

Veritas™ Cluster Server Bundled Agents Reference Guide

Linux

5.0 Maintenance Pack 3



Veritas Cluster Server Bundled Agents Reference Guide

The software described in this book is furnished under a license agreement and may be used only in accordance with the terms of the agreement.

Product version: 5.0 MP3

Document version: 5.0MP3.0

Legal Notice

Copyright © 2008 Symantec Corporation. All rights reserved.

Symantec, the Symantec Logo, Veritas and Veritas Storage Foundation are trademarks or registered trademarks of Symantec Corporation or its affiliates in the U.S. and other countries. Other names may be trademarks of their respective owners.

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 Symantec Corporation 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. SYMANTEC CORPORATION 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, "Rights in Commercial Computer Software or Commercial Computer Software Documentation", as applicable, and any successor regulations. 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.

Symantec Corporation
20330 Stevens Creek Blvd.
Cupertino, CA 95014
<http://www.symantec.com>

Technical Support

Symantec Technical Support maintains support centers globally. Technical Support's primary role is to respond to specific queries about product features and functionality. The Technical Support group also creates content for our online Knowledge Base. The Technical Support group works collaboratively with the other functional areas within Symantec to answer your questions in a timely fashion. For example, the Technical Support group works with Product Engineering and Symantec Security Response to provide alerting services and virus definition updates.

Symantec's maintenance offerings include the following:

- A range of support options that give you the flexibility to select the right amount of service for any size organization
- Telephone and Web-based support that provides rapid response and up-to-the-minute information
- Upgrade assurance that delivers automatic software upgrade protection
- Global support that is available 24 hours a day, 7 days a week
- Advanced features, including Account Management Services

For information about Symantec's Maintenance Programs, you can visit our Web site at the following URL:

www.symantec.com/techsupp

Contacting Technical Support

Customers with a current maintenance agreement may access Technical Support information at the following URL:

http://www.symantec.com/business/support/assistance_care.jsp

Before contacting Technical Support, make sure you have satisfied the system requirements that are listed in your product documentation. Also, you should be at the computer on which the problem occurred, in case it is necessary to replicate the problem.

When you contact Technical Support, please have the following information available:

- Product release level
- Hardware information
- Available memory, disk space, and NIC information
- Operating system
- Version and patch level
- Network topology
- Router, gateway, and IP address information
- Problem description:

- Error messages and log files
- Troubleshooting that was performed before contacting Symantec
- Recent software configuration changes and network changes

Licensing and registration

If your Symantec product requires registration or a license key, access our technical support Web page at the following URL:

www.symantec.com/techsupp

Customer service

Customer service information is available at the following URL:

www.symantec.com/techsupp

Customer Service is available to assist with the following types of issues:

- Questions regarding product licensing or serialization
- Product registration updates, such as address or name changes
- General product information (features, language availability, local dealers)
- Latest information about product updates and upgrades
- Information about upgrade assurance and maintenance contracts
- Information about the Symantec Buying Programs
- Advice about Symantec's technical support options
- Nontechnical presales questions
- Issues that are related to CD-ROMs or manuals

Documentation feedback

Your feedback on product documentation is important to us. Send suggestions for improvements and reports on errors or omissions to clustering_docs@symantec.com.

Include the title and document version (located on the second page), and chapter and section titles of the text on which you are reporting.

Maintenance agreement resources

If you want to contact Symantec regarding an existing maintenance agreement, please contact the maintenance agreement administration team for your region as follows:

Asia-Pacific and Japan	contractsadmin@symantec.com
Europe, Middle-East, and Africa	semea@symantec.com
North America and Latin America	supportsolutions@symantec.com

Additional enterprise services

Symantec offers a comprehensive set of services that allow you to maximize your investment in Symantec products and to develop your knowledge, expertise, and global insight, which enable you to manage your business risks proactively.

Enterprise services that are available include the following:

Symantec Early Warning Solutions	These solutions provide early warning of cyber attacks, comprehensive threat analysis, and countermeasures to prevent attacks before they occur.
Managed Security Services	These services remove the burden of managing and monitoring security devices and events, ensuring rapid response to real threats.
Consulting Services	Symantec Consulting Services provide on-site technical expertise from Symantec and its trusted partners. Symantec Consulting Services offer a variety of prepackaged and customizable options that include assessment, design, implementation, monitoring, and management capabilities. Each is focused on establishing and maintaining the integrity and availability of your IT resources.
Educational Services	Educational Services provide a full array of technical training, security education, security certification, and awareness communication programs.

To access more information about Enterprise services, please visit our Web site at the following URL:

www.symantec.com

Select your country or language from the site index.

Contents

Chapter 1	Introduction	
	Resources and their attributes	17
	Modifying agents and their resources	18
	Attributes	18
Chapter 2	Storage agents	
	About the storage agents	21
	DiskGroup agent	22
	Dependencies	22
	Agent functions	23
	State definitions	24
	Attributes	25
	Resource type definition	27
	DiskGroup agent notes	28
	High availability fire drill	28
	Configuring the Fiber Channel adapter	28
	Sample configurations	28
	DiskGroup resource configuration	28
	DiskGroupSnap agent	29
	Platforms	29
	Dependencies	29
	Agent functions	30
	State definitions	30
	Attributes	30
	DiskGroupSnap agent notes	32
	Configuration considerations	32
	Agent limitations	32
	Resource type definition	33
	Sample configurations	33
	DiskReservation agent	36
	Agent functions	36
	State definitions	36
	Attributes	37
	Resource type definition	39
	DiskReservation agent notes	39

The DiskReservation agent does not reserve disks that have multiple paths	39
Configuring the MonitorTimeout attribute for more than three disks	39
Sample configurations	40
Configuration 1	40
Configuration 2	40
Volume agent	42
Dependencies	42
Agent functions	42
State definitions	43
Attributes	43
Resource type definition	43
Sample configurations	44
Configuration	44
Configuration	44
LVMLogicalVolume agent	46
Dependencies	46
Agent functions	46
State definitions	47
Attributes	47
Resource type definition	47
Sample configurations	48
Configuration 1	48
Configuration 2	48
LVMVolumeGroup agent	49
Dependencies	49
Agent functions	49
State definitions	50
Attributes	50
Resource type definition	50
Sample configurations	51
Configuration 1	51
Configuration 2	51
Mount agent	52
Dependencies	52
Agent functions	52
State definitions	54
Attributes	55
Resource type definition	57
Mount agent notes	58
High availability fire drill	58
VxFS file system lock	58

Sample configurations	59
SANVolume agent	60
Agent functions	60
State definitions	60
Attributes	61
Resource type definition	62
Sample configuration	63

Chapter 3 Network agents

About the network agents	65
Agent comparisons	66
IP and NIC agents	66
IPMultiNIC and MultiNICA agents	66
802.1Q trunking	66
IP agent	67
High availability fire drill	67
Dependencies	67
Agent functions	68
State definitions	68
Attributes	69
Resource type definition	70
Sample configurations	70
Configuration 1	70
Configuration using specified NetMask	70
NIC agent	71
High availability fire drill	71
Dependencies	71
Bonded network interfaces	72
Agent functions	72
State definitions	72
Attributes	73
Resource type definition	74
Monitoring bonded NICs	75
Setting Mii and miimon	75
Sample configurations	76
Configuration for using Mii	76
Configuration for using network hosts	76
IPMultiNIC agent	77
Dependencies	77
Agent functions	77
State definitions	78
Attributes	79
Resource type definition	79

Sample configuration: IPMultiNIC and MultiNICA	80
MultiNICA agent	81
Dependencies	81
IP Conservation Mode (ICM)	82
Configuration	82
Operation	82
Performance Mode (PM)	82
Configuration	82
Operation	82
Agent function	83
Attributes	84
Resource type definition	86
Sample configurations	86
MultiNICA and IPMultiNIC Performance Mode configuration	86
MultiNICA and IPMultiNIC IP Conservation Mode	
Configuration	87
DNS agent	89
Dependencies	89
Agent functions	90
State definitions	91
Attributes	92
Resource type definition	96
DNS agent notes	96
High availability fire drill	96
Monitor scenarios	97
Sample Web server configuration	97
Secure DNS update for BIND 9	97
Setting up secure updates using TSIG keys for BIND 9	97

Chapter 4 File share agents

About the file service agents	99
NFS agent	100
Dependencies	100
Prerequisites for NFS lock recovery	100
Agent functions	101
State definitions	101
Attributes	102
Resource type definition	103
Sample configurations	103
Setting Nproc configuration	103
Configuring NFS lock on shared storage	103
NFSv4Support attribute configuration	105

NFSRestart agent	110
Dependencies	110
Agent functions	110
State definitions	111
Attributes	112
Resource type definition	112
NFSRestart agent notes	113
High availability fire drill	113
Providing a fully qualified host name	113
Sample configurations	114
Share agent	115
Dependencies	115
Agent functions	115
State definitions	116
Attributes	116
Resource type definition	117
Share agent notes	117
High availability fire drill	117
Sample configurations	118
Configuration 1	118
Configuration 2	118
About the Samba agents	121
The Samba agents	121
Before using the Samba agents	121
Supported versions	122
Configuring the Samba agents	122
SambaServer agent	123
Dependencies	123
Agent functions	123
State definitions	124
Attributes	125
Resource type definitions	127
Sample configurations	127
SambaShare agent	128
Dependencies	128
Agent functions	128
State definitions	129
Attributes	129
Resource type definition	130
Sample configuration	130
NetBIOS agent	131
Dependencies	131
Agent functions	132

State definitions	132
Attributes	133
Resource type definition	134
Sample configuration	134

Chapter 5 Service and application agents

About the service and application agents	135
Apache Web server agent	136
Dependencies	136
Agent functions	136
State definitions	137
Attributes	138
Resource type definition	142
Apache Web server notes	142
Tasks to perform before you use the Apache Web server agent	142
Detecting application failure	143
About bringing an Apache Web server online outside of VCS control	144
About the ACC Library	144
High Availability fire drill	145
Sample configurations	145
Application agent	149
High availability fire drill	149
Dependencies	149
Agent functions	150
State definitions	151
Attributes	152
Resource type definition	154
Sample configurations	155
Configuration 1	155
Configuration 2	155
Process agent	156
High availability fire drill	156
Dependencies	156
Agent functions	157
State definitions	157
Attributes	158
Resource type definition	159
Sample configurations	159
Configuration	159
ProcessOnOnly agent	160
Dependencies	160
Agent functions	160

State definitions	160
Attributes	161
Resource type definition	162
Sample configurations	163
Configuration 1	163
Configuration 2	163
Chapter 6	Infrastructure and support agents
About the infrastructure and support agents	165
NotifierMngr agent	166
Dependency	166
Agent functions	166
State definitions	166
Attributes	167
Resource type definition	170
Sample configuration	171
Configuration	171
VRTSWebApp agent	173
Agent functions	173
State definitions	173
Attributes	173
Resource type definition	174
Sample configuration	174
Proxy agent	175
Dependencies	175
Agent functions	175
Attributes	176
Resource type definition	177
Sample configurations	177
Configuration 1	177
Configuration 2	177
Configuration 3	177
Phantom agent	179
Dependencies	179
Agent functions	179
Resource type definition	179
Sample configurations	179
Configuration 1	179
Configuration 2	180
RemoteGroup agent	181
Dependency	181
Agent functions	182
State definitions	182

Attributes	183
Resource type definition	188

Chapter 7

Testing agents

About the program support agents	189
ElifNone agent	190
Dependencies	190
Agent function	190
Attributes	191
Resource type definition	191
Sample configuration	191
FileNone agent	192
Dependencies	192
Agent functions	192
Attribute	193
Resource type definition	193
Sample configuration	193
FileOnOff agent	194
Dependencies	194
Agent functions	194
Attribute	195
Resource type definition	195
Sample configuration	195
FileOnOnly agent	196
Dependencies	196
Agent functions	196
Attribute	197
Resource type definition	197
Sample configuration	197

Glossary	199
----------------	-----

Index	201
-------------	-----

Introduction

Bundled agents are Veritas 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 agents, see the VCS User's Guide.

Resources and their attributes

Resources are parts of a system and are known by their type, such as: 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 Cluster Server Management Console, or the command line to dynamically modify the configuration of the resources managed by an agent.

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

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

Attributes

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

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

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

Table 1-1 Attribute data types

Data Type	Description
string	Enclose strings, which are a sequence of characters, in double quotes ("). Optionally enclose strings in quotes when they begin with a letter, and contains only letters, numbers, dashes (-), and underscores (_). A string can contain double quotes, but the quotes must be immediately preceded by a backslash. In a string, represent a backslash with two slashes (//).
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: 21471183247.

Table 1-1 Attribute data types

Data Type	Description
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 SntpConsoles{}.

Storage agents

This chapter contains:

- [“DiskGroup agent”](#) on page 22
- [“DiskGroupSnap agent”](#) on page 29
- [“DiskReservation agent”](#) on page 36
- [“Volume agent”](#) on page 42
- [“LVMLogicalVolume agent”](#) on page 46
- [“LVMVolumeGroup agent”](#) on page 49
- [“Mount agent”](#) on page 52
- [“SANVolume agent”](#) on page 60

About the storage agents

Use storage agents to Monitor shared storage.

DiskGroup agent

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

When the value of the StartVolumes and StopVolumes attribute is 1, the DiskGroup agent brings the volumes online and takes them offline during the import and deport operations of the disk group.

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

The agent protects data integrity by disabling failover when data is written to a volume in the disk group.

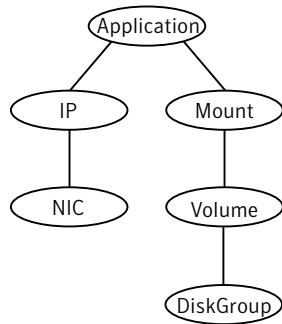
For important information on this agent, refer to:

[“DiskGroup agent notes”](#) on page 28

Dependencies

This type of resource can depend on DiskReservation resources, provided that Dynamic Multipathing is *not* configured in Veritas Volume Manager. The DiskGroup resource does not necessarily depend on any other resource.

