# Cluster Server 7.3 Bundled Agents Reference Guide - AIX

VERITAS™

Last updated: 2017-05-31

## Legal Notice

Copyright © 2017 Veritas Technologies LLC. All rights reserved.

Veritas and the Veritas Logo are trademarks or registered trademarks of Veritas Technologies LLC or its affiliates in the U.S. and other countries. Other names may be trademarks of their respective owners.

This product may contain third party software for which Veritas is required to provide attribution to the third party (“Third Party Programs”). Some of the Third Party Programs are available under open source or free software licenses. The License Agreement accompanying the Software does not alter any rights or obligations you may have under those open source or free software licenses. Refer to the third party legal notices document accompanying this Veritas product or available at:

https://www.veritas.com/about/legal/license-agreements

The product described in this document is distributed under licenses restricting its use, copying, distribution, and decompilation/reverse engineering. No part of this document may be reproduced in any form by any means without prior written authorization of Veritas Technologies LLC and its licensors, if any.

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:

doc.feedback@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 ..... 14

About Bundled agents ..... 14
Resources and their attributes ..... 15
Modifying agents and their resources ..... 15
Attributes ..... 15
WPAR-aware agents ..... 16
IMF-aware agents ..... 17
Enabling debug log messages ..... 17
VCS support for multi-pathing solutions ..... 18

Chapter 2 Storage agents ..... 19

About the storage agents ..... 19
DiskGroup agent ..... 19
IMF awareness ..... 20
Support for online migration for DiskGroup agent ..... 20
Dependencies for DiskGroup agent ..... 20
Agent functions for DiskGroup agent ..... 20
State definitions for DiskGroup agent ..... 22
Attributes for DiskGroup agent ..... 23
Resource type definition ..... 27
Notes for DiskGroup agent ..... 28
Sample configurations for DiskGroup agent ..... 31
Debug log levels for DiskGroup agent ..... 31
DiskGroupSnap agent ..... 31
Dependencies for DiskGroupSnap agent ..... 32
Agent functions for DiskGroupSnap agent ..... 33
State definitions for DiskGroupSnap agent ..... 33
Attributes for DiskGroupSnap agent ..... 34
Notes for DiskGroupSnap agent ..... 36
Resource type definition for DiskGroupSnap agent ..... 40
Sample configurations for DiskGroupSnap agent ..... 40
Debug log levels for DiskGroupSnap agent ..... 47
Volume agent ..... 47
Dependencies for Volume agent ..... 47
Agent functions for Volume agent ..... 47

State definitions for Volume agent ..... 48
Attributes for Volume agent ..... 48
Resource type definition for Volume agent ..... 49
Sample configuration for Volume agent ..... 49
Debug log levels for Volume agent ..... 49
VolumeSet agent ..... 50
Dependencies for VolumeSet agent ..... 50
Agent functions for VolumeSet agent ..... 50
State definitions for VolumeSet agent ..... 50
Attributes for VolumeSet agent ..... 51
Resource type definition for VolumeSet agent ..... 51
Sample configurations for VolumeSet agent ..... 51
Agent notes for VolumeSet agent ..... 52
Inaccessible volumes prevent the VolumeSet agent from coming online ..... 52
Debug log levels for VolumeSet agent ..... 52
LVMVG agent ..... 52
Dependencies for LVMVG agent ..... 53
Agent functions for LVMVG agent ..... 53
State definitions for LVMVG agent ..... 54
Attributes for LVMVG agent ..... 55
Resource type definition for LVMVG agent ..... 57
Notes for LVMVG agent ..... 57
Sample configuration for LVMVG agent ..... 65
Debug log levels for LVMVG agent ..... 66
Mount agent ..... 66
IMF awareness ..... 66
Dependencies for Mount agent ..... 66
Agent functions for Mount agent ..... 67
State definitions for Mount agent ..... 69
Attributes for Mount agent ..... 70
Resource type definition for Mount agent ..... 77
Notes for Mount agent ..... 78
High availability fire drill ..... 79
VxFS file system lock ..... 79
IMF usage notes ..... 79
IPv6 usage notes ..... 80
Bringing a Mount resource online in the WPAR ..... 80
Selecting the attribute values for a Mount resource for the WPAR's root file system for NFS mounts ..... 80
Support for namefs file system ..... 81
Taking a group with the Mount resource offline can take several minutes if the file system is busy ..... 82

Example 1 ..... 82
Example 2 ..... 83
Example 3 ..... 83
Enabling Level two monitoring for the Mount agent ..... 83
Sample configurations for Mount agent ..... 84
Debug log levels for Mount agent ..... 85
SFCache agent ..... 85
Resource dependency ..... 86
Agent functions ..... 87
State definitions ..... 87
Attributes ..... 87
Resource type definition ..... 89
Notes for SFCache agent ..... 90
Debug log levels ..... 91

Chapter 3 Network agents ..... 92

About the network agents ..... 92
Agent comparisons ..... 92
IP agent ..... 94
High availability fire drill for IP agent ..... 94
Dependencies for IP agent ..... 94
Agent functions for IP agent ..... 95
State definitions for IP agent ..... 95
Attributes for IP agent ..... 96
Resource type definition for IP agent ..... 98
Sample configurations for IP agent ..... 99
Debug log levels for IP agent ..... 99
NIC agent ..... 99
Dependencies for NIC agent ..... 100
Agent functions for NIC agent ..... 100
State definitions for NIC agent ..... 101
Attributes for NIC agent ..... 101
Resource type definition for NIC agent ..... 103
Notes for the NIC agent ..... 103
Sample configurations for NIC agent ..... 103
IPv6 configuration for NIC agent ..... 104
Debug log levels for NIC agent ..... 105
IPMultiNIC agent ..... 105
Dependencies for IPMultiNIC agent ..... 105
Agent functions for IPMultiNIC agent ..... 105
State definitions for IPMultiNIC agent ..... 106
Attributes for IPMultiNIC agent ..... 106

Resource type definition for IPMultiNIC agent ..... 107
Sample configuration: IPMultiNIC and MultiNICA ..... 108
Debug log levels ..... 109
MultiNICA agent ..... 109
Dependencies for MultiNICA agent ..... 109
Agent function for MultiNICA agent ..... 110
State definitions for MultiNICA agent ..... 110
Attributes for MultiNICA agent ..... 110
Resource type definition for MultiNICA agent ..... 113
Notes for MultiNICA agent ..... 114
Sample configurations for MultiNICA agent ..... 115
Debug log levels for MultiNICA agent ..... 116
About the IPMultiNICB and MultiNICB agents ..... 117
Checklist to ensure the proper operation of MultiNICB ..... 117
IPMultiNICB agent ..... 117
Dependencies for IPMultiNICB agent ..... 118
Requirements for IPMultiNICB ..... 118
The haipswitch utility for IPMultiNICB agent ..... 118
Agent functions for IPMultiNICB agent ..... 118
State definitions for IPMultiNICB agent ..... 119
Attributes for IPMultiNICB agent ..... 120
Resource type definition for IPMultiNICB agent ..... 121
Sample configurations for IPMultiNICB agent ..... 122
Debug log levels for IPMultiNICB agent ..... 122
MultiNICB agent ..... 123
The haping utility for MultiNICB agent ..... 123
Dependencies for MultiNICB agent ..... 123
Agent functions for MultiNICB agent ..... 124
State definitions for MultiNICB agent ..... 124
Attributes for MultiNICB agent ..... 124
Resource type definition for MultiNICB agent ..... 128
Trigger script for MultiNICB agent ..... 128
Sample configurations for MultiNICB agent ..... 129
Debug log levels for MultiNICB agent ..... 130
DNS agent ..... 130
Dependencies for DNS agent ..... 130
Agent functions for DNS agent ..... 131
State definitions for DNS agent ..... 132
Attributes for DNS agent ..... 133
Resource type definition for DNS agent ..... 138
Agent notes for DNS agent ..... 139
Sample configurations for DNS agent ..... 144
Debug log levels for DNS agent ..... 146

Chapter 4 File share agents ..... 147

About the file service agents ..... 147
NFS agent ..... 147
Dependencies for NFS agent ..... 148
Agent functions for NFS agent ..... 148
State definitions for NFS agent ..... 149
Attributes for NFS agent ..... 149
Resource type definition for NFS agent ..... 151
Notes for NFS agent ..... 151
Sample configurations for NFS agent ..... 152
Debug log levels for NFS agent ..... 152
NFSRestart agent ..... 152
Dependencies for NFSRestart agent ..... 153
Agent functions for NFSRestart agent ..... 154
State definitions ..... 155
Attributes for NFSRestart agent ..... 155
Resource type definition for NFSRestart agent ..... 156
Notes for NFSRestart agent ..... 156
Sample configurations for NFSRestart agent ..... 158
Debug log levels for NFSRestart agent ..... 159
Share agent ..... 159
Dependencies for Share agent ..... 159
Agent functions for Share agent ..... 160
State definitions for Share agent ..... 160
Attributes for Share agent ..... 161
Resource type definition for Share agent ..... 161
Notes for Share agent ..... 161
Sample configurations for Share agent ..... 162
Debug log levels for Share agent ..... 162
About the Samba agents ..... 162
The Samba agents ..... 162
Before using the Samba agents ..... 163
Supported versions for Samba agents ..... 163
Notes for configuring the Samba agents ..... 163
SambaServer agent ..... 164
Dependencies for SambaServer agent ..... 164
Agent functions for SambaServer agent ..... 164
State definitions for SambaServer agent ..... 165
Attributes for SambaServer agent ..... 165
Resource type definitions for SambaServer agent ..... 167
Sample configurations for SambaServer agent ..... 167
Debug log levels for SambaServer agent ..... 168

SambaShare agent ..... 168
Dependencies for SambaShare agent ..... 168
Agent functions for SambaShare agent ..... 168
State definitions for SambaShare agent ..... 169
Attributes for SambaShare agent ..... 169
Resource type definition for SambaShare agent ..... 169
Sample configuration for SambaShare agent ..... 170
Debug log levels for SambaShare agent ..... 170
NetBios agent ..... 170
Dependencies for NetBios agent ..... 170
Agent functions for NetBios agent ..... 171
State definitions for NetBios agent ..... 171
Attributes for NetBios agent ..... 172
Resource type definition for NetBios agent ..... 174
Sample configuration for NetBios agent ..... 174
Debug log levels for NetBios agent ..... 174

Chapter 5 Service and application agents ..... 175

About the services and applications agents ..... 175
Apache HTTP server agent ..... 175
Dependencies ..... 176
Agent functions ..... 177
State definitions ..... 178
Attributes ..... 178
Resource type definition ..... 182
Apache HTTP server notes ..... 183
Sample configurations ..... 186
Debug log level ..... 189
Application agent ..... 189
IMF awareness ..... 189
High availability fire drill for Application agent ..... 190
Dependencies for Application agent ..... 190
Agent functions ..... 191
State definitions for Application agent ..... 193
Agent attributes ..... 193
Resource type definition for Application agent ..... 196
Notes for Application agent ..... 197
Sample configurations for Application agent ..... 200
Debug log levels for Application agent ..... 201
CoordPoint agent ..... 201
Coordination Point server as a coordination point ..... 201
SCSI-3 based disk as a coordination point ..... 201

Dependencies ..... 202
Agent functions ..... 202
State definitions ..... 202
Attributes ..... 204
Resource type definition ..... 205
Notes for the CoordPoint agent ..... 205
Sample configuration ..... 207
Debug log levels ..... 207
Process agent ..... 207
IMF awareness ..... 208
High availability fire drill for Process agent ..... 208
Dependencies for Process agent ..... 208
Agent functions for Process agent ..... 209
State definitions for Process agent ..... 209
Attributes for Process agent ..... 210
Resource type definition for Process agent ..... 210
Usage notes for Process agent ..... 211
Sample configurations for Process agent ..... 211
Debug log levels for Process agent ..... 212
ProcessOnOnly agent ..... 212
Dependencies ..... 212
Agent functions ..... 212
State definitions ..... 213
Attributes ..... 213
Resource type definition ..... 214
ProcessOnOnly agent usage notes ..... 214
Sample configurations ..... 214
Debug log levels ..... 215
WPAR agent ..... 215
IMF awareness ..... 215
Dependencies ..... 215
Agent functions ..... 216
Attributes ..... 216
Resource type definition ..... 218
Sample configuration ..... 219
WPAR agent notes ..... 220
Debug log levels ..... 221
MemCPUAllocator agent ..... 221
Dependencies ..... 221
Agent functions ..... 222
Attributes ..... 222
Resource type definition ..... 224
MemCPUAllocator agent notes ..... 224

Debug log levels ..... 229
LPAR agent ..... 229
Dependencies for LPAR agent ..... 230
Agent functions for LPAR agent ..... 230
State definitions for LPAR agent ..... 230
Required attributes for LPAR agent ..... 231
Optional attributes for LPAR agent ..... 232
Group attribute for LPAR agent ..... 233
System attribute for LPAR agent ..... 234
Resource type definition for LPAR agent ..... 234
Sample configurations for LPAR agent ..... 235
Notes for LPAR agent ..... 235
Debug log levels for LPAR agent ..... 237

Chapter 6 Infrastructure and support agents ..... 238

About the infrastructure and support agents ..... 238
NotifierMngr agent ..... 238
Dependency ..... 239
Agent functions ..... 239
State definitions ..... 239
Attributes ..... 239
Resource type definition ..... 242
Sample configuration ..... 243
Debug log levels ..... 245
Proxy agent ..... 245
Dependencies ..... 245
Agent functions ..... 246
Attributes ..... 246
Resource type definition ..... 247
Sample configurations ..... 247
Debug log levels ..... 248
Phantom agent ..... 248
Dependencies ..... 249
Agent functions ..... 249
Resource type definition ..... 249
Sample configurations ..... 249
RemoteGroup agent ..... 250
Dependency ..... 251
Agent functions ..... 251
State definitions ..... 251
Attributes ..... 252
Resource type definition ..... 257

Debug log levels ..... 257

Chapter 7 Testing agents ..... 258

About the testing agents ..... 258
ElifNone agent ..... 258
Dependencies for ElifNone agent ..... 258
Agent function for ElifNone agent ..... 259
State definitions for ElifNone agent ..... 259
Attributes for ElifNone agent ..... 259
Resource type definition for ElifNone agent ..... 260
Sample configuration for ElifNone agent ..... 260
Debug log levels for ElifNone agent ..... 260
FileNone agent ..... 260
Dependencies for FileNone agent ..... 260
Agent functions for FileNone agent ..... 261
State definitions for FileNone agent ..... 261
Attribute for FileNone agent ..... 261
Resource type definition for FileNone agent ..... 262
Sample configuration for FileNone agent ..... 262
Debug log levels for FileNone agent ..... 262
FileOnOff agent ..... 262
Dependencies for FileOnOff agent ..... 262
Agent functions for FileOnOff agent ..... 263
State definitions for FileOnOff agent ..... 263
Attribute for FileOnOff agent ..... 264
Resource type definition for FileOnOff agent ..... 264
Sample configuration for FileOnOff agent ..... 264
Debug log levels for FileOnOff agent ..... 264
FileOnOnly agent ..... 264
Dependencies for FileOnOnly agent ..... 264
Agent functions for FileOnOnly agent ..... 265
State definitions for FileOnOnly agent ..... 265
Attribute for FileOnOnly agent ..... 266
Resource type definition for FileOnOnly agent ..... 266
Sample configuration for FileOnOnly agent ..... 266
Debug log levels for FileOnOnly agent ..... 266

Chapter 8 Replication agents ..... 267

About the replication agents ..... 267
RVG agent ..... 267
Dependencies ..... 268
Agent functions ..... 269

State definitions ..... 269
Attributes ..... 269
Resource type definitions ..... 270
Sample configurations ..... 271
RVGPrimary agent ..... 271
Dependencies ..... 272
Agent functions ..... 272
State definitions ..... 273
Attributes ..... 274
Resource type definitions ..... 278
Sample configurations ..... 278
RVGSnapshot ..... 279
Dependencies ..... 279
Agent functions ..... 280
State definitions ..... 280
Attributes ..... 280
Resource type definitions ..... 281
Sample configurations ..... 282
RVGShared agent ..... 282
Dependencies ..... 282
Agent functions ..... 283
State definitions ..... 283
Attributes ..... 284
Resource type definitions ..... 284
Sample configurations ..... 284
RVGLogowner agent ..... 285
Dependencies ..... 285
Agent functions ..... 286
State definitions ..... 286
Attributes ..... 286
Resource type definitions ..... 287
RVGLogowner agent notes ..... 288
Sample configurations ..... 288
RVGSharedPri agent ..... 288
Dependencies ..... 289
Agent functions ..... 289
State definitions ..... 290
Attributes ..... 290
Resource type definitions ..... 291
Sample configurations ..... 292

Index ..... 293

Chapter 1

# Introducing Bundled agents

This chapter includes the following topics:


- About Bundled agents
- Resources and their attributes
- Modifying agents and their resources
- Attributes
- WPAR-aware agents
- IMF-aware agents
- Enabling debug log messages
- VCS support for multi-pathing solutions


## 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 (_).<br>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{}. |

# WPAR-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 |
| IP | 0 | 1 |
| IPMultiNICB | 0 | 1 |
| Mount | 0 | 0 |
| Process | 1 | 0 |
| WPAR | 0 | 1 |

For more information on using WPARs in your VCS environment, refer to the *Cluster Server Administrator's 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 175.
- Application agent. See "Application agent" on page 189.
- DiskGroup agent. See "DiskGroup agent" on page 19.
- Mount agent. See "Mount agent" on page 66.
- Process agent. See "Process agent" on page 207.
- WPAR agent. See "WPAR agent" on page 215.

# 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 85.

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

# VCS support for multi-pathing solutions

This section applies to LVMVG 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
- DiskGroup agent
- DiskGroupSnap agent
- Volume agent
- VolumeSet agent
- LVMVG agent
- Mount agent
- SFCache agent


## 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 28.

## 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 20.

For more information about using the DiskGroup agent with IMF, see See “Using the DiskGroup agent with IMF” on page 30.

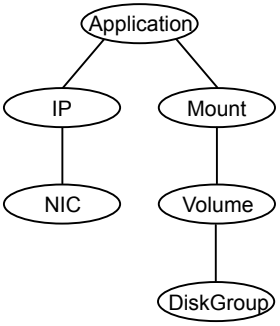
## Support for online migration for DiskGroup agent

VCS supports online migration of data from LVM volumes to VxVM volumes in a Veritas InfoScale environment. For more details, refer to the *Veritas InfoScale Solutions Guide*.

## 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:<br>■ license.vfd<br>Checks for valid Veritas Volume manager license–if one is not found use the vxlicinst utility to install a valid license key.<br>■ disk.vfd<br>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.<br>■ udid.vfd<br>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.<br>■ verifyplex.vfd<br>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.<br>■ volinuse<br>Checks if open volumes are in use or file systems on volumes that are mounted outside of VCS configuration.<br>See "High availability fire drill" on page 28. |
| 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.<br>Type and dimension: string-scalar |

**Table 2-2** Optional attributes

| Optional attributes | Description |
|---|---|
| StartVolumes | If the value of this attribute is 1, the DiskGroup online function starts all volumes belonging to that disk group after importing the group.<br>**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.<br>Type and dimension: boolean-scalar<br>Default: 1 |
| StopVolumes | If the value of this attribute is 1, the DiskGroup offline function stops all volumes belonging to that disk group before it deports the disk group.<br>Type and dimension: boolean-scalar<br>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:<br>■ 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.<br>■ The agent attempts to forcefully unmount the file systems to close the volumes.<br>Type and dimension: integer-scalar<br>Default: 0 |
| MonitorReservation | If the value of this attribute 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.<br>Type and dimension: boolean-scalar<br>Default: 0<br>**Note:** If the MonitorReservation attribute is set to 0, 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 program times out. A loss of storage connectivity can cause the disk group to become disabled. VxVM commands not responding properly can cause monitor program to timeout.<br>**Note:** System administrators may want to set a high value for FaultOnMonitorTimeout to increase system tolerance.<br>This attribute accepts the following values:<br>■ 0: Do not halt the system<br>■ 1: halt the system if either disk group goes into disabled state or the disk group resource faults due to monitor timeout<br>■ 2: halt the system if disk group goes into disabled state<br>■ 3: halt the system if disk group resource faults due to monitor timeout<br>If the value of the attribute is 0 and the disk group becomes disabled, the following occurs:<br>■ If the cluster has I/O fencing enabled, the DiskGroup resource is marked as FAULTED. This state results in the agent attempting to take the service group offline. As part of bringing the DiskGroup resource offline, the agent attempts to deport the disabled disk group. Even if disabled disk group fails to deport, the DiskGroup resource enters the FAULTED state. This state enables the failover of the service group that contains the resource. To fail back the DiskGroup resource, manually deport the disk group after restoring storage connectivity.<br>■ 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<br>(Continued) | **Note:** The PanicSystemOnDGLoss attribute does not depend on the MonitorReservation attribute.<br>**Note:** If PanicSystemOnDGLoss is set to non-zero value, the system panic is initiated using `sysdumpstart -p` command. This command reboots the system.<br>Type and dimension: integer-scalar<br>Default: 0 |
| Reservation | Determines if you want to enable SCSI-3 reservation. This attribute can have one of the following three values:<br>■ 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.<br>■ SCSI3—The disk group is imported with SCSI-3 reservation if the value of the cluster-level UseFence attribute is SCSI3.<br>■ NONE—The disk group is imported without SCSI-3 reservation.<br>To import a disk group with SCSI-3 reservations, ensure that the disks of the disk group are SCSI-3 persistent reservation (PR) compliant.<br>Type and dimension: string-scalar<br>Default: ClusterDefault<br>Example: "SCSI3" |

**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.<br>For more information about the '-c' option, refer to the *VxVM manual page*.<br>**Note:** For hardware cloning devices, do not set this attribute to 1.<br>Type and dimension: boolean-scalar<br>Default: 0 |

**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.<br>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.<br>Type and dimension: static integer-scalar<br>Default: 1<br>**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

The resource definition for this agent on AIX follows:

```
type DiskGroup (
static keylist SupportedActions = { "license.vfd", "disk.vfd", "udid.vfd",
"verifyplex.vfd", checkudid, numdisks, campusplex, volinuse,
joindg, splitdg, getvxvminfo }
```

```
static int OnlineRetryLimit = 1
static str ArgList[] = { DiskGroup, StartVolumes, StopVolumes, MonitorOnly,
MonitorReservation, tempUseFence, PanicSystemOnDGLoss, UmountVolumes,
Reservation, ConfidenceLevel, ClearClone }
static str IMFRegList[] = { DiskGroup, Reservation }
static int IMF{} = { Mode = 3, MonitorFreq = 5, RegisterRetryLimit = 3 }
str DiskGroup
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
)
```

# 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 Cluster Server with virtual SCSI devices
- 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.

### 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

You must set FC adapter tunables appropriately to avoid excessive waits for monitor timeouts. One FS adapter tunable is FC error recovery policy.

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 Cluster Server with virtual SCSI devices

VCS supports disk groups and volume groups created on virtual Small Computer System Interface (SCSI) devices. If you enable fencing and if you configure private disk groups created on virtual SCSI devices, the Reservation attribute of the DiskGroup resources must be set to NONE for these disk groups.

To set the Reservation attribute value to NONE, enter:

```
#hares -modify dg_res Reservation "NONE"
```

## 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 driver, enter:

```
# haremajor -v
```

For example:

```
# haremajor  -v
43
```

To determine the available major numbers on the system, enter:

```
# haremajor -a
```

For example:

```
# haremajor  -a
51...
```

To reassign the vxio driver number on the system to a specified number, enter:

```
# haremajor -s vxio-major-number
```

For example:

```
# haremajor  -s 51
vxio major number changed to 51
reboot the system so VxVM can restart its volumes using this major number
```

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.

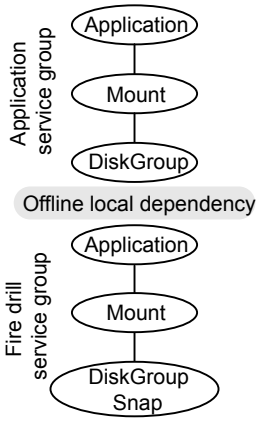
The DiskGroupSnap agent provides Gold and Bronze configurations for the fire drill, which can be specified using the agent's FDType attribute. The configuration decides the site to be detached from the DiskGroup for fire drill testing. The Gold configuration is the default option in which the agent selects 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 36.

## 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.<br>Type-dimension: string-scalar<br>Example: "dgres" |
| FDType | Specifies the configuration to be used for the fire drill. The possible values for this attribute are:<br>■ Bronze<br>■ Gold (default)<br>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.<br>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.<br>■ 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.<br>■ 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.<br>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 |

**Table 2-7** Internal attribute

| Internal 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.<br>Do not modify this attribute for this agent.<br>Setting this attribute to a higher value may result in agent function timeouts due to serialization of underlying commands.<br>Type and dimension: integer-scalar<br>Default: 1 |

# Notes for DiskGroupSnap agent

The DiskGroupSnap agent has the following notes:

- See "Fire drill configuration after upgrading VCS" on page 36.
- See "Configuring the SystemZones attribute for the fire drill service group" on page 36.
- See "Configuring the FireDrill service group" on page 37.
- See "Adding the ReuseMntPt attribute to the ArgList attribute for the Mount agent type" on page 37.
- See "Configuration considerations" on page 38.
- See "Agent limitations" on page 39.

