_rvg
DiskGroup = oradatadg
)
```

RVGLogowner agent

Assigns and unassigns a node as the logowner in the CVM cluster; this is a failover resource. The RVGLogowner agent assigns or unassigns a node as a logowner in the cluster. To replicate data, VVR requires network connectivity between the Primary and the Secondary. In a shared disk group environment, only one node, that is, the logowner, can replicate data to the Secondary.

For replication to be highly available, the logowner must be highly available. To make the logowner highly available, the RVGLogowner resource must be configured as a resource in a failover group. Also, a virtual IP must be set up on the logowner to enable replication and failover of the logowner from one node to another in a cluster. The virtual IP must be configured as an IP resource.

For more information about the logowner, see the *Veritas InfoScale Replication Administrator's Guide*. An example configuration file for this agent that can be used as a guide when creating your configuration, is located at /etc/VRTSvcs/conf/sample_vvr/RVGLogowner.

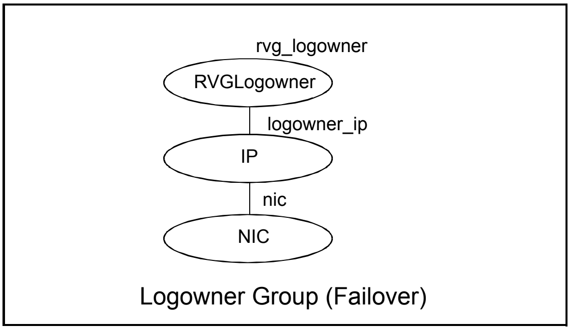
Dependencies

The RVGLogowner resource represents the logowner for RVG in the cluster. The RVGLogowner resource is dependent on the IP resource that it uses for replication.

The RVGLogowner resource must be configured in a failover group. The RVGLogowner group is used in conjunction with the RVGSharedPri and RVGShared agents in separate groups, with the appropriate service group dependencies.

For more information on dependencies, refer to the *Cluster Server Administrator's Guide*

Figure 8-5 Sample service group for an RVGLogowner resource



Agent functions

The RVGLogowner agent has the following agent functions:

Online Assigns the logowner on the node.

Offline Unassigns the logowner on the node.

Monitor Returns `ONLINE` if the node is the logowner and the RVG is in `ENABLED/ACTIVE` state. Returns `OFFLINE` if the node is the logowner and the state is not `ENABLED/ACTIVE`, or if the node is not the logowner (regardless of the state). The RVG for which the logowner is monitored must be configured as the RVGShared resource type.

Clean Unassigns the logowner on the node.

State definitions

The RVGLogowner agent has the following state definitions:

ONLINE	Indicates that the node is the logowner for the RVG in the cluster.
OFFLINE	Indicates that the node is not the logowner for the RVG in the cluster.

Attributes

Table 8-10 Required attributes

Attribute	Description
RVG	The name of the RVG being monitored. Type and dimension: string-scalar Example: "hr_rvg"
DiskGroup	The disk group with which this RVG is associated. Type and dimension: string-scalar Example: "hrbg"

Table 8-11 Optional attribute

Attribute	Description
NumThreads	Number of threads used within the agent process for managing resources. This number does not include threads used for other internal purposes. Do not modify this attribute for this agent. Setting this attribute to a higher value may result in agent function timeouts due to serialization of underlying commands. Default: 1

Table 8-12 Internal attributes

Attribute	Description
StorageDG	For internal use only, do not modify. The name of the bunker disk group. Type and dimension: string-scalar Example: "hr_bdg"

Table 8-12 Internal attributes *(continued)*

Attribute	Description
StorageRVG	For internal use only, do not modify. The name of the bunker RVG. Type and dimension: string-scalar Example: "hr_brvg"
StorageHostIds	For internal use only, do not modify. A space-separated list of the host IDs of each node in the bunker cluster. Type and dimension: string-keylist Example: "bunker_host"

Resource type definitions

The RVGLogowner resource type definition follows.

```
type RVGLogowner (
    static int NumThreads = 1
    static str ArgList[] = { RVG, DiskGroup }
    static int OnlineRetryLimit = 5
    str RVG
    str DiskGroup
    str StorageRVG
    str StorageDG
    str StorageHostIds
)
```

RVGLogowner agent notes

The RVGLogowner agent has the following notes:

Sample configurations