Figure 2-1 Sample service group for a DiskGroup resource



Agent functions

Online	Imports the disk group using the <code>vxdg</code> command.
Offline	Deports the disk group using the <code>vxdg</code> command.
Monitor	<p>Determines if the disk group is online or offline using the <code>vxdg</code> command.</p> <p>The Monitor function changes the value of the VxVM <code>noautoimport</code> flag from off to on. This action allows VCS to maintain control of importing the disk group. The following command changes the <code>autoimport</code> flag back to off:</p> <pre># vxdg -g disk_group set autoimport=no</pre>
Clean	Terminates all ongoing resource actions and takes the resource offline—forcibly when necessary.
Info	<p>The DiskGroup info agent function gets information from the Volume Manager and displays the type and free size for the DiskGroup resource.</p> <p>Initiate the info agent function by setting the <code>InfoInterval</code> timing to a value greater than 0.</p> <p>In the following example, the info agent function executes every 60 seconds:</p> <pre># haconf -makerw # hatype -modify DiskGroup InfoInterval 60</pre> <p>The command to retrieve information about the <code>DiskType</code> and <code>FreeSize</code> of the DiskGroup resource is:</p> <pre># hares -value diskgroupres ResourceInfo</pre> <p>Output includes:</p> <pre>DiskType sliced FreeSize 35354136</pre>

Action	Different action agent functions follow: <ul style="list-style-type: none">■ <code>license.vfd</code> Checks for valid Veritas Volume manager license—if one is not found use the <code>vxlicinst</code> utility to install a valid license key.■ <code>disk.vfd</code> 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.■ <code>udid.vfd</code> Checks the UDIDs 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.■ <code>verifyplex.vfd</code> 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.■ <code>volinuse</code> Checks if open volumes are in use or file systems on volumes that are mounted outside of VCS configuration. See “High availability fire drill” on page 28.
--------	---

State definitions

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.

Attributes

Table 2-1 Required attributes

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

Table 2-2 Optional attributes

Optional attributes	Description
MonitorReservation	If the value is 1, and SCSI-3 fencing is used, the agent monitors the SCSI reservation on the disk group. If the reservation is missing, the monitor agent function takes the resource offline. Type and dimension: boolean-scalar Default: 0

Table 2-2 Optional attributes

Optional attributes	Description
PanicSystemOnDGLoss	<p>Determines whether to panic the node if the disk group becomes disabled. A loss of storage connectivity can cause the disk group to become disabled.</p> <p>If the value of this attribute is 1 and the disk group becomes disabled, the node panics.</p> <p>If the value of the attribute is 0 and the disk group becomes disabled, the following occurs:</p> <ul style="list-style-type: none"> ■ If the cluster has I/O fencing enabled, the DiskGroup resource is marked <code>FAULTED</code>. 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 a <code>FAULTED</code> 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 ■ If the cluster does not use I/O fencing, a message is logged and the resource is reported <code>ONLINE</code>. <p>Type and dimension: boolean-scalar Default: 1</p>
StartVolumes	<p>If value is 1, the DiskGroup online function starts all volumes belonging to that disk group after importing the group.</p> <p>Type and dimension: boolean-scalar Default: 1</p>
StopVolumes	<p>If value is 1, the DiskGroup offline function stops all volumes belonging to that disk group before it deports the group.</p> <p>Type and dimension: boolean-scalar Default: 1</p>

Table 2-2 Optional attributes

Optional attributes	Description
UmountVolumes	<p>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:</p> <ul style="list-style-type: none"> ■ The agent attempts to unmount the file systems on open volumes. If required, the agent attempts to kill all VCS managed and un-managed applications using the file systems on those open volumes. ■ The agent attempts to forcefully unmount the file systems to close the volumes. <p>Type and dimension: integer-scalar Default: 0</p>
TempUseFence	Do not use. For internal use only.
DiskGroupType	Do not use. For internal use only.

Resource type definition

```

type DiskGroup (
    static keylist SupportedActions = { "license.vfd", "disk.vfd",
    "udid.vfd", "verifyplex.vfd", checkudid, numdisks, campusplex,
    joindg, splitdg, getvxvminfo, volinuse }
    static int NumThreads = 1
    static int OnlineRetryLimit = 1
    static str ArgList[] = { DiskGroup, StartVolumes, StopVolumes,
    UmountVolumes, MonitorOnly, MonitorReservation, tempUseFence,
    PanicSystemOnDGLoss }
    str DiskGroup
    boolean StartVolumes = 1
    boolean StopVolumes = 1
    int UmountVolumes = 0
    boolean MonitorReservation = 0
    boolean PanicSystemOnDGLoss = 1
    temp str tempUseFence = INVALID
)

```

DiskGroup agent notes

The DiskGroup agent has the following notes:

- [“High availability fire drill”](#) on page 28
- [“Configuring the Fiber Channel adapter”](#) on page 28

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 *Veritas Cluster Server User's Guide*.

Configuring the Fiber Channel adapter

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

Refer to the Fiber Channel adapter's configuration guide for further information.

Sample configurations

DiskGroup resource configuration

Example of a disk group resource in the Share Out mode.

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

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 Veritas Volume Manager (VxVM) stretch mirroring.

For more information on fire drills, refer to the *Veritas Cluster Server User's Guide*.

For important information about this agent, refer to:

“[DiskGroupSnap agent notes](#)” on page 32

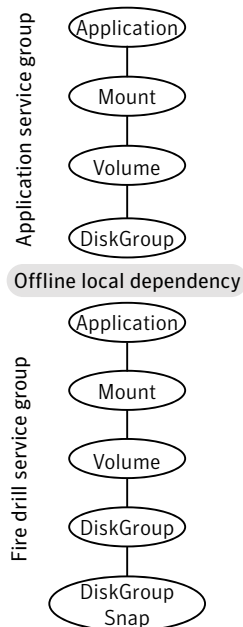
Platforms

AIX, HP-UX, Linux, and Solaris

Dependencies

The DiskGroupSnap resource does not depend on any other resources. The service group that contains the DiskGroupSnap agent has an offline local dependency on the application's service group.

Figure 2-2 Sample service group for a DiskGroupSnap resource



Agent functions

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.
Offline	This re-attaches the site that the value of the FDSiteName attribute specifies back to the application's disk group.
Monitor	Monitors the DiskGroupSnap resource.
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. In all other cases, the DiskGroupSnap resource performs no action.

State definitions

ONLINE	The DiskGroupSnap resource functions normally.
OFFLINE	The DiskGroupSnap resource is not running.
UNKNOWN	A configuration error exists.

Attributes

Table 2-3 Required attributes

Required attribute	Description
TargetResName	The name of the DiskGroup resource from the application's service group. Type-dimension: string-scalar Example: "dgres1"

Table 2-3 Required attributes

Required attribute	Description
FDSiteName	<p>This is the site name that fire drill disks use. This name must be distinct for each site. You need to assign this local (per system) values as it maps to the SystemZone of the application service group. For more information about the SystemZone attribute, refer to the <i>Veritas Cluster Server User's Guide</i>.</p> <p>You can run the fire drill in the following two configurations:</p> <ul style="list-style-type: none">■ Use a dedicated set of disks at the secondary that have been set aside for fire drill use. In this case, you must set the FDSiteName attribute to the VxVM site name given to this set of disks. This setup is commonly referred to as the Gold configuration.■ Use the same disks that make up the mirror at the secondary site. In this case, you must set the FDSiteName attribute to the VxVM site name of the secondary site. This setup is commonly referred to as the Bronze configuration. <p>Type and dimension: string-scalar</p> <p>Example:</p> <p>When the application service group has the following values for the SystemZones attribute:</p> <pre>SystemZones = { n1 = 0, n2 = 0, n3 = 1, n4 = 1 }</pre> <p>Where n1 (node 1) and n2 (node 2) comprise the first site and where the second site has n3 (node 3) and n4 (node 4). The FDSiteName definitions in the fire drill service group resemble the following:</p> <ul style="list-style-type: none">■ "FDSiteName@n1=fdpri"■ "FDSiteName@n2=fdpri"■ "FDSiteName@n3=fdsec"■ "FDSiteName@n4=fdsec" <p>The fdpri and fdsec values are the site names of dedicated fire drill site disks at the primary and secondary sites respectively.</p>

DiskGroupSnap agent notes

Configuration considerations

Keep the following recommendations in mind:

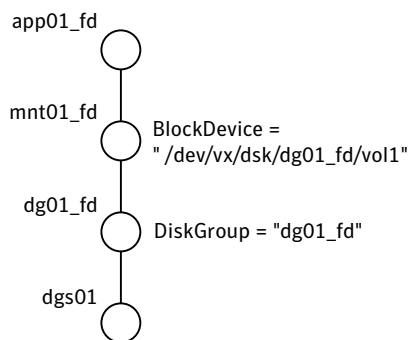
- Do not bring the DiskGroupSnap resource online in the SystemZone where the application service group is online.

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

The BlockDevice attribute of the Mount resource and the DiskGroup attribute of the DiskGroup resource change between the application's service group and the fire drill's service group. You must append an `_fd` to the original disk group name for the disk group name that the fire drill uses. For example, if `dg01` is the disk group's name in the application service group use `dg01_fd` in the fire drill service group. See [Figure 2-2](#).

[Figure 2-2](#) shows the changes to resource values for the fire drill service group, note that the Volume resource is not included.

Table 2-4 Sample resource values for a DiskGroupSnap resource



Agent limitations

The following limitations apply to the DiskGroupSnap agent:

- 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.

- If you attempt to shut down the node where you brought the fire drill service group online, the node goes into a `LEAVING` state and the VCS engine attempts to take all the service groups offline on that node. At this point, the VCS engine rejects all action entry point requests. Therefore, during offline the `DiskGroupSnap` agent cannot invoke the action to reattach the fire drill site to the target diskgroup. The agent logs a message that the node is in a leaving state and then removes the lock file. The agent's monitor function declares that the resource is offline. After the node restarts, the fire drill site still remains detached from the diskgroup and you must manually reattach it.
- If you halt the node while the `DiskGroupSnap` resource's service group is still online, the VxVM site used for the fire drill remains detached after the node is brought up. You must manually reattach the fire drill site to the original diskgroup at the primary site.
- Before you shut down or stop VCS locally on the node where the fire drill service group is online, take the fire drill service group offline. Otherwise, after the node restarts you must manually reattach the fire drill site to the disk group that is imported at the primary site.

Resource type definition

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

Sample configurations

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

```
group dgfdsg (  
    SystemList = { thoribm32 = 0, thoribm31 = 1 }  
    SystemZones = { thoribm32 = 1, thoribm31 = 0 }  
)  
  
DiskGroup dgfdres (  
    DiskGroup = newdg1_fd  
)  
  
DiskGroupSnap dgsres (  
    TargetResName = dgres
```

```
FDSiteName @thoribm32 = firedrill
FDSiteName @thoribm31 = firedrill_31
)

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

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

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
dgfdres requires dgsres
mntfdres1 requires dgfdres
mntfdres2 requires dgfdres
procfdres1 requires mntfdres1
procfdres2 requires mntfdres2
```

The application's service group follows:

```
group dgsg (
  SystemList = { thoribm32 = 0, thoribm31 = 1 }
  SystemZones = { thoribm31 = 0, thoribm32 = 1 }
)

DiskGroup dgres (
  DiskGroup = newdg1
)

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

```
Mount mntres2 (  
  MountPoint = "/dgsfs2"  
  BlockDevice = "/dev/vx/dsk/newdg1/newvol2"  
  FSType = vxfs  
  FsckOpt = "-y"  
)  
  
Process proces1 (  
  PathName = "/usr/bin/ksh"  
  Arguments = "/scrib.sh /dgsfs1"  
)  
  
Process proces2 (  
  PathName = "/usr/bin/ksh"  
  Arguments = "/scrib.sh /dgsfs2"  
)  
  
mntres1 requires dgres  
mntres2 requires dgres  
proces1 requires mntres1  
proces2 requires mntres2
```

DiskReservation agent

Reserves and monitors SCSI disks for a system, enabling a resource to go online on that system. This agent enables you to specify a list of raw disk devices, and reserve all or a percentage of accessible disks. The reservations prevent disk data corruption by restricting other nodes from accessing and writing to the disks. The DiskReservation agent supports all SCSI-II compliant disks.

An automatic probing feature allows systems to maintain reservations even when the disks or bus are reset. The optional FailFast feature minimizes data corruption in the event of a reservation conflict by causing the system to panic.

Note: The DiskReservation agent cannot be used to reserve disks that have multiple paths.

For important information on this agent, refer to:

[“DiskReservation agent notes”](#) on page 39

Agent functions

Online	Brings the resource online after reserving all or a specified percentage of accessible disks.
Offline	Releases reservations on reserved disks.
Monitor	Monitors the accessibility and reservation status of the reserved disks.
Clean	Terminates all ongoing resource actions and takes the resource offline—forcibly when necessary.

State definitions

ONLINE	Indicates that the number of reserved disks is greater than or equal to the percentage specified in the resource definition.
OFFLINE	Disks are not reserved.
UNKNOWN	Indicates that a problem exists with the configuration.

Attributes

Table 2-5 Required attributes

Required attribute	Description
Disks	<p>A list of raw disk devices. Use the absolute or relative device path. The absolute or relative device path allows a maximum of 64 characters. The relative path is assumed to start from the <code>/dev</code> directory.</p> <p>The order of the disks in the list must be the same across all systems in the cluster, even if the same device has a different name on different systems.</p> <p>Note: You must change this attribute before bringing a resource online. An online device must be taken offline before altering this attribute because disk reservation occurs during the process of bringing a resource online.</p> <p>Type and dimension: string-vector</p> <p>Example: "c1t2d0s4"</p>

Table 2-6 Optional attributes

Optional attribute	Description
FailFast	<p>If enabled, FailFast causes the system to panic when a reservation conflict is detected, reducing the chance of further data corruption.</p> <p>Type and dimension: boolean-scalar</p> <p>Default: 0</p>

Table 2-6 Optional attributes

Optional attribute	Description
Percentage	<p>Minimum percentage of configured disks that can be reserved before a resource can go online. The percentage must be greater than or equal to 51, and less than or equal to 100.</p> <p>If the value specified is less than 51, the percentage is set to 51.</p> <p>If the value specified is greater than 100, the percentage is set to 100.</p> <p>Type and dimension: integer-scalar</p> <p>Default: 100</p>
ProbeInterval	<p>Alters the periodicity (in seconds) of the automatic probe function that checks the reservation status of the disks. The value must be greater than or equal to three, and less than or equal to 15.</p> <p>If the value specified is less than 3, the interval is set to 3.</p> <p>If the value specified is greater than 15, the interval is set to 15.</p> <p>A lower value for ProbeInterval specifies more frequent probes and provides for quicker discovery of reservation conflicts. Symantec recommends a value is between 3 and 8.</p> <p>Type and dimension: integer-scalar</p> <p>Default: 3</p>

Resource type definition

```
type DiskReservation (
  static str ArgList[] = { Disks, FailFast, Percentage,
    ProbeInterval }
  str Disks[]
  boolean FailFast = 0
  int Percentage = 100
  int ProbeInterval = 3
)
```

DiskReservation agent notes

The DiskReservation agent has the following notes:

- [“The DiskReservation agent does not reserve disks that have multiple paths”](#) on page 39
- [“Configuring the MonitorTimeout attribute for more than three disks”](#) on page 39

The DiskReservation agent does not reserve disks that have multiple paths

You cannot use the DiskReservation agent to reserve disks that have multiple paths. The LVMVolumeGroup and the LVMLogicalVolume agents can only be used with the DiskReservation agent, Symantec does not support the configuration of logical volumes on disks that have multiple paths. To ensure data protection on such a configuration, Symantec recommends the use of Veritas Volume Manager (VxVM) disk groups. Note that VxVM requires the use of SCSI-3 compliant disks.

Configuring the MonitorTimeout attribute for more than three disks

The MonitorTimeout attribute’s setting of 60 is adequate for up to three disks. When configuring the MonitorTimeout attribute for more than three disks, use the following formula:

Set MonitorTimeout to be equal or greater than 15 times the total number of disks. (MonitorTimeout \geq 15 * Number of disks).

For example, if you have eight disks, MonitorTimeout is 120 or greater.

Sample configurations

Configuration 1

In this example, the DiskReservation agent reserves a disk. The disk is mounted with the Veritas File System.