## 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

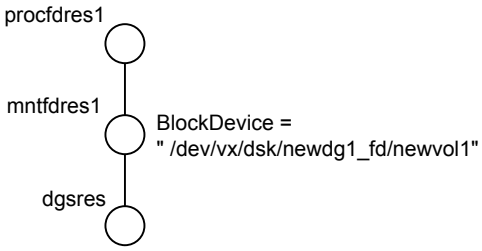
Keep the following recommendations in mind:

- 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 39. 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
```

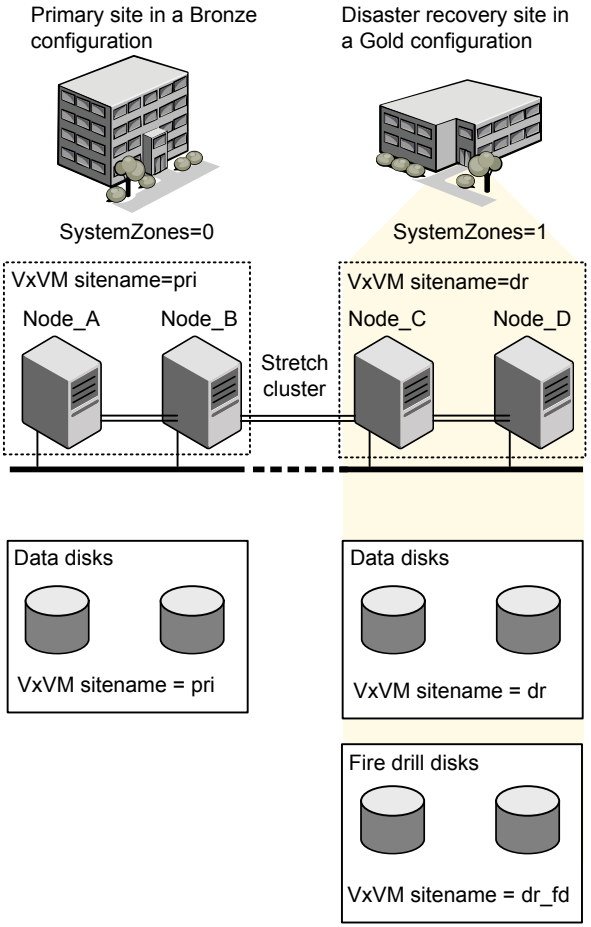
The Disaster Recovery site is in a Gold configuration as it has dedicated fire drill disks at the site. For the FDSiteName attribute, use the VxVM site tag given to the fire drill disks. For this configuration, the FDSiteName attribute values for the nodes at the Disaster recovery site follow:

```
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 = en0
        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 = en0
  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 = en0
       Address = "10.198.95.192"
       Netmask = "255.255.255.0"
        )

  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 = en0
                )

        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

# Volume agent

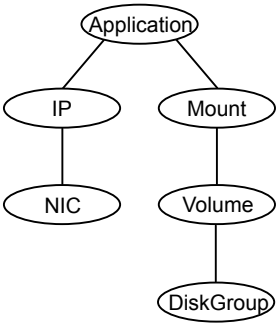
The Volume agent brings online, takes offline, and monitors a Veritas Volume Manager (VxVM) volume. Use the agent to make a volume highly available.

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

## Dependencies for Volume agent

Volume resources depend on DiskGroup resources.

**Figure 2-5** 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.<br>Type and dimension: string-scalar<br>Example: `"DG1"` |
| Volume | Name of the volume from disk group specified in DiskGroup attribute.<br>Type and dimension: string-scalar<br>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.<br>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.<br>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.<br>Veritas recommends that you should not modify this attribute.<br>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, DBG_3, DBG_5

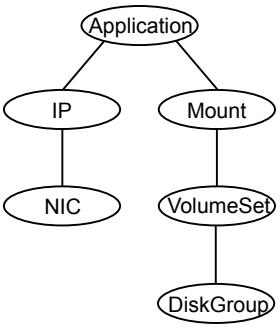
# VolumeSet agent

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 2-6** 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.<br>Type and dimension: string-scalar<br>Example: "DG1" |
| VolumeSet | The name of the volume set from the disk group that you specified in the DiskGroup attribute.<br>Type and dimension: string-scalar<br>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

# LVMVG agent

The LVMVG agent activates, deactivates, and monitors a Logical Volume Manager (LVM) volume group. The LVMVG agent supports JFS or JFS2. It does not support VxFS. This agent ensures that the ODM is in sync with changes to the volume group, specifically from the last time that the volume group was imported on the system. The LVMVG agent requires that date and time on all cluster nodes should be synchronized.

The LVMVG agent is also capable of ensuring high availability for AIX scalable volume group.

See "VCS support for multi-pathing solutions" on page 18.

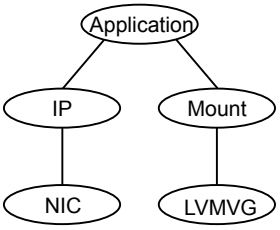
For important information on this agent, refer to:

Notes for LVMVG agent.

# Dependencies for LVMVG agent

No dependencies exist for the LVMVG resource.

**Figure 2-7** Sample service group for an LVMVG resource




# Agent functions for LVMVG agent

| | |
|---|---|
| Online | Activates the volume group. The Online agent function expects that the volume group is already imported on the system. If the volume group had been modified on a system where it was previously active, the online agent function detects the modification. It then syncs up the ODM on the system where you want to bring the volume group resource online. |
| Offline | Deactivates the volume group. |
| Monitor | Determines the volume group's state (activated or deactivated) and availability for read/write operations. |
| Clean | Terminates all ongoing resource actions and takes the resource offline, forcibly when necessary. |

| | |
|---|---|
| Action | Different action agent functions follow:<br>■ pv.vfd<br>Checks if all the disks in the volume group are visible on a host. If it fails, check if the path to disks exists from the host and check if LUN masking and zoning are set properly.<br>■ autoon.vfd<br>Checks if the flag to automatically activate volume group on system restart is set to yes. If it fails, set the "auto on" flag of volume group to "no".<br>■ volinuse<br>Checks if open volumes are in use or file systems on volumes that are mounted outside of VCS configuration.<br>■ updatepv<br>Updates the volume group's physical volumes (PV) information on all the other nodes in the cluster. You must run this action whenever there are changes in the disk configuration, such as addition, deletion, or replacement of physical volumes of the volume group. Running this action ensures that the other nodes are updated with the new information, which is used when the agent brings the volume group online. |

# State definitions for LVMVG agent

| | |
|---|---|
| ONLINE | Indicates that the volume group is activated. |
| OFFLINE | Indicates that the volume group is deactivated. |
| FAULTED | Indicates that the volume group has unexpectedly deactivated or deported or been disabled. |
| UNKNOWN | Indicates that a problem exists either with the configuration or the ability to determine the status of the resource. |

# Attributes for LVMVG agent

**Table 2-11** Required attributes for AIX

| Required attribute | Description |
|---|---|
| MajorNumber | Integer that represents the major number of the volume group. To ensure NFS functions properly, assign the same major number to the volume group on each system in the cluster.<br>Type and dimension: integer-scalar |
| NumThreads | The number of threads that are used within the agent process for managing resources. This number does not include the threads that are used for other internal purposes.<br>This resource type attribute is for internal use only. This value of this attribute must be set to 1.<br>Type and dimension: integer-scalar<br>Default: 1 |
| VolumeGroup | Name of the volume group that is configured with LVM.<br>Type and dimension: string-scalar<br>Example: "testvg1" |

**Table 2-12** Optional attributes

| Optional attribute | Description |
|---|---|
| GroupName | Attribute used to specify the volume's group.<br>If set, the groups's name is applied to the volume group and all of its logical volumes.<br>Type and dimension: string-scalar<br>Default: system |
| ImportvgOpt | Attribute used to specify options for the importvg command.<br>The default option, "n", indicates the volume group is not automatically activated when imported.<br>Type and dimension: string-scalar<br>Default: n |

**Table 2-12** Optional attributes *(continued)*

| Optional attribute | Description |
|---|---|
| Mode | Attribute used to specify permissions for a volume group and its logical volumes.<br>If set, these permissions are applied to the volume group and all of its logical volumes.<br>Type and dimension: string-scalar<br>Default: 640 |
| OwnerName | Attribute used to specify the volume owner's name.<br>If set, the owner's name is applied to the volume group and all of its logical volumes.<br>Type and dimension: string-scalar<br>Default: root |
| SyncODM | Integer that specifies whether or not the agent ensures that the ODM is in sync with any changes to the volume group.<br>If the value of this attribute is 1, the agent ensures that the ODM is in sync with the changes to the volume group in situations where the volume group was modified on another system in the cluster. The sync operation occurs on the system where the agent brings the volume group online.<br>If you run the updatepv action for the volume group, set the value of this attribute to 1 to ensure that the ODM entries are in sync with the changes in the volume group.<br>If the value of this attribute is 0, the changes to the volume group are independent of the ODM.<br>See "SyncODM Attribute" on page 61.<br>Type and dimension: integer-scalar<br>Default: 1 |
| VaryonvgOpt | Attribute used to specify options for the varyonvg command.<br>Type and dimension: string-scalar |
| ModePermSyncFlag | This attribute is deprecated. The agent uses an advanced algorithm to apply the Owner, Group and Permissions for the VG and its volumes. It does not use this attribute anymore. |

# Resource type definition for LVMVG agent

The resource type definition for this agent follows:

```
type LVMVG (
    static keylist SupportedActions = { "pv.vfd", numdisks,
"autoon.vfd", volinuse, updatepv }
static int NumThreads = 1
static str ArgList[] = { VolumeGroup, MajorNumber, OwnerName,
GroupName, Mode, ImportvgOpt, VaryonvgOpt, SyncODM,
ModePermSyncFlag }
str VolumeGroup
int MajorNumber
str OwnerName
str GroupName
str Mode
str ImportvgOpt = n
str VaryonvgOpt
int SyncODM = 1
int ModePermSyncFlag = 1
)
```

# Notes for LVMVG agent

The LVMVG agent for AIX has the following notes:

- Disks renamed after device renaming
- About the updatepv action
- LVMVG support in a VIO server environment
- Deactivation failure using the varyoffvg command on losing storage connectivity
- LVMVG Agent Supports JFS or JFS2
- Volume group needs to be imported
- Varyonvg options
- SyncODM Attribute
- Major Numbers
- Autoactivate Options
- LVMVG agent support for the Subsystem Device Driver (SDD)
- LVMVG agent support for the Hitachi's HiCommand Dynamic Link Manager (HDLM)

- LVMVG agent support for the EMC PowerPath
- The hadevice utility
- Removing a ghost disk from VxVM control

## Disks renamed after device renaming

If the disks used to create the volume group are renamed after device renaming, you need to export and re-import the volume group for the LVMVG type resources.

## About the updatepv action

The updatepv action updates the volume group's physical volumes (PV) information on all the other nodes in the cluster.

If a volume group's disk configuration is changed on a node, the updated information is not automatically propagated to other nodes of the cluster, and thus some or all of the ODM entries on those nodes become stale. This may cause the online operation of the volume group resource to fail.

You must run this action whenever there are changes in the disk configuration, such as addition, deletion, or replacement of physical volumes of the volume group. Running this action ensures that the other nodes are updated with the new information, which is used when the agent brings the volume group online. Use the following command to run this action.

```
# hares -action res_name updatepv -sys system_name
```

where system_name is the name of the node on which the disks were added to or removed from the volume group.

Running this action sends the updated physical volume information from the local node to all the other nodes. This information is stored in the file /var/VRTSvcs/log/tmp/resource_name.volume_group_name.pvid on all the nodes. If this file is present on that node, and if the SyncODM attribute is set to 1, then the online entry point uses the PVIDs from the file, exports the volume group, breaks the reservations on all these disks, and uses any one PVID to re-import the volume group. The file is deleted from that node after the volume group is successfully brought online on that node and the ODM is synchronized.

The SyncODM attribute must the set to 1 if the updatepv action has been executed for that volume group.

To ensure the high availability of the LVMVG resource, you must run the updatepv action immediately after adding, deleting, or replacing the physical volumes in the volume group. When the new disks are added or replaced in the volume group, these disks must be visible and have same PVID on all the cluster nodes.

You must run the updatepv action again for the nodes that were down when updatepv was last run.

---

**Note:** The updatepv action does not support the GCO environment.

---

## LVMVG support in a VIO server environment

The LVMVG agent supports volume groups created with virtual SCSI devices.

AIX and VIOS must be at the following required levels:

- The AIX operating system level must be AIX 6.1 TL5 or later and AIX 7.1 TL0 or later.
  For more information about supported AIX versions, refer to the *Veritas InfoScale Installation Guide*.
- The VIOS version must be VIOS 1.3 Fix Pack 8.1 or later.

## Deactivation failure using the varyoffvg command on losing storage connectivity

In certain circumstances, the varyoffvg command does not deactivate all the volume groups on a node. This failure can prevent the failback of the LVMVG resource.

In situations where storage connectivity is lost, the LVMVG resources fails over. Failback for the LVMVG resource requires the deactivation of the volume groups on the node that lost its connectivity to storage. VCS uses the varyoffvg command to deactivate the volume groups. The LVMVG resource cannot fail back, however, when deactivation is unsuccessful.

When the volume group loses its storage connectivity, the clean function executes the varyoffvg command. Deactivation using the varyoffvg command can fail, however, if the volume group is busy.

Criteria that can cause this failure can include:

- when the volume group has pending I/O operations, or
- when an application or upper-level resources in the resource dependency tree uses the volume group.

To overcome this deactivation failure, a post offline trigger has been added to issue the `varyoffvg` command. A side effect of the post offline trigger is that you must set the value of the OnlineRetryLimit attribute to 0.

**Following steps are performed to enable the lvmvg_postofline trigger:**

1. Set the POSTOFFLINE value in TriggersEnabled attribute of service group containing the LVMVG resource.
2. Install the lvmvg_postoffline trigger script from the sample triggers directory into the triggers directory:

   ```
   # cp /opt/VRTSvcs/bin/sample_triggers/VRTSvcs/lvmvg_postoffline
   /opt/VRTSvcs/bin/triggers/postoffline
   ```

   Change the file permissions to make it executable.

After the restoration of storage connectivity, you must ensure that the volume groups are deactivated on the node. You can then clear the fault on the resources. If you find active volume groups, deactivate them using the varyoffvg command.

The LVMVG resource must be the bottom-most resource in the resource dependency tree in the service group. A resource under the LVMVG resource can potentially fail to go offline if the volume group's deactivation fails.

## LVMVG Agent Supports JFS or JFS2

The LVMVG agent supports JFS and JFS2 file systems. It does not support VxFS.

## Volume group needs to be imported

The LVMVG agent relies on the ODM to find out the names of the disk devices that a volume group is created on. Unless a volume group is imported on the system, the ODM on that system does not contain any information about that volume group. Therefore, you must import the volume group on all the systems in the group's SystemList for the LVMVG agent to function properly.

For example, the volume groups (vg1 and vg2) must be imported on the specified systems (sysA and sysB).

See "Notes for LVMVG agent" on page 57.

## Varyonvg options

By default, the agent checks the state of the disk devices underneath the volume group. If the disk device is in a defined state, the agent resets it to an available state. You can use the VaryonvgOpt attribute to change this default behavior.

You can tell the agent not to check for the state of the disk devices. Set the VaryonvgOpt attribute in the main.cf file to a value of "u". This option to the `varyonvg` command ensures that the disks underneath the volume group are not reserved when the volume group is activated.

**Note:** When you activate a volume group with the "u" option, ghost disks are not created. Therefore, you do not have to reset disks for these volume groups.

## SyncODM Attribute

The LVMVG agent ensures that the ODM is in sync with any changes to the volume group since it was last imported on the system. This sync happens only if this attribute is set to 1. The agent uses the volume group's timestamp ODM entry to get the time when the volume group was last imported on the system.

The sync operation occurs when the timestamp value in the volume group's timestamp ODM entry is older than the time stamp value in the volume group's descriptor area. The timestamp value in the VGDA area of a volume group is updated after creating or deleting logical volumes, and adding or removing physical volumes. The sync operation also occurs if PVID file is found on the node and SyncODM attribute is set to 1. The PVID file is present if updatepv action has been executed. In this case the sync operation is performed without comparing the time stamp values.

## Major Numbers

If a file system on a volume group is shared for NFS, make sure that the volume group is imported with the same major number. The volume group is imported on all of the nodes in the cluster.

To view a list of available major numbers on the system, enter the lvlstmajor command. For example:

```
# lvlstmajor
  49, 60 ...
```

To import volume group vg00 with major number 60, enter:

```
# importvg -V 60 -y vg00 hdisk3
```

To view the major number that is assigned to a volume group, use the `ls` command with the -l option. For example:

```
# ls -l /dev/vg00
crw-r-----   1 root     system    60,  0 Apr  2 16:05 /dev/vg00
```

Assign the same major number to the volume group on each system in the cluster. Specify this major number in the MajorNumber attribute of the LVMVG configuration.

**Note:** Do not specify the V option in the ImportvgOpt attribute string, the agent specifies this option.

## Autoactivate Options

The "Concurrent Capable" options for the `importvg` and `mkvg` commands that are used with HACMP are not required for VCS. If an LVM volume group is placed under VCS control, the autoactivate options should be turned off. Do this using SMIT or through the command line.

From SMIT, set the following field values when creating or altering the volume group:

```
Activate volume group AUTOMATICALLY                 no
     at system restart?
 Create VG Concurrent Capable?                       no
 Auto-varyon in Concurrent Mode?                     no
```

From the command line, to view the current value for these fields, use the `lsattr` command.

For example:

```
# lsattr -El vg00
     vgserial_id  0001632f00004c00000000ee092b3bd8 N/A False
     auto_on      y                                N/A True
     conc_capable n                                N/A True
     conc_auto_on n                                N/A True
     timestamp    3ceff3390a8b1379                 N/A True
```

From the command line, to change the value for these fields, use the `chvg` command.

**To change the value of** `auto_on` **to n:**

1. Activate the volume group vg00 (if the volume group is not already activated):

```
# varyonvg vg00
```

2. Run the `chvg` command:

```
# chvg -a 'n' vg00
```

3. Verify the changes:

```
# lsattr -El vg00
    vgserial_id  0001632f00004c00000000ee092b3bd8 N/A False
    auto_on      n                                N/A True
    conc_capable n                                N/A True
    conc_auto_on n                                N/A True
    timestamp    3ceff3390a8b1379                 N/A True
```

## LVMVG agent support for the Subsystem Device Driver (SDD)

The LVMVG agent supports the IBM Multi-pathing SDD version 1.4.0.0 and later. If disks are under SDD control, create a volume group with vpath devices. Refer to the SDD Documentation for configuration and migration of volume groups.

SDD support requires the /usr/sbin/lquerypr command, which provides a set of persistent reserve functions. The lquerypr command tool comes with the SDD installation fileset.

## LVMVG agent support for the Hitachi's HiCommand Dynamic Link Manager (HDLM)

The LVMVG agent supports the Hitachi's HiCommand Dynamic Link Manager. For the details of the array and HDLM versions supported, refer to the HCL.

Note that if disks are under HDLM control, create a volume group with HDLM devices (dlmfdrv*n*). Refer to the HDLM documentation for configuration and migration of volume groups.

## LVMVG agent support for the EMC PowerPath

The LVMVG agent supports the EMC PowerPath. For the details of the array and PowerPath versions supported, refer to the HCL.

Note that if disks are under PowerPath control, create a volume group with PowerPath devices (hdiskpower*n*). Refer to the EMC PowerPath documentation for configuration and migration of volume groups.

## The hadevice utility

The LVMVG agent provides the hadevice utility. This utility checks the status of a disk device and resets a disk device to an available state. The utility then breaks any SCSI reservations on a disk device. Its syntax is:

```
# hadevice -c | -r | -b -p device_name
```

The five possible states of a disk device are: AVAILABLE, DEFINED AND RESERVED, DEFINED AND UNRESERVED, PERSISTENT RESERVATION, and AVAILABLE AND OPEN.

To check the state of a disk device, enter:

```
# hadevice -c device_name
```

The following commands locate and remove ghost disks for a disk device and break any SCSI reservation on the disk device. When the -p flag follows the -b flag, it breaks any previous SCSI reservation on the device. It then obtains and retains a new reservation on the device. For SDD (vpath) disks, ghost disks are not created. Both the -b and -r flags remove any persistent reservation and clear all reservation key registration on the device. The -p flag (retain reservation) is not applicable for SDD disks.

To break any SCSI reservations on the disk device, enter:

```
# hadevice -b device_name
```

To break any SCSI reservations on the disk device, and obtain and retain a new reservation on the device, enter:

```
# hadevice -b -p device_name
```

To locate and remove ghost disks, reset a disk device that is in a DEFINED state and put it into an AVAILABLE state, enter:

```
# hadevice -r device_name
```

## Removing a ghost disk from VxVM control

If VxVM 5.0 is installed, you may need to remove a ghost disk from VxVM control before using hadevice utility (except -r option).

If you check the ghost disk's status using the `hadevice -c hdisk#` command, you get an error. The error reads: `V-16-10011-10237 Error opening the device /dev/hdisk# (The file access permissions do not allow the specified action.)` Check if the ghost disk is under VxVM control. You can do this using the `vxdisk -eq list` command. If the disk is under VxVM control, remove it using the `vxdisk rm vxvm_disk_name`.

In this example, hdisk4 is a ghost disk.

```
sysA# vxdisk -eq list
 Disk_0          auto      -          -         LVM         disk0
 HDS9500-ALUA0_0 auto      -          -         error       hdisk4
 HDS9500-ALUA0_1 auto      -          -         online      hdisk2
 HDS9500-ALUA0_2 auto      -          -         online      hdisk3

 sysA# vxdisk rm HDS9500-ALUA0_0
```

## Sample configuration for LVMVG agent

The following is a sample configuration:

```
system sysA

system sysB

system sysC

group  lvmgroup (
    SystemList = { sysA, sysB }
    AutoStartList = { sysA }

LVMVG lvmvg_vg1 (
    VolumeGroup = vg1
    MajorNumber = 50
    )

LVMVG lvmvg_vg2 (
    VolumeGroup = vg2
    MajorNumber = 51
    ImportvgOpt = "f"
    )
```

## Debug log levels for LVMVG agent

The LVMVG agent uses the following debug log levels:

DBG_1, DBG_2, DBG_3, DBG_5

# 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 WPAR-aware. The ContainerOpts resource type attribute for this type has a default value of 0 for RunInContainer and a default value of 0 for PassCInfo. Veritas recommends that you do not change these values. Refer to the *Cluster Server Administrator's Guide* for more information.

This agent also supports high availability fire drills.

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 78.

## 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.

---

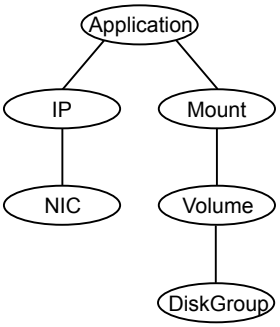
For more information about IMF-related Mount agent functions, see Agent functions for Mount agent.

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.