```
RVGLogowner vvr_rvglogowner (
  RVG = app_rvg
  DiskGroup = vvrdg
  )
```

RVGSharedPri agent

Attempts to migrate or takeover a Secondary to a Primary when a parallel service group fails over. The RVGSharedPri agent enables migration and takeover of a VVR Replicated Data Set (RDS) in parallel groups in a VCS environment. Bringing a resource of type RVGSharedPri online causes the RVG on the local host to become a primary if it is not already. The agent is useful when hosts in both the primary and secondary side are clustered using a VCS global cluster, to completely automate the availability of writable replicated disks to an application managed by VCS.

You cannot use the primary-elect feature with this agent. For a detailed description of the primary-elect feature, see *Veritas InfoScale Replication Administrator's Guide*.

The RVGSharedPri agent includes the following key features:

- Removes manual steps of migrating a VVR primary and secondary roles when failing over applications across a wide area.
- Minimizes the need for resynchronizing replicated volumes by attempting a migration before attempting a hard takeover.
- Waits for the two sides of a replicated data set to become completely synchronized before migrating roles.
- Supports an automatic fast failback resynchronization of a downed primary if it later returns after a takeover.
- After successful migration or takeover of a Secondary RVG, the agent automatically starts the replication from the new Primary to any additional Secondary(s) that exists in the RDS.

Sample configuration files are located in the `/etc/VRTSvcs/conf/sample_rac/` directory and include `CVR` in the filename. These sample files are installed as part of the `VRTSdbac` package, and can be used as a guide when creating your configuration.

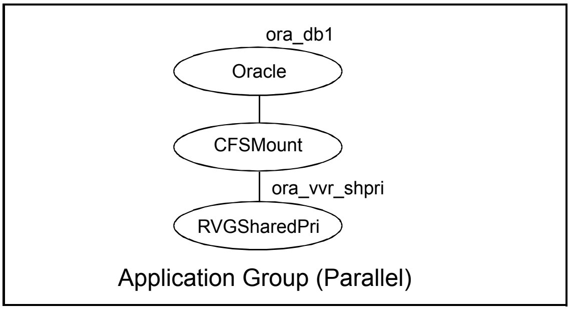
Dependencies

The RVGSharedPri agent is used in conjunction with the RVGShared and RVGLogowner agents in separate groups, with the appropriate service group dependencies.

Refer to the *Veritas InfoScale Replication Administrator's Guide* for information on configuring parallel applications for highly availability.

The RVGSharedPri agent must be configured in a parallel service group. The application service group contains the resources managing the actual application and file systems as well as the RVGSharedPri agent.

Figure 8-6 Sample service group for an RVGSharedPri resource



Agent functions

The RVGSharedPri agent has the following agent functions:

Online
Determines the current role of the RVG; if Secondary, attempt a migrate, waiting for any outstanding writes from the original Primary; if the original Primary is down attempt a takeover; if the RVG is a Primary, perform no actions and go online

Offline
Performs no actions.

Monitor
Performs no actions; monitoring of the actual RVG is done by the RVGShared agent.

Clean
Performs no actions.

fbsync
This is an action entry point.

It resynchronizes the original Primary with the new Primary that has taken over with fast-failback, after the original Primary had become unavailable.

This needs to be executed when the original Primary becomes available and starts acting as a Secondary.

resync
This is an action entry point.

It resynchronizes the Secondaries with the Primary using DCM.

State definitions

The RVGSharedPri agent has the following state definitions:

FAULTED	Monitoring of the actual RVG is done by the RVGShared agent; accidental migration of a VVR Primary outside of VCS would cause other resources to fault immediately, such as Mount, so no special monitoring by this agent is necessary.

Attributes

Table 8-13 Required attributes

Attribute	Description
RvgResourceName	The name of the RVGShared resource type that this agent will promote, that is, the name RVG resource type which has been configured using the RVGShared agent. Type and dimension: string-scalar
AutoTakeover	A flag to indicate whether the agent should perform a takeover on online if the original Primary is down. Type and dimension: integer-scalar Default: 1
AutoResync	A flag to indicate whether the agent should attempt to automatically perform a fast-failback resynchronization of the original Primary after a takeover and after the original Primary returns. Type and dimension: integer-scalar Default: 0
VCSResLock	This attribute is reserved for internal use by VCS. Type and dimension: string-scalar

Table 8-14 Optional attribute

Attribute	Description
NumThreads	Number of threads used within the agent process for managing resources. This number does not include threads used for other internal purposes. Do not modify this attribute for this agent. Setting this attribute to a higher value may result in agent function timeouts due to serialization of underlying commands. Default: 1

Note: Default settings of AutoTakeover=1 and AutoResync=0 cause the first failover to succeed when the original Primary goes down, and upon the original Primary's return, the RDS has a Primary-Primary configuration error. Set the default value of the AutoResync attribute of the RVGPrimary and RVGSharedPri agents to 1 if you want the agent to automatically attempt a fast-failback resynchronization of the original Primary after a takeover and after the original Primary returns. This prevents the Primary-Primary configuration error.

Resource type definitions

The RVGSharedPri resource type definition follows.

```
type RVGSharedPri (
    static keylist SupportedActions = { fbsync, resync }
    static int NumThreads = 1
    static int OnlineRetryLimit = 1
    static str ArgList[] = { RvgResourceName, "RvgResourceName:RVG",
    "RvgResourceName:DiskGroup", AutoTakeover, AutoResync }
    str RvgResourceName
    int AutoTakeover = 1
    int AutoResync = 0
    temp str VCSResLock
)
```

Sample configurations

```
RVGSharedPri ora_vvr_shpri (
RvgResourceName = racdata_rvg
OnlineRetryLimit = 0
)
```