```
system sysA

system sysB

group groupx (
    SystemList = { sysA, sysB }
    AutoStartList = { sysA }
)

DiskReservation diskres1 (
    Disks = { "/dev/sdc" }
    FailFast = 1
)

Mount mount (
    MountPoint = "/mnt/tmp"
    BlockDevice = "/dev/sdc1"
    FSType = vxfs
    MountOpt = rw
)

mount requires diskres1

// resource dependency tree
//
// group groupx
// {
//   Mount mount
//   {
//     DiskReservation diskres1
//   }
// }
```

Configuration 2

In this example, the DiskReservation agent reserves several disks. The disk group defined on these disks is imported only if the system can reserve the disks.

Volumes can be enabled and mounted on the Disk Group. Refer to the Volume agent for a sample configuration.

```
group groupy (
    SystemList = { sysA, sysB }
    AutoStartList = { sysA }
```

```
)  
  
DiskGroup resdg (  
  DiskGroup = resdg  
)  
  
DiskReservation diskres2 (  
  Disks = { "/dev/sdc", "/dev/sdd", "/dev/sde", "/dev/  
  sdf", "/dev/sdg" }  
  ProbeInterval = 5  
  Percentage = 60  
)  
  
resdg requires diskres2  
  
// resource dependency tree  
//  
// group groupy  
// {  
//   DiskGroup resdg  
//   {  
//     DiskReservation diskres2  
//   }  
// }
```

Volume agent

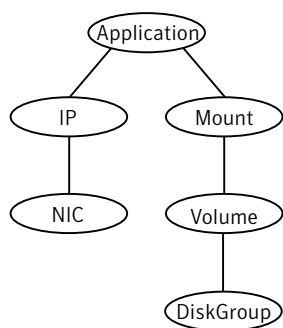
The Volume agent brings online, takes offline, and monitors a Veritas Volume Manager (VxVM) volume. You can use the agent to make a volume highly available or to monitor it.

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

Dependencies

Volume resources depend on DiskGroup resources.

Figure 2-3 Sample service group for a Volume resource



Agent functions

Online	Starts the volume using the <code>vxrecover</code> command.
Offline	Stops the volume using the <code>vxvol</code> command.
Monitor	Determines if the volume is online or offline by reading a block from the raw device interface to the volume.
Clean	Terminates all ongoing resource actions and takes the resource offline—forcibly when necessary.

State definitions

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

Attributes

Table 2-7 Required attributes

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

Resource type definition

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

Sample configurations

Configuration

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

Configuration

In this example, the DiskReservation resource is used to verify that disks are available only to one system. The volumes on the disk groups that are imported are started if the reservation is confirmed. The volumes can then be mounted at a mount point.

```
group groupy (
  SystemList = { sysA, sysB }
  AutoStartList = { sysA }
)

DiskGroup resdg (
  DiskGroup = resdg
)

DiskReservation diskres2 (
  Disks = { "/dev/sdc", "/dev/sdd", "/dev/sde", "/dev/sdf",
  "/dev/sdg" }
  ProbeInterval = 5
  Percentage = 60
)

Mount mountvol (
  BlockDevice = "/dev/vx/dsk/resdg/resvol"
  MountPoint = "/share"
  FSType = vxfs
  MountOpt = rw
)

Volume resdg_resvol (
  DiskGroup = resdg
  Volume = resvol
)

mountvol requires resdg_resvol
resdg requires diskres2
resdg_resvol requires resdg

// resource dependency tree
//
// group groupy
```

```
// {  
// Mount mountvol  
// {  
//   Volume resdg_resvol  
//     {  
//       DiskGroup resdg  
//         {  
//           DiskReservation diskres2  
//         }  
//       }  
//     }  
//   }  
// }
```

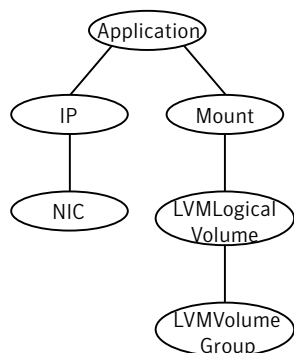
LVMLogicalVolume agent

The LVMLogicalVolume agent brings online, takes offline, and monitors a Logical Volume Manager (LVM2) volume. This agent uses LVM2 commands. You can use this agent to make volume groups and logical volumes highly available and to monitor them.

Dependencies

LVMLogicalVolume resources depend on LVMVolumeGroup resources.

Figure 2-4 Sample service group for a LVMLogicalVolume resource



Agent functions

Online	Starts the volume using the <code>lvchange</code> command.
Offline	Stops the volume using the <code>lvchange</code> command.
Monitor	Determines if the volume is online or offline by reading a block from the raw device interface to the volume.
Clean	Terminates all ongoing resource actions and takes the resource offline, forcibly when necessary.

State definitions

ONLINE	Indicates that the specified volume is started and that I/O is permitted.
OFFLINE	Indicates that the specified volume is not started—and I/O is not permitted.
UNKNOWN	Indicates that the agent could not determine the state of the resource or that the resource attributes are invalid.

Attributes

Table 2-8 Required attributes

Required attribute	Description
LogicalVolume	Name of the volume that is configured with Logical Volume Manager (LVM2). Type and dimension: string-scalar Example: "volume1"
VolumeGroup	Name of the volume group that is configured with Logical Volume Manager (LVM2), which contains the volume. Type and dimension: string-scalar Example: "volumegroup1"

Resource type definition

```
type LVMLogicalVolume (  
  static str ArgList[] = { LogicalVolume, VolumeGroup }  
  str LogicalVolume  
  str VolumeGroup  
)
```

Sample configurations

Configuration 1

In this example, /dev/sdc and /dev/sdd are the disks where the volume group testvg_1 is created.

```
LVMLogicalVolume lvoll (
    LogicalVolume = testvol_1
    VolumeGroup = testvg_1
)

LVMVolumeGroup lvg1 (
    VolumeGroup = testvg_1
)

DiskReservation dr1 (
    Disks = { "/dev/sdc", "/dev/sdd" }
)

lvoll requires lvg1
lvg1 requires dr1
```

Configuration 2

In this example, you use the DiskReservation resource to verify that disks are available only to one system. The LVM2 logical volumes on the LVM2 volume groups that are imported are started if the reservation is confirmed. The logical volumes can then be mounted at a mount point.

```
DiskReservation dr_cde (
    Disks = { "/dev/sdc", "/dev/sdd", "/dev/sde" }
)

Mount mnt_lvmlvol01 (
    MountPoint = "/mnt/lvmlvol01"
    BlockDevice = "/dev/mapper/lvmlvg01-lvmlvol01"
    FSType = "reiserfs"
    FsckOpt = "-y"
)

LVMLogicalVolume lvmlvol01 (
    LogicalVolume = lvmlvol01
    VolumeGroup = lvmlvg01
)

LVMVolumeGroup lvmlvg01 (
    VolumeGroup = lvmlvg01
)

mnt_lvmlvol01 requires lvmlvol01
lvmlvol01 requires lvmlvg01
lvmlvg01 requires dr_cde
```

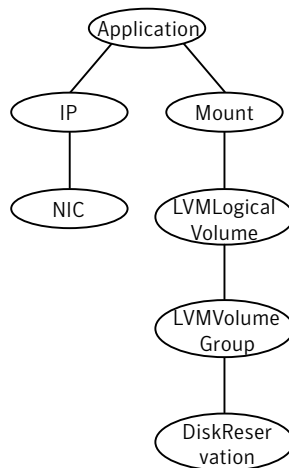
LVMVolumeGroup agent

The LVMVolumeGroup agent brings online, takes offline, and monitors a Logical Volume Manager (LVM2) volume group. This agent uses LVM2 commands. You can use this agent to make volume groups and logical volumes highly available and to monitor them.

Dependencies

LVMVolumeGroup resources depend on DiskReservation resources. If an LVMVolumeGroup does not have a corresponding DiskReservation resource on which it depends, the LVMVolumeGroup does not function.

Figure 2-5 Sample service group for a LVMVolumeGroup resource



Agent functions

Online	Imports the volume group using the <code>vgimport</code> command.
Offline	Exports the volume group using the <code>vgexport</code> command.
Monitor	Determines if the volume group is online or offline using the <code>vgdisplay</code> command.
Clean	Terminates all ongoing resource actions and takes the resource offline, forcibly when necessary.

State definitions

ONLINE	Indicates that the volume group is imported.
OFFLINE	Indicates that the volume group is not imported.
UNKNOWN	Indicates that a problem exists either with the configuration or the ability to determine the status of the resource.

Attributes

Table 2-9 Required attributes

Required attribute	Description
VolumeGroup	The name of the volume group that is configured with Logical Volume Manager (LVM2) that contains the volume. Type and dimension: string-scalar Example: "volumegroup1"

Table 2-10 Optional attributes

Optional attribute	Description
StartVolumes	If the value of this attribute is 1, the LVMVolumeGroup online function imports the group. It then starts all the volumes that belong to that volume group. Type and dimension: boolean-scalar Default: 0

Resource type definition

```

type LVMVolumeGroup (
    static str ArgList[] = { VolumeGroup, StartVolumes }
    str VolumeGroup
    boolean StartVolumes = 0
)

```

Sample configurations

Configuration 1

In this example, /dev/sdc and /dev/sdd are the disks where the volume group testvg_1 is created.

```
LVMVolumeGroup lvg1 (  
    VolumeGroup = testvg_1  
)  
  
DiskReservation dr1 (  
    Disks = { "/dev/sdc", "/dev/sdd" }  
)  
  
lvg1 requires dr1
```

Configuration 2

In this example, the DiskReservation resource is used to verify that disks are available only to one system. All LVM2 logical volumes on the LVM2 volume groups that are imported are started if the reservation is confirmed. You can then mount the logical volumes at a mount point.

```
DiskReservation dr_cde (  
    Disks = { "/dev/sdc", "/dev/sdd", "/dev/sde" }  
)  
  
Mount mnt_lvmvol01 (  
    MountPoint = "/mnt/lvmvol01"  
    BlockDevice = "/dev/mapper/lvmvg01-lvmvol01"  
    FSType = "reiserfs"  
    FsckOpt = "-y"  
)  
  
LVMVolumeGroup lvmvg01 (  
    VolumeGroup = lvmvg01  
    StartVolumes = 1  
)  
  