**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.<br><br>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.<br><br>**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.<br><br>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.<br><br>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:<br>`# df -k mount_point`<br>The output displays Mount resource information:<br>`Size Used Avail Use%`<br>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:<br>`# haconf -makerw`<br>`# hatype -modify Mount InfoInterval 60`<br>The command to retrieve information about the Mount resource is:<br>`# hares -value mountres ResourceInfo`<br>Output includes:<br>`Size 2097152`<br>`Used 139484`<br>`Available 1835332`<br>`Used% 8%` |

| | |
|---|---|
| Action | ■ chgmntlock<br>Resets the VxFS file system lock to a VCS-defined lock.<br>■ mountpoint.vfd<br>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.<br>■ mounted.vfd<br>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.<br>■ vxfslic.vfd<br>Checks for valid Veritas File System (VxFS) licenses. If it fails, you need to update the license for VxFS.<br>■ mountentry.vfd<br>Checks that the mount point is not listed in auto file system tables.<br>For example,<br>/etc/filesystems<br>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.<br>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.<br>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.<br><br>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 2-13** Required attributes

| Required attribute | Description |
|---|---|
| BlockDevice | Block device for mount point.<br><br>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.<br><br>**Note:** If the block device to be mounted is renamed after device renaming, you must update the value of the BlockDevice attribute for Mount type resources.<br><br>Type and dimension: string-scalar<br><br>Examples:<br><br>■ "/dev/vx/dsk/myvcs_dg/myvol"<br>■ IPv4 "10.209.70.90:/dirname/anotherdir"<br>■ IPv6 "[fe80::1:2:3]:/dirname/anotherdir" |

**Table 2-13** Required attributes *(continued)*

| Required attribute | Description |
|---|---|
| FsckOpt | Mandatory for the following file systems types:<br>■ jfs<br>■ jfs2<br>■ vxfs<br>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`, `-n`, or `-p`. The -p option is only for jfs or jfs2 file systems on AIX. Refer to the fsck manual page for more information.<br>For NFS mounts, the value of this attribute is not applicable and is ignored.<br>Type and dimension: string-scalar<br>Example: "-n"<br>Example: "-y"<br>**Note:** When you use the command line, add the % sign to escape '-'. For example: hares -modify MntRes FsckOpt %-y |
| FSType | Type of file system.<br>Supports jfs, jfs2, nfs, namefs, or vxfs.<br>Type and dimension: string-scalar<br>Example: "vxfs" |
| MountPoint | Directory for mount point<br>Type and dimension: string-scalar<br>Example: "/tmp/mnt" |

**Table 2-14** Optional attributes for AIX

| Optional attribute | Description |
|---|---|
| MountOpt | Options for the `mount` command. Refer to the `mount` manual page for more information.<br>Do not set the VxFS mount option "mntlock=*key*". The agent uses this option only when bringing a Mount resource online.<br>Type and dimension: string-scalar<br>Example: "rw" |
| SnapUmount | If the value of this attribute is 1, this attribute automatically unmounts VxFS snapshots when the file system is unmounted.<br>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.<br>Type and dimension: integer-scalar<br>Default: 0 |
| CkptUmount | If the value of this attribute is 1, this attribute automatically unmounts VxFS Storage Checkpoints when file system is unmounted.<br>If the value of this attribute is 0, and Storage Checkpoints are mounted, then failover does not occur.<br>Type and dimension: integer-scalar<br>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.<br>If any of these do not match the values that you have defined, a message is logged.<br>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.<br>Type and dimension: integer-scalar<br>Default: 0 |

**Table 2-14** Optional attributes for AIX *(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.<br><br>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.<br><br>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.<br><br>Type and dimension: integer-scalar<br><br>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.<br><br>Type and dimension: string-scalar<br><br>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.<br><br>Type and dimension: string-scalar<br><br>Example: "usr1" |

**Table 2-14** Optional attributes for AIX *(continued)*

| Optional attribute | Description |
|---|---|
| MntPtPermission | This attribute specifies the permissions of the mounted file system in an absolute format of a four-digit octal.<br>The agent verifies the mode of the mounted file system every monitor cycle if the value of the AccessPermissionChk attribute is not 0.<br>Type and dimension: string-scalar<br>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.<br>If the value of this attribute is 0 (default), the mount options are not checked.<br>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.<br>If the value of the attribute is 1, the state of the resource is unaffected.<br>If the value is 2, the state of the resource is set to offline.<br>If the value is 3, state of the resource is set to unknown.<br>Type and dimension: integer-scalar<br>Default: 0 |
| RecursiveMnt | If the value of this attribute is 1, VCS creates all the parent directories of the mount point if necessary.<br>Type and dimension: boolean-scalar<br>Default: 0 |

**Table 2-14** Optional attributes for AIX *(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.<br><br>To use this attribute, the cluster administrator needs to add this attribute to the ArgList resource type attribute of the agent. Set the appropriate group and resource dependencies such that only one resource can come online on a system at a time.<br><br>Type and dimension: integer-scalar<br><br>Default: 0 |

**Table 2-14** Optional attributes for AIX *(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.<br><br>This attribute can take three values: 0, 1, or 2.<br><br>VxFSMountLock=0<br><br>The resource does not detect any changes to the lock when VCS reports that it is online after you set the value to zero.<br><br>■ If the mount point is initially locked with the mntlock="VCS", the monitor agent function unlocks it.<br>■ If the mount point is initially locked with a key that is not equal to "VCS", the agent logs a message once.<br>■ If the mount point is initially not locked, no action is performed.<br><br>VxFSMountLock=1<br><br>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.<br><br>■ If the mount point is initially locked with the mntlock="VCS", no action is performed.<br>■ If the mount point is initially locked with a key that is not equal to "VCS", the agent logs a message once.<br>■ If the mount point is initially not locked, the monitor agent function locks it with the mntlock="VCS".<br><br>VxFSMountLock=2<br><br>When the value of the VxFSMountLock is 2, the file system is locked and the agent monitors any change to mntlock.<br><br>■ If the mount point is locked with the mntlock="VCS", no action is performed.<br>■ 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.<br>■ If the mount point is not locked, the agent locks it with the mntlock="VCS".<br><br>Type and dimension: integer-scalar<br><br>Default: 1 |

**Table 2-14** Optional attributes for AIX *(continued)*

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

**Table 2-15** 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 AIX 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
- Bringing a Mount resource online in the WPAR
- Selecting the attribute values for a Mount resource for the WPAR's root file system for NFS mounts
- Support for namefs file system
- Taking a group with the Mount resource offline can take several minutes if the file system is busy
- Example 1
- Example 2
- Example 3
- Enabling Level two monitoring for the Mount agent

## 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/filesystems 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.
- AIX defaults to NFSv3, which does not work across IPv6.
- Note that AIX's mount command refuses to accept IP addresses unless they are resolvable to a hostname.

## Bringing a Mount resource online in the WPAR

The Mount resource is brought online in the global environment by default (RunInContainer = 0).

If you want to bring a mount resource online inside the WPAR, perform the following:

- Make sure the resource is in a service group that has the ContainerInfo attribute configured.
- Override this attribute at the resource level.
- Set the value of the RunInContainer key to 1.

## Selecting the attribute values for a Mount resource for the WPAR's root file system for NFS mounts

For NFS mounts, you can run the SecondLevelMonitor in a container if you configure the following:

- RunInContainer = 0
- PassCInfo = 1
- Use the absolute path for the value of the MountPoint attribute for the Mount resource. The MountPoint attribute should not have the path relative to the WPAR root with this combination.
- Use a value of 1 for the SecondLevelMonitor attribute for the Mount resource.

The following are examples of relative and absolute paths:

- The file system is mounted on: /wpar/p1/mnt
- The MountPoint attribute's value absolute path: /wpar/p1/mnt

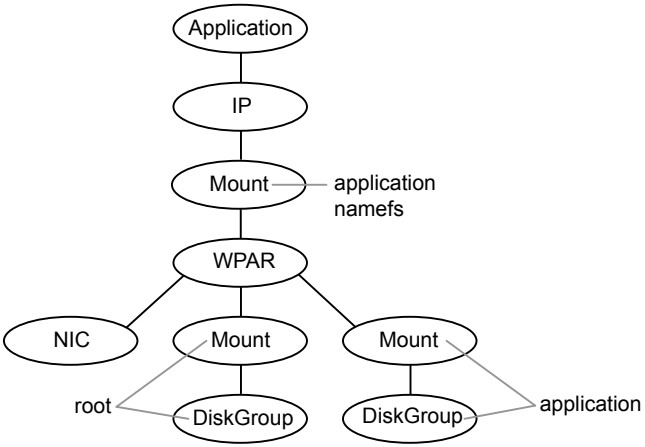
- The MountPoint attribute's value relative path to WPAR root: /mnt

For more information on the ContainerOpts resource attribute, and is RunInContainer and PassCInfo keys, refer to the *Cluster Server Administrator's Guide*.

## Support for namefs file system

The Mount agent provides namefs file system support. You can manage the namefs file system as a Mount resource. Use namefs support to mount a file system in the global environment and share it in the WPAR. For namefs support, configure the FSType attribute to use a value of namefs.

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




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

```
group namefssg (
    SystemList = { sysA = 0, sysB = 1 }
    ContainerInfo@sysA = { Name = wpar1, Type = WPAR, Enabled = 1 }
    ContainerInfo@sysB = { Name = wpar1, Type = WPAR, Enabled = 1 }
    )
    Mount namefs_mnt_global_to_local (
                MountPoint = "/wpars/wpar1/namefs_mnt"
                BlockDevice = "/mnt1/m1"
                FSType = namefs
                )
    WPAR w1 (
```

```
                )
        Mount base_mnt (
             MountPoint = "/mnt1"
             BlockDevice = "/dev/vx/dsk/tdg/tvol1"
             FSType = vxfs
             FsckOpt = "-y"
)
    namefs_mnt_global_to_local requires w1
    namefs_mnt_global_to_local requires base_mnt
```

## Taking a group with the Mount resource offline can take several minutes if the file system is busy

When a file system has heavy I/O, the `umount` command can take several minutes to respond. However, the `umount` command temporarily deletes the mount point from mount command output while processing. Per IBM, this is the expected and supported behavior on AIX. The `umount` command's processing later puts the mount point back if the mount point is found busy. Meanwhile, the default OfflineTimeout value of the Mount agent can get exceeded, which in turn invokes the Clean agent function. The Clean function can find the mount point's entry absent from the mount command output and exit with success.

The unmounting, however, may not have happened yet. If unmounting did not occur, offlining resources below the Mount resource (for example the LVMVG or DiskGroup resources) can fail.

The Mount resource's Offline agent function then proceeds to unmount the mount point. After several attempts, the Clean scripts that clean the resources below the Mount resource succeed and the group goes offline.

See the *Cluster Server Administrator's Guide* for more information about the OfflineTimeout attribute.

## Example 1

In this /etc/filesystems entry for a VxFS file system created on a VxVM volume, /mount_point is the mount point for the file system, /dev/vx/dsk/*Diskgroup_name*/*Volume_name* is the block device on which the file system is created, and `vxfs` is the file system type.

```
/etc/filesystems:
/mount_point:
    dev     = /dev/vx/dsk/Diskgroup_name/Volume_name
    vfs     = vxfs    mount    = false
    check   = false
```

## Example 2

In this /etc/filesystems entry for a JFS file system created on an LVM logical volume, /mount_point2 is the mount point for the file system, /dev/*LVMlogical_volume* is the block device on which the file system is created, /dev/*LVMlogical_volumelog* is the log device for the file system automatically created by the crfs command, and `jfs` is the file system type.

```
/etc/filesystems:
/mount_point2:
           dev    = /dev/LVMlogical_volume
           vfs    = jfs
           log    = /dev/LVMlogical_volumelog
           mount  = false
           check  = false
```

## Example 3

Use the crfs and mkfs commands to create file systems.

VCS supports the following configurations for the Mount agent:

- LVM volume group with a JFS or JFS2 file system.
- VxVM volume with a VxFS file system.

# 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*.

# Sample configurations for Mount agent

## Configuration 1 for Mount agent

In the following configuration, vg00 is a LVM volume group. The mount resource mnt requires the lvmvg_vg00 LVMVG resource.

```
LVMVG lvmvg_vg00 (
 VolumeGroup = vg00
 MajorNumber = 50
)
Mount mnt (
 MountPoint = "/lvm_testmnt"
 BlockDevice = "/dev/lv00"
 FSType = jfs
 FsckOpt = "-p"
)

mnt requires lvmvg_vg00
```

## Configuration 2 for Mount agent

In the following configuration, vol0 is a volume in diskgroup testdg_1 created with VxVM. Mount resource m0 requires the dg1 diskgroup resource.

```
DiskGroup  dg1 (
    DiskGroup = testdg_1
)

Mount m0 (
 MountPoint = "/tmp/m0"
 BlockDevice = "/dev/vx/dsk/testdg_1/vol0"
 FSType = vxfs
 FsckOpt = "-y"
)

m0 requires dg1
```

## Configuration 3 for AIX for Mount agent

Configuration 3 for AIX follows:

In the following configuration, sysA is the remote NFS server and /home/xyz is the remote directory.

```
Mount mnt3 (
    MountPoint = "/tmp/m1"
    BlockDevice = "sysA:/home/xyz"
    FSType = nfs
)
```

## 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

# 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 2-10 shows sample SFCache resource dependency for VxFS caching. The SFCache resource can depend on the Mount or CFSMount resource.

**Figure 2-10** Sample SFCache resource dependency (VxFS caching)

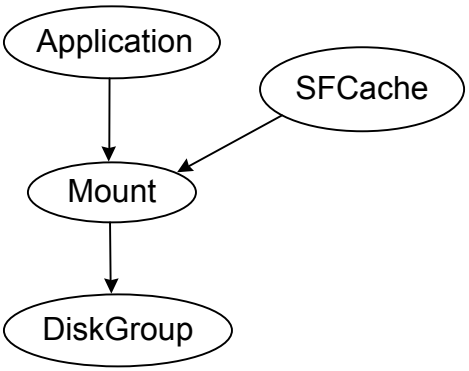
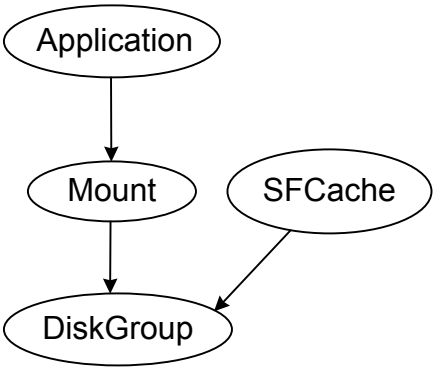



Figure 2-11 shows sample SFCache resource dependency for VxVM caching. The SFCache resource can depend on the DiskGroup, Volume, VolumeSet, or CVMVolDg resource.

**Figure 2-11** 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:<br>■ VxFS<br>■ 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 2-16** Required attributes

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

**Table 2-16** Required attributes *(continued)*

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

**Table 2-17** Optional attributes

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

**Table 2-18** 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.<br><br>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.<br><br>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.<br><br>**Note:** If the CacheFaultPolicy attribute is set to FAILOVER, Veritas recommends to set the FaultOnMonitorTimeouts attribute to 4.<br><br>Default: 0 |
| 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.<br><br>**Note:** If the NumThreads value is greater than 1, then there is a possibility that the `sfcache` command may fail.<br><br>Type and dimension: static integer-scalar<br><br>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
- IP agent
- NIC agent
- IPMultiNIC agent
- MultiNICA agent
- About the IPMultiNICB and MultiNICB agents
- IPMultiNICB agent
- MultiNICB agent
- DNS agent


## About the network agents

Use network agents to provide high availability for networking resources.

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

All these agents also support EtherChannel.

### 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

## IPMultiNIC and MultiNICA agents

The IPMultiNIC and MultiNICA agents:

- Monitor single or multiple NICs
- Check the backup NICs at fail over
- Use the original base IP address when failing over
- Provide slower failover compared to MultiNICB but can function with fewer IP addresses
- Have only one active NIC at a time

## 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
- Provide faster fail over compared to MultiNICA but require more IP addresses
- Have more than one active NIC at a time

## 802.1Q trunking

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

To use 802.1Q trunking, create 802.1Q trunked interfaces over a physical interface using the SMIT. The physical interface is connected to a 802.1Q trunked port on the switch.

The NIC, MultiNICA, and MultiNICB agents can monitor these trunked interfaces. The IP, IPMultiNIC, and IPMultiNICB agents monitor the virtual IP addresses that are configured on these interfaces.

For example, create a 802.1Q interface called en6 over a physical interface called en0. Do not configure an IP address on en0. You connect en0 to a trunked port on the switch. The NIC and IP agents can then monitor en6 and the virtual IP address configured on en6.

### EtherChannel support

EtherChannel aggregates multiple network interfaces so that they appear as a single interface. The IP/NIC, IPMultiNIC/MultiNICA, and IPMultiNICB/MultiNICB agents support EtherChannel use with VCS. For example you can combine en0 and en1 into an EtherChannel and call the combined interface en2. You then use the NIC (or MultiNICA or MultiNICB) agent to monitor this en2 interface. You use the IP (or IPMultiNIC or IPMultiNICB) agent to configure and monitor an IP address on the en2 interface. Note that you use the en2 interface configured through EtherChannel for the Device attribute. EtherChannel is responsible for providing local adapter swapping, which is outside of VCS control. EtherChannel backup and active-active modes are supported.

# 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 WPAR-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 *Cluster Server Administrator's Guide* for more information.

## 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

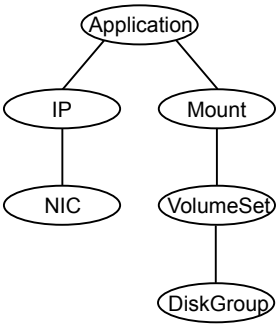
For more information about using the high availability fire drill see the *Cluster Server Administrator's Guide*.

## Dependencies for IP agent

IP resources depend on NIC resources.

They can also depend on WPAR resources.

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




## Agent functions for IP agent

| | |
|---|---|
| Online | Uses the `ifconfig` command to set the IP address as an alias on the interface. |
| Action | ■ route.vfd<br>Checks for the existence of a route to the IP from the specified NIC.<br>■ device.vfd<br>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 different from the base IP address, and 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.<br>Type and dimension: string-scalar<br>Example:<br>IPv4: "192.203.47.61"<br>IPv6: "2001::10" |
| Device | The name of the NIC device that is associated with the IP address.<br>When a network interface or a network adapter of the type IP under VCS control is renamed, you must update the value of the Device attribute of the IP resource.<br>**Note:** Veritas recommends to offline the service groups containing the network resources before renaming the network interfaces and adapters and to update the VCS configuration to avoid any undesired behaviour.<br>Type and dimension: string-scalar<br>Example: "en0" |
| One of the following attributes:<br>■ NetMask: Mandatory only if you configure an IPv4 address.<br>■ PrefixLen: Mandatory only if you configure an IPv6 address. | |
| NetMask | For IPv4 protocol, the subnet mask that is associated with the IP address.<br>Type and dimension: string-scalar<br>"255.255.255.0" |

**Table 3-1** Required attributes *(continued)*

| Required attribute | Description |
|---|---|
| PrefixLen | Prefix for the IPv6 address represented as the CIDR value.<br>When you use the IPv6 protocol, you must configure values for this attribute and the corresponding NIC resource's Device and Protocol attributes.<br>Type-dimension: integer-scalar<br>Range: 1 - 128<br>Example: 64 |

**Table 3-2** Optional attributes

| Optional attribute | Description |
|---|---|
| Options | Options for the `ifconfig` command.<br>For complete list of ifconfig options refer to `ifconfig` manpage.<br>Type and dimension: string-scalar<br>Example: "metric 4 mtu 1400" |

**Table 3-2** Optional attributes *(continued)*

| 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`*".<br>RouteOptions for an IP resource configured inside WPAR is formed as "`[attribute=value ...]`" pairs.<br>For details about the `route` command, refer to the man page for your operating system.<br>When the value of this string is null, the agent does not add routes.<br>Type and dimension: string-scalar<br>Example 1: "192.100.201.0 192.100.13.7"<br>In this example, the agent executes the following command when it configures an interface:<br>`"route add 192.100.201.0 192.100.13.7"`<br>Example 2: "rtdest=10.209.60.57 rtgateway=10.209.60.55 rttype=host"<br>In this example, the agent executes the following command to configure route for an IP address in a WPAR:<br>`"chwpar -i -I rtdest=10.209.60.57 rtgateway=10.209.60.55 rttype=host wpar_name"` |

# Resource type definition for IP agent

The resource definition for this agent on AIX follows:

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

```
        int PrefixLen
)
```

## Sample configurations for IP agent

The sample configurations for this agent follow:

### NetMask in decimal (base 10)

Configuration with decimal NetMask:

```
IP ipres (
Device = en0
Address = "192.203.47.61"
NetMask = "255.255.248.0"
)
```

### NetMask in hexadecimal (base 16)

Configuration with hexadecimal NetMask:

```
IP ipres (
Device = en0
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_2, DBG_4

# 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.

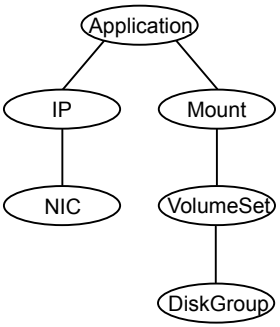
This agent is compatible with AIX WPARs. The ContainerOpts resource type attribute is not specified for this type. Veritas recommends that you do not change the values for the ContainerOpts keys.

Refer to the *Cluster Server Administrator's 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.

  **Note:** For AIX, the systems do not respond to broadcast pings by default. Run the `no -o bcastping=1` command to enable response to broadcast pings.

| | |
|---|---|
| Action | ■ device.vfd<br>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 | Specifies the name of the NIC that you want to monitor.<br>Use the `lsdev` command to check for all available network adapters.<br>When a network interface or a network adapter of the type NIC under VCS control is renamed, you must update the value of the Device attribute of the NIC resource.<br>**Note:** Veritas recommends to offline the service groups containing the network resources before renaming the network interfaces and adapters and to update the VCS configuration to avoid any undesired behaviour.<br>Type and dimension: string-scalar<br>Example: "en0" |
| NetworkHosts | Required for virtual devices.<br>See Table 3-4 on page 102. |
| Protocol | Required to use the IPv6 protocol.<br>See Table 3-4 on page 102. |

**Table 3-4** Optional attributes

| Optional attribute | Description |
|---|---|
| NetworkHosts | List of hosts on the same network that are pinged to determine if the network connection is alive. Enter the IP address of the host, instead of the host name, to prevent the monitor from timing out. DNS lookup 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 reachable.<br>If you do not specify network hosts, the monitor tests the NIC by sending pings to the broadcast address on the NIC.<br>For a virtual device, you must configure the NetworkHosts attribute. Veritas recommends configuring more than one host to take care of the NetworkHost itself failing.<br>Type and dimension: string-vector<br>Example: { "166.96.15.22", "166.97.1.2" } |
| NetworkType | Specifies the type of network.<br>Type and Dimension: string-scalar<br>Example: "ether" |
| PingOptimize | Determines whether to ping every monitor cycle.<br>A value of 0 means that the agent pings either the network hosts or the broadcast address every monitor cycle. It pings each cycle to determine the state of the network interface.<br>A value of 1 means that the agent uses the device statistics from the netstat output to determine the state of the interface. If no activity exists on the interface, the agent then pings the broadcast address to double-check the state of the network interface.<br>Type and dimension: integer-scalar<br>Default: 1 |
| Protocol | Specifies the type of IP protocol (IPv4 or IPv6) that you want to use with the agent.<br>When you use the IPv6 protocol, you must configure values for this attribute, the Device attribute, and the corresponding IP resource's PrefixLen attribute.<br>Type-dimension: string-scalar<br>Default: IPv4<br>Example: IPv6 |

# Resource type definition for NIC agent

The resource definition for this agent on AIX follows:

```
type NIC (
static keylist SupportedActions = { "device.vfd" }
static int OfflineMonitorInterval = 60
static str ArgList[] = { Device, Protocol, PingOptimize,
NetworkHosts, NetworkType }
static int ContainerOpts{} = { RunInContainer=0, PassCInfo=0 }
static str Operations = None
str Device
str Protocol = "ipv4"
int PingOptimize = 1
str NetworkType
str NetworkHosts[]
)
```

# Notes for the NIC agent

The NIC agent has the following notes:

- High availability fire drill for NIC agent

## 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*.

# Sample configurations for NIC agent

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

```
NIC nicres (
Device = en0
PingOptimize = 0
)
```

### Configuration with network hosts for NIC agent

```
NIC nicres (
    Device = en0
    NetworkHosts = { "10.182.1.1", "10.182.1.2" }
    )
```

## 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 = en0
Device@sysB = en1
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" }
Protocol = IPv6
)
Phantom phantom_resource (
)
group ip_group (
SystemList = { sysA = 0, sysB = 1 }
)
IP ip_resource (
Device@sysA = en0
Device@sysB = en1
Address = "2001:db8:c18:2:214:4fff:fe96:102"
PrefixLen = 64
Protocol = IPv6
)
Proxy proxy_resource (
TargetResName = nic_resource
)
ip_resource requires proxy_resource
```

## Debug log levels for NIC agent

The NIC agent uses the following debug log levels:

DBG_1, DBG_2

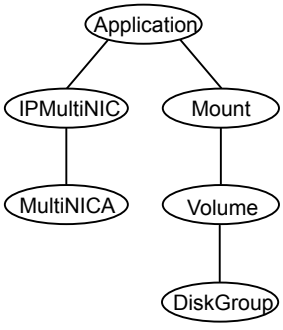
# IPMultiNIC agent

The IPMultiNIC agent manages the virtual IP address that is configured as an alias on one interface of a MultiNICA resource. If the interface faults, the agent works with the MultiNICA resource to fail over the virtual IP to a backup interface. If multiple service groups have IPMultiNIC resources associated with the same MultiNICA resource, only one group must have the MultiNICA resource. The other groups have Proxy resources pointing to it. You can use this agent for IP addresses on multiple-adapter systems.

## Dependencies for IPMultiNIC agent

IPMultiNIC resources depend on MultiNICA resources. They can also depend on WPAR resources.

**Figure 3-3** Sample service group that includes an IPMultiNIC resource




## Agent functions for IPMultiNIC agent

| | |
|---|---|
| Online | Configures a virtual IP address on the active interface of the MultiNICA resource. Also sends out a gratuitous ARP. |
| Offline | Removes the virtual IP address from the active interface of the MultiNICA resource. |
| Monitor | Checks if the virtual IP address is configured on one interface of the MultiNICA resource. |

# State definitions for IPMultiNIC agent

The state definitions for this agent follow:

| | |
|---|---|
| ONLINE | Indicates that the specified IP address is assigned to one of the interfaces specified in the corresponding MultiNICA resource. |
| OFFLINE | Indicates that the specified IP address is not assigned to any interface of the MultiNICA resource. |
| UNKNOWN | Indicates that the agent can not determine the state of the resource. This state may be due to an incorrect configuration. |
| FAULTED | Indicates that the IP address could not be brought online, usually because all the NICs in the MultiNICA resource are faulted or the IP address was removed out of VCS control. |

# Attributes for IPMultiNIC agent

**Table 3-5** Required attributes

| Required attribute | Description |
|---|---|
| Address | The virtual IP address that is assigned to the active interface.<br>Type and dimension: string-scalar<br>Example:<br>IPv4: "10.128.10.14"<br>IPv6: "2001:DB8::" |
| MultiNICAResName | Name of the associated MultiNICA resource that determines the active interface.<br>Type and dimension: string-scalar<br>Example: "MultiNICA_res1" |
| One of the following attributes:<br>■ NetMask: Mandatory only if you configure an IPv4 address.<br>■ PrefixLen: Mandatory only if you configure an IPv6 address. | |
| NetMask | The IPv4 protocol netmask for the virtual IP address.<br>Required to use the IPv4 protocol.<br>Type and dimension: string-scalar<br>Example: "255.255.255.0" |

**Table 3-5** Required attributes *(continued)*

| Required attribute | Description |
|---|---|
| PrefixLen | Specifies the prefix for the IPv6 address represented as the CIDR value.<br>When you use the IPv6 protocol, you must configure values for this attribute and the corresponding MultiNICA agent's Device, Protocol and PrefixLen attributes.<br>Type-dimension: integer-scalar<br>Range: 1 - 128<br>Example: 64 |

**Table 3-6** Optional attributes

| Optional attribute | Description |
|---|---|
| Options | The `ifconfig` command options for the virtual IP address.<br>Type and dimension: string-scalar<br>Example: "mtu 2000" |

**Note:** The default value of the ToleranceLimit static attribute is 3. A value higher than zero helps to prevent the IPMultiNIC agent from performing a failover of the virtual IP address to another system before the MultiNICA agent does a local failover of the virtual IP address.

## Resource type definition for IPMultiNIC agent

The resource definition for this agent on AIX follows:

```
type IPMultiNIC (
static str ArgList[] = { "MultiNICAResName:Device", Address,
NetMask, Options, "MultiNICAResName:Probed",
"MultiNICAResName:Protocol", MultiNICAResName, PrefixLen }
static int MonitorTimeout = 120
static int ToleranceLimit = 3
str Address
str NetMask
str Options
int PrefixLen
```

```
str MultiNICAResName
)
```

# Sample configuration: IPMultiNIC and MultiNICA

Refer to the MultiNICA agent for more information.

## Configuration for IPMultiNIC agent

Configuration for this agent on AIX follows:

```
group grp1 (
SystemList = { sysa = 0, sysb = 1 }
AutoStartList = { sysa }
)
MultiNICA mnic (
Device@sysa = { en0 = "10.128.8.42", en1 = "10.128.8.42" }
Device@sysb = { en0 = "10.128.8.43", en1 = "10.128.8.43" }

NetMask = "255.255.255.0"
Gateway = "10.128.1.1"
BroadcastAddr = "10.128.8.255"
)
IPMultiNIC ip1 (
Address = "10.128.10.14"
NetMask = "255.255.255.0"
MultiNICAResName = mnic
)

ip1 requires mnic
group grp2 (
SystemList = { sysa = 0, sysb = 1 }
AutoStartList = { sysa }
)
IPMultiNIC ip2 (
Address = "10.128.9.4"
NetMask = "255.255.255.0"
MultiNICAResName = mnic
Options = "mtu 1500"
)
Proxy proxy (
TargetResName = mnic
)
ip2 requires proxy
```

## Debug log levels

The IPMultiNIC agent uses the following debug log levels:

DBG_1, DBG_2, DBG_4, DBG_5

# MultiNICA agent

The MultiNICA agent represents a set of network interfaces and provides failover capabilities amongst them. You can use the agent to make IP addresses on multiple-adapter systems highly available or to monitor them. Each interface in a MultiNICA resource has a base IP address. You can use one base IP address for all interfaces, or you can specify a different IP address for use with each interface. The MultiNICA agent configures one interface at a time. If it does not detect activity on the configured interface, it configures a new interface and migrates IP aliases to it.

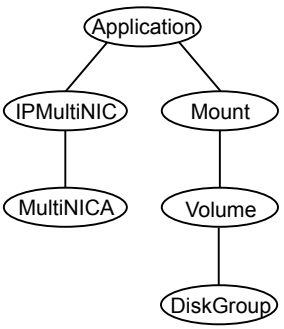
If an interface is associated with a MultiNICA resource, do not associate it with any other MultiNICA, MultiNICB, or NIC resource. If the same set of interfaces must be a part of multiple service groups, configure a MultiNICA resource in one of the service groups. In the other service groups, configure Proxy resources that point to the MultiNICA resource in the first service group.

For more information on this agent, refer to See “Notes for MultiNICA agent” on page 114.

## Dependencies for MultiNICA agent

The MultiNICA resource does not depend on any other resources.

**Figure 3-4** Sample service group that includes a MultiNICA resource



# Agent function for MultiNICA agent

| | |
|---|---|
| Monitor | Checks the status of the active interface. If the agent detects a failure, it tries to migrate the IP addresses that are configured on that interface. If possible, it tries to migrate the addresses to the next available interface that is configured in the Device attribute. |

**Note:** Systems do not respond to broadcast pings by default. You must run `"no -o bcastping=1"` to enable response to broadcast pings.

# State definitions for MultiNICA agent

The state definitions for this agent follow:

| | |
|---|---|
| ONLINE | Indicates that one or more of the network interfaces listed in the Device attribute of the resource is in working condition. |
| FAULTED | Indicates that all of the network interfaces listed in the Device attribute failed. |
| UNKNOWN | Indicates that the agent cannot determine the state of the network interfaces that are specified in the Device attribute. This state may be due to incorrect configuration. |

# Attributes for MultiNICA agent

**Table 3-7** Required attributes

| Required attribute | Description |
|---|---|
| BroadcastAddr | Broadcast address<br>Type and dimension: string-scalar<br>Example: "10.192.15.255" |

**Table 3-7** Required attributes *(continued)*

| Required attribute | Description |
|---|---|
| Device | List of interfaces and their base IP addresses.<br>When a network interface or a network adapter of the type MultiNICA under VCS control is renamed, you must update the value of the Device attribute of the MultiNICA resource.<br>**Note:** Veritas recommends to offline the service groups containing the network resources before renaming the network interfaces and adapters and to update the VCS configuration to avoid any undesired behaviour.<br>For each system you must localize the attribute with a separate base IP address.<br>Type and dimension: string-association<br>Example: { en0 = "10.128.8.42", en1 = "10.128.8.42" } |
| Gateway | IP address for the default gateway.<br>Type and dimension: string-scalar<br>Example: "10.192.1.7" |
| One of the following attributes:<br>■ NetMask: Mandatory only if you configure an IPv4 address.<br>■ PrefixLen: Mandatory only if you configure an IPv6 address. | |
| NetMask | Netmask for the base IP address.<br>Required to use the IPv4 protocol.<br>Type and dimension: string-scalar<br>Example: "255.255.255.0"<br>This attribute is required when the resource is configured for the IPv4 protocol. |
| PrefixLen | Specifies the prefix for the IPv6 address represented as the CIDR value.<br>When you use the IPv6 protocol, you must configure values for this attribute, the Device, Protocol attributes, and the corresponding IPMultiNIC resources' PrefixLen attributes.<br>Type-dimension: integer-scalar<br>Range: 1 - 128<br>Example: 64 |

**Table 3-7** Required attributes *(continued)*

| Required attribute | Description |
|---|---|
| Protocol | Required to use the IPv6 protocol.<br>See Table 3-8 on page 112. |

**Table 3-8** Optional attributes

| Optional attribute | Description |
|---|---|
| HandshakeInterval | Specifies the maximum number of tries the agent makes either to:<br>■ ping a host (listed in the NetworkHosts attribute) before it fails over to a new interface, or<br>■ ping the default broadcast address (depending on the attribute configured) before it fails over to a new interface.<br>To prevent spurious failovers, the agent must try to contact a host on the network several times before it marks an interface as FAULTED. Increased values result in longer failover times, whether between the interface or from system to system in the case of FAULTED interfaces.<br>Type and dimension: integer-scalar<br>Default: 1 |
| NetworkHosts | The list of hosts on the network that are pinged to determine if the network connection is alive. Enter the IP address of the host, instead of the host name, to prevent the monitor from timing out. DNS lookup causes the ping to hang. If this attribute is unspecified, the monitor tests the NIC by pinging the broadcast address on the interface. If more than one network host is listed, the monitor returns online if at least one of the hosts is alive.<br>Type and dimension: string-vector<br>Example: {"128.93.2.1", "128.97.1.2"} |
| Options | The `ifconfig` command options for the base IP address.<br>Type and dimension: string-scalar<br>Example: "metric 4 mtu 1400" |

**Table 3-8** Optional attributes *(continued)*

| Optional attribute | Description |
|---|---|
| PingOptimize | Determines whether to ping every monitor cycle.<br>A value of 0 means that the agent pings either the network hosts or the broadcast address every monitor cycle. It pings every cycle to determine the state of the network interface.<br>A value of 1 means that the agent uses the device statistics from the netstat output to determine the state of the interface. If no activity exists on the interface, the agent then pings the broadcast address to double-check the state of the network interface.<br>Type and dimension: integer-scalar<br>Default: 1 |
| 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`*".<br>For details about the `route` command, refer to the man page for your operating system.<br>When the value of this string is null, the agent does not add routes.<br>Type and dimension: string-scalar<br>Example: "192.100.201.0 192.100.13.7" |
| FailoverInProgress | For internal use only. |
| Protocol | Specifies the type of IP protocol (IPv4 or IPv6) that you want to use with the agent.<br>When you use the IPv6 protocol, you must configure values for this attribute, the Device attribute, and the corresponding IPMultiNIC resources' PrefixLen attributes.<br>Type-dimension: string-scalar<br>Default: IPv4<br>Example: IPv6 |

# Resource type definition for MultiNICA agent

The resource definition for this agent on AIX follows:

```
type MultiNICA (
static int OfflineMonitorInterval = 60
```

```
static int MonitorTimeout = 300
static str ArgList[] = { Device, NetMask, Gateway,
BroadcastAddr, Options, RouteOptions, PingOptimize,
MonitorOnly, HandshakeInterval, NetworkHosts, PrefixLen,
Protocol }
static str Operations = None
str Device{}
str NetMask
str Gateway
str BroadcastAddr
str Options
str RouteOptions
int PingOptimize = 1
int HandshakeInterval = 1
int PrefixLen
str Protocol = "ipv4"
str NetworkHosts[]
temp boolean FailoverInProgress = 0
)
```

## Notes for MultiNICA agent

- If all interfaces configured in the Device attribute are down, the MultiNICA agent faults the resource after a two-three minute interval. This delay occurs because the MultiNICA agent tests the failed interface several times before it marks the resource OFFLINE. Engine logs record a detailed description of the events during a failover.
- For a single main.cf configuration file, you can only have one MultiNICA resource, which uses either the IPv4 or the IPv6 protocol for a given set of devices. For example, you can have a MultiNICA resource configured as follows:

  ```
  MultiNICA mnic (
      Device@sysa = { en0 = "10.128.8.42", en1 = "10.128.8.42" }
      Device@sysb = { en0 = "10.128.8.43", en1 = "10.128.8.43" }
  ```

- The MultiNICA agent supports only one active interface on one IP subnet; the agent does not work with multiple active interfaces on the same subnet.
  For example, you have two active interfaces, en0 (10.128.2.5) and en1 (10.128.2.8). You configure a third interface, en2, as the backup interface to en1. The agent does not fail over from en1 to en2 because some ping tests are redirected through en0 on the same subnet. The redirect makes the MultiNICA monitor return an online status.

# Sample configurations for MultiNICA agent

The sample configurations for this agent follow:

## MultiNICA and IPMultiNIC

In the following example, two systems, sysa and sysb, each have a pair of network interfaces, en0 and en1. In this example, the two interfaces, en0 and en1, have the same base, or physical, IP address. Note the lines beginning Device@sysa and Device@sysb; the use of different physical addresses shows how to localize an attribute for a particular host.

The MultiNICA resource fails over the IP addresses to the backup interface in the event of a failure of the active interface. The resources ip1 and ip2, shown in the following example, have the Address attribute that contains the virtual IP address. In the event of an interface failure on sysa, the physical IP address and the two virtual IP addresses fail over from en0 to en1.

However, if both the interfaces on sysa are disconnected, the MultiNICA and IPMultiNIC resources work in tandem to fault the group on sysa. The entire group now fails over to sysb.

If you have more than one group using the MultiNICA resource, the other groups can use a Proxy resource. The Proxy resource points to the MultiNICA resource in the first group. The Proxy resource prevents redundant monitoring of the interfaces on the same system. The IPMultiNIC resource is always made dependent on the MultiNICA resource.

```
group grp1 (
SystemList = { sysa = 0 , sysb = 1 }
AutoStartList = { sysa }
)
MultiNICA mnic (
Device@sysa = { en0 = "10.128.8.42", en1 = "10.128.8.42" }
Device@sysb = { en0 = "10.128.8.43", en1 = "10.128.8.43" }
NetMask = "255.255.0.0"
Gateway = "10.128.1.1"
BroadcastAddr = "10.128.255.255"
Options = "mtu 1500"
)
IPMultiNIC ip1 (
Address = "10.128.10.14"
NetMask = "255.255.0.0"
MultiNICAResName = mnic
Options = "mtu 1500"
)
```

```
ip1 requires mnic
group grp2 (
SystemList = { sysa = 0 , sysb = 1 }
AutoStartList = { sysa }
)
IPMultiNIC ip2 (
Address = "10.128.9.4"
NetMask = "255.255.0.0"
MultiNICAResName = mnic
Options = "mtu 1500"
)
Proxy proxy (
TargetResName = mnic
)
ip2 requires proxy
```

### IPv6 configuration for MultiNICA agent

The following is a basic configuration for IPv6.

```
group mnica_group (
SystemList = { sysA = 0, sysB = 1 }
)
IPMultiNIC ipmnic_res (
Address = "2007:192::1627:161"
MultiNICAResName = mnica_res
PrefixLen = 64
)
MultiNICA mnica_res (
Device@sysA = { en0 = "fe80::214:4fff:fe96:ae0a",
en1 = "fe80::214:4fff:fe96:ae0a" }
Device@sysB = { en0 = "fe80::214:4fff:fe98:aeFb",
en1 = "fe80::214:4fff:fe98:aeFb" }
PrefixLen = 64
Protocol = ipv6
)
ipmnic_res requires mnica_res
```

## Debug log levels for MultiNICA agent

The MultiNICA agent uses the following debug log levels:

DBG_2, DBG_3, DBG_4, 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.
- 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.
- If you specify the NetworkHosts attribute, then that host must be on the same subnet as the base IP addresses for the MultiNICB resource.
- If any network host is meant to respond to a broadcast ping, run `no -o bcastping=1` on the network host.
- You must use the AIX SMIT configuration tool to configure the base IP addresses and to make them persistent across reboots. If you do not use SMIT to configure the IP addresses the agent may failover incorrectly.
- Ensure that media speed settings are the same for both the interface and the corresponding switch port. Veritas recommends setting the media speed to full duplex mode.

# 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.

This agent is WPAR-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 *Cluster Server Administrator's Guide* for more information.

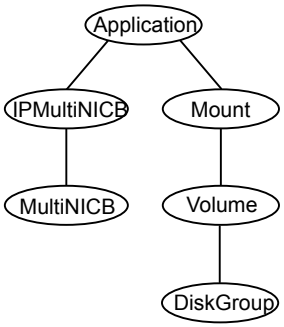
Refer to the *Veritas InfoScale Virtualization Guide*.

## Dependencies for IPMultiNICB agent

IPMultiNICB resources depend on MultiNICB resources.

They can also depend on WPAR resources.

**Figure 3-5** 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.

## The haipswitch utility for IPMultiNICB agent

You can use the haipswitch utility to switch IP addresses between MultiNICB interfaces on the same system. Running the utility with the -h flag gives an example of usage.

## 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. |
| Open | Data structures necessary for monitoring the network interfaces are created. |
| Close | Data structures that the monitor agent function uses are freed. |
| Attr_Changed | Updates the data structures that are used for monitoring the interfaces. |

## State definitions for IPMultiNICB agent

The state definitions for this agent follow:

| | |
|---|---|
| ONLINE | Indicates that the virtual IP address is up on one of the working network interfaces of the MultiNICB resource. The virtual IP address is specified in the Address attribute. The MultiNICB resource is specified in the MultiNICBResName attribute. |
| OFFLINE | Indicates that the virtual IP address is not up on any of the working network interfaces of the MultiNICB resource. The virtual IP address is specified in the Address attribute. The MultiNICB resource is specified in the MultiNICBResName 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 3-9** Required attributes

| Required attribute | Description |
|---|---|
| Address | Defines the dotted decimal virtual IP address.<br>This IP address must be different than the base IP addresses in the MultiNICB resource.<br>The IPMultiNICB agent automatically assigns the virtual IP address. Do not configure this IP address before the IPMultiNICB agent goes online. If the IP address is already configured, the agent returns an error.<br>Type and dimension: string-scalar<br>Example: "10.118.10.15" |
| MultiNICBResName | Contains the name of the MultiNICB resource that the IPMultiNICB resource depends on.<br>Type and dimension: string-scalar<br>Example: "MultiNICB_res1" |
| One of the following attributes:<br>■ NetMask: Mandatory only if you configure an IPv4 address.<br>■ PrefixLen: Mandatory only if you configure an IPv6 address. | |
| NetMask | The netmask that is associated with the virtual IP address. If you do not specify a netmask, the agent uses the operating system's default netmask.<br>This attribute is required if you configure this resource for IPv4 protocol.<br>Type and dimension: string-scalar<br>Example: "255.255.255.0" |
| PrefixLen | This is the prefix for the IPv6 address represented as the CIDR value.<br>When you use the IPv6 protocol, you must configure values for this attribute and the corresponding MultiNICB agent's Device and Protocol attributes.<br>Type-dimension: integer-scalar<br>Range: 1 - 128<br>Example: 64 |

**Table 3-10** 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`*".<br>For details about the `route` command, refer to the man page for your operating system.<br>When the value of this string is null, the agent does not add routes.<br>Type and dimension: string-scalar<br>Example: "192.100.201.0 192.100.13.7"<br>In this example, the agent executes the "`route add 192.100.201.0 192.100.13.7`" command when it configures an interface. |
| Options | Options for the `ifconfig` command.<br>Type and dimension: string-scalar<br>Example: "mtu 1500" |

# Resource type definition for IPMultiNICB agent

The resource definition for this agent on AIX follows:

```
type IPMultiNICB (
static int MonitorTimeout = 120
static int OfflineMonitorInterval = 60
static int MonitorInterval = 10
static str ArgList[] = { Address, NetMask, MultiNICBResName,
"MultiNICBResName:Probed", RouteOptions, PrefixLen, Options }
static int ContainerOpts{} = { RunInContainer=0, PassCInfo=1 }
str Address
str NetMask
str MultiNICBResName
str RouteOptions
int PrefixLen
str Options
)
```

# Sample configurations for IPMultiNICB agent

## IPMultiNICB and MultiNICB sample configuration

The sample configuration for the IPMultiNICB and MultiNICB agent follows:

```
group grp1 (
SystemList = { sysa = 0 , sysb = 1 }
AutoStartList = { sysa }
)
MultiNICB MNICB_grp1 (
Device@sysa = { en0 = "10.128.8.42", en1 = "10.128.8.43" }
Device@sysb = { en0 = "10.128.8.44", en1 = "10.128.8.45" }
NetworkHosts = "10.128.8.10"
)
IPMultiNICB ip1 (
Address = "10.128.8.14"
Netmask = "255.255.255.0"
MultiNICBResName = MNICB_grp1
)
ip1 requires MNICB_grp1
group grp2 (
SystemList = { sysa = 0 , sysb = 1 }
AutoStartList = { sysa }
)
IPMultiNICB ip2 (
Address = "10.128.8.15"
Netmask = "255.255.255.0"
MultiNICBResName = MNICB_grp1
)
Proxy MNICB_proxy (
TargetResName = MNICB_grp1
)
ip2 requires MNICB_proxy
```

## 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_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.

You need to set the MONITOR flag for each interface that the agent controls. Use the ifconfig command to set the flag. For example:

```
# ifconfig en0 monitor
```

## 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.

Some examples of the command syntax are as follows:

Link test only on interface en0:

```
haping -l en0
```

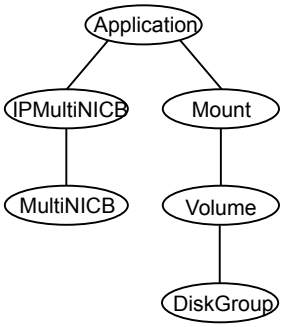
Ping a remote host 10.10.10.10 from interface en0:

```
haping -g 10.10.10.10 en0
```

## Dependencies for MultiNICB agent

The MultiNICB resource does not depend on any other resources.