mnt_lvmvol01 requires lvmvg01  
lvmvg01 requires dr_cde
```

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 or to monitor them. This agent also supports high availability fire drills.

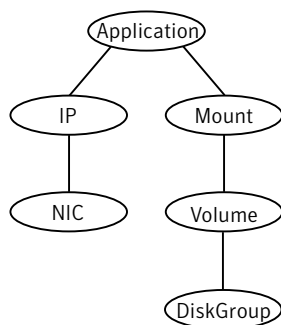
For important information about this agent, refer to:

“[Mount agent notes](#)” on page 58

Dependencies

No dependencies exist for the Mount resource.

Figure 2-6 Sample service group for a Mount resource



Agent functions

Online	Mounts a block device on the directory. If the mount process fails for non-NFS mounts, the agent attempts to run the <code>fsck</code> command on the device to remount the block device. If file system type is NFS, agent mounts the remote NFS file system to a specified directory. The remote NFS file system is specified in the <code>BlockDevice</code> attribute.
Offline	Unmounts the mounted file system gracefully.
Monitor	Determines if the file system is mounted.
Clean	Unmounts the mounted file system forcefully.

Info	<p>The Mount info agent function executes the command:</p> <pre>df -h <i>mount_point</i></pre> <p>The output displays Mount resource information:</p> <pre>Size Used Avail Use%</pre> <p>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:</p> <pre>haconf -makerw hatype -modify Mount InfoInterval 60</pre> <p>The command to retrieve information about the Mount resource is:</p> <pre>hares -value mountres ResourceInfo</pre> <p>Output includes:</p> <pre>Size 2097152 Used 139484 Available 1835332 Used% 8%</pre>
Action	<ul style="list-style-type: none">■ chgmtlock Invoke this action to reset the VxFS file system lock to a VCS-defined lock.■ mountpoint.vfd Checks if the specified mount point exists on the offline node. If it fails, it creates the mount point directory using <code>mkdir</code> command.■ mounted.vfd Checks if the mount point is already mounted on the offline node. If it fails, you need to unmount all the file systems from the specified mount point directory.■ vxfslic.vfd Checks for valid Veritas File System (VxFS) licenses. If it fails, you need to update the license for VxFS.■ mountentry.vfd Checks if the mount point is not listed in file system tables (e.g. <code>/etc/fstab</code>). This is to prevent the auto mount of mount points on system reboot. If it fails, you need to remove mount point from file system tables.
attr_changed	Unlocks the mounts when you change the value of the VxFSMountLock attribute from 1 to 0.

State definitions

ONLINE	Indicates that the file system is properly mounted on the given mount point.
OFFLINE	Indicates that the file system is not mounted properly on the mount point.
FAULTED	Indicates that the file system unexpectedly unmounted.
UNKNOWN	Indicates that a problem exists either with the configuration or the ability to determine the status of the resource.

Attributes

Table 2-11 Required attributes

Required attribute	Description
BlockDevice	<p>Block device for mount point.</p> <p>Type and dimension: string-scalar</p> <p>Examples:</p> <p><code>/dev/sdc1</code>, <code>/dev/vx/dsk/dg/volume</code></p> <p>If the device is LVM2 volume, then you must specify the BlockDevice as:</p> <p><code>/dev/mapper/volume-group-logical-volume</code></p>
FsckOpt	<p>Use this attribute to specify options for the <code>fsck</code> command. You must correctly set this attribute for local mounts. If the mount process fails, the <code>fsck</code> command is executed with the specified options before it attempts to remount the block device. Its value must include either <code>-y</code> or <code>-n</code>. Refer to the <code>fsck</code> manual page for more information.</p> <p>The <code>-y</code> argument enables the VxFS file systems to perform a log replay before a full <code>fsck</code> operation.</p> <p>For NFS mounts, the value of this attribute is not applicable and is ignored.</p> <p>Type and dimension: string-scalar</p> <p>VxFS example: <code>-y -o full</code></p>
FSType	<p>Type of file system.</p> <p>Supports <code>vxfs</code>, <code>bind</code>, <code>ext2</code>, <code>ext3</code>, <code>nfs</code>, or <code>reiserfs</code>.</p> <p>Type and dimension: string-scalar</p>
MountPoint	<p>Directory for mount point.</p> <p>Type and dimension: string-scalar</p> <p>Example: <code>"/mnt/apache1"</code></p>

Table 2-12 Optional attributes

Optional attribute	Description
CkptUmount	<p>If the value of this attribute is 1, this attribute automatically unmounts VxFS checkpoints when the file system is unmounted.</p> <p>If the value of this attribute is 0, and checkpoints are mounted, then failover does not occur.</p> <p>Type and dimension: boolean-scalar</p> <p>Default: 1</p>
MountOpt	<p>Options for the <code>mount</code> command. Refer to the <code>mount</code> manual page for more information.</p> <p>Do not specify <code>-o</code> in the <code>MountOpt</code> field.</p> <p>Type and dimension: string-scalar</p> <p>Example: "rw"</p>
VxFSMountLock	<p>This attribute is applicable to Veritas (VxFS) file systems. It controls the agent's use of the locking feature provided by <code>vxfs</code> to prevent accident unmounts.</p> <p>If the value of this attribute is 0, the agent does not lock the mount point when the resource is brought online. It does not monitor the status of the lock when the resource is online. No warnings appear if the mount has been locked with a key different than "VCS".</p> <p>If the value of this attribute is 1, during online, the agent uses the key "VCS" to lock the mount point. The monitor agent function monitors the locks during every cycle.</p> <ul style="list-style-type: none"> ■ If the mountpoint is not locked, the agent locks it. ■ If the mountpoint is already locked with a key other than "VCS", the agent logs a warning. It then requests that you run the <code>Chgmtlock</code> action agent function. <p>During offline, the agent, as required, unlocks using whatever key needed.</p> <p>Type and dimension: boolean-scalar</p> <p>Default: 0</p>

Table 2-12 Optional attributes

Optional attribute	Description
SecondLevelMonitor	<p>This attribute is only applicable to NFS client mounts.</p> <p>If the value of this attribute is 1, this attribute enables detailed monitoring of an NFS mounted file system.</p> <p>Type and dimension: boolean-scalar</p> <p>Default: 0</p>
SecondLevelTimeout	<p>This attribute is only applicable for an NFS client mount.</p> <p>This attribute is the timeout (in seconds) for the SecondLevelMonitor/Detail monitoring of NFS Mounts.</p> <p>Type and dimension: integer-scalar</p> <p>Default: 30</p>
SnapUmount	<p>If the value of this attribute is 1, this attribute automatically unmounts VxFS snapshots when the file system is unmounted.</p> <p>If the value of this attribute is 0, and snapshots are mounted, then failover does not occur.</p> <p>Type and dimension: boolean-scalar</p> <p>Default: 0</p>

Resource type definition

```

type Mount (
    static keylist RegList = { VxFSMountLock }
    static keylist SupportedActions = { "mountpoint.vfd",
    "mounted.vfd", "vxfslic.vfd", "chgmtlock", "mountentry.vfd" }
    static str ArgList[] = { MountPoint, BlockDevice, FSType,
    MountOpt, FsckOpt, SnapUmount, CkptUmount, SecondLevelMonitor,
    SecondLevelTimeout, VxFSMountLock }
    str MountPoint
    str BlockDevice
    str FSType
    str MountOpt
    str FsckOpt
    boolean SnapUmount = 0
    boolean CkptUmount = 1
    boolean SecondLevelMonitor = 0

```

```
    int SecondLevelTimeout = 30
    boolean VxFSMountLock = 0
)
```

Mount agent notes

The Mount agent has the following notes:

- [“High availability fire drill”](#) on page 58
- [“VxFS file system lock”](#) on page 58

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 (VxFS) file system licenses
- Checks if the mount point exists in the `/etc/fstab` file

For more information about using the high availability fire drill see the *Veritas Cluster Server User'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 this option has a 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 lock, you can run the following command where `key` is the key's name and `mount_point_name` is the name of the mount point.

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

You can run the `vxumount` command with the option "mntunlock=key", for example:

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

Sample configurations

```
Mount MountSCSI1 (  
  MountPoint= "/scsi1"  
  BlockDevice = "/dev/sda1"  
  FSType = ext2  
  MountOpt = rw  
  FsckOpt = "-y"  
)
```

SANVolume agent

Use this agent as a resource to control access to a SAN volume, and to monitor the health of a SAN volume. You can configure the agent as part of a VCS service group.

The SAN volumes must reside on storage arrays that support SCSI-3 persistent reservations.

Note: Storage Foundation Volume Server (SF Volume Server) is a separately licensed feature of Veritas Storage Foundation™ by Symantec. An SF Volume Server license is currently available only through the Symantec customer access program. For information about participating in the access program and obtaining an SF Volume Server license, visit the following Symantec website: <http://cap.symantec.com>

Agent functions

Online	Attaches the SAN volume to the volume client host, and creates a device node for the SAN volume on the volume client host.
Offline	Unattaches a SAN volume. It deletes the device node for the already attached SAN volume to the volume client host.
Clean	Forcibly detaches the SAN volume from a volume client.
Monitor	Checks the state of the SAN volume on a volume client. It checks the health of the SAN volume and determines whether it is online or offline.

State definitions

ONLINE	Indicates that state of the SAN volume is attached.
OFFLINE	Indicates that the SAN volume is unattached.
UNKNOWN	Indicates that a problem exists either with the configuration or the ability to determine the status of the resource.

Attributes

Table 2-13 Required attributes

Required attribute	Description
SANDiskGroup	The name of the SAN disk group that contains the volume. Type and dimension: string-scalar Example: "dg1"
Domain	The name of the storage domain that the SAN volume belongs to. Type and dimension: string-scalar Example: "domain1"
SANVolume	The name of the SAN volume Type and dimension: string-scalar Example: "sanvol_1"
VolumeServer	The name of the SAN volume server. <ul style="list-style-type: none">■ If the volume server is not centrally managed, then this is required. If the volume server is made highly available using VCS, then the name of the volume server should be a virtual IP address or the host name associated with the virtual IP address.■ For a centrally managed volume server, then this attribute is not required. Type and dimension: string-scalar Example: "myserver.symantec.com"

Table 2-14 Optional attributes

Optional attribute	Description
ExclusiveUse	ExclusiveUse enforces volume to be opened by only one node in the cluster at a time. Type and dimension: boolean-scalar Default: 1
Preempt	Preempt enforces an exclusive attach of the volume by a node in the cluster. Type and dimension: integer-scalar Default: 1
AccessPolicy	The access policy for the volume: {RDONLY RDWR} Type and dimension: string-scalar Default: RDWR

Resource type definition

```
type SANVolume (  
    static int OnlineRetryLimit = 4  
    static str ArgList[] = { SANVolume, SANDiskGroup, VolumeServer,  
        Domain, ExclusiveUse, Preempt, AccessPolicy }  
    str Domain  
    str SANDiskGroup  
    str SANVolume  
    str VolumeServer  
    boolean ExclusiveUse = 1  
    boolean Preempt = 1  
    str AccessPolicy = RDWR  
)
```

Sample configuration

This example shows all the required attributes.

```
SANVolume svol (  
  SANDiskGroup = vsdg  
  SANVolume = vsvol  
  VolumeServer = "sysA.symantec.com"  
  Domain = "domain1"  
)
```


Network agents

This chapter contains the following:

- [“About the network agents”](#) on page 65
- [“IP agent”](#) on page 67
- [“NIC agent”](#) on page 71
- [“IPMultiNIC agent”](#) on page 77
- [“MultiNICA agent”](#) on page 81
- [“DNS agent”](#) on page 89

About the network agents

Use network agents to provide high availability for networking resources.

Agent comparisons

IP and NIC agents

The IP and NIC agents:

- Monitor a single NIC

IPMultiNIC and MultiNICA agents

The IPMultiNIC and MultiNICA agents:

- Operate in two modes:
 - IP Conservation (IPC) Mode, which uses fewer IP addresses
 - Performance Mode (PM), which provides faster failover, but uses more IP addresses
- Monitor single or multiple NICs
- Check the backup NICs at fail over
- Use the original base IP address when failing over
- Have only one active NIC at a time

802.1Q trunking

The IP/NIC and IPMultiNIC/MultiNICA agents support 802.1Q trunking.

The underlying utility to manage 802.1Q trunk interfaces is `vconfig`. For example, you can create a trunk interface on the physical interface:

```
# vconfig add eth2 10
```

This creates a trunk interface called `eth2.10` in the default configuration. In this case, the physical NIC `eth2` must be connected to a trunk port on the switch. You can now use `eth2.10` like a regular physical NIC in a NIC, IP, and MultiNICA resource configuration. You can remove it with the following command.

```
# vconfig rem eth2.10
```

VCS does not create nor remove trunk interfaces. The administrator should set up the trunking as per the operating system vendor's documentation rather than using `vconfig` directly.

IP agent

The IP agent manages the process of configuring a virtual IP address and its subnet mask on an interface. The interface must be enabled with a physical (or administrative) base IP address before you can assign it a virtual IP address. 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.

High availability fire drill

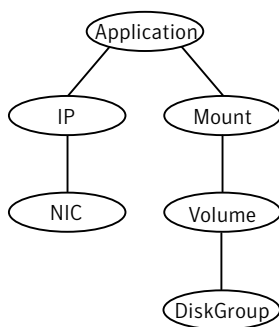
The high availability fire drill detects discrepancies between the VCS configuration and the underlying infrastructure on a node. These discrepancies might prevent a service group from going online on a specific node. For IP resources, the high availability fire drill checks for the existence of a route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

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

Dependencies

IP resources depend on NIC resources.

Figure 3-1 Sample service group for an IP resource



Agent functions

Online	Configures the IP address to the NIC. Checks if another system is using the IP address. Uses the <code>ifconfig</code> command to set the IP address on a unique alias on the interface. Sends out a gratuitous ARP.
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 associated with the specified interface.

State definitions

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.

Attributes

Table 3-1 Required attributes

Required attribute	Description
Address	<p>A virtual IP address, different from the base IP address, which is associated with the interface. Note that the address you specify must not be the same as the configured physical IP address.</p> <p>Type and dimension: string-scalar</p> <p>Example: "192.203.47.61"</p>
Device	<p>The name of the NIC device that is associated with the IP address. Requires the device name without an alias.</p> <p>Type and dimension: string-scalar</p> <p>Example: Specify eth0 to assign the IP address to the next available alias. Use the <code>ifconfig -a</code> command to display a list of NICs that are up and the IP addresses assigned to each NIC.</p>

Table 3-2 Optional attributes

Optional attribute	Description
NetMask	<p>The subnet mask that is associated with the IP address. Specify the value of NetMask in decimal (base 10).</p> <p>If Netmask is not specified, the agent uses the operating system's default netmask.</p> <p>Type and dimension: string-scalar</p> <p>Example: "255.255.255.0"</p>
Options	<p>Options for the <code>ifconfig</code> command.</p> <p>Type and dimension: string-scalar</p> <p>Example: "broadcast 172.20.9.255"</p>

Resource type definition

```
type IP (  
    static keylist SupportedActions = { "device.vfd", "route.vfd" }  
    static str ArgList[] = { Device, Address, NetMask, Options }  
    str Device  
    str Address  
    str NetMask  
    str Options  
)
```

Sample configurations

Configuration 1

```
IP          IP_192_203_47_61 (  
    Device = eth0  
    Address = "192.203.47.61"  
)
```

Configuration using specified NetMask

```
IP          IP_192_203_47_61 (  
    Device = eth0  
    Address = "192.203.47.61"  
    NetMask = "255.255.248.0"  
)
```

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 or to monitor it.

Some NICs maintain their connection status in a hardware register. For NICs that maintain their connection status, the agent uses MII to determine the status of the NIC resource. For NICs that do not maintain their connection status, the agent uses a ping or a broadcast to determine the status of the resource.

High availability fire drill

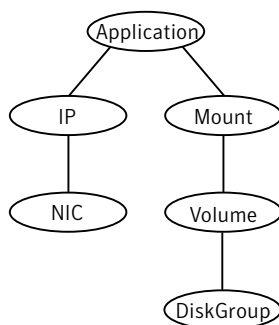
The high availability fire drill detects discrepancies between the VCS configuration and the underlying infrastructure on a node. These discrepancies might prevent a service group from going online on a specific node. For NIC resources, the high availability fire drill checks for the existence of the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

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

Dependencies

No dependencies exist for the NIC resource.

Figure 3-2 Sample service group for a NIC resource



Bonded network interfaces

The NIC agent now supports using bonded network interfaces.

See “[Monitoring bonded NICs](#)” on page 75.

Agent functions

Monitor	<p>If the NIC maintains its connection status, the agent uses MII to determine the status of the resource.</p> <p>If the NIC does not maintain its connection status, the agent verifies that the NIC is configured. The agent then sends a ping to all the hosts that are listed in the NetworkHosts attribute. If the ping test is successful, it marks the NIC resource ONLINE.</p> <p>If the NetworkHosts attribute list is empty, or the ping test fails, the agent counts the number of packets that the NIC received. The agent compares the count with a previously stored value. If the packet count increases, the resource is marked ONLINE. If the count remains unchanged, the agent sends a ping to the broadcast address of the device to generate traffic on the network.</p> <p>The agent counts the number of packets that the NIC receives before and after the broadcast. If the count increases, the resource is marked ONLINE. If the count remains the same or decreases over a period of five broadcast cycles, the resource is marked OFFLINE.</p>
---------	---

State definitions

ONLINE	Indicates that the NIC resource is working.
OFFLINE	The NIC resource can go OFFLINE if the NIC it represents has failed or is unavailable.
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

Table 3-3 Required attributes

Required attribute	Description
Device	<p>Name of the NIC that you want to monitor.</p> <p>Use the <code>ifconfig -a</code> command to list all network adapters and the IP addresses assigned to each NIC.</p> <p>Type and dimension: string-scalar</p> <p>Example: "eth0" or "eth1"</p>

Table 3-4 Optional attributes

Optional attribute	Description
Mii	<p>Flag that defines whether the NIC maintains its connection status.</p> <p>If this flag is set to 1, the agent uses MII hardware registers, instead of the ping and packet count method. The agent uses this method to determine the health of the network card.</p> <p>If the flag is set to 0, the agent does not use Mii to monitor the status of the NIC.</p> <p>Type and dimension: integer-scalar</p> <p>Default: 1</p>

Table 3-4 Optional attributes

Optional attribute	Description
NetworkHosts	<p>List of hosts on the network that receive pings to determine the state of the NIC. Specify the IP address of the host—not the host name.</p> <p>The specified hosts must be pingable:</p> <ul style="list-style-type: none"> ■ from all the AppNodes that are specified in the SystemList attribute for the service group to which the resource belongs ■ through all the devices that are specified in the Device attribute <p>The command to ping the host (hostip) via a NIC device (nicdev) is:</p> <pre># ping -I nicdev hostip</pre> <p>If more than one network host is listed, the monitor returns ONLINE if the ping test is successful with at least one of the hosts.</p> <p>Type and dimension: string-vector</p>
PingOptimize	<p>Attribute that defines whether the agent sends a broadcast ping before it retrieves the received packet statistics. This attribute is used when Mii is not set and no network hosts are specified.</p> <p>If the value of this attribute is 1, the agent retrieves received packet statistics from the netstat command and compare them with previously stored values. The agent sends a broadcast ping to the network only if the packet count remains unchanged.</p> <p>If the value of this attribute is 0, the agent sends a broadcast ping before it checks the network statistics.</p> <p>Type and dimension: integer-scalar</p> <p>Default: 1</p>

Resource type definition