**Figure 3-6** 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. |

## 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

For AIX:

**Table 3-11** Required attributes

| Required attribute | Description |
|---|---|
| Device | Lists the interfaces that you want the agent to monitor. You can assign a unique base IP address to each interface.<br><br>Use the AIX SMIT configuration tool to configure the base IP addresses and to make them persistent across reboots.<br><br>When you use the IPv6 protocol, you must configure the value for this attribute with base IPv6 addresses. You need to also configure the corresponding IPMultiNICB agent's PrefixLen attribute.<br><br>When a network interface or a network adapter of the type MultiNICB under VCS control is renamed, you must update the value of the Device attribute of the MultiNICB resource.<br><br>**Note:** Veritas recommends to offline the service groups containing the network resources before renaming the network interfaces and adapters and to update the VCS configuration to avoid any undesired behaviour.<br><br>Type and dimension: string-association<br><br>IPv4 example:<br><br>■ { en1= "10.182.9.34", en2 = "10.182.10.34" }<br><br>IPv6 example:<br><br>■ { en1 = "2001:db8::1", en2 = "2001:db8::2" } |

**Table 3-12** Optional attributes

| Optional attribute | Description |
|---|---|
| LinkTestRatio | Controls the frequency of the ping test in relation to the link test. The ping test may be run at a lesser frequency to reduce network traffic.<br><br>If this attribute is set to 1, packets are sent during every monitor cycle. If this attribute is set to 0, packets are never sent during a monitor cycle. Veritas does not recommend setting the value to zero. The agent determines link status without transmitting any ping packets.<br><br>For other values greater than 1, packets are sent at a lower frequency. For example, if LinkTestRatio=2, then ping packets are sent out during every other monitor cycle. In other words, packets are sent out half as often than if LinkTestRatio were equal to one.<br><br>When using IPv6 protocol, set the LinkTestRatio attribute to 0. If you set the attribute to another value, the MultiNICB resource may show unexpected behavior.<br><br>Type and dimension: integer-scalar<br><br>Default: 0<br><br>Example: 1 |
| NetworkHosts | The NetworkHosts attribute is a list of hosts on the local network that are pinged to determine if the network connection is available. These must be IP addresses, and not host names.<br><br>If you do not specify this attribute, the agent monitors the interface by pinging the broadcast address on the interface. If you specify one or more network hosts, and at least one host responds to a ping, the agent reports the MultiNICB resource online. The IP addresses for the NetworkHosts attribute must be on the same subnet as the base IP addresses for the MultiNICB resource. If an invalid network host address is specified or if there is mismatch in protocol of the network host and the Protocol attribute of resource, the resource enters an UNKNOWN state.<br><br>Type and dimension: string-vector<br><br>Example: { "10.128.8.10" , "10.128.8.45" } |

**Table 3-12** Optional attributes *(continued)*

| Optional attribute | Description |
|---|---|
| NoBroadcast | If the value of this attribute is 1, NoBroadcast prevents the agent from sending broadcast pings. ARP requests may still be generated.<br>**Note:** If no NetworkHosts are specified and NoBroadcast is set to 1, the agent cannot function properly. Veritas does not recommend setting NoBroadcast to 1.<br>Type and dimension: integer-scalar<br>Default: 0 |
| 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 NetworkHosts attribute 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 increases the response time.<br>Type and dimension: integer-scalar<br>Default: 3 |
| OnlineTestRepeatCount | The number of times that the test is repeated if the interface changes from down to up. This test helps to prevent oscillations in the status of the interface.<br>Type and dimension: integer-scalar<br>Default: 3 |
| NetworkTimeout | Timeout for ARP and ICMP packets in milliseconds. MultiNICB waits for the response to ICMP and ARP packets only during this time period.<br>Assign the NetworkTimeout a value in the order of tens of milliseconds, given that the ICMP and ARP destinations must be on the local network. Increasing this value increases the time for failover.<br>Type and dimension: integer-scalar<br>Default: 100 |

**Table 3-12** Optional attributes *(continued)*

| Optional attribute | Description |
|---|---|
| Gateway | IP address for the default gateway on the local network.<br>Type and dimension: string-scalar<br>Example: "136.22.1.1" |

# Resource type definition for MultiNICB agent

The resource definition for this agent on AIX follows:

```
type MultiNICB (
static int OfflineMonitorInterval = 60
static int MonitorInterval = 10
static str ArgList[] = { Device, NetworkHosts, Gateway,
LinkTestRatio, NoBroadcast, NetworkTimeout,
OnlineTestRepeatCount, OfflineTestRepeatCount }
static str Operations = None
str Device{}
str NetworkHosts[]
str Gateway
int LinkTestRatio = 0
int NoBroadcast
int NetworkTimeout = 100
int OnlineTestRepeatCount = 3
int OfflineTestRepeatCount = 3
)
```

# 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:
  - en0
- 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 agent Res. Name: Device en0 status
changed from Down to Up.
```

# Sample configurations for MultiNICB agent

## IPMultiNICB and MultiNICB configuration for MultiNICB agent

```
group grp1 (
SystemList = { sysa = 0 , sysb = 1 }
AutoStartList = { sysa }
)
MultiNICB MNICB_grp1 (
Device@sysa = { en0 = "10.128.8.42", en1 = "10.128.8.43" }
Device@sysb = { en0 = "10.128.8.44", en1 = "10.128.8.45" }
NetworkHosts = { "10.128.8.10", "10.128.8.11" }
LinkTestRatio = 1
)
IPMultiNICB ip1 (
Address = "10.128.8.14"
Netmask = "255.255.255.0"
MultiNICBResName = MNICB_grp1
)
ip1 requires MNICB_grp1
group grp2 (
SystemList = { sysa = 0 , sysb = 1 }
AutoStartList = { sysa }
)
IPMultiNICB ip2 (
Address = "10.128.8.15"
Netmask = "255.255.255.0"
MultiNICBResName = MNICB_grp1
```

```
)
Proxy MNICB_proxy (
TargetResName = MNICB_grp1
)
ip2 requires MNICB_proxy
```

## 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.

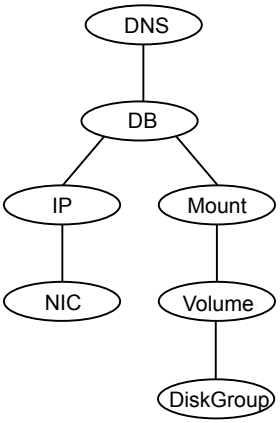
Use the DNS agent if the Resource Records need to be dynamically added and deleted from the DNS servers during failover. The agent updates the name server with the new resource record mappings while failing over and allows the clients to connect to the failed over instance of the application.

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 3-7** 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.<br><br>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:<br><br>■ keyfile.vfd<br>This action entry point checks if the key file as specified in the TSIGKeyFile attribute exists either locally or on shared storage.<br>■ dig.vfd<br>This action entry point checks if dig and nsupdate binaries exist and are executable.<br>■ master.vfd<br>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:<br><br>■ The online lock does not exist.<br>■ 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 3-13** Required attributes

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

**Table 3-13** Required attributes *(continued)*

| Required attribute | Description |
|---|---|
| ResRecord | 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.<br><br>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.<br><br>Type and dimension: string-association<br><br>Examples:<br><br>■ For forward mapping, where the zone is demo.example.com:<br>- aix901 = "192.168.2.191"<br>- ww2 = aix901<br>- aix9ip6 = "2007::1:2:3:abc"<br>■ 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:<br>aix902 = "192.168.2.102 10.87.13.22"<br>A multi-home AAAA DNS record can be configured as follows:<br>aix902 = "1234::5678 1234::AABB:CCDD"<br>■ For reverse IPv4 address mapping, where the zone is 2.168.192.in-addr.arpa:<br>191 = "aix901.demo.example.com"<br>■ For reverse IPv6 address mapping, where the zone is 3.0.0.0.2.0.0.0.1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.7.0.0.2.ip6.arpa:<br>cba = "aix9ip6.demo.example.com"<br><br>Use only partial host names. If you use a fully qualified domain name, append a period "." at the end of the name.<br><br>For CNAME records, use:<br><br>■ ResRecord = { www = mydesktop }<br>or<br>■ ResRecord = { www = "mydesktop.marketing.example.com." }<br>Where the Domain attribute is "marketing.example.com" |

**Table 3-13** Required attributes *(continued)*

| Required attribute | Description |
|---|---|
| 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:<br>■ PTR: if the Domain attribute ends with .arpa<br>■ 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.)<br>■ AAAA: if the record data fields are in multiple sets of hexadecimal format, then this record is an IPv6 associated type AAAA record.<br>■ CNAME: for any other valid record data.<br>**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 3-14** 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.<br><br>A lower value means more hits on your DNS server, while a higher value means more time for your clients to learn about changes.<br><br>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.<br><br>Type and dimension: integer-scalar<br><br>Default: 86400<br><br>Example: 3600 |
| StealthMasters | The list of primary master name servers in the domain.<br><br>This attribute is optional since the first name server is retrieved from the zone's SOA (Start of Authority) record.<br><br>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.<br><br>Type and dimension: string-vector<br><br>Example: { "10.190.112.23" } |
| 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.<br><br>Type and dimension: string-scalar<br><br>Example:<br><br>/var/tsig/example.com.+157+00000.private |

**Table 3-14** 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.<br>Type and dimension: boolean-scalar<br>Default: 0<br>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.<br>Type and dimension: boolean-scalar<br>Default: 0<br>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.<br>**Note:** Do not set this attribute if the Windows DNS server accepts non-secure updates.<br>If this attribute is set to 1, the agent uses the -g option with the nsupdate command.<br>See "Agent notes for DNS agent" on page 139. for more information on requirements to use the DNS agent with the secure Windows DNS server.<br>Type and dimension: boolean-scalar<br>Default: 0<br>Example: 1 |

**Table 3-14** 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.<br>**Note:** The refresh request is sent in the next monitor cycle after the RefreshInterval period is reached.<br>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.<br>Type and dimension: integer-scalar<br>Default: 0<br>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.<br>**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.<br>Type and dimension: boolean-scalar<br>Default: 0<br>Example: 1 |

# Resource type definition for DNS agent

The resource definition for this agent on AIX 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 3-15 summarizes the monitor scenarios for the Online lock files.

**Table 3-15** 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: en0

```
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

Chapter 4

# File share agents

This chapter includes the following topics:


- About the file service agents
- NFS agent
- NFSRestart agent
- Share agent
- About the Samba agents
- SambaServer agent
- SambaShare agent
- NetBios agent


## 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 subsystem processes required by all exported NFS file systems.

---

**Note:** The attributes NFSv4root and NFSSecurity require AIX 5.3 TL7 SP6 or later and AIX 6.1 TL5 or later.

---

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.

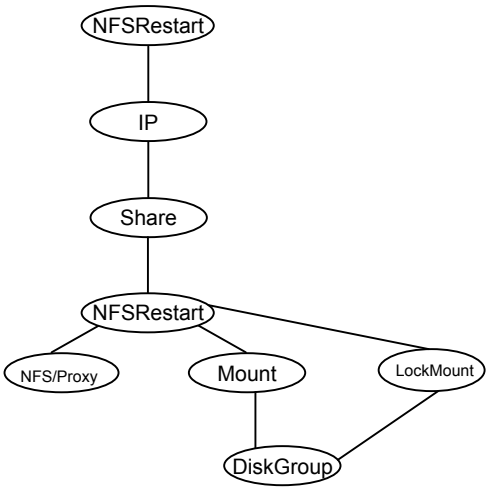
For important information about this agent,

See "Notes for NFS agent" on page 151.

## 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 are running. If they are not running, the agent starts the daemons and exits.
- The nfsrgyd daemon is started if NFSv4Root is specified.
- The gssd daemon is started if NFSSecurity is set to 1.

Offline — Not applicable.

| | |
|---|---|
| Monitor | ■ Monitors nfsd and mountd by checking whether or not the daemons are active.<br>■ The nfsrgyd daemon is monitored if NFSv4Root is specified.<br>■ The gssd daemon monitored if NFSSecurity is set to 1. |
| Clean | Stops and restarts nfsd and mountd 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 AIX

| Optional attribute | Description |
|---|---|
| Nservers | Specifies the number of concurrent NFS requests the server can handle.<br>Type and dimension: integer-scalar<br>Default: 10 |

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

| Optional attribute | Description |
|---|---|
| NFSv4Root | Root directory of the NFSv4 pseudo file system to be exported. All exports should have a path relative to the path specified by this attribute. You can explicitly create the NFSv4 pseudo file system by specifying the exname option of the `exportfs` command in the Options attribute of the Share resource.<br>If you want to export file systems with NFSv4 protocols and do not want to explicitly create NFSv4 pseudo file system by using the exname option, then set NFSv4Root to "/".<br>Required for filesystems to be exported with v4 protocol.<br>Type and dimension: string-scalar |
| NFSSecurity | If the value of this attribute is 1, the gssd daemon starts.<br>You must configure the type of security that NFS supports, for example: Kerberos.<br>Type and dimension: boolean-scalar<br>Default: 0 |
| GracePeriod | Specifies the grace period, in seconds, for which the server allows lock recovery.<br>Required for NFS lock recovery.<br>Type and dimension: integer-scalar<br>Default: 90 |
| LockFileTimeout | Specify the amount of time required, in seconds, for the service group to go online. The agent uses this attribute to synchronize the starting and stopping of daemons between multiple service groups.<br>Type and dimension: integer-scalar<br>Default: 180<br>Example: "240" |

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

| Optional attribute | Description |
|---|---|
| Protocol | Specify the protocol to run the nfsd daemon. The following are the valid values:<br>■ tcp<br>■ udp<br>■ all<br>The agent uses this attribute to ensure that the NFS daemon is running using the specified protocol.<br>**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.<br>Type and dimension: string-scalar<br>Default: all<br>Example: tcp |

# Resource type definition for NFS agent

```
type NFS (
    static keylist RegList = { Protocol }
    static int RestartLimit = 1
    static str ArgList[] = { Nservers, GracePeriod, NFSv4Root,
    NFSSecurity, LockFileTimeout, Protocol }
    static str Operations = OnOnly
    int Nservers = 10
    int GracePeriod = 90
    str NFSv4Root
    boolean NFSSecurity = 0
    int LockFileTimeout = 180
    str Protocol = all
)
```

# Notes for NFS agent

The NFS agent has the following notes:

- Using NFSv4 on AIX

### Using NFSv4 on AIX

For NFS v4 support, you must specify the NFSv4Root attribute. You must include `vers=4` in the Option attribute of the Share resource.

Set up Enterprise Identity Mapping (EIM) in the NFS environment, if:

- Mapping of userids and username is not same on both client and server
- Client and server belong to different domains

If either of the above points are true, and EIM is not set up, the client has minimal rights (user=nobody, group=nobody).

If you want to use the NFSv4 security feature, set the NFSSecurity attribute of the NFS resource to `1`. Manually configure Kerberos or any other security environment that is supported by NFSv4.

### Caveats

You export filesystems with NFSv4Root="/exp/exports1", and you forcefully stop the engine so that exports are still valid and existing. If you change configurations on NFS to set NFSv4Root="/newexport", the NFS Agent is not able to come online with this new root, because the already exported filesystem is using an older NFS pseudo file system root. To avoid this problem bring all Share resources down properly before changing NFSv4Root.

If you create a pseudo file system, a client can access the filesystem. After the NFS server fails over to the other system in the cluster, the client can not see the filesystem. The client needs to remount it.

## 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_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.

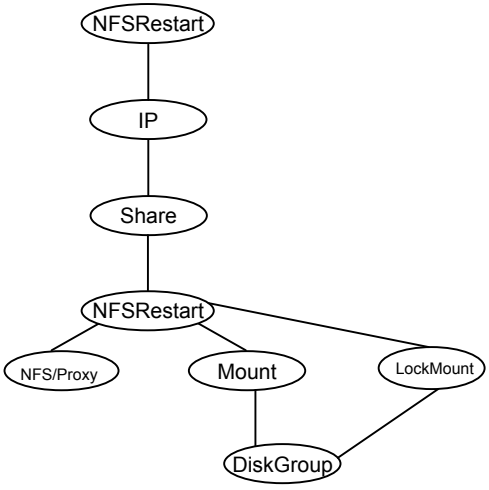
For important information about this agent, refer to Notes for NFSRestart agent

## 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:<br>■ Restarts all the NFS daemons that the upper NFSRestart resource stopped previously.<br>For the upper NFSRestart resource:<br>■ Terminates the statd and lockd daemons to clear the lock state.<br>■ Terminates the nfsd and mountd daemons to close the TCP/IP connections.<br>■ Terminates the smsyncd daemon if the daemon is running. |
| Action | ■ nfsconf.vfd<br>Checks the runlevel information of the system service nfslock to confirm that the lock daemons do not come online automatically after reboot.<br>■ lockdir.vfd<br>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 on the system.<br>Type and dimension: string-scalar |

**Table 4-3** Optional attributes

| Optional attribute | Description |
|---|---|
| LocksPathName | The path name of the directory to store the NFS lock information. This attribute is required when the value of the NFSLockFailover attribute is 1. The path that you specify for the LocksPathName attribute should be on shared storage. This is to ensure that it is accessible to all the systems where the NFSRestart resource fails over.<br>Type and dimension: string-scalar<br>Example: "/share1x" |
| NFSLockFailover | A flag that specifies whether the user wants NFS locks to be recovered after a failover.<br>Type and dimension: boolean-scalar<br>Default: 0 |
| Lower | Defines the position of NFSRestart resource in the service group. The NFSRestart resource below the Share resource needs a value of 1.<br>The NFSRestart resource on the top of the resource dependency tree has a Lower attribute value of 0.<br>Type and dimension: integer-scalar<br>Default: 0 |

## Resource type definition for NFSRestart agent

```
type NFSRestart (
    static str ArgList[] = { NFSLockFailover, LocksPathName,
    "NFSRes:GracePeriod", "NFSRes:LockFileTimeout",
    "NFSRes:Nservers", "NFSRes:NFSv4Root", Lower, State  }
    static keylist SupportedActions = { "lockdir.vfd",
    "nfsconf.vfd" }
    str LocksPathName
    str NFSRes
    int Lower = 0
    boolean NFSLockFailover = 0
)
```

## Notes for NFSRestart agent

The NFSRestart agent has the following notes:

- About high availability fire drill
- Providing a fully qualified host name

## 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*.

## 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. NFS lock recovery may fail 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).

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 netsvc.conf files:

Changes in `/etc/hosts` file

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. Make the following changes:

Changes in `/etc/netsvc.conf` file

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 netsvc.conf file might affect other services running on the system.

For example:

```
hosts = local,bind,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.

# 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:

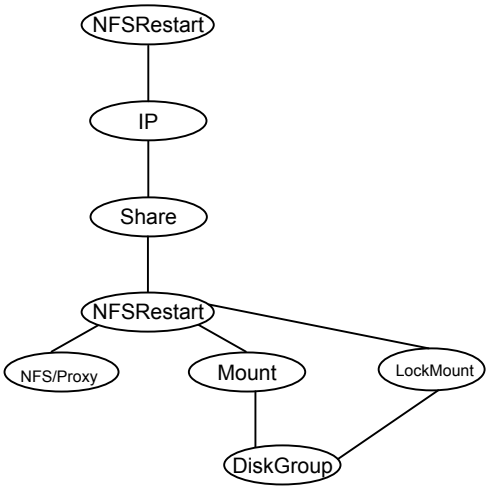
Notes for Share agent

## 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 | Exports (shares) a directory to the specified client. |
| Offline | Unshares the exported directory from the client. |
| Monitor | Verifies that the shared directory is exported to the client. |
| Clean | Terminates all ongoing resource actions and takes the resource offline, forcibly when necessary. |
| Action | direxists.vfd<br>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.<br>Type and dimension: string-scalar<br>Example: "/share1x" |
| NFSRes | This attribute has been deprecated. |

**Table 4-5** Optional attributes

| Optional attribute | Description |
|---|---|
| Options | Options to the `exportfs` command. When specifying multiple options, separate them with commas, for example:<br>"rw,vers=4"<br>For more information about the `exportfs` command and its options, refer to the exportfs manual page.<br>Type and dimension: string-scalar |

# Resource type definition for Share agent

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

# 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*.

## 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

# 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

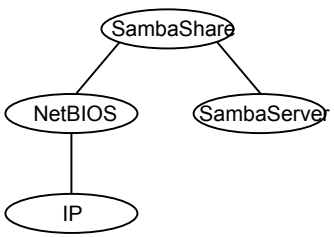
The SambaServer agent starts, stops, and monitors the smbd process as a daemon. Only one resource of this type is permitted. You can use the agent to make a smbd daemon highly available.

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.<br>Type and dimension: string-scalar<br>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.<br>Type and dimension: string-scalar<br>Example:" /usr/local/samba/var/locks" |

**Table 4-6** Required attributes *(continued)*

| Required attribute | Description |
|---|---|
| SambaTopDir | Parent path of Samba daemon and binaries.<br>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:<br>`# hares -modify <res> Enabled 0`<br>`# hares -modify <res> Enabled 1`<br>Example:"/usr/local/samba" |

**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.<br>Type and dimension: integer-scalar<br>Default: 5 |
| Ports | Ports where Samba accepts connections.<br>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.<br>Type and dimension: integer-vector<br>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.<br>Type and dimension: integer-scalar<br>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.<br>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.<br>Type and dimension: string-scalar<br>Example: "/usr/local/samba/var/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.<br>**Note:** Only IPv4 addresses are supported.<br>Type and Dimension: string-scalar<br>Example: “10.128.10.14” |

# Resource type definitions for SambaServer agent

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

# Sample configurations for SambaServer agent

The sample configurations for this agent follow:

```
SambaServer samba_server (
ConfFile = "/etc/smb.conf"
LockDir = "/usr/local/samba/var/locks"
```

```
SambaTopDir = "/usr/local/samba"
IndepthMonitorCyclePeriod = 3
ResponseTimeout = 15
)
```

## 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

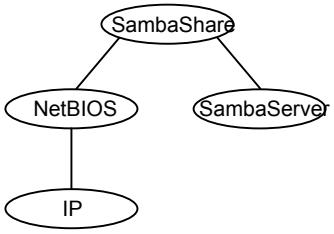
The SambaShare agent adds, removes, and monitors a share by modifying the specified Samba configuration file. You can use the agent to make a Samba Share highly available.

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.<br>Type and dimension: string-scalar<br>Example: "smb_res1" |
| ShareName | Name of the share resource as exported by samba.<br>**Note:** This name can be different from the SambaShare resource name.<br>Type and dimension: string-scalar<br>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 (;).<br>Type and dimension: string-scalar<br>Example: "path=/shared; public=yes; writable=yes" |

## 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" }
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.

---

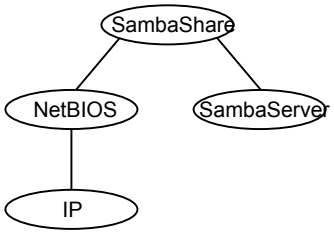
**Note:** The nmbd broadcasts the NetBIOS name, or the name by which the Samba server is known in the network.

---

## Dependencies for NetBios agent

The NetBios resource depends on the IP, IPMultiNIC, or IPMultiNICB resource if the virtual IP address configured in the IP/IPMultiNIC 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:<br>■ NetBIOS name is not advertised.<br>■ A NetBIOS alias is not advertised.<br>■ Samba is not handling requests on any of the specified interfaces.<br>■ If WINS support is configured, Samba is not providing WINS service.<br>■ 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 4-9** Required attributes

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

**Table 4-10** Optional attributes

| Optional attribute | Description |
|---|---|
| Interfaces | List of network interfaces on which Samba handles browsing.<br>Type and dimension: string-vector<br>Example: "172.29.9.24/16"<br>**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.<br>Type and dimension: string-vector<br>Example: { host1_samba, myname }<br>**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.<br>Type and dimension: integer-scalar<br>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.<br>Type and dimension: integer-scalar<br>Default: 0 |
| PidFile | The absolute path to the NetBIOS daemon pid file. This file contains the process ID of the monitored nmbd process.<br>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.<br>Type and dimension: string-scalar<br>Example: "/usr/local/samba/var/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
- Apache HTTP server agent
- Application agent
- CoordPoint agent
- Process agent
- ProcessOnOnly agent
- WPAR agent
- MemCPUAllocator agent
- LPAR agent


## 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.

---

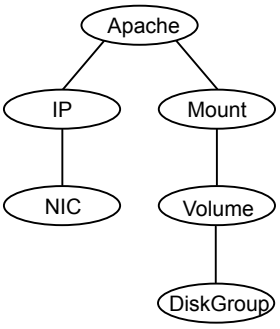
The agent performs detailed monitoring on the Apache resource. You can tune the frequency of detailed monitoring with the LevelTwoMonitorFreq attribute. By default, the agent monitors the Apache resource after every fifth monitor cycle to confirm the health of the resource.

See "Apache HTTP server notes" on page 183. 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:<br>■ 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.<br>■ Alternatively, if the httpdDir attribute specifies the full path of the Apache HTTP server binary file, the binary file is executed with appropriate arguments.<br>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:<br>■ 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.<br>■ 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.<br>■ Sends a TERM signal to the HTTP Server parent process (Apache).<br>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<br>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.<br>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.<br>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.<br>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.<br>Type and dimension: string-scalar<br>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.<br>Type and dimension: string-scalar<br>Example: "/usr/apache2/bin" |
| PidFile | This attribute is required when you want to enable the detection of a graceful shutdown outside of VCS control.<br>See Table 5-2 on page 179. |
| EnvFile | This attribute may be required when you use IBM HTTP Server.<br>See Table 5-2 on page 179. |

**Table 5-2** Optional attributes

| Optional attribute | Description |
|---|---|
| DirectiveAfter | A list of directives that httpd processes after reading the configuration file.<br>Type and dimension: string-association<br>Example: DirectiveAfter{} = { KeepAlive=On } |
| DirectiveBefore | A list of directives that httpd processes before it reads the configuration file.<br>Type and dimension: string-association<br>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.<br>Type and dimension: string-scalar<br>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.<br>For example: `/usr/sbin/httpd -f path_to_httpd.conf -k start -DSSL`<br>Where path_to_httpd.conf file is the path to the httpd.conf file.<br>If this attribute is set to 0 (false) the agent excludes the SSL support.<br>Type and dimension: boolean-scalar<br>Default: 0<br>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.<br>You can use IPv4 or IPv6 addresses for the HostName attribute.<br>**Note:** The HostName attribute is required only if you enable in-depth monitoring by setting the LevelTwoMonitorFreq attribute.<br>Type and dimension: string-scalar<br>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.<br>Type and dimension: integer-scalar<br>Default: 80<br>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.<br>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.<br>Type and dimension: string-scalar<br>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.<br>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).<br>Type and dimension: string-scalar<br>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.<br>Type and dimension: boolean-scalar<br>Example: "/apache/server1/libexec" |

**Table 5-2** Optional attributes *(continued)*

| Optional attribute | Description |
|---|---|
| ResLogLevel | This attribute has been deprecated.<br>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.<br>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.<br>For example, the value 5 indicates that the agent will monitor Apache in detail after every five online monitor intervals.<br>Type and dimension: integer-scalar<br>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 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

## 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 184.
- 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.

## 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.

# Sample configurations

```
group ApacheG1(
        SystemList = { host1 = 0, host2 = 1 }
        )

        Apache httpd_server (
                httpdDir = "/apache/bin"
                HostName = vcsaix1
                Port = 8888
                User = root
                ConfigFile = "/apache/conf/httpd.conf"
                )

        DiskGroup Apache_dg (
                DiskGroup = apc1
                )

        IP Apache_ip (
                Device = en0
                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 = en0

        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 = en0


)
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 WPAR-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 about ContainerOpts attribute, refer to the *Cluster Server Administrator's Guide*.

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)

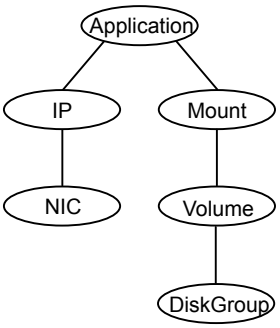
For more information, refer to the *Cluster Server Administrator's Guide*.

## 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 IPMultiNIC or IPMultiNICB resource.

**Figure 5-2** Sample service group that includes an Application resource



# Agent functions

| | |
|---|---|
| Online | Runs the executable that you specify in the value of the StartProgram attribute. Runs the executable with the specified parameters in the context of the specified user.<br>To bring the resource online, the agent function performs the command:<br>`su [-] user -c executable_to_online_resource` |
| Offline | Runs the executable that you specify in the value of the StopProgram attribute. Runs the executable with the specified parameters in the context of the specified user.<br>To take the resource offline, the agent function performs the command:<br>`su [-] user -c executable_to_offline_resource` |
| Monitor | If you specify the MonitorProgram attribute, the agent executes the user defined MonitorProgram 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 you specify.<br>Use any combination among these attributes (MonitorProgram, PidFiles, or MonitorProcesses) to monitor the application.<br>If any of the processes that are specified in either PidFiles or MonitorProcesses is determined not to be running, the monitor returns OFFLINE. If the process terminates ungracefully, the monitor returns OFFLINE and failover occurs.<br>To monitor the resource, the agent function performs the command:<br>`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.<br>To forcefully stop the resource, the agent function performs the command:<br>`su [-] user -c executable_to_clean_resource`<br>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:<br>■ program.vfd<br>Checks the availability of the specified program and the execution permissions for the specified program.<br>■ user.vfd<br>Checks the existence of the specified user on the host.<br>■ cksum.vfd<br>Checks the existence of the same binary on all nodes.<br>■ propcv<br>[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.<br>■ getcksum<br>Returns the checksum of the specified program |

For all agent functions on AIX: If the User attribute is set to a non-root user, the agent checks if this non-root user has any home directory that exists and is accessible. If this non-root user has a home directory and is accessible, then it does the command: `su - user -c command`. If this non-root user does not have a home

directory or the home directory is unaccessible, then it uses the command: `su user -c command`.

## 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. |
| FAULTED | Indicates that the process has terminated unexpectedly or MonitorProgram returns OFFLINE unexpectedly. |

## Agent attributes

**Table 5-4** Required attributes

| 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.<br>**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.<br>**Note:** Do not use the opening and closing ({ }) brace symbols in this string.<br>**Note:** In the script, specify a return value that is between 0 and 255.<br>Type and dimension: string-scalar<br>Example: "/usr/sbin/sample_app start" |

**Table 5-4** Required attributes *(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.<br>**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.<br>**Note:** Do not use the opening and closing ({ }) brace symbols in this string.<br>**Note:** In the script, specify a return value that is between 0 and 255.<br>Type and dimension: string-scalar<br>Example: "/usr/sbin/sample_app stop" |
| At least one of the following attributes:<br>■ MonitorProcesses<br>■ MonitorProgram<br>■ PidFiles | See Table 5-5 on page 194. |

**Table 5-5** Optional attributes

| 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.<br>**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.<br>**Note:** If the CleanProgram executable returns a non-zero value, the agent treats it as a clean failure and the resource will not fault.<br>Type and dimension: string-scalar<br>Example: "/usr/sbin/sample_app stop" |

**Table 5-5** Optional attributes *(continued)*

| Optional attribute | Description |
|---|---|
| MonitorProcesses | A list of processes that you want monitored and cleaned. Each process name is the name of an executable. Qualify the executable name with its complete path if the path starts the executable.<br><br>The process name must be the full command line argument that the `ps -u user -eo pid,args` command displays for the process.<br><br>Type and dimension: string-vector<br><br>Example: { "/usr/bin/sh /user/app/process1", "/usr/bin/sh /user/app/process2" } |
| 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.<br><br>MonitorProgram can return the following states: 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.<br><br>If MonitorProgram is configured and not available, then resource state will be:<br><br>■ OFFLINE if the resource was in OFFLINE state and not waiting for any action.<br>■ UNKNOWN if the resource was in any other state or waiting for some action.<br><br>**Note:** Do not use the opening and closing ({ }) brace symbols in this string.<br><br>Type and dimension: string-scalar<br><br>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.<br><br>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.<br><br>Type and dimension: string-vector<br><br>Example: "/etc/sample/sample_app.pid" |

**Table 5-5** Optional attributes *(continued)*

| Optional attribute | Description |
|---|---|
| User | The user ID 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.<br>**Note:** If the configured user does not exist, the resource state will be UNKNOWN.<br>Type and dimension: string-scalar<br>Default: root<br>Example: user1 |
| EnvFile | The environment file that should get sourced before running any of the StartProgram, StopProgram, MonitorProgram or CleanProgram.<br>**Note:** Make sure that the EnvFile adheres the default shell syntax of the configured user.<br>Type and dimension: string-scalar<br>Default: ""<br>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.<br>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.<br>Type and dimension: boolean-scalar<br>Default: 0<br>Example: 1 |

# Resource type definition for Application agent

```
type Application (
        static keylist SupportedActions = { "program.vfd",
        "user.vfd", "cksum.vfd", getcksum, propcv }
        static str ArgList[] = { User, StartProgram, StopProgram,
```

```
        CleanProgram, MonitorProgram, PidFiles, MonitorProcesses,
        EnvFile, UseSUDash, State, IState }
        static int ContainerOpts{} = { RunInContainer=1, PassCInfo=0 }
        static str IMFRegList[] = { MonitorProcesses, User, PidFiles,
        MonitorProgram, LevelTwoMonitorFreq }
        static int LevelTwoMonitorFreq = 1
        str User = "root"
        str StartProgram
        str StopProgram
        str CleanProgram
        str MonitorProgram
        str PidFiles[]
        str MonitorProcesses[]
        str EnvFile
        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.

For a process that is registered with AMF for offline monitoring, AMF may not detect the process being executed if the length of the process and related arguments exceeds 80 characters.

## 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.
- The combined length of the process along with its arguments and the interpreter path (if the process is a script) does not exceed 80 characters.
- 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.

## 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" }
)
```

## 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.

---

For important information about this agent, refer to:

See "Notes for the CoordPoint agent" on page 205.

## 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.<br><br>CoordPoint resources are persistent, which means that they cannot be brought online or taken offline. They can only monitor the coordination point registrations.<br><br>The CoordPoint agent also performs I/O fencing reporting activities.<br><br>See "CoordPoint agent I/O fencing reporting activities" on page 206. |

## 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:<br>■ The number of coordination points with missing keys (or registrations) has exceeded the value of the FaultTolerance attribute.<br>■ The number of unreachable coordination points has exceeded the value of the FaultTolerance attribute.<br>■ Coordinator disks are deleted from or added to the Coordinator Disk Group.<br>■ 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.<br><br>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.<br><br>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.<br><br>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.<br><br>To view the current FaultTolerance value, enter the following command:<br><br># hares -display coordpoint -attribute FaultTolerance<br><br>Type and dimension: integer-scalar<br><br>Default: "0" |

**Table 5-8** Optional attribute

| Optional attribute | Description |
|---|---|
| ActionOnCoordPointFault | 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.<br>By default, the attribute is disabled. To enable the attribute set its value to `RefreshRegistrations`.<br>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.<br>`# /opt/VRTS/bin/hares -action coordpoint enable_refresh -sys <sys_name>`<br>Type and dimension: string-scalar<br>Default: None<br>To enable the attribute: Set its value to `RefreshRegistrations` |

# Resource type definition

```
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

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 WPAR-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 *Cluster Server Administrator's Guide*.

# IMF awareness

The Process agent is IMF-aware and uses Asynchronous Monitoring Framework (AMF) kernel driver for IMF notification.

Note that the AMF kernel driver also monitors the kernel processes if you have enabled intelligent monitoring for Process agent.

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:

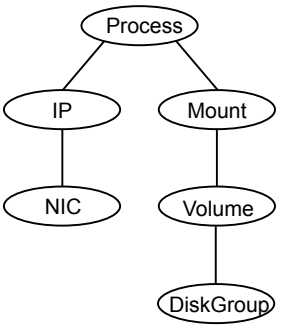
- The existence of a binary executable for the specified process (program.vfd)
- The existence of the same binary on all nodes (program.vfd)

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, IPMultiNIC, IPMultiNICB, WPAR, 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.<br>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 for AIX

| 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.<br>Type and dimension: string-scalar<br>Example: "/usr/sbin/sendmail" |

**Table 5-10** Optional attributes for AIX

| 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.<br>This attribute must not exceed 80 characters.<br>Type and dimension: string-scalar<br>Example: "bd q1h" |

**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 AIX follows:

```
Process usr_lib_sendmail (
    PathName = "/usr/lib/sendmail"
    Arguments = "bd q1h"
    )
```

## 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_2, DBG_3, 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 WPAR-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 *Cluster Server Administrator's Guide* for more information.

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.<br>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 for AIX

| 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.<br>The value configured for this attribute needs to match the process listing from the ps output for the agent to display as ONLINE.<br>Type and dimension: string-scalar |

**Table 5-12** Optional attributes for AIX

| 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.<br>Arguments must not exceed 80 characters (total).<br>Type and dimension: string-scalar |

**Table 5-12** Optional attributes for AIX *(continued)*

| Optional attribute | Description |
|---|---|
| IgnoreArgs | A flag that indicates whether monitor ignores the argument list.<br>■ If the value is 0, it checks the process pathname and argument list.<br>■ If the value is 1, it only checks for the executable pathname and ignores the rest of the argument list.<br>Type and dimension: boolean-scalar<br>Default: 0 |

# Resource type definition

```
type ProcessOnOnly (
static str ArgList[] = { IgnoreArgs, PathName, Arguments }
static str Operations = OnOnly
static int ContainerOpts{} = { RunInContainer=1, PassCInfo=0 }
int IgnoreArgs
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, DBG_2, DBG_3, DBG_4, DBG_5

# WPAR agent

The WPAR agent brings online, takes offline, and monitors system workload partitions. You can use the agent to make WPARs highly available and to monitor them.

This agent is WPAR-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 the values for these keys.

For more information on ContainerOpts attribute, refer to the *Cluster Server Administrator's Guide* for more information.

## IMF awareness

The WPAR 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*.

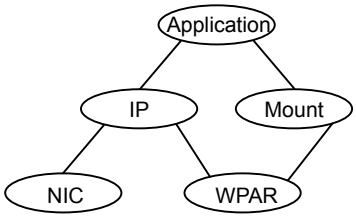
For more information about IMF-related WPAR agent functions, see Agent functions.

For more information about using the WPAR agent with IMF, see Using the WPAR agent with IMF.

## Dependencies

No dependencies exist for the WPAR resource.

**Figure 5-4** Sample service group that includes a WPAR resource




# Agent functions

| | |
|---|---|
| Online | Brings a WPAR up and running. |
| Offline | Takes a WPAR down gracefully. |
| Monitor | Checks if the specified WPAR is up and running.<br>If IMF is enabled for the WPAR agent, the resource is monitored asynchronously and any change in the resource state is immediately sent to VCS for appropriate action. |
| Clean | Brings down a WPAR forcefully. |
| 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). |

# Attributes

Table 5-13 summarizes the optional attributes for the WPAR agent.

**Table 5-13** Optional attributes for AIX

| Optional attribute | Description |
|---|---|
| ShutdownGracePeriod | Allows the root user to set the number of seconds before the shut down of a WPAR.<br>Veritas recommends to check the time required for WPAR to stop outside VCS control and configure this attribute accordingly.<br>**Note:** Offline fails if the value of this attribute is 0 as the WPAR takes some time to shut down fully.<br>Type and dimension: integer-scalar<br>Default: 90<br>Example: "120" |
| ResourceSet | A resource set is used to define a subset of processors in the system. If a resource set is specified for a workload partition, it can use the processors within the specified resource set only. The value of the ResourceSet attribute is the name of the resource set created using the `mkrset` command. If set, the agent configures the WPAR to use only the resource set specified by this attribute.<br>Type and dimension: string-scalar<br>Default: ""<br>Example: ResourceSet = “myrset” |
| WorkLoad | Allows modification of resource control attributes shares_CPU and shares_memory. The key CPU is used to specify the number of processor shares that are available to the workload partition. The key MEM is used to specify the number of memory shares that are available to the workload partition.<br>Valid values for the WorkLoad attribute keys CPU and MEM can be any value from 1 to 65535. If the values are not specified, unlimited share of CPU and memory is assigned to the WPAR.<br>Type and dimension: integer-association<br>Default: WorkLoad{} = { }<br>Example: { CPU = 50, MEM = 30 } |

**Table 5-13** Optional attributes for AIX *(continued)*

| Optional attribute | Description |
|---|---|
| DROpts | The value of this attribute consists of the following keys that define the disaster recovery (DR) options for the WPAR.<br>■ DNSDomain<br>The domain name to use within the WPAR at this site<br>■ DNSSearchPath<br>The domain search path used by this WPAR at this site. The value of this key must contain a list of DNS domain names that are used for the DNS lookup of a hostname in case the domain name of the hostname is not specified. Use spaces to separate the domain names.<br>■ DNSServers<br>The list of DNS servers used by this WPAR at this site. The value of this key must contain a list of IP addresses of DNS servers that are used for the DNS lookup of a hostname. Use spaces to separate the IP addresses.<br>In a DR configuration, if one or more of these keys are set, the resource is considered to be DR-enabled. If all the keys stay at their default value (""), then the resource is not DR-enabled even if it is in a disaster recovery configuration.<br>Type and dimension: string-association<br>Example: DROpts = {DNSSearchPath = "vx1.example.com example.com", DNSServers = "10.216.16.101", Gateway = "10.209.72.1", DNSDomain = "vx1.example.com" } |

# Resource type definition

The resource type definition for this agent follows:

```
type WPAR (
static str ArgList[] = { ShutdownGracePeriod, ResourceSet,
WorkLoad, DROpts }
static boolean AEPTimeout = 1
static str IMFRegList[] = { ContainerInfo }
static str IMF{} = { Mode = 3, MonitorFreq = 5,
RegisterRetryLimit = 3 }
static int ContainerOpts{} = { RunInContainer=0,
PassCInfo=1 }
int ShutdownGracePeriod = 90
str ResourceSet
```

```
int WorkLoad{}
str DROpts{}
)
```

For more information about configuring WPARs, refer to the *Cluster Server Administrator's Guide*.

# Sample configuration

The sample configuration for the WPAR agent follows:

```
include "types.cf"
cluster vcswpars_clus (
UserNames = { w_wpar_res = aHEbEDdEEeEGdHEg }
MeterUnit = { CPU = CPU, Mem = MB, Swap = MB }
MeterWeight = { CPU = 10, Mem = 5, Swap = 1 }
)

system sys1 (
)
system sys2 (
)



group wparapp_grp (
      SystemList = { sys1 = 0, sys2= 1 }
      ContainerInfo = { Name = wpar, Type = WPAR, Enabled = 1 }
      Administrators = { w_wpar_res }
        )

         Application samba_app (
StartProgram = "/usr/sbin/samba start"
StopProgram = "/usr/sbin/samba stop"
PidFiles = { "/localzone1/root/var/lock/samba/smbd.pid" }
MonitorProcesses = { "nmbd" }
    )
      WPAR wpar_res (
           )


 samba_app requires wpar_res
```

```
// resource dependency tree
        //
        //      group wparapp_grp
        //      {
        //      Application samba_app
        //          {
        //          WPAR wpar_res
        //          }
        //      }
```

# WPAR agent notes

- Using the WPAR agent with IMF
- Configuring the WPAR agent for DR in a Global Cluster environment
- Using the IP agent with WPAR

## Using the WPAR agent with IMF

The WPAR agent supports intelligent monitoring during the online and offline operation. The agent registers the following for online monitoring:

- WPAR name
- /etc/VRTSvcs/.vcshost file of WPAR
- Directory name of WPAR

The agent registers the WPAR name for offline monitoring.

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 WPAR root is created, using one of the storage agents such as Mount or Volume. The WPAR agent should have a dependency on the storage agent as depicted in Dependencies

## Configuring the WPAR agent for DR in a Global Cluster environment

For information about configuring the WPAR agent for DR in a Global Cluster environment, refer to the *Veritas InfoScale Virtualization Guide*.

### Using the IP agent with WPAR

When you want to monitor an IP within a WPAR, the IP resource should depend on the WPAR resource. Any other resources within WPAR should be made dependent on the IP resource. For a pictorial representation, see Dependencies.

## Debug log levels

The WPAR agent uses the following debug log levels:

DBG_1, DBG_5

# MemCPUAllocator agent

Use the MemCPUAllocator agent to allocate CPU and memory to an IBM AIX dedicated partition. Set this resource's attribute values to specify the amount of CPU and memory that you want to allocate to a service group on a DLPAR. Configure this resource as a leaf node in the service group dependency tree.

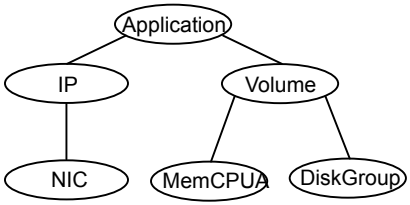
For prerequisites and other important information about this agent:

See "MemCPUAllocator agent notes" on page 224.

## Dependencies

Set the MemCPUAllocator resource as a leaf node in a resource dependency tree. Select the amount of CPU and memory that you want the DLPAR to have before it comes online.

**Figure 5-5** Sample service group that includes a MemCPUAllocator resource, where the MemCPUA resource represents the MemCPUAllocator resource



# Agent functions

| | |
|---|---|
| Online | The MemCPUAllocator agent dynamically allocates the required amount of memory and CPU to the DLPAR through the Hardware Management Console (HMC).<br>■ The agent does not allocate additional memory if there is memory already allocated to the DLPAR that was not allocated by any other MemCPUAllocator resource and that extra memory is more than what is required by MemCPUAllocator resource.<br>■ The agent does not allocate additional CPU if there is CPU already allocated to the DLPAR that was not allocated by any other MemCPUAllocator resource and that extra CPU is more than what is required by MemCPUAllocator resource. |
| Offline | The agent deallocates the amount of memory and CPU it acquired during the online agent function. It then returns the resources back to the pool. |
| Monitor | Checks that the online agent function succeeded. If it succeeded, then the monitor agent function reports the resource state as ONLINE. If it did not succeed, then the monitor agent function reports the resource state as OFFLINE.<br>If the agent is not able to allocate the required resources during the online agent function, the subsequent monitor reports OFFLINE and the resource faults. Because the resource is a leaf node, VCS engine stops bringing other resources online and marks the group as FAULTED. The VCS engine then tries to bring the group online on some other DLPAR. This check ensures that the agent can dynamically allocate the resources that the service group requires for the DLPAR. |

# Attributes

Table 5-14 summarizes the required attributes for the MemCPUAllocator agent.

**Table 5-14** Required attributes

| Required attribute | Description |
|---|---|
| ManagedSystem | The name of the managed system that contains the partition.<br>Type-dimension: string-scalar<br>Example: mymachine |

**Table 5-14** Required attributes *(continued)*

| Required attribute | Description |
|---|---|
| HMC | Name of the HMC<br>The list of HMCs that control the managed systems. The agent tries to connect to any HMC on this list in the order that they are specified.<br>Type-dimension: string-vector<br>Example: HMC = { myhmc1, myhmc2 } |

Table 5-15 summarizes the optional attributes for the MemCPUAllocator agent.

**Table 5-15** Optional attributes

| Optional attribute | Description |
|---|---|
| MemoryRequired | Amount of RAM (in MB) that you want to allocate.<br>Type-dimension: string-scalar<br>Default: 0<br>Example: 256 |
| User | Specifies the user of HMC(s) which has the permission to allocate and deallocate the resources of the DLPAR on which the resource is configured.<br>Type-dimension: string-scalar<br>Default: hscroot<br>Example: hmcuser |
| MemoryCritical | Specifies whether the memory allocation is critical. A value of 0 indicates that the online agent function should go ahead even when the required memory was not successfully allocated.<br>Type-dimension: boolean-scalar<br>Default: 0<br>Example: 1 |
| CPURequired | The number of dedicated CPUs that you want to allocate.<br>Type-dimension: string-scalar<br>Example: 2 |

**Table 5-15** Optional attributes *(continued)*

| Optional attribute | Description |
|---|---|
| CPUCritical | Specifies whether the CPU allocation is critical. A value of 0 indicates that the online agent function should proceed even when the required CPU was not successfully allocated.<br>Type-dimension: boolean-scalar<br>Default: 0<br>Example: 1 |

# Resource type definition

The resource type definition for this agent:

```
type MemCPUAllocator (
        static int NumThreads = 1
        static str ArgList[] = { ManagedSystem, HMC,
        MemoryRequired, MemoryCritical, CPUCritical, CPURequired, User }
        str ManagedSystem
        str HMC[]
        str User = "hscroot"
        str MemoryRequired
        str CPURequired
        boolean CPUCritical = 0
        boolean MemoryCritical = 0
)
```

# MemCPUAllocator agent notes

The MemCPUAllocator agent has the following notes:

- See "Configuring password free SSH communication between VCS nodes and HMC" on page 224.
- See "Dynamic resource allocation scenarios" on page 225.
- See "Configuring MemCPUAllocator" on page 228.

## Configuring password free SSH communication between VCS nodes and HMC

To use remote command operations on the HMC, you must have SSH installed on the DLPAR nodes in the VCS cluster. You must configure the HMC to allow

password free SSH access from these partitions. Refer to the appropriate IBM AIX documentation for information.

**To verify that you have password free SSH access**

- From each DLPAR in the cluster, execute the following command to test if the password free access works.

```
Eagle> ssh -l hscroot hmc2.veritas.com
Last login:Thur Jun 16 22:46:51 2005 from 10.182.9.34
hscroot@hmc2:~>
```

  Once each node can connect to the HMC using SSH without a password, you can start to use the MemCPUAllocator agent.

## Dynamic resource allocation scenarios

This section describes different examples of the resource allocation scenarios that the MemCPUAllocator agent can handle. For ease of explanation, consider only the memory resource in these examples. CPU resource implementation is similar.

Consider two DLPARs named Eagle and Vulture. These DLPARs are configured with the following minimum and maximum values memory values.

Table 5-16 summarizes the minimum and maximum memory for the DLPARs Eagle and Vulture.

**Table 5-16** The minimum and maximum memory for the DLPARs Eagle and Vulture

| DLPAR | Minimum | Maximum |
|---|---|---|
| Eagle | 512 MB | 2 GB |
| Vulture | 512 MB | 2 GB |

Two service groups SG1 and SG2 have the following resource requirements.

Table 5-17 summarizes the memory that is required for the service group SG1 and SG2.

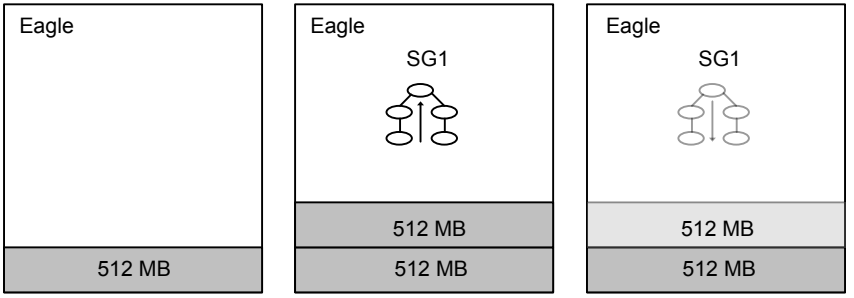
**Table 5-17** The memory that is required for service group SG1 and SG2

| Service group | Required memory |
|---|---|
| SG1 | 512 MB |
| SG2 | 512 MB |

## Scenario 1: A DLPAR node has minimum resources

Assume that the DLPARs start with the minimum values for memory. When SG1 is brought online on Eagle, the online agent function for the agent attempts to allocate 512 MB to Eagle from the free pool. The agent retains the minimum resources for the DLPAR's overhead operations and allocates resources for the service group in addition to the existing memory. For SG1 to come online the agent allocates an additional 512 MB to Eagle. After this allocation the total current memory for eagle is 1 GB. If SG1 goes offline, the agent deallocates the 512 MB that it allocated when the service group came online. This deallocation brings back the current memory of Eagle to 512 MB.

**Figure 5-6** Bringing a service group online and taking it offline on a DLPAR




Eagle DLPAR has no service groups.

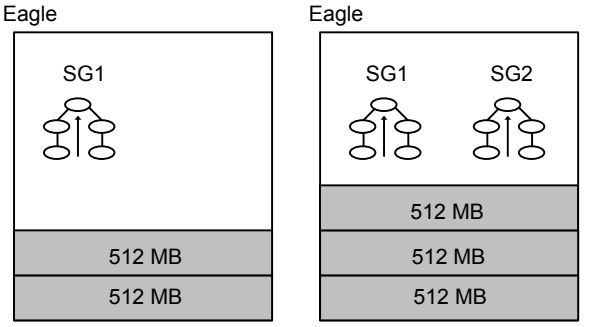
Service group 1 comes online on Eagle and boosts demand for memory.

Service group 1 goes offline and memory is returned to the resource pool.

## Scenario 2: Bringing another service group online

In this scenario, the Eagle DLPAR starts with 512 MB, and has SG1 online on it. It uses a total of 1 GB of memory. If SG2 is brought up on Eagle, the agent allocates an additional 512 MB of memory to Eagle. This reallocation brings the total memory to 1.5 GB.