```

type NIC (
    static keylist SupportedActions = { "device.vfd" }
    static int OfflineMonitorInterval = 60
    static str ArgList[] = { Device, PingOptimize, Mii,
        NetworkHosts }
    static str Operations = None
    str Device
    int PingOptimize = 1
    int Mii = 1
    str NetworkHosts[]
)

```

Monitoring bonded NICs

The NIC agent can monitor the network interfaces (bond0, bond1, etc.) that the bonding driver exports. Refer to operating system vendor documentation to set up the bonds and to configure your system to load the bonding driver correctly.

For monitoring a bond interface, the two important settings are:

- The value of the `miimon` parameter, which you set while loading the bonding driver. `miimon` is a parameter to the bonding module and has a default setting of 0.
- The value of the `Mii` attribute (`Mii`) of the NIC resource, which you set at runtime. `Mii` is an attribute of the NIC resource and has a default setting of 0.

Setting `Mii` and `miimon`

For the following cases, the name of the monitored bond interface is `B`. If you do not use one of the following cases to set up bonding, the bonding driver can potentially provide incorrect health status. This incorrect health status can result in VCS failing to fault the resource appropriately.

Case 1

Accept defaults—`miimon` is 0 and `Mii` is 1. Each of `B`'s slaves must support the `netif_carrier_ok` in-kernel call.

Case 2

When you set `miimon` to anything except 0 (`miimon!=0`) and `Mii` to 1, both the hardware and the drivers of each of `B`'s slaves must support the MII-based health monitoring.

Case 3

When you set `Mii` to 0, the NIC agent uses ping, which each card supports. In this case, the `miimon` setting is irrelevant.

Sample configurations

Configuration for using Mii

If the NIC does not respond to Mii, the agent uses network statistics to monitor the device.

```
NIC groupx_eth0 (  
  Device = eth0  
  Mii = 1  
  PingOptimize = 1  
)
```

Configuration for using network hosts

```
NIC groupx_eth0 (  
  Device = eth0  
  NetworkHosts = { "166.93.2.1", "166.99.1.2" }  
)
```

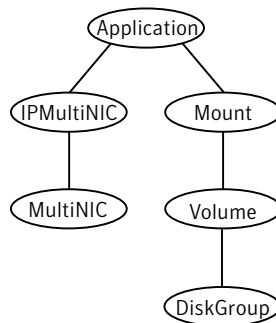
IPMultiNIC agent

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 to a backup NIC. If multiple service groups have IPMultiNICs associated with the same MultiNICA resource, only one group has 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

IPMultiNIC resources depend on MultiNICA resources.

Figure 3-3 Sample service group for an IPMultiNIC resource



Agent functions

Online	Configures a virtual IP address on one interface of the MultiNICA resource.
Offline	Removes the virtual IP address from one interface of the MultiNICA resource.
Monitor	Checks if the virtual IP address is configured on one interface of the MultiNICA resource.

State definitions

ONLINE	Indicates that the specified IP address is assigned to the device. Sends out a gratuitous ARP.
OFFLINE	Indicates that the specified IP address is not assigned to the device.
UNKNOWN	Indicates that the agent can not determine the state of the resource. This state may be due to an incorrect configuration.

Attributes

Table 3-5 Required attributes

Required attribute	Description
Address	The virtual IP address that is assigned to the active NIC. Type and dimension: string-scalar Example: "10.128.10.14"
MultiNICAResName	Name of the associated MultiNICA resource that determines the active NIC. Type and dimension: string-scalar Example: "MultiNICA_grp1"

Table 3-6 Optional attributes

Optional attribute	Description
NetMask	Netmask for the virtual IP address. Specify the value of NetMask in decimal (base 10). Type and dimension: string-scalar Example: "255.255.255.0"
Options	The <code>ifconfig</code> command options for the virtual IP address. Type and dimension: string-scalar Example: "mtu m"

Resource type definition

```
type IPMultiNIC (
    static int MonitorTimeout = 200
    static int OfflineMonitorInterval = 120
    static int ToleranceLimit = 2
    static str ArgList[] = { Address, NetMask, MultiNICAResName,
        Options, "MultiNICAResName:Probed" }
```

```
    str Address
    str MultiNICAResName
    str NetMask
    str Options
)
```

Sample configuration: IPMultiNIC and MultiNICA

Refer to the MultiNICA agent for more information.

```
cluster foo (
    UserNames = { admin = "cDRpdxPmHpzS." }
    CounterInterval = 5
)
system vcslx3 (
)
system vcslx4 (
)
group grp1 (
    SystemList = { vcslx3 = 1, vcslx4 = 2 }
)

IPMultiNIC ip1 (
    Address = "192.123.10.177"
    MultiNICAResName = mnic
    NetMask = "255.255.248.0"
)

MultiNICA mnic (
    Device @vcslx3 = { eth0 = "192.123.10.127", eth1 =
"192.123.11.127" }
    Device @vcslx4 = { eth0 = "192.123.10.128", eth2 =
"192.123.11.128" }
    NetMask = "255.255.248.0"
    NetworkHosts = { "192.123.10.129", "192.123.10.130" }
)

ip1 requires mnic

// resource dependency tree
//
//          group grp1
//          {
//          IPMultiNIC ip1
//          {
//          MultiNICA mnic
//          }
//          }
//          }
```

MultiNICA agent

The MultiNICA agent represents a set of network interfaces, and provides failover capabilities between them. You can use the agent to make IP addresses on multiple-adapter systems highly available and to monitor them.

The IPMultiNIC agent depends upon the MultiNICA agent to select the most preferred NIC on the system. IPMultiNIC brings the virtual IP online or offline. However, if the MultiNICA resource changes its active device, the MultiNICA agent handles the shifting of IP addresses.

If a NIC on a system fails, the MultiNICA agent selects another active NIC. The agent then shifts the virtual IP address to the newly selected active NIC. Only in a case where all the NICs that form a MultiNICA agent fail, does the virtual IP address shift to another system.

If you associate an interface with a MultiNICA resource, do not associate it with any other MultiNICA or NIC resource. If the same set of interfaces must be a part of multiple service groups, configure the following:

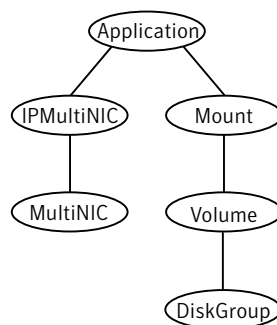
- A MultiNICA resource in one of the service groups, and
- The Proxy resources that point to the MultiNICA resource in the other service groups.

The MultiNICA agent can operate in two modes “[IP Conservation Mode \(ICM\)](#)” on page 82 and “[Performance Mode \(PM\)](#)” on page 82. With sufficient IP addresses, use PM.

Dependencies

The IPMultiNIC resources depend on the MultiNICA resources.

Figure 3-4 Sample service group for a MultiNICA resource



IP Conservation Mode (ICM)

Requires fewer IP addresses than Performance Mode, but provides slower failover.

Configuration

When a MultiNICA resource controls all the NICs of a cluster node, the NICs must have the same base IP address. This IP address must be unique, and cannot appear on any other NIC on any other node. You do not need to enable the base IP addresses beforehand. This mode does not support failing back the NIC, see the optional Failback attribute.

Operation

When you specify all the NICs with the same base IP address, the agent runs in ICM. It enables the base IP address on the active NIC.

In case of a failover, it moves the base IP address to the new active NIC. It also moves all the virtual IP addresses that are configured on that NIC. It then tries to find the first working NIC in the order of priority.

Performance Mode (PM)

Requires more IP addresses than ICM, but provides faster failover. You do not have to spend time enabling and disabling base IP addresses and reinstating lost routes, thus no resultant service disruption occurs.

Configuration

The MultiNICA agent controls each NIC, and each NIC must have a unique base IP address. The base IP address cannot appear on any other NIC on the same node or any other node. The base IP address of all the devices in a single MultiNICA resource must belong to the same subnet in the configuration.

When you configure a single NIC under a MultiNICA resource, the MultiNICA agent uses PM. The base IP addresses have to be enabled on each NIC under MultiNICA control. The addresses need to be enabled before starting VCS and handing over the management of the NICs to the agent.

Operation

The agent uses this mode when all NICs under the MultiNICA agent have separate base IP addresses specified. The mode requires that you enable the base IP addresses before starting VCS. When a NIC goes down, the agent migrates only virtual IP addresses.

In this mode, you can set the Failback attribute to 1 or 0:

- If you set the Failback attribute to 1, in each monitor cycle the agent checks to see if a preferred NIC is up. If the NIC is up, it selects that NIC as the active NIC and moves the virtual IP addresses to the preferred NIC.
- If you set the Failback attribute to 0, the agent selects a new active NIC only if the current active NIC fails. It selects the new active NIC in the order of priority.

Agent function

Monitor	Uses Medium Independent Interface (MII) to request the device status. If the hardware does not respond, the agent sends a ping to the hosts that are listed in the NetworkHosts attribute. If the ping test fails, the agent checks for activity on a configured interface by sampling the input packets that are received on that interface. If the agent does not detect activity, it forces activity by sending out a broadcast ping. If the agent does not receive a network reply, it migrates to the most suitable next interface.
---------	--

Attributes

Table 3-7 Required attributes

Required attribute	Description
Device	<p>List of devices and associated base IP addresses. This attribute must be specified separately for each system in the SystemList. Specify the devices in the list in the order of priority. The first device that the agent determines is “up” becomes the active device, to which the agent assigns a corresponding IP address.</p> <p>For IP Conservation Mode (ICM): if all the NICs configured in the Device attribute are down, the MultiNICA agent faults the resource after a 2-3 minute interval. This delay occurs because the MultiNICA agent tests the failed NIC several times before it marks the resource offline. The engine log records messages that provide a detailed description of the failover events. Find the engine log in /var/VRTSvcs/log/engine_A.log.</p> <p>Type and dimension: string-association</p> <p>Example:</p> <pre>Device@vcslinux1={ eth1 = "10.212.100.178", eth2 = "10.212.102.178" } Device@vcslinux2 = { eth2 = "10.212.100.179", eth3 = "10.212.102.179" }</pre>
NetMask	<p>The netmask that is associated with the base IP address. The value must be specified in decimal (base 10).</p> <p>Type and dimension: string-scalar</p>

Table 3-8 Optional attributes

Optional attribute	Description
Failback	<p>For Performance Mode (PM): this attribute determines if the active NIC should be changed to a preferred NIC, even though the current NIC is healthy. If operating in the ICM mode, change the value to 0.</p> <p>Type and dimension: boolean-scalar</p> <p>Default: 1</p>

Table 3-8 Optional attributes

Optional attribute	Description
NetworkHosts	<p>List of hosts on the network that receive pings to determine the state of the NICs. Specify the IP address of the host, not the host name. Include the hosts that all the NICs in the Device list can reach. If more than one network host is listed, monitor returns ONLINE if the ping test is successful with at least one of the hosts.</p> <p>Type and dimension: string-vector</p>
Options	<p>The <code>ifconfig</code> options that you want to use when you assign the base IP address to the active device.</p> <p>Type and dimension: string-scalar</p> <p>Example: "broadcast 10.212.100.255"</p>
PingOptimize	<p>Determines whether or not a broadcast ping is sent before checking network statistics, which are used to determine the state of the NIC (if MII is not supported and the ping to NetworkHosts does not confirm the NIC is up.) A value of 1 indicates a broadcast ping does not occur, a value of 0 indicates a broadcast ping occurs.</p> <p>Type and dimension: integer-scalar</p> <p>Default: 1</p>
RouteOptions	<p>Assignment of a base IP address to a device, which is followed by a route <code>add</code> command. The command has the options specified for this attribute. RouteOptions are applicable only when configuring the local host as the default gateway. No routes are added if this string is set to NULL.</p> <p>Type and dimension: string-scalar</p> <p>Example: "default 166.98.16.103"</p>

Resource type definition

```

type MultiNICA (
    static int MonitorTimeout = 240
    static str ArgList[] = { Device, NetMask, Options,
RouteOptions, PingOptimize, MonitorOnly, NetworkHosts,
Failback }
    static str Operations = None
    str Device{}
    str NetMask
    str Options
    str RouteOptions
    int PingOptimize = 1
    str NetworkHosts[]
    boolean Failback = 1
)

```

Sample configurations

MultiNICA and IPMultiNIC Performance Mode configuration

In this example, two systems (vcslx3 and vcslx4) each have a pair of network interfaces (eth0 and eth1, eth0 and eth2). These interfaces have different physical IP addresses and the agent behaves in Performance Mode (PM).

The MultiNICA resource fails over only the logical IP address to the backup NIC in the event of a failure. The resource ip1 has the Address attribute, which contains the logical IP address. In the event of a NIC failure on vcslx3, the logical IP address fails over from eth0 to eth1. In the event that eth1 fails—the address fails back to eth0—as long as eth0 is reconnected.

However, if both the NICs on vcslx3 are disconnected, the MultiNICA and IPMultiNIC resources work in tandem to fault the group on vcslx3. The entire group fails over to vcslx4.

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

See “[IPMultiNIC agent](#)” on page 64.