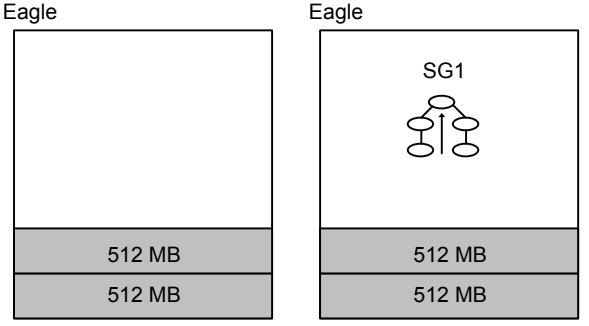
**Figure 5-7** Bringing another service group online on a DLPAR




## Scenario 3: DLPAR has required resources

Instead of starting with 512 MB, Eagle starts with 1 GB of initial memory. Eagle has 512 MB more than its minimum amount. If SG1 is brought online on Eagle, the agent determines that Eagle has an extra 512 MB more than its minimum. No service groups use this extra 512 MB. The agent does not allocate any additional memory to Eagle. SG1 is brought online on Eagle and the current memory for Eagle stays 1 GB.

**Figure 5-8** DLPAR Eagle starting with 1 GB of initial memory

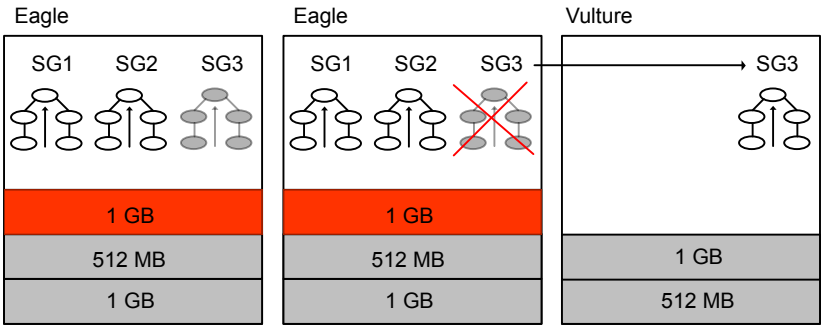



## Scenario 4: Cannot allocate required resources

Consider the stage in Scenario 2, where SG1 and SG2 are both online on Eagle, which brings its current memory to 1.5 GB. An additional service group SG3 enters the picture and requires 1 GB memory. SG3 tries to come up on Eagle. The agent determines that allocating 1 GB more memory to Eagle exceeds its maximum limit of 2 GB. The agent therefore does not allocate the memory and the online agent function fails, which leads to a resource fault. This resource fault makes the VCS engine stop the online of SG3 on Eagle and try it on Vulture. If Vulture starts with

512 MB and the agent allocates an additional 1 GB to Vulture, its current memory is 1.5 GB. SG3 can fail over and come online on Vulture.

**Figure 5-9** Exceeding the maximum amount of memory on a DLPAR




## Scenario 5: Service group failover

As in Scenario 2, SG1 and SG2 are both online on Eagle, which brings its current memory to 1.5 GB. Vulture has a current memory configuration of 512 MB.

If you switch the service groups from Eagle to Vulture:

- The MemCPUAllocator agent's offline agent function deallocates 1 GB from Eagle (512 MB for SG1 and 512 MB for SG2).
- The VCS engine migrates SG1 and SG2 to Vulture and the agent's online agent function allocates 1 GB to Vulture. This allocation brings Vulture's memory to 1.5 GB.

## Configuring MemCPUAllocator

Before you can use the MemCPUAllocator agent, you need to set up SSH access between the HMC and the DLPAR nodes. You must also make sure to configure the MemCPUAllocator resource as a leaf node in the service group's dependency tree in the main.cf file.

See Figure 5-5 on page 221.

Provide values to the MemCPUAllocator resource to specify the resource requirements for that service group. For example, if a service group needs 512 MB memory and two CPUs to start with, the MemCPUAllocator resource definition resembles:

```
MemCPUAllocator mymem (
    ManagedSystem @eagle = eagle-server
    ManagedSystem @vulture = vulture-server
```

```
    HMC = { testhmc }
    RequiredMemory = 512
    RequiredCPU = 2
    MemoryCritical = 1
    CPUCritical = 1
)
```

## Debug log levels

The MemCPUAllocator agent uses the following debug log levels:

DBG_1, DBG_2

# LPAR agent

The LPAR agent brings online, takes offline, and monitors AIX logical partitions (LPARs). The LPAR agent can also migrate the LPAR from one physical server to another physical server. VCS monitors and manages LPARs and uses the LPAR agent to make LPARs highly available. The LPAR which is a VCS node and controls other LPARs on the physical server is referred to as management LPAR.

Each managed system (physical server with AIX logical partitions) uses a Hardware Management Console (HMC) to manage the software configuration and operation of LPARs. The HMC also monitors and identifies hardware problems. The LPAR agent communicates with the HMCs to manage and check the status of LPARs.

The LPAR agent supports deployment of a redundant HMC. AIX virtualization supports two HMCs and you can use two HMCs in your LPAR resource configuration. If one HMC fails, the LPAR agent communicates with the other HMC. Hence, you need to set up password-less SSH access for HMCs on all VCS systems in the cluster.

See “Required attributes for LPAR agent” on page 231.

The LPAR agent also provides protection against failure of the virtual input output server (VIO server). You can run multiple VIO servers on a single system. If one VIO server fails, the LPAR agent enables communication through the other VIO servers. Disaster Recovery of LPARs is supported.

See “Optional attributes for LPAR agent” on page 232.

All the nodes of the cluster on which the LPAR is configured to failover must have access to the OS image.

See “ Notes for LPAR agent” on page 235.

**Note:** The PhysicalServer attribute of the system and the SysDownPolicy attribute of the group must be configured for LPAR resources.

For more information, see the following sections:

- Group attribute for LPAR agent
- System attribute for LPAR agent

## Dependencies for LPAR agent

There exists no dependency for the LPAR resource.

## Agent functions for LPAR agent

| | |
|---|---|
| Online | Starts the LPAR |
| Offline | Stops the LPAR |
| Monitor | Monitors the status of the LPAR |
| Clean | Stops the LPAR forcefully |
| Open | Blocks migration of the management LPAR |
| Shutdown | Unblocks migration of the management LPAR |
| Migrate | Migrates the LPAR from one physical server to another physical server |
| attr_changed | Creates Ethernet adapters required for the LPAR DR operation. This entry point is invoked when you modify the DROpts attribute. |

## State definitions for LPAR agent

| | |
|---|---|
| ONLINE | Indicates that the LPAR is in running state and the reference code is blank. |
| OFFLINE | Indicates that either the LPAR is not in running state or the LPAR is in running state and the reference code is not blank. |
| FAULTED | Indicates that the LPAR has gone down unexpectedly. |
| UNKNOWN | Indicates an invalid configuration. |

# Required attributes for LPAR agent

**Table 5-18**

| Required attribute | Description |
|---|---|
| LPARName | Name of the logical partition (LPAR).<br>Type-dimension: string-scalar<br>Default: NA<br>Example: "lpar1" |
| MCName | Names of the HMCs that manage LPARs.<br>The value of this attribute cannot be local to a management LPAR because all LPARs in a VCS cluster must use the same HMC names.<br>**Note:** The value is a string that can be IPv4 or IPV6 addresses or host names of HMCs.<br>Type-dimension: string-keylist<br>Default: NA<br>Example: { hmc1, hmc2 } |
| MCUser | Names of the users to communicate with HMCs using password-less SSH.<br>You must provide the MCName and MCUser value pairs in the proper order. For example, if you specify MCName as { hmc1, hmc2 } and MCUser as { hmcuser1, hmcuser2 }, hmcuser1 can communicate with hmc1 and hmcuser2 can communicate with hmc2.<br>Type-dimension: string-keylist<br>Default: { hscroot }<br>Example: { hmcuser1, hmcuser2 } |

# Optional attributes for LPAR agent

**Table 5-19**

| Optional attributes | Description |
|---|---|
| ProfileName | Name of the profile used to start LPAR. If you do not provide the profile name, the agent uses the default profile.<br>Type-dimension: string-scalar<br>Default: NA<br>Example: {lpar1} |
| VIOSName | Names of the virtual input output servers (VIO servers) that provide virtual resources to the LPAR. If there are multiple VIO servers available for an LPAR, you must specify names of all the VIO servers in the attribute so that VCS does not treat the LPAR as faulted till all the VIO servers go down.<br>If one VIO server fails, LPARs can perform input and output operations through the other VIO servers. When all the VIO servers specified in the VIOSName attribute are down, VCS fails over the managed LPAR to another system.<br>If you do not specify VIOSName, the LPAR agent does not fail over the LPAR after a VIO server crash.<br>Type-dimension: string-keylist<br>Default: NA<br>Example: {vios1, vios2 } |
| ProfileFile | Absolute path of the LPAR profile file on the management LPAR.<br>Type-dimension: string-scalar<br>Default: NA<br>Example: "/config/lpar-profile.profile"<br>For the steps to create a profile file, refer to the *Veritas InfoScale Virtualization Guide* for AIX. |
| RemoveProfileOnOffline | If this attribute is set to 1, LPAR configuration and its VIOS mappings (as specified in the profile file present in the path mentioned in the ProfileFile attribute) are deleted after the LPAR resource is brought offline or is faulted.<br>Type-dimension: boolean-scalar<br>Default: 0 |

**Table 5-19** *(continued)*

| Optional attributes | Description |
|---|---|
| DROpts | Defines the DR options. The value of this attribute consists of the following keys:<br>■ DNSSearchPath: The domain search path used by the LPAR in this site. The value of this key must contain a list of DNS domain names that are used for the DNS lookup of a hostname in case the domain name of the hostname is not specified. Use spaces to separate the domain names.<br>■ DNSServers: The list of DNS servers used by the LPAR in this site. The value of this key must contain a list of IP addresses of DNS servers that are used for the DNS lookup of a hostname. Use spaces to separate the IP addresses.<br>■ Gateway: The default gateway used by the LPAR in this site.<br>■ Device: Interface name on which to plumb the IP address.<br>■ IPAddress: The IP address to be assigned to the LPAR in this site after a cross-site failover.<br>■ Netmask: The netmask to be used by the LPAR in this site after a cross-site failover.<br>■ ConfigureNetwork: Setting used by the LPAR. The DROpts attribute value is applied to the LPAR only if this key is set to 1.<br>■ DomainName: Name of the domain for this site.<br>■ HostName: Host name of the LPAR to be applied after a cross-site failover.<br>If any of the keys are not specified or have default value, the `vcsnr` script retains all the configuration that is inside the LPAR on which this script is installed.<br>Type and dimension: string-association |

# Group attribute for LPAR agent

**Table 5-20**

| Group attribute | Description |
|---|---|
| SysDownPolicy | SysDownPolicy is a service group attribute that is relevant only if LPAR resources are configured.<br>For more information, refer to the *Veritas InfoScale Virtualization Guide* for AIX and the *Cluster Server Administrator's Guide*. |

# System attribute for LPAR agent

**Table 5-21**

| System attribute | Description |
|---|---|
| PhysicalServer | Name of the physical Power server on which LPAR is running.<br>For more information, refer to the *Cluster Server Administrator's Guide*. |

# Resource type definition for LPAR agent

The resource type definition for the agent is as follows:

```
type LPAR (
        static keylist RegList = { DROpts }
        static boolean IntentionalOffline = 1
        static int MigrateWaitLimit = 2
        static int MigrateTimeout = 600
        static str AgentFile = "/opt/VRTSvcs/bin/Script60Agent"
        static int CleanTimeout = 600
        static int OfflineTimeout = 600
        static int OfflineWaitLimit = 3
        static int OnlineTimeout = 600
        static int OnlineWaitLimit = 3
        static str ArgList[] = { LPARName, MCUser, MCName,
         ProfileName, State, IState, ProfileFile,
         RemoveProfileOnOffline, VIOSName, DROpts }
        static boolean AEPTimeout = 1
        static keylist SupportedOperations = { migrate }
        str LPARName
        str MCUser[] = { hscroot }
        str MCName[]
        str ProfileName
        str ProfileFile
        boolean RemoveProfileOnOffline = 0
        keylist VIOSName
        str DROpts{} = { ConfigureNetwork=0, IPAddress=NONE,
         Netmask=NONE, Gateway=NONE, DNSServers=NONE,
         DNSSearchPath=NONE, Device=NONE, Domain=NONE, HostName=NONE }
)
```

# Sample configurations for LPAR agent

## Minimal LPAR configuration

```
LPAR lpar1 {
          LPARName = testlpar1
          MCUser = { hscuser }
          MCName = { hmcserver1 }
}
```

## LPAR configuration with VIOS servers

```
LPAR lpar1 {
          LPARName = testlpar1
          MCUser = { hscuser }
          MCName = { hmcserver1 }
          VIOSName = {viosserver1 viosserver2}
}
```

## LPAR configuration for live migration

```
LPAR lpar1 {
          LPARName = testlpar1
          MCUser = { hscuser }
          MCName = { hmcserver1 }
          VIOSName = {viosserver1 viosserver2}
          ProfileFile = "/var/testlpar1.profile"
          RemoveProfileOnOffline = 1
}
```

# Notes for LPAR agent

The LPAR agent has the following notes:

## VCS requirements to manage the LPAR agent

Supported hardware and software versions:

- VIOS version 2.2.0.0 and above

- HMC version 7.7.0 and above
- Power5, Power6, or Power7

## Configuring password-less SSH communication between VCS nodes and HMC

To perform remote command operations on HMC:

- SSH must be installed on each LPAR that is a VCS system in the cluster.
- Configured HMCs must allow password-less SSH access from LPARs.

For more information, refer to the appropriate *IBM AIX documentation*.

## Verifying password-less SSH status

Verify if each VCS system can connect to HMC using password-less SSH

```
Eagle> ssh -l hscroot hmc2.domain.com
Last login:Thur Jun 16 22:46:51 2005 from 10.182.9.34
hscroot@hmc2:~>
```

You can start using the LPAR agent when each VCS system can connect to HMC using SSH without a password.

## Live Partition Mobility support for LPARs managed by VCS

Live Partition Mobility (LPM) functionality allows you to migrate an LPAR and the hosted applications from one physical server to another physical server.

VCS supports live migration of LPAR using the VCS command line interface (CLI).

You can use the `hagrp -migrate` command to migrate LPARs.

For more information, refer to the *Veritas InfoScale Virtualization Guide* for AIX.

VCS supports LPAR failover with live migration setup.

For more information about the ProfileFile and RemoveProfileOnOffline attributes, see Optional attributes for LPAR agent. For steps to create the profile file, refer to the *Veritas InfoScale Virtualization Guide* for AIX.

The LPAR agent blocks LPM functionality for the management LPAR that hosts VCS when it manages and monitors LPAR resources. When VCS stops managing LPARs, LPM functionality is available for the VCS system. The LPM functionality to migrate the management LPAR remains blocked if the LPAR agent crashes or is terminated.

**To allow live migration of the management LPAR**

1. Check if the `/usr/lib/dr/scripts/all/.vcs_blockmigrate.sh` file exists.
2. If the file exists, unblock LPM`/usr/sbin/drmgr -u vcs_blockmigrate.sh.`

### Setting the SysDownPolicy attribute for the LPAR service group

Veritas recommends to set the SysDownPolicy attribute for the LPAR service group. If the SysDownPolicy attribute is not set and the managed LPAR on one system goes down (but the physical server and managed LPAR are alive), VCS tries to start the managed LPAR on the other node. If the managed LPAR configuration exists on the other node, it leads to concurrency violation and might corrupt the managed LPAR data.

### Configuring the LPAR agent for DR in a Global Cluster environment

For information about configuring the LPAR agent for DR in a Global Cluster environment, refer to the *Veritas InfoScale Virtualization Guide*.

## Debug log levels for LPAR agent

The LPAR agent uses the following debug log levels:

DBG_1, DBG_2, DBG_3, DBG_4, DBG_5

Chapter 6

# Infrastructure and support agents

This chapter includes the following topics:


- About the infrastructure and support agents
- NotifierMngr agent
- Proxy agent
- Phantom agent
- RemoteGroup agent


## 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.

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 AIX

| 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.<br>**Note:** SnmpConsoles is a required attribute if SmtpServer is not specified; otherwise, SnmpConsoles is an optional attribute. Specify both SnmpConsoles and SmtpServer if desired.<br>Type and dimension: string-association<br>Example:<br>"172.29.10.89" = Error, "172.29.10.56" = Information |

**Table 6-1** Required attributes for AIX *(continued)*

| Required attribute | Description |
|---|---|
| SmtpServer | Specifies the machine name of the SMTP server.<br>**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.<br>Type and dimension: string-scalar<br>Example: "smtp.example.com" |

**Table 6-2** Optional attributes for AIX

| Optional attribute | Description |
|---|---|
| MessagesQueue | Size of the VCS engine's message queue. Minimum value is 30.<br>Type and dimension: integer-scalar<br>Default: 30 |
| NotifierListeningPort | Any valid, unused TCP/IP port number.<br>Type and dimension: integer-scalar<br>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.<br>**Note:** Make sure that the Source IP given in this attribute is present in the /etc/hosts file or is DNS-resolvable.<br>Type and dimension: string-scalar<br>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.<br>Type and dimension: string-scalar<br>Example: "usera@example.com" |

**Table 6-2** Optional attributes for AIX *(continued)*

| Optional attribute | Description |
|---|---|
| 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.<br>**Note:** SmtpRecipients is a required attribute if you specify SmtpServer.<br>Type and dimension: string-association<br>Example:<br>`"james@example.com" = SevereError,`<br>`"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.<br>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.<br>Type and dimension: string-scalar<br>Example: "usera@example.com" |
| 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.<br>Type and dimension: integer-scalar<br>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.<br>Type and dimension: boolean-scalar<br>Default: 0 |

**Table 6-2** Optional attributes for AIX *(continued)*

| Optional attribute | Description |
|---|---|
| SnmpCommunity | Specifies the community ID for the SNMP manager.<br>Type and dimension: string-scalar<br>Default: public |
| SnmpdTrapPort | Port on the SNMP console machine where SNMP traps are sent.<br>If you specify more than one SNMP console, all consoles use this value.<br>Type and dimension: integer-scalar<br>Default: 162 |
| EngineListeningPort | Change this attribute if the VCS engine is listening on a port other than its default port.<br>Type and dimension: integer-scalar<br>Default: 14141 |
| 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.<br>Minimum value: 3600<br>Type and dimension: integer-scalar<br>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, south }
    AutoStartList = { north }
    Parallel = 1
    )
```

```
    Phantom my_phantom (
    )

    NIC    NicGrp_en0 (
        Enabled = 1
        Device  = en0
        NetworkType = ether
        )

group Grp1 (
    SystemList = { north, south }
    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
    //              }
    //      }
```

## IPv6 configuration for AIX

While the NotifierMngr resource can work without the NIC resource, Veritas recommends this dependency.

If the “en0” is a virtual device on AIX, then the NetworkHosts attribute is required, otherwise this resource takes an UNKNOWN state.

```
group ClusterService (
    SystemList = { sysA = 0, sysB = 1 }
    AutoStartList = { sysA, sysB }
    OnlineRetryLimit = 3
    OnlineRetryInterval = 120
    )

    NIC csgnic (
        Device = en0
        NetworkHosts = {"fe80::88c4:e0ff:fe00:c002"}
        )

    NotifierMngr ntfr (
        SnmpConsoles = { "3ffe:556::1000:5761" = SevereError }
        SmtpServer = "sample.domain.com"
        SmtpRecipients = { "user@domain.com" =
        SevereError }
        )
        ntfr requires csgnic
```

## Debug log levels

The NotifierMngr agent uses the following debug log levels:

DBG_1, DBG_2, DBG_3, DBG_5

# Proxy agent

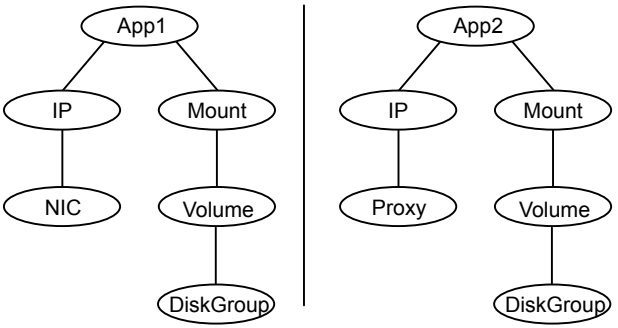
The Proxy agent mirrors the state of another resource on a local or remote system. It provides a means to specify and modify one resource and have its state reflected by its proxies. You can use the agent when you need to replicate the status of a resource.

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.<br>The target resource must be in a different resource group than the Proxy resource.<br>Type and dimension: string-scalar<br>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.<br>Type and dimension: string-scalar<br>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; note that target resource is in grp1, and the proxy is in grp2; a target resource and its proxy cannot be in the same group.

```
group grp1 (
    SystemList = { sysa, sysb }
    AutoStartList = { sysa }
    )

MultiNICA mnic (
        Device@sysa = { en0 = "10.128.8.42", en1 = "10.128.8.42" }
        Device@sysb = { en0 = "10.128.8.43", en1 = "10.128.8.43" }
        NetMask = "255.255.255.0"
        Gateway = "10.128.8.1"
```

```
        BroadcastAddr = "10.128.8.255"
        Options = "mtu 1400"
        )

        IPMultiNIC ip1 (
            Address = "10.128.8.78"
            NetMask = "255.255.255.0"
            MultiNICAResName = mnic
            Options = "mtu 1400"
            )
    ip1 requires mnic

group grp2 (
        SystemList = { sysa, sysb }
        AutoStartList = { sysa }
        )

        IPMultiNIC ip2 (
            Address = "10.128.8.79"
            NetMask = "255.255.255.0"
            MultiNICAResName = mnic
            Options = "mtu 1400"
            )
        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.

---

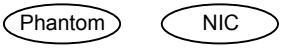
**Note:** Do not attempt manual online or offline operations on the Phantom resource at the resource level. Do not use `hares` commands on the Phantom resource at the resource level. Unpredictable behavior results when you try a manual online or offline procedure or an `hares` command on a Phantom resource. 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:<br>See Table 6-5 on page 252. |
| Offline | Takes the remote service group offline. For more information:<br>See Table 6-5 on page 252. |
| Monitor | Monitors the state of the remote service group.<br>The true state of the remote service group is monitored only on the online node in the local cluster. For more information:<br>See Table 6-5 on page 252. |
| Clean | If the RemoteGroup resource faults, the Clean function takes the remote service group offline. For more information:<br>See Table 6-5 on page 252. |

## State definitions

| | |
|---|---|
| ONLINE | Indicates that the remote service group is in an ONLINE state.<br>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.<br>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.<br>When configuring a virtual IP address of a remote cluster, do not configure the IP resource as a part of the remote service group.<br>Type and dimension: string-scalar<br>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.<br>See Table 6-6 on page 255. |
| GroupName | The name of the service group on the remote cluster that you want the RemoteGroup agent to monitor or manage.<br>Type and dimension: string-scalar<br>Example: "DBGrp" |

**Table 6-5** Required attributes *(continued)*

| Required attribute | Description |
|---|---|
| VCSSysName | You must set this attribute to either the VCS system name or the ANY value.<br>■ ANY<br>The RemoteGroup resource goes online if the remote service group is online on any node in the remote cluster.<br>■ *VCSSysName*<br>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.<br>Type and dimension: string-scalar<br>Example: "vcssys1" or "ANY" |
| ControlMode | Select only one of these values to determine the mode of operation of the RemoteGroup resource: MonitorOnly, OnlineOnly, or OnOff.<br>■ OnOff<br>The RemoteGroup resource brings the remote service group online or takes it offline.<br>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.<br>■ MonitorOnly<br>The RemoteGroup resource only monitors the state of the remote service group. The RemoteGroup resource cannot online or offline the remote service group.<br>Make sure that you bring the remote service group online before you online the RemoteGroup resource.<br>■ OnlineOnly<br>The RemoteGroup resource only brings the remote service group online. The RemoteGroup resource cannot take the remote service group offline.<br>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.<br>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.<br><br>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.<br><br>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.<br><br>For a secure remote cluster:<br><br>■ Local Unix user<br>user@nodename—where the nodename is the name of the node that is specified in the IpAddress attribute. Do not set the DomainType attribute.<br>■ NIS or NIS+ user<br>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.<br><br>Type and dimension: string-scalar<br><br>Example:<br><br>■ For a cluster without the Veritas Product Authentication Service: "johnsmith"<br>■ 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 with the `vcsencrypt -agent` command.<br><br>**Note:** Do not use the vcsencrypt utility when entering passwords from a configuration wizard or the Cluster Manager (Java Console).<br><br>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.<br>For users who have the domain type unixpwd, you do not have to set this attribute.<br>Type: string-scalar<br>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.<br>Type: string-scalar<br>Example: "128.11.295.51" |
| Port | The port where the remote engine listens for requests.<br>This is an optional attribute, unless the remote cluster listens on a port other than the default value of 14141.<br>Type and dimension: integer-scalar<br>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.<br>Type and dimension: integer-scalar<br>Default: 0 |

**Table 6-6** Optional attributes *(continued)*

| Optional attribute | Description |
|---|---|
| ReturnIntOffline | Select one of the following values for RemoteGroup to return IntentionalOffline:<br>■ RemotePartial—Indicates that the RemoteGroup resource returns an IntentionalOffline if the remote service group is in an ONLINE\|PARTIAL state.<br>■ RemoteOffline—Indicates that the RemoteGroup resource returns an IntentionalOffline if the remote service group is in an OFFLINE state.<br>■ RemoteFaulted—Indicates that the RemoteGroup resource returns an IntentionalOffline if the remote service group is OFFLINE\|FAULTED.<br>You can use these values in combinations with each other.<br>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*.<br>Type and dimension: string-vector<br>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<br>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.<br>See the *Cluster Server Administrator's Guide* for more information about these attributes. |
| ToleranceLimit<br>MonitorInterval | If you expect the RemoteGroup agent to support sudden offline of the remote service group, modify the ToleranceLimit attribute.<br>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.<br>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
- ElifNone agent
- FileNone agent
- FileOnOff agent
- FileOnOnly agent


## 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

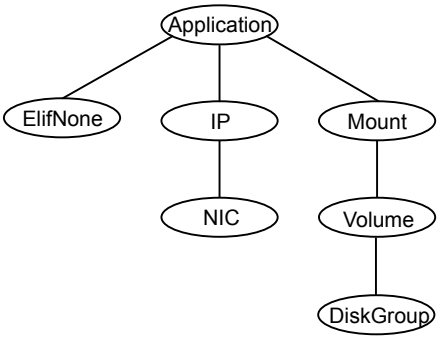
The ElifNone agent monitors a file. It checks for the file's absence.