```

cluster foo (
    UserNames = { admin = "cDRpdxPmHpzS." }
    CounterInterval = 5
)

system vcslx3 (
)
system vcslx4 (

```

```

    )

group grp1 (
    SystemList = { vcslx3 = 1, vcslx4 = 2 }
)

IPMultiNIC ip1 (
    Address = "192.123.10.177"
    MultiNICAResName = mnic
    NetMask = "255.255.248.0"
)

MultiNICA mnic (
    Device @vcslx3 = { eth0 = "192.123.10.127", eth1 =
"192.123.11.127" }
    Device @vcslx4 = { eth0 = "192.123.10.128", eth2 =
"192.123.11.128" }
    NetMask = "255.255.248.0"
    NetworkHosts = { "192.123.10.129", "192.123.10.130" }
)

ipl requires mnic

// resource dependency tree
//
// group grp1
//   {
//     IPMultiNIC ip1
//       {
//         MultiNICA mnic
//       }
//   }
// }

```

MultiNICA and IPMultiNIC IP Conservation Mode Configuration

In this example, two systems (vcslx3 and vcslx4) each have a pair of network interfaces (eth0 and eth1, eth0 and eth2). These interfaces have a common physical IP address and the agent behaves in IP Conservation Mode (ICM).

The MultiNICA resource fails over both the physical IP and the logical IP addresses to the backup NIC in the event of a failure. The resource ip1 has the Address attribute, which contains the logical IP address. In the event of a NIC failure on vcslx3, the IP addresses fail over from eth0 to eth1. In the event that eth1 fails—the addresses fail back to eth0—if eth0 is reconnected.

However, if both the NICs on vcslx3 are disconnected, the MultiNICA and IPMultiNIC resources work in tandem to fault the group on vcslx3. The entire group fails over to vcslx4.

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

See “[IPMultiNIC agent](#)” on page 64.

```
cluster foo (
    UserNames = { admin = "cDRpdxPmHpzS." }
    CounterInterval = 5
)

system vcslx3 (
)
system vcslx4 (
)

group grp1 (
    SystemList = { vcslx3 = 1, vcslx4 = 2 }
)
IPMultiNIC ip1 (
    Address = "192.123.10.177"
    MultiNICAResName = mnic
    NetMask = "255.255.248.0"
)

MultiNICA mnic (
    Device @vcslx3 = { eth0 = "192.123.10.127", eth1 =
"192.123.10.127" }
    Device @vcslx4 = { eth0 = "192.123.10.128", eth2 =
"192.123.10.128" }
    NetMask = "255.255.248.0"
    NetworkHosts = { "192.123.10.129", "192.123.10.130" }
    Failback = 0
)

ip1 requires mnic

// resource dependency tree
//
// group grp1
// {
// IPMultiNIC ip1
// {
// MultiNICA mnic
// }
// }
```

DNS agent

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

- The host name to IP address (A, AAAA, or PTR record)
- The 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, name server, SOA, and PTR records.

Use the DNS agent when the failover source and target nodes are on different subnets. The agent updates the name server and allows clients to connect to the failed over instance of the application service.

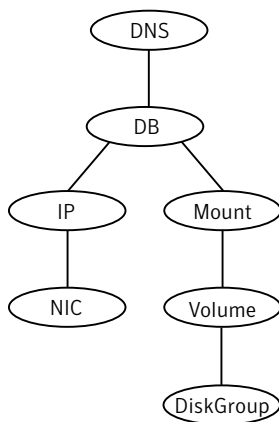
For important information about this agent, refer to:

“[DNS agent notes](#)” on page 96

Dependencies

No dependencies exist for the DNS resource.

Figure 3-5 Sample service group for a DNS resource



Agent functions

Online	<p>Sends a DNS query to retrieve the Start of Authority (SOA) record of the zone that the Domain agent attribute defines. The master server's name is in the SOA field. Unless you define the StealthMasters attribute, it is the only server for the update. When you define the StealthMasters attribute, only the servers that the attribute defines are updated.</p> <p>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 servers serve the forward (A or AAAA) and reverse zones.</p>
Offline	<p>Removes the Online lock file.</p> <p>If attribute OffDelRR is true, offline removes all records that the ResRecord keys define.</p>
Monitor	<p>Returns the ONLINE state if at least one name server reports all mappings that ResRecord or Hostname and Alias defines. The name servers are the master or StealthMaster, and all the servers for which an NS record for the zone exists.</p>
Clean	<p>Removes the Online lock file, if it exists.</p>
Open	<p>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 RRs that the ResRecord or Hostname and Alias attributes define. If one name server cannot report all the RRs, the agent function removes the Online lock file.</p>
Action	<p>Different action agent functions follow:</p> <ul style="list-style-type: none">■ keyfile.vfd This action entry point checks if the key file as specified in the TSIGKeyFile attribute exists either locally or on shared storage.■ dig.vfd This action entry point checks if dig and nsupdate binaries exist and are executable.■ master.vfd This action entry point checks if stealth masters are pingable from the node.

State definitions

ONLINE	All the RRs that one name server reports.
OFFLINE	Indicates an offline state when either of the following is true: <ul style="list-style-type: none">■ The online lock does not exist.■ At least one server cannot report all of the RRs' mappings.
UNKNOWN	A problem exists with the configuration. 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

Table 3-9 Required attributes

Required attribute	Description
Domain	<p>A string representing the DNS zone that the agent administers. The domain name can only contain alphanumeric symbols and the dash.</p> <p>Type and dimension: string-scalar</p> <p>Examples:</p> <ul style="list-style-type: none"> ■ Forward mapping "demo.symantec.com" ■ IPv4 reverse mapping "2.168.192.in-addr.arpa"
<ul style="list-style-type: none"> ■ Hostname and Alias or ■ ResRecord 	<p>You must use either the ResRecord attribute only or the HostName and Alias attributes. Do not use all three attributes together.</p>
Alias	<p>A string representing the alias to the canonical name.</p> <p>Type and dimension: string-scalar</p> <p>Example: "www"</p> <p>Where www is the alias to the canonical name mtv.symantec.com.</p>
Hostname	<p>A string that represents the canonical name of a system.</p> <p>Type and dimension: string-scalar</p> <p>Example: "mtv.symantec.com"</p>

Table 3-9 Required attributes

Required attribute	Description
ResRecord	<p>You can use the ResRecord attribute alone, or you can use the Hostname and Alias attributes.</p> <p>ResRecord is an association of DNS resource record values. Each ResRecord attribute consists of two values: <i>DNS record key</i> = <i>DNS record data</i>. Note that the record key must be a unique value.</p> <p>If the resource record list contains any invalid value as a part of the record key or a record value of the ResRecord attribute, the resource enters an UNKNOWN state.</p> <p>Type and dimension: association-scalar</p> <p>Examples:</p> <ul style="list-style-type: none"> ■ For forward mapping, where the zone is demo.symantec.com: <ul style="list-style-type: none"> - sles901 = "192.168.2.191" - ww2 = sles901 - sles9ip6 = "2007::1:2:3:abc" ■ For forward mapping, where the zone is demo.symantec.com. A multi-home DNS record, typically for one host with two network interfaces, different address, but the same DNS name. This results in two-A records, or a single A record with continuation lines. <ul style="list-style-type: none"> sle902 = "192.168.2.102 10.87.13.22" <p>A multi-home AAAA DNS record can be configured as below:</p> <ul style="list-style-type: none"> sle902 = "1234::5678 1234::AABB:CCDD" ■ For reverse IPv4 address mapping, where the zone is 2.168.192.in-addr.arpa: <ul style="list-style-type: none"> 191 = "sles901.demo.symantec.com" ■ For reverse IPv6 address mapping, where the zone is 3.0.0.0.2.0.0.0.1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.7.0.0.2.ip6.arpa: <ul style="list-style-type: none"> cba = "sles9ip6.demo.symantec.com" ■ Use only partial host names. If you use a fully qualified domain name, append a period "." at the end of the name. For CNAME records, use: <ul style="list-style-type: none"> - ResRecord = { www = mydesktop } or - ResRecord = { www = "mydesktop.marketing.db.com." } <p>Where the Domain attribute is "marketing.db.com"</p>

Table 3-10 Required attributes

Required attribute	Description
ResRecord (cont.)	<p>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 type is as follows:</p> <ul style="list-style-type: none"> ■ PTR: if the Domain attribute ends with .arpa ■ A: if the record data field is four sets of numbers, where a space separates each set. The following details the pattern it tries to match: [1-223].[0-255].[0-255].[0-255] Hexadecimal is not supported. ■ AAAA: if the record data fields are in multiple sets of hexadecimal format, then this record is an IPv6 associated type AAAA record. ■ CNAME: for any other results. <p>Note: If a name in the ResRecord attribute does not comply with RFC 1035, then a warning is issued to the log file. The ResRecord association is not used.</p>

Table 3-11 Optional attributes

Optional attribute	Description
TTL	<p>A non-zero integer represents the “Time To Live” value, in seconds, for the DNS entries in the zone that you want to update.</p> <p>A lower value means more hits on your DNS server, while a higher value means more time for your clients to learn about changes.</p> <p>The time-in-seconds value 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.</p> <p>Type and dimension: integer-scalar</p> <p>Default: 86400</p> <p>Example: "3600"</p>

Table 3-11 Optional attributes

Optional attribute	Description
StealthMasters	<p>The list of primary master name servers in the domain.</p> <p>This attribute is optional since the first name server is retrieved from the zones SOA (Start of Authority) record.</p> <p>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 zone's SOA record. It is hidden to prevent direct attacks from the Internet.</p> <p>Type and dimension: string-keylist</p> <p>Example: { "10.190.112.23" }</p>
TSIGKeyFile	<p>Required when you configure DNS for secure updates. Specifies the absolute path to the file containing the private TSIG (Transaction Signature) key.</p> <p>Type and dimension: string-scalar</p> <p>Example:</p> <p><code>/var/tsig/Kexample.com.+157+00000.private</code></p>
CreatePTR	<p>Use the CreatePTR attribute to direct the online agent function 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 record. Before you can use this attribute, the same master or stealth servers must serve the forward (A or AAAA) and reverse zones.</p> <p>Type and dimension: boolean-scalar</p> <p>Default: 0</p> <p>Example: 1</p>
OffDelRR	<p>Use the OffDelRR attribute to direct the offline agent function to remove all the records that the ResRecord key defines. You must set the value of this attribute to true (1) to have the agent remove all the records.</p> <p>The online agent function always adds records if they do not exist.</p> <p>Type and dimension: boolean-scalar</p> <p>Default: 0</p> <p>Example: 1</p>

Resource type definition

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

DNS agent notes

The DNS agent has the following notes:

- [“High availability fire drill”](#) on page 96
- [“Monitor scenarios”](#) on page 97
- [“Sample Web server configuration”](#) on page 97
- [“Secure DNS update for BIND 9”](#) on page 97
- [“Setting up secure updates using TSIG keys for BIND 9”](#) on page 97

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 DNS resources, the high availability drill performs the following, it:

- 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 are pingable 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 *Veritas Cluster Server User’s Guide*.

Monitor scenarios

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 3-12 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

Take the former Veritas corporate web server as an example. A person using a web browser specifies the URL `www.veritas.com` to view the Veritas Web page. Where `www.veritas.com` maps to the canonical name `mtv.veritas.com`, which is a host in Mountain View running the web server. The browser, in turn, retrieves the IP address for the web server by querying the domain name servers. If VCS fails the web server for `www.veritas.com` from Mountain View to Heathrow, the domain name servers must be updated with the new canonical name mapping. This update occurs so that the web browsers are directed to Heathrow instead of Mountain View. The DNS agent should update the name server to change the mapping of `www.veritas.com`. From `mtv.veritas.com` to the canonical name of the standby system in Heathrow, `hro.veritas.com`, in case of a failover.

Secure DNS update for BIND 9

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, which is available in 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

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

To use secure updates using TSIG keys

- 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 -n HOST example.com.  
Kexample.com.+157+00000
```

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

```
# cat Kexample.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-update` 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 is 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 a= "/var/tsig/Kexample.com.+157+00000.private"  
)
```

File share agents

This chapter contains the following:

- [“About the file service agents”](#) on page 99
- [“NFS agent”](#) on page 100
- [“NFSRestart agent”](#) on page 110
- [“Share agent”](#) on page 115
- [“About the Samba agents”](#) on page 121
- [“NetBIOS agent”](#) on page 131
- [“SambaServer agent”](#) on page 123
- [“SambaShare agent”](#) on page 128

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`, `mountd`, `statd`, and `lockd` daemons required by all exported NFS file systems. Always configure the NFSRestart agent along with NFS agent. Do not configure the NFS resource alone in a group.

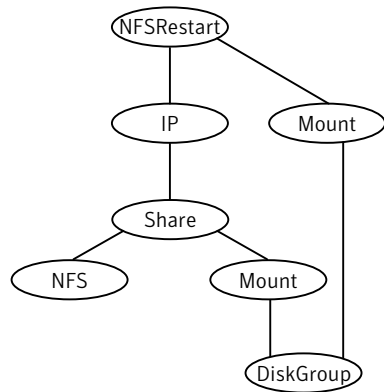
Symantec recommends that you configure only one NFS resource on a system. If you want to configure multiple NFS resources on a system, create multiple proxies pointing to the NFS resource that you already created.

Symantec recommends that you configure only one 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 can use a Proxy resource. The Proxy resource can point to the NFS resource in the first group. This use of the Proxy resource prevents redundant monitoring of the NFS daemons on the same system.

Dependencies

The NFS resource does not depend on other resources.

Figure 4-1 Sample service group for an NFS resource



Prerequisites for NFS lock recovery

If you plan on using lock recovery on a Linux system, store locking information on shared storage so that it is accessible to the system where NFS fails over. Using this information, NFS carries out lock recovery. See [“Configuring NFS lock on shared storage”](#) on page 103 for an NFS lock recovery example configuration.

Agent functions

Online	<p>Starts nfsd and mountd daemons for all kernels. If daemons are not running, the agent starts the daemons and exits.</p> <ul style="list-style-type: none">■ For Red Hat, the agent also starts the lockd and statd daemons.■ For Suse, the agent also starts the lockd daemon. If the Address attribute is set, then sm-notify is started. The sm-notify daemon sends lock recovery notifications, which then perform their jobs and terminate automatically.
Monitor	<p>Monitors versions 2 and 3 of the nfsd daemon and versions 1, 2, and 3 of the mountd daemon for all kernels.</p> <ul style="list-style-type: none">■ For Red Hat, the agent also monitors the lockd and statd daemons. Monitors versions 1, 3, 4 of the lockd daemon and version 1 of the statd daemon. When the value of the NFSv4Support attribute is 1, nfsd version 4 is also monitored.■ For Suse, the agent also monitors the lockd daemon. Monitors versions 1, 3, 4 of the lockd daemon. Monitors version 1 of statd.
Clean	<p>Stops nfsd and mountd daemons for all kernels.</p> <ul style="list-style-type: none">■ For Red Hat, the agent also stops the lockd and statd daemons.■ For Suse, the agent also stops the lockd daemon.

State definitions

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

Table 4-1 Optional attributes

Optional attributes	Description
Address	<p>Set the attribute to the name or the IP address that the clients use to mount file systems.</p> <p>Type and dimension: string-scalar</p> <p>Example: "11.124.205.20"</p>
GracePeriod	<p>Required when the value of the NFSRestart attribute is 1. GracePeriod specifies the amount of time that lock recovery is allowed by the NFS server after its reboot.</p> <p>Type and dimension: integer-scalar</p> <p>Default: 90</p>
LockFileTimeout	<p>The NFS and the NFSRestart agents require a synchronization mechanism when the group to which they belong is in transition, for example going online or coming offline. A file serves as this synchronization mechanism. The LockFileTimeout attribute specifies the maximum time that the synchronization file exists.</p> <p>Type and dimension: integer-scalar</p> <p>Default: 180</p>
Nproc	<p>Specifies the number of concurrent NFS requests that the server can handle.</p> <p>Type and dimension: integer-scalar</p> <p>Default: 8</p> <p>Example: 16</p>
NFSSecurity	<p>Specifies whether to start the NFS security daemon rpc.svcgssd or not.</p> <p>Type and dimension: boolean-scalar</p> <p>Default: 0</p>

Table 4-1 Optional attributes

Optional attributes	Description
NFSv4Support	<p>Specifies whether to start the NFSv4 daemon <code>rpc.idmapd</code> or not and whether to monitor <code>nfsd</code> version 4.</p> <p>Type and dimension: boolean-scalar</p> <p>Default: 0</p>

Resource type definition

```

type NFS (
    static int RestartLimit = 1
    static str Operations = OnOnly
    static str ArgList[] = { Nproc, Address, GracePeriod,
        NFSSecurity, NFSv4Support, LockFileTimeout }
    int Nproc = 8
    str Address
    int GracePeriod = 90
    boolean NFSSecurity = 0
    boolean NFSv4Support = 0
    int LockFileTimeout = 180
)

```

Sample configurations

Setting Nproc configuration

Refer to the configuration for Share agent for more information.

```

NFS nfs_group_16 (
    Nproc = 16
)

```

Configuring NFS lock on shared storage

```

include "types.cf"

cluster vcs_cluster (
    CounterInterval = 5
)

system sysa(
)

system sysb(
)

```

```
)  
  
group test_grp (  
  SystemList = { sysa=0, sysb=1 }  
)  
  
DiskGroup test_dg (  
  DiskGroup = test_dg  
)  
  
IP test_ip (  
  Device = eth0  
  Address = "10.182.13.28"  
  NetMask = "255.255.240.0"  
)  
  
Mount test_mnt (  
  MountPoint = "/test_mnt"  
  BlockDevice = "/dev/vx/dsk/test_dg/test_vol"  
  FSType = ext3  
  MountOpt = rw  
  FsckOpt = "-y"  
)  
  
Mount test_lockinfo_mnt (  
  MountPoint = "/lockinfo"  
  BlockDevice = "/dev/vx/dsk/test_dg/test_lockinfo_vol"  
  FSType = ext3  
  MountOpt = rw  
  FsckOpt = "-y"  
)  
  
NFS test_nfs (  
  Address = "10.182.13.28"  
)  
  
NFSRestart test_nfsrestartres (  
  NFSLockFailover = 1  
  LocksPathName = "/test_mnt"  
  NFSRes = test_nfs  
)  
  
Share test_share (  
  PathName = "/test_mnt"  
  Options = "-o rw"  
)  
  
Volume test_lockinfo_vol (  
  Volume = test_lockinfo_vol  
  DiskGroup = test_dg  
)  
  
Volume test_vol (  
  Volume = test_vol
```

```

        DiskGroup = test_dg
    )

    test_nfsrestartres requires test_ip
    test_nfsrestartres requires test_lockinfo_mnt
    test_lockinfo_mnt requires test_lockinfo_vol
    test_lockinfo_vol requires test_dg
    test_ip requires test_share
    test_share requires test_nfs
    test_share requires test_mnt
    test_mnt requires test_vol
    test_vol requires test_dg

```

NFSv4Support attribute configuration

This sample configuration is for NFS when the NFSv4Support attribute is 1. Note that the share test_share0 has fsid = 0 in Options attribute. This indicates that /home/export is root of all the exports. The client needs to mount only this root filesystem instead of mounting all shares individually.

The syntax is:

```
mount -t nfs4 <server>:/ <mountpoint>
```

Note that path after colon(:) is always /

Also note that all the filesystems other than the root filesystem should have the nohide option set in Options attribute of share resources. Set the nohide option so that authentic clients can seamlessly move through the tree of exported filesystems just by mounting the root filesystem.

```

include "types.cf"
cluster vcs_139 (
    UserNames = { admin = IJKcJEjGKfKKiSKeJH, a = gJd }
    Administrators = { admin, a }
)
system vcslinux139 (
)
system vcslinux140 (
)
group nfstest (
    SystemList = { vcslinux139 = 0, vcslinux140 = 1 }
)
DiskGroup test_dg1 (
    DiskGroup = nfsdg
)
IP test_ip (
    Device = eth0
    Address = "10.212.88.37"
    NetMask = "255.255.254.0"
)
Mount test_mnt0 (
    MountPoint = "/home/export"
    BlockDevice = "/dev/vx/dsk/nfsdg/vol0"
)

```

```
FSType = vxfs
MountOpt = rw
FsckOpt = "-n"
)
Mount test_mnt1 (
  MountPoint = "/home/export/nshare1"
  BlockDevice = "/dev/vx/dsk/nfsdg/vol1"
  FSType = vxfs
  MountOpt = rw
  FsckOpt = "-n"
)
Mount test_mnt2 (
  MountPoint = "/home/export/nshare2"
  BlockDevice = "/dev/vx/dsk/nfsdg/vol2"
  FSType = vxfs
  MountOpt = rw
  FsckOpt = "-n"
)
Mount test_mnt3 (
  MountPoint = "/home/export/nshare3"
  BlockDevice = "/dev/vx/dsk/nfsdg/vol3"
  FSType = vxfs
  MountOpt = rw
  FsckOpt = "-n"
)
NFS test_nfs (
  Nproc = 10
  NFSv4Support = 1
)
NFSRestart test_nfsres (
  NFSRes = test_nfs
)
NIC test_nic (
  Device = eth0
)
Share test_share0 (
  PathName = "/home/export"
  Options = "rw,nohide,fsid=0"
)
Share test_share1 (
  PathName = "/home/export/nshare1"
  Options = "rw,nohide"
)
Share test_share2 (
  PathName = "/home/export/nshare2"
  Options = "rw,nohide"
)
Share test_share3 (
  PathName = "/home/export/nshare3"
  Options = "rw,nohide"
)
Volume test_vol0 (
```

```
        DiskGroup = nfsdg
        Volume = vol0
    )
Volume test_vol1 (
    DiskGroup = nfsdg
    Volume = vol1
)
Volume test_vol2 (
    DiskGroup = nfsdg
    Volume = vol2
)
Volume test_vol3 (
    DiskGroup = nfsdg
    Volume = vol3
)
test_ip requires test_nic
test_ip requires test_share0
test_ip requires test_share1
test_ip requires test_share2
test_ip requires test_share3
test_mnt0 requires test_vol0
test_mnt1 requires test_mnt0
test_mnt1 requires test_vol1
test_mnt2 requires test_mnt0
test_mnt2 requires test_vol2
test_mnt3 requires test_mnt0
test_mnt3 requires test_vol3
test_nfsres requires test_ip
test_share0 requires test_mnt0
test_share0 requires test_nfs
test_share1 requires test_mnt1
test_share1 requires test_nfs
test_share2 requires test_mnt2
test_share2 requires test_nfs
test_share3 requires test_mnt3
test_share3 requires test_nfs
test_vol0 requires test_dg1
test_vol1 requires test_dg1
test_vol2 requires test_dg1
test_vol3 requires test_dg1

// resource dependency tree
//
//     group nfstest
//     {
//     NFSRestart test_nfsres
//     {
//     IP test_ip
//     {
//     NIC test_nic
//     Share test_share0
```

```
//      {
//      Mount test_mnt0
//      {
//      Volume test_vol0
//      {
//      DiskGroup test_dg1
//      }
//      }
//      NFS test_nfs
//      }
//      Share test_share1
//      {
//      Mount test_mnt1
//      {
//      Mount test_mnt0
//      {
//      Volume test_vol0
//      {
//      DiskGroup test_dg1
//      }
//      }
//      Volume test_vol1
//      {
//      DiskGroup test_dg1
//      }
//      }
//      NFS test_nfs
//      }
//      Share test_share2
//      {
//      Mount test_mnt2
//      {
//      Mount test_mnt0
//      {
//      Volume test_vol0
//      {
//      DiskGroup test_dg1
//      }
//      }
//      Volume test_vol2
//      {
//      DiskGroup test_dg1
//      }
//      }
//      NFS test_nfs
//      }
//      Share test_share3
//      {
//      Mount test_mnt3
//      {
//      Mount test_mnt0
//      {
```

```
//          Volume test_vol0
//          {
//              DiskGroup test_dg1
//          }
//          }
//          Volume test_vol3
//          {
//              DiskGroup test_dg1
//          }
//          }
//          NFS test_nfs
//          }
//      }
//  }
```

NFSRestart agent

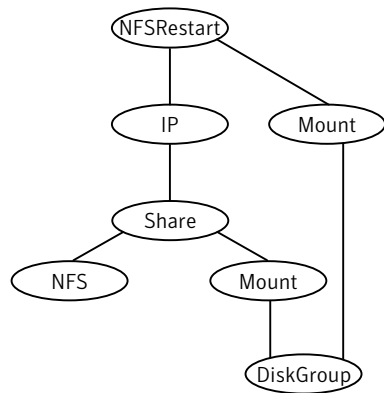
The NFSRestart agent recovers NFS record locks after sudden reboots or crashes on clients and servers. This avoids file corruption and provides high availability for NFS record locks.

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 shared-storage to the local directory (/var/lib/nfs) and vice-versa.

Dependencies

This resource must be at the top of the resource dependency tree of a service group. Only one NFSRestart resource should be configured in a service group. The NFSRestart, NFS, and Share agents must be in same service group.

Figure 4-2 Sample service group for an NFSRestart resource



Agent functions

- Online
- Terminates statd and lockd.
 - If the value of the NFSLockFailover attribute is 1, it copies the locks from the shared storage to the /var/lib/nfs directory.
 - Starts the statd and lockd daemons.
 - Starts the smsyncd daemon to copy the contents of the /var/lib/nfs directory for Linux to the shared storage (LocksPathName) at regular, two-second intervals if the value of the NFSLockFailover attribute is 1.

Monitor	It monitors the smsyncd daemon if the value of the NFSLockFailover attribute is 1.
Offline	<ul style="list-style-type: none">■ 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	<ul style="list-style-type: none">■ Terminates the statd and lockd daemons to clear the lock state.■ Terminates the nfsd and mountd daemons to close TCP/IP connections.■ Terminates the smsyncd daemon if the daemon is running.
Action	<ul style="list-style-type: none">■ nfsconf.vfd Checks the runlevel information of the system service nfslock to confirm that the lock daemons do not come online automatically after reboot.■ lockdir.vfd Verifies that the NFS lock directory (which is specified by the LocksPathName attribute of NFSRestart) is on shared storage.
nfs_postoffline	■ Restarts all the required NFS daemons if needed.

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.
FAULTED	Indicates that the NFS daemons are not running properly.

Attributes

Table 4-2 Optional attributes

Optional attribute	Description
LocksPathName	<p>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.</p> <p>Type and dimension: string-scalar</p>
NFSLockFailover	<p>NFS Lock recovery is done for all the Share resources that are configured in the group of this resource.</p> <p>Type and dimension: boolean-scalar</p> <p>Default: 0</p>
NFSRes	<p>Name of the NFS resource on the system. This attribute is required if the value of the NFSLockFailover attribute is 1.</p> <p>Type and dimension: string-scalar</p>

Resource type definition

```

type NFSRestart (
    static keylist SupportedActions = { "lockdir.vfd",
    "nfsconf.vfd" }
    static str ArgList[] = { "NFSRes:Nproc", "NFSRes:Address",
    "NFSRes:GracePeriod", "NFSLockFailover", LocksPathName}
    str NFSRes
    str LocksPathName
    boolean NFSLockFailover = 0
)

```

NFSRestart agent notes

The NFSRestart agent has the following notes:

- “[High availability fire drill](#)” on page 113
- “[Providing a fully qualified host name](#)” on page 113

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 *Veritas Cluster Server User's Guide*.

Providing a fully qualified host name

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

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

```
/etc/hosts
```

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

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

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

For example:

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

You have to make sure that the NFS client stores the same information for the NFS server as the client uses while mounting the file system. For example, if the NFS client mounts the file system using fully qualified domain names for the NFS server, then the NFS client directory: `/var/statmon/sm` directory should

also have a fully qualified domain name after the acquisition of locks. Otherwise, you need to start and stop the NFS client twice using the `/etc/init.d/nfs.client` script 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 NFS lock recovery:

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

For no NFS lock recovery:

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

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 shared are on shared disks.

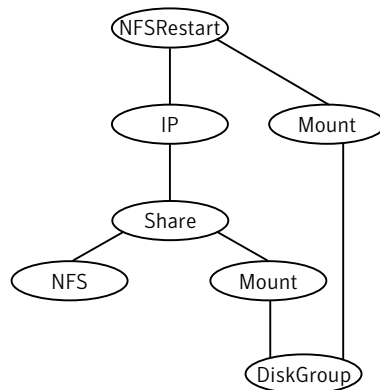
For important information on this agent, refer to:

“[Share agent notes](#)” on page 117

Dependencies

Share resources depend on NFS. In NFS service group IP and IPMultiNIC resources depend on Share resources.