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.<br>Type and dimension: string-scalar<br>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

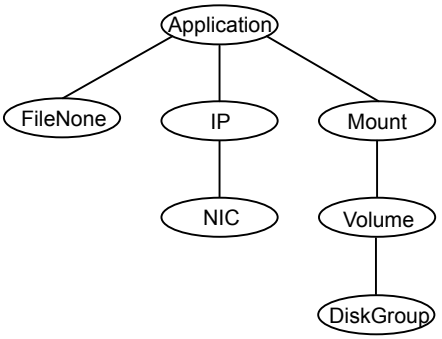
Monitors a file, checks for the file's existence.

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.<br>Type and dimension: string-scalar<br>Example: "/tmp/file01" |

## Resource type definition for FileNone agent

```
type FileNone (
    static int AutoRestart = 1
    static int OfflineMonitorInterval = 60
    static str ArgList[] = { PathName }
    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

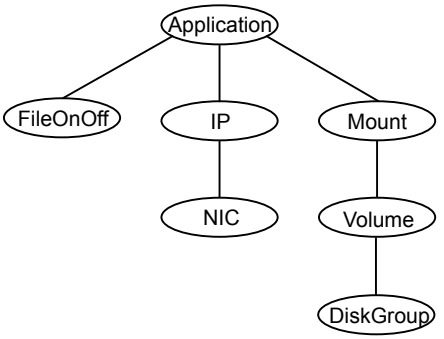
The FileOnOff agent creates, removes, and monitors a file.

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.<br>Type and dimension: string-scalar<br>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

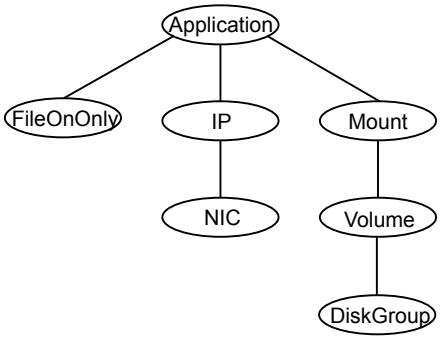
The FileOnOnly agent creates and monitors a file.

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.<br>Type and dimension: string-scalar<br>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

Chapter 8

# Replication agents

This chapter includes the following topics:


- About the replication agents
- RVG agent
- RVGPrimary agent
- RVGSnapshot
- RVGShared agent
- RVGLogowner agent
- RVGSharedPri agent


## 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:

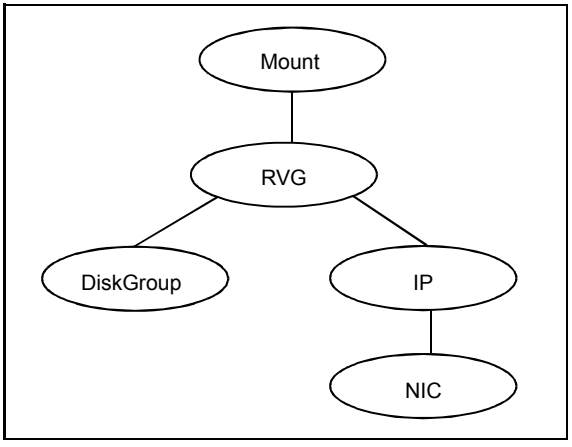
/etc/VRTSvcs/conf/sample_vvr/RVG

## 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.<br>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.<br>Type and dimension: string-scalar<br>Example: "hr_rvg" |
| DiskGroup | The disk group that this RVG is associated with.<br>Type and dimension: string-scalar<br>Example: "hrbg" |

**Table 8-1** Required attributes *(continued)*

| Attribute | Description |
|---|---|
| StorageDG | The name of the bunker disk group.<br>Type and dimension: string-scalar<br>Example: "hr_bdg" |
| StorageRVG | The name of the bunker RVG.<br>Type and dimension: string-scalar<br>Example: "hr_brvg" |
| StorageHostIds | A space-separated list of the hostids of each node in the bunker cluster.<br>Type and dimension: string-keylist<br>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.<br>Do not modify this attribute for this agent.<br>Setting this attribute to a higher value may result in agent function timeouts due to serialization of underlying commands.<br>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.

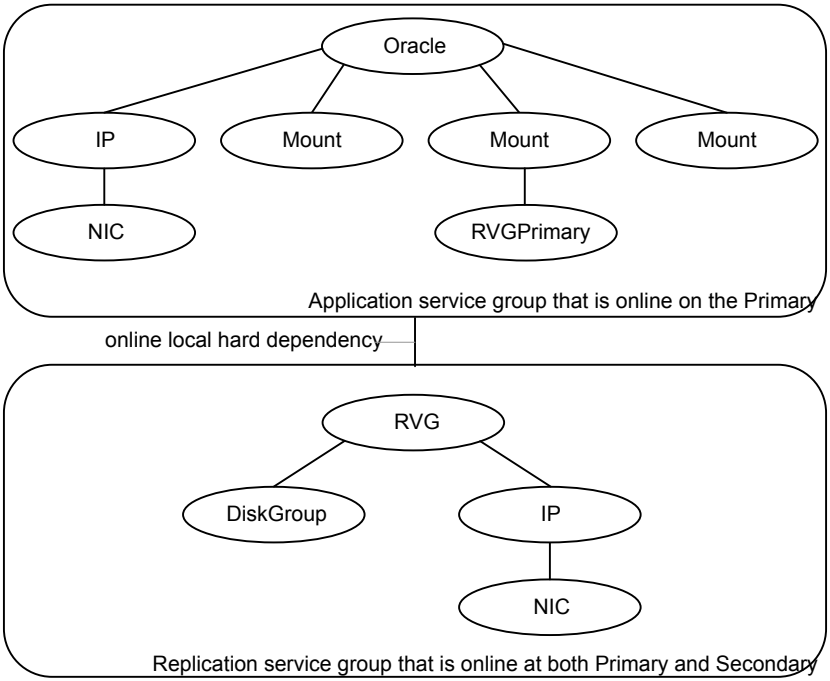
A sample configuration file for this agent that you can use as a guide to create the configuration is located at /etc/VRTSvcs/conf/sample_vvr/RVGPrimary.

## 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




## 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.<br>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.<br>It can be executed to retain the specified RVG as the Primary in a Primary-Primary configuration.<br>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.<br>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.<br>AutoTakeover and AutoResync are mutually exclusive attributes.<br>When AutoTakeover=0, the primary-elect feature is not applicable; therefore, it is not supported.<br>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.<br>You can use the following values for this attribute:<br>■ 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.<br>■ 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.<br>■ 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.<br>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.<br>AutoTakeover and AutoResync are mutually exclusive attributes.<br>When AutoTakeover=0, the primary-elect feature is not applicable; therefore, it is not supported.<br>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.<br><br>Use one of the following values for the BunkerSyncTimeOut attribute:<br><br>■ 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.<br>■ 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.<br>■ 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.<br><br>The RVGPrimary agent's OnlineTimeout and OnlineRetryLimit attribute values determine the available time for an RVGPrimary resource to complete its online operation.<br><br>Use the following formula to get the Time Available for Online to Complete (TAOC):<br><br>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.<br><br>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.<br><br>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).<br><br>Type and dimension: string-scalar<br><br>Default value: "" |

**Table 8-4** 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.<br><br>Do not modify this attribute for this agent.<br><br>Setting this attribute to a higher value may result in agent function timeouts due to serialization of underlying commands.<br><br>Default: 1 |

**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.<br>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
    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.

---

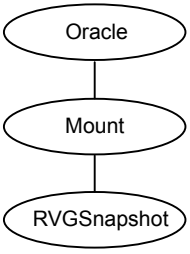
**Note:** The RVGSnapshot agent does not support Volume Sets.

---

## 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.<br>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<br>Type and dimension: string-scalar |
| Prefix | Token put before the name of the actual volume when creating the snapshotted volumes.<br>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).<br>Type and dimension: integer-scalar<br>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.<br>Type and dimension: string-scalar |
| NumThreads | Number of threads used within the agent process for managing resources. This number does not include threads used for other internal purposes.<br>Do not modify this attribute for this agent.<br>Setting this attribute to a higher value may result in agent function timeouts due to serialization of underlying commands.<br>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

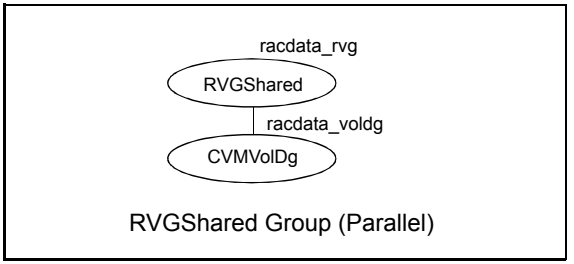
The RVGShared resource represents the RVG of the RDS. The RVGShared resource is dependent on the CVMVolDg resource.

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.<br>Type and dimension: string-scalar |
| DiskGroup | The shared-disk group with which this RVG is associated.<br>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.<br>Do not modify this attribute for this agent.<br>Setting this attribute to a higher value may result in agent function timeouts due to serialization of underlying commands.<br>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.

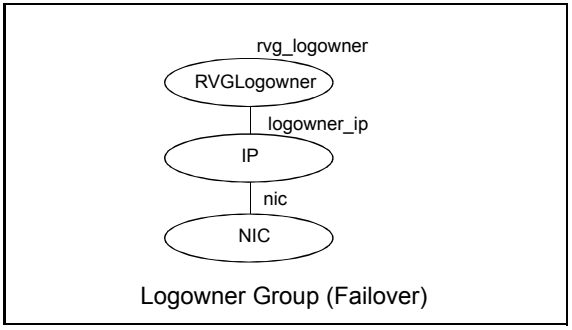
## Dependencies

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.<br>Type and dimension: string-scalar<br>Example: "hr_rvg" |
| DiskGroup | The disk group with which this RVG is associated.<br>Type and dimension: string-scalar<br>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.<br>Do not modify this attribute for this agent.<br>Setting this attribute to a higher value may result in agent function timeouts due to serialization of underlying commands.<br>Default: 1 |

**Table 8-12** Internal attributes

| Attribute | Description |
|---|---|
| StorageDG | For internal use only, do not modify. The name of the bunker disk group.<br>Type and dimension: string-scalar<br>Example: "hr_bdg" |
| StorageRVG | For internal use only, do not modify. The name of the bunker RVG.<br>Type and dimension: string-scalar<br>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.<br>Type and dimension: string-keylist<br>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 fileset, and can be used as a guide when creating your configuration.

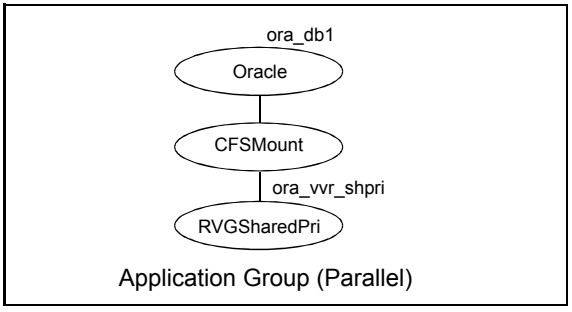
# Dependencies

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.<br>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.<br>Type and dimension: integer-scalar<br>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.<br>Type and dimension: integer-scalar<br>Default: 0 |

**Table 8-13** Required attributes *(continued)*

| Attribute | Description |
|---|---|
| VCSResLock | This attribute is reserved for internal use by VCS.<br>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.<br>Do not modify this attribute for this agent.<br>Setting this attribute to a higher value may result in agent function timeouts due to serialization of underlying commands.<br>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
)
```

# Index

## Symbols

802.1Q trunking 93

## A

about
    Network agents 92
    Samba agents 162
agent
    modifying 15
agent functions 160
    Apache HTTP server agent 177
    Application agent 191
    CoordPoint agent 202
    DiskGroup agent 20
    DiskGroupSnap agent 33
    DNS agent 131
    ElifNone agent 259
    FileNone agent 261
    FileOnOff agent 263
    FileOnOnly agent 265
    IP agent 95
    IPMultiNIC agent 105
    IPMultiNICB agent 118
    LVMVG agent 53
    MemCPUAllocator agent 222
    Mount agent 67
    MultiNICA agent 110
    MultiNICB agent 124
    NetBIOS agent 171
    NFS agent 148
        AIX 148
    NFSRestart agent 154
    NIC agent 100
    NotifierMngr agent 239
    Phantom agent 249
    Process agent 209
    ProcessOnOnly agent 212
    Proxy agent 246
    RemoteGroup agent 251
    SambaServer agent 164
    SambaShare agent 168
agent functions *(continued)*
    Share agent 160
    Volume agent 47
    VolumeSet agent 50
    Zone agent 216
agents. *See* RVG agent
    Apache HTTP server 175
    Application 189
    CoordPoint 201
    DiskGroup 19
    DiskGroupSnap 31
    DNS 130
    ElifNone 258
    FileNone 260
    FileOnOff 262
    FileOnOnly 264
    IP 94
    IPMultiNIC 105
    IPMultiNICB 117
    LVMVG 52
    MemCPUAllocator 221
    Mount 66
    MultiNICA 109
    NetBIOS 170
    NFS 147
    NFSRestart 152
    NIC 99
    NotifierMngr 238
    Phantom 248
    Process 207
    ProcessOnOnly 212
    Proxy 245
    RemoteGroup 250
    RVGLogowner. *See* RVGLogowner agent
    RVGPrimary. *See* RVGPrimary agent
    RVGShared. *See* RVGShared agent
    RVGSharedPri. *See* RVGSharedPri agent
    RVGSnapshot. *See* RVGSnapshot agent
    SambaServer 164
    SambaShare 168
    Share 159
    Volume 47

agents *(continued)*
Zone 215
Apache HTTP server agent
agent functions 177
attributes 178
description 175
detecting application failure 184
sample command output 188
sample configuration 186
state definitions 178
Application agent
agent functions 191
AIX attributes 193
attributes 193
description 189
high availability fire drill 190
resource type definition 196
sample configurations 200
state definitions 193
association dimension 15
attribute data types 15
attributes
Application agent 193
AIX 193
CoordPoint agent 204
DiskGroup agent
AIX 23
DiskGroupSnap agent 34
DNS agent 133
ElifNone agent 259
FileNone agent 261
FileOnOff agent 264
FileOnOnly agent 266
IP agent
AIX 96
IPMultiNIC agent 106
AIX 106
IPMultiNICB agent 120
AIX 120
modifying 15
Mount agent 70
AIX 70
MultiNICA agent 110, 222
MultiNICB agent 124
NFS agent 149
AIX 149
NFSRestart agent 155
AIX 155
attributes *(continued)*
NIC agent 101
AIX 101
NotifierMngr agent 239
AIX 239
Process agent 210
AIX 210
ProcessOnOnly
AIX 213
ProcessOnOnly agent 213
Proxy agent 246
RemoteGroup agent 252
SambaServer agent 165
Share agent 161
AIX 161
Volume agent 48
VolumeSet agent 51
AutoResync attribute
RVGPrimary agent 290

## B

boolean data types 15

## C

Checklist to ensure the proper operation of MultiNICB 117
Cluster Manager (Java Console)
modifying attributes 15
CNAME record 141
configuration files
main.cf 249
modifying 15
types.cf 15
CoordPoint agent
agent functions 202
attributes 204
description 201
resource type definition 205
sample configurations 207
state definitions 202

## D

data type
boolean 15
string 15
data types
integer 15

dependency graphs
RVGLogowner agent 285
RVGPrimary agent 272
RVGShared agent 282
RVGSharedPri agent 289
description
resources 15
dimensions
keylist 15
scalar 15
vector 15
DiskGroup agent
agent functions 20
AIX attributes 23
description 19
high availability fire drill 28
resource type definition 27
sample configurations 31
state definitions 22
DiskGroupSnap agent
agent functions 33
attributes 34
description 31
resource type definition 40
sample configurations 40
state definitions 33
DNS agent 132
agent functions 131
attributes 133
description 130
resource type definition 138
sample web server configuration 142

## E

ElifNone agent
agent functions 259
attributes 259
description 258
resource type definition 260
sample configuration 260
state definitions 259

## F

failover group
RVGLogowner agent 285
fast failback
AutoResync attribute of RVGPrimary 290
fast failback resynchronization
RVGPrimary 271
RVGSharedPri 288
Fiber Channel adapter 29
FileNone agent
agent functions 261
attribute 261
description 260
resource type definition 262
sample configurations 262
state definitions 261
FileOnOff agent
agent functions 263
attribute 264
description 262
state definitions 263
FileOnOnly agent
agent functions 265
attribute 266
description 264
resource type definition 266
sample configuration 266
state definitions 265
fire drill
RVGSnaphot agent 279

## H

haipswitch utility 118
AIX 118
high availability fire drill 28, 79, 94, 103, 141, 157, 190, 208

## I

integer data types 15
IP agent
agent functions 95
AIX attributes 96
description 94
high availability fire drill 94
resource type definitions 98
sample configurations 99
state definitions 95
IPMultiNIC agent
agent functions 105
AIX attributes 106
attributes 106
description 105
resource type definitions 107

IPMultiNIC agent *(continued)*
sample configuration 108
state definitions 106
IPMultiNICB agent 122
agent functions 118
AIX attributes 120
attributes 120
description 117
requirements 118
resource type definition 121
state definitions 119

## K

keylist dimension 15

## L

logowner
virtual IP requirement 285
LVMVG agent
agent functions 53
attributes 55
autoactivate options 62
description 52
hadevice utility 64
importing volume group 60
JFS 60
JFS or JFS2 support 60
JFS2 60
major numbers 61
resource type definition 57
sample configurations 65
state definitions 54
Subsystem Device Driver support 63
SyncODM attribute 61
varyonvg options 60
LVMVG notes 57

## M

main.cf 15, 249
main.xml 15
MemCPUAllocator agent
agent functions 222
description 221
migrating
RVGPrimary 271
RVGSharedPri 288
modifying
configuration files 15
modifying agents 15
monitor scenarios
DNS agent 141
Mount agent
agent functions 67, 69
AIX attributes 70
attributes 70
description 66
high availability fire drill 79, 141, 157
notes 78
offline 82
resource type definition 77
sample configurations 84
MultiNICA agent 114
agent functions 110
attributes 110, 222
description 109
resource type attributes 113
resource type definitions 224
sample configurations 115
state definitions 110
MultiNICB agent 124
agent functions 124
attributes 124
resource type definition 128
sample configurations 129
state definitions 124

## N

NetBIOS agent
agent functions 171
description 170
resource type definition 172
sample configurations 174
state definitions 171
NFS agent
agent functions 148
AIX 148
attributes 149
AIX 149
description 147
resource type definition 151
sample configurations 152
state definitions 149
NFSRestart agent
agent functions 154
attributes 155
AIX 155
description 152

NFSRestart agent *(continued)*
resource type definition 156
sample configuration 158
state definitions 155
NIC agent
agent functions 100
attributes 101
AIX 101
description 99
high availability fire drill 103
resource type definitions 103
sample configurations 103
state definitions 101
noautoimport flag 29
Notes on using NFSv4 152
NotifierMngr agent
agent functions 239
AIX attributes 239
attributes 239
description 238
resource type definition 242
sample configurations 243
state definitions 239

## O

offline
Mount agent 82
online query 141

## P

parallel group
RVGShared agent 282
Phantom agent
agent functions 249
description 248
resource type definition 249
sample configurations 249
prerequisites
Samba agents 163
Process agent
agent functions 209
AIX attributes 210
attributes 210
description 207
high availability fire drill 208
resource type definition 210
sample configurations 211
state definitions 209
ProcessOnOnly agent
agent functions 212
AIX attributes 213
attributes 213
description 212
resource type definition 214
sample configurations 214
state definitions 213
Proxy agent
agent functions 246
attributes 246
description 245
resource type definition 247
sample configurations 247

## R

RemoteGroup agent
agent functions 251
attributes 252
description 250
resource type definition 257
state definitions 251
resource type definition 49
SambaShare agent 169
resource type definitions
Application agent 196
CoordPoint agent 205
DiskGroup agent 27
DiskGroupSnap agent 40
DNS agent 138
ElifNone agent 260
FileNone agent 262
FileOnOnly agent 266
IP agent 98
IPMultiNIC agent 107
IPMultiNICB agent 121
LVMVG agent 57
Mount agent 77
MultiNICA agent 113, 224
MultiNICB agent 128
NetBIOS agent 172
NFS agent 151
NFSRestart agent 156
NIC agent 103
NotifierMngr agent 242
Phantom agent 249
Process agent 210
ProcessOnOnly agent 214
Proxy agent 247

resource type definitions *(continued)*
RemoteGroup agent 257
SambaServer agent 167
Share agent 161
Volume agent 49
Zone agent 218
resources
description of 15
RVG agent
described 267
RVGLogowner agent
dependency graph 285
described 285
failover group 285
RVGPrimary agent
dependency graph 272
described 271
migrating 271
takeover 271
RVGShared agent
dependency graph 282
described 282
parallel group 282
RVGSharedPri agent
dependency graph 289
described 288
migrating 288
takeover 288
RVGSnapshot agent
described 279
fire drill 279

## S

Samba agents 162
overview 162
prerequisites 163
SambaServer agent
agent functions 164
attributes 165
description 164
resource type definition 167
sample configuration 167
state definitions 165
SambaShare agent 168
agent functions 168
attributes 169
resource type definition 169
sample configurations 170
state definitions 169
sample configurations 122
Apache HTTP server agent 186
Application agent 200
CoordPoint agent 207
DiskGroup agent 31
DiskGroupSnap agent 40
ElifNone agent 260
FileNone agent 262
FileOnOff agent 264
FileOnOnly agent 266
IP agent 99
IPMultiNIC 108
IPMultiNICB agent 122
LVMVG agent 65
Mount agent 84
MultiNICA agent 115
MultiNICB agent 129
NetBIOS agent 174
NFS agent 152
NFSRestart agent 158
NIC agent 103
NotifierMngr agent 243
Phantom agent 249
Process agent 211
ProcessOnOnly agent 214
Proxy agent 247
SambaServer agent 167
SambaShare agent 170
Share agent 162
Volume agent 49
scalar dimension 15
secure DNS update 142
Share agent 160
agent functions 160
attributes 161
AIX 161
description 159
resource type definitions 161
sample configurations 162
state definitions 160
snapshots
using RVGSnapshot agent for 279
State definitions
VolumeSet agent 50
state definitions 132
Apache HTTP server agent 178
Application agent 193
CoordPoint agent 202
DiskGroup agent 22

state definitions *(continued)*
DiskGroupSnap agent 33
DNS agent 132
ElifNone agent 259
FileNone agent 261
FileOnOff agent 263
FileOnOnly agent 265
IP agent 95
IPMultiNIC agent 106
IPMultiNICB agent 119
LVMVG agent 54
Mount agent 69
MultiNICA agent 110
MultiNICB agent 124
NetBIOS agent 171
NFS agent 149
NFSRestart agent 155
NIC agent 101
NotifierMngr agent 239
Process agent 209
ProcessOnOnly agent 213
RemoteGroup agent 251
SambaServer agent 165
SambaShare agent 169
Share agent 160
Volume agent 48
string data type 15

## T

takeover
RVGPrimary 271
RVGSharedPri 288
trigger script 128
trunking 93
types.cf 15

## V

varyoffvg command 59
VCS
resource types 15
vector dimension 15
virtual IP
RVGLogowner agent requirement 285
Volume agent
agent functions 47
attributes 48
description 47
sample configurations 49
Volume agent *(continued)*
state definitions 48
volume sets 29
VolumeSet agent
agent functions 50
attributes 51
State definitions 50

## Z

Zone agent
agent functions 216
attributes 216
description 215
resource type definition 218