Figure 4-3 Sample service group for a Share resource



Agent functions

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`

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

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.

Attributes

Table 4-3 Required attributes

Required attribute	Description
<code>PathName</code>	Pathname of the file system to be shared. Type and dimension: string-scalar Example: <code>"/share1x"</code>

Table 4-4 Optional attributes

Optional attribute	Description
<code>Client</code>	The Share agent accepts as many clients as the user wishes provided all the clients are exported the same ' <code>PathName</code> '. Client or host where the directory specified by <code>PathName</code> is exported. The client can be a wild card (*) or a fully qualified domain name (FQDN) including the host name. Type and dimension: string-scalar Example: If "outland" is the host name, the FQDN hostname is <code>outland.symantec.com</code> .

Table 4-4 Optional attributes

Optional attribute	Description
Options	<p>Options to the <code>exportfs</code> command. When specifying multiple options, separate them with commas, for example: <code>"rw, no_root_squash"</code></p> <p>For more information about the <code>exportfs</code> command and its options, refer to the <code>exportfs</code> manpage.</p> <p>Type and dimension: string-vector</p> <p>Default = <code>"ro, async, wdelay, root_squash"</code></p>
OtherClients	<p>The Client attribute can be assigned one FQDN host name, whereas multiple FQDN host names can be assigned to the 'OtherClients' field.</p> <p>A combination of 'Client' and 'OtherClients' can be used to specify the host names.</p> <p>If both of the Client and OtherClients attributes are left unspecified, the PathName is exported to the world (*).</p> <p>Type and dimension: string-vector</p>

Resource type definition

```

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

```

Share agent notes

The following section contains notes on the Share agent.

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 *Veritas Cluster Server User's Guide*.

Sample configurations

Configuration 1

```
Share ShareMnt (  
    PathName = "/mnt"  
    Client = vcslinux2  
    Options = rw  
)
```

Configuration 2

In this example, a disk group and the volumes defined on it form the resource dg_dg1. Mount volumes defined on dg_dg1 at /extdisk1 and /extdisk2 and share them with clients for NFS mounting. This configuration ensures that the mounting process occurs prior to exporting, that NFS is online prior to sharing the directories, and that the IP address is online after sharing the directories.

```
system sysA  
  
system sysB  
  
group groupA (  
    SystemList = { sysA = 0, sysB = 1 }  
    AutoStartList = { sysA }  
)  
  
DiskGroup dg_dg1 (  
    DiskGroup = dg1  
    StartVolumes = 1  
    StopVolumes = 1  
)  
  
IP ip_172_29_9_100 (  
    Device = eth0  
    Address = "172.29.9.100"  
)  
  
Mount mount_extdisk1 (  
    MountPoint = "/extdisk1"  
    BlockDevice = "/dev/vx/dsk/dg1/vol1"  
    FSType = vxfs  
)
```

```
Mount mount_extdisk2 (
    MountPoint = "/extdisk2"
    BlockDevice = "/dev/vx/dsk/dg1/vol2"
    FSType = vxfs
)

NFS nfs_groupA (
    Nproc = 16
)

NIC nic_groupA_eth0 (
    Device = eth0
)

Share share_extdisk1 (
    PathName = "/extdisk1"
    Client = "172.29.9.93"
    Options = "rw,no_root_squash"
)

Share share_extdisk2 (
    PathName = "/extdisk2"
    Client = "172.29.9.93"
    Options = "rw,no_root_squash"
)

ip_172_29_9_100 requires nic_groupA_eth0
ip_172_29_9_100 requires share_extdisk1
ip_172_29_9_100 requires share_extdisk2
mount_extdisk1 requires dg_dg1
mount_extdisk2 requires dg_dg1
share_extdisk1 requires mount_extdisk1
share_extdisk1 requires nfs_groupA
share_extdisk2 requires mount_extdisk2
share_extdisk2 requires nfs_groupA

// resource dependency tree
//
// group groupA
// {
// IP ip_172_29_9_100
//     {
//         NIC nic_groupA_eth0
//         Share share_extdisk1
//             {
//                 Mount mount_extdisk1
//                     {
//                         DiskGroup dg_dg1
//                     }
//             }
//         NFS nfs_groupA
//     }
// }
```

```
//          Share share_extdisk2
//          {
//          Mount mount_extdisk2
//              {
//                  DiskGroup dg_dg1
//              }
//          NFS nfs_groupA
//          }
//      }
// }
```

About the Samba agents

Samba is a suite of programs that allows a system running a UNIX or Linux 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 start using meta-daemon `inetd`.
- Verify that the `smbd` and `nmbd` daemons are in the path environment variable.
- If they are not, verify that they run from the default directory `/usr/bin`.
 - The path of `smbd` and `nmbd` is `/usr/bin`.
- 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 configuration ensures that the WINS server database and Domain Master are created on the shared disk.

Supported versions

[Table 4-5](#) provides the support matrix for the Samba agents.

Table 4-5 Supported platforms, architectures, and Samba versions

Platform	Architecture	Supported Samba versions
Linux	RHEL 4.0 U3 x86	Version 3.0.10-1.4E.6
	RHEL 4.0 U3 x86_64	
	RHEL 4.0 U3 ppc64	
	SLES9 SP3 x86	Version 3.0.20b-3.4-SUSE
	SLES9 SP3 x86_64	
	SLES9 SP3 ppc64	

Configuring the Samba agents

If Samba is configured properly, and the configuration file is identical on all cluster systems, configure resources of type SambaServer and NetBios only. This ensures that all shares in the Samba configuration file are failed over when the SambaServer resource fails over. Note that the Samba shares are not monitored. To monitor the Samba shares, configure the agents with the following dependencies:

```
SambaShare requires NetBios
SambaShare requires SambaServer
NetBios requires IP
```

For example, use the following configuration to monitor Samba shares SambaShare1 and SambaShare2. Use multiple resources of type SambaShare (if necessary), but only one resource each of type NetBios and SambaServer.

```
SambaShare1 requires NetBios1
SambaShare1 requires SambaServer1
SambaShare2 requires NetBios1
SambaShare2 requires SambaServer1
NetBios1 requires IP_1
```

SambaServer agent

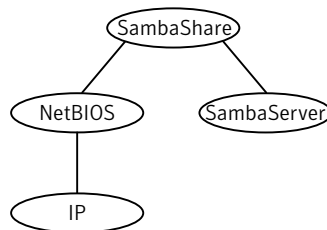
The SambaServer agent starts, stops, and monitors the `smbd` process as a daemon. Only one resource of this type is permitted. You can use the agent to make a `smbd` daemon highly available or to monitor it.

The `smbd` daemon provides Samba share services. The agent makes a copy of `smbd` for each client and verifies that Samba is running by reading the `pid` of this 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

No dependencies exist for the SambaServer resource.

Figure 4-4 Sample service group for a SambaServer resource



Agent functions

Online	Starts the <code>smbd</code> daemon at specified ports.
Offline	Stops the <code>smbd</code> daemon.
Monitor	Verifies that the <code>smbd</code> daemon is running by reading its <code>pid</code> file. Does in-depth monitoring periodically, if configured, by establishing a socket connection to Samba and sending it a NetBIOS session request.
Clean	Stops the <code>smbd</code> daemon.

State definitions

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.

Attributes

Table 4-6 Required attributes

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

Table 4-7 Optional attributes

Optional attribute	Description
IndepthMonitorCyclePeriod	Number of monitor cycles after which the in-depth monitoring is performed. For example, the value 5 indicates that the agent monitors the resource in-depth every five monitor cycles. The value 0 indicates that the agent will not perform in-depth monitoring for the resource. Type and dimension: integer-scalar Default: 5

Table 4-7 Optional attributes

Optional attribute	Description
Ports	<p>Ports where Samba accepts connections.</p> <p>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.</p> <p>For Samba version less than 3.0, exactly one value must be provided.</p> <p>Type and dimension: integer-vector</p> <p>Default: 139, 445</p>
ResponseTimeout	<p>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.</p> <p>Type and dimension: integer-scalar</p> <p>Default: 10</p>

Resource type definitions

```
type SambaServer (  
  static int RestartLimit = 5  
  static str ArgList [] = {ConfF, LockDir,  
    IndepthMonitorCyclePeriod}  
  str ConfFile = "etc/smb.conf"  
  str LockDir = "/var/lock/samba"  
  int IndepthMonitorCyclePeriod = 5  
  int ResponseTimeout = 10  
)
```

Sample configurations

```
SambaServer samba_server (  
  ConfFile = "/etc/smb.conf"  
  LockDir = "/usr/lock/samba"  
  IndepthMonitorCyclePeriod = 3  
  ResponseTimeout = 15  
)
```

SambaShare agent

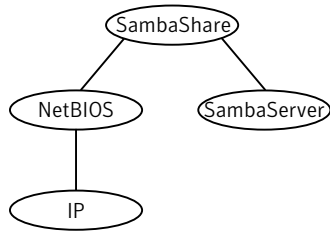
The SambaShare agent adds, removes, and monitors a share by modifying the specified Samba configuration file. You can use the agent to make a Samba Share highly available or to monitor it.

Each filesystem or printer 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

SambaShare resources depend on SambaServer, NetBios and Mount resources.

Figure 4-5 Sample service group for a SambaShare resource



Agent functions

Online	Edits the samba configuration file and adds the shares.
Offline	Removes the shares from the configuration file.
Monitor	Issues the command <code>smbclient</code> to check if the specified shares exist.
Clean	Terminates all ongoing resource actions and takes the resource offline, forcibly when necessary.

State definitions

ONLINE	Indicates that the share is available and that the share path exists.
OFFLINE	Indicates that the share is not available, or that the share has a non-existent path.
UNKNOWN	Indicates that the agent could not determine the state of the resource.

Attributes

Table 4-8 Required attributes

Required attribute	Description
SambaServerRes	Name of the SambaServer resource. Type and dimension: string-scalar Example: "SG.smb_res1" Where SG is the service group to which the resource smb_res1 belongs.
"SambaServerRes:ConfFile"	Complete path of the configuration file that is specified in the SambaServer resource.
"SambaServerRes:LockDir"	Complete path of the lock directory that is specified in the SambaServer resource.
ShareName	Name of the share resource. Type and dimension: string-scalar Example: "share1"
ShareOptions	List of parameters for the share attributes. These parameters are specified as name=value pairs, with each pair separated by a semicolon (;). Type and dimension: string-scalar Example: "path=/shared; public=yes; writable=yes"

Resource type definition

```
type SambaShare (  
    static str ArgList[] = { "SambaServerRes:ConfFile",  
        "SambaServerRes:LockDir", ShareName, ShareOptions,  
        str SambaServerRes  
    str ShareName  
    str ShareOptions  
)
```

Sample configuration

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

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 or to monitor it.

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 that `nmbd` broadcasts the NetBIOS name, or the name by which the Samba server is known in the network.

Before using this agent:

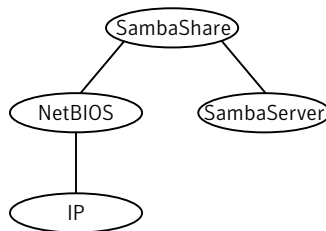
- Set the NetBIOS name.
- Set the NetBIOS interface.

Dependencies

The NetBios resource depends on the IP or the IPMultiNIC resource.

Note: You can configure only one NetBios resource on a system.

Figure 4-6 Sample service group for a NetBIOS resource



Agent functions

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

State definitions

ONLINE	Indicates that the specified NetBIOS 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: <ul style="list-style-type: none">■ NetBIOS name is not advertised.■ A NetBIOS alias is not advertised.■ Samba is not handling requests on one of the specified interfaces.■ If WINS support is configured, Samba is not providing WINS service.■ If domain support is set, Samba is not providing Domain Master service.
UNKNOWN	Indicates that the agent could not determine the state of the resource.

Attributes

Table 4-9 Required attributes

Required attribute	Description
NetBiosName	Name by which the Samba server is known in the network. Type and dimension: string-scalar
"SambaServerRes:ConfFile"	Complete path of the configuration file that is specified in the SambaServer resource. Type and dimension: string-scalar
"SambaServerRes:LockDir"	Complete path of the lock directory that is specified in the SambaServer resource.

Table 4-10 Optional attributes

Optional attribute	Description
Interfaces	List of network interfaces on which Samba handles browsing. Type and dimension: string-vector Example: "172.29.9.24/16"
NetBiosAliases	List of additional names by which the Samba server is known in the network. Type and dimension: string-vector Example: "host1_samba, myname"
WinsSupport	If set to 1, this flag causes the agent to configure Samba as a WINS server. Type and dimension: integer-scalar Default: 0

Resource type definition

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

Sample configuration

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

Service and application agents

This chapter contains the following agents:

- [“Apache Web server agent”](#) on page 136
- [“Application agent”](#) on page 149
- [“Process agent”](#) on page 156
- [“ProcessOnOnly agent”](#) on page 160

About the service and application agents

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

Apache Web server agent

The Apache Web server agent brings an Apache Server online, takes it offline, and monitors its processes. The Apache Web server agent consists of resource type declarations and agent scripts. You use the Apache Web server agent, in conjunction with other agents, to make an Apache Web server highly available. This agent supports the Apache HTTP server 1.3, 2.0, and 2.2. It also supports the IBM HTTP Server 1.3 and 2.0.

This agent can detect when an Apache Web 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.

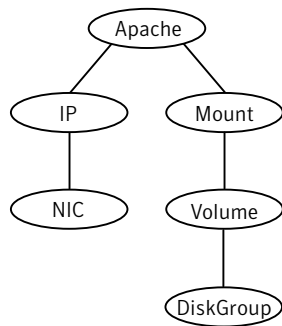
Note: The Apache agent requires an IP resource for operation.

For more information regarding this agent:
See [“Apache Web server notes”](#) on page 142.

Dependencies

This type of resource depends on IP and Mount resources.

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



Agent functions

Online Starts an Apache server by executing the httpdDir/httpd program with the appropriate arguments. When you specify a file with the EnvFile attribute, the file is sourced before the agent executes the httpd command.

Offline	<p>To stop the Apache HTTP server, the agent:</p> <ul style="list-style-type: none">■ Executes the httpdDir/httpd program with the appropriate arguments (Apache v2.0), or■ Sends a TERM signal to the HTTP Server parent process (Apache v1.3). <p>When you specify a file with the EnvFile attribute, the file is sourced before the agent executes the httpd command.</p>
Monitor	<p>Monitors the state of the Apache server. First it checks for the processes, next it can perform an optional state check.</p>
Clean	<p>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.</p>
Action	<p>checkconffile.vfd</p> <p>Checks for the existence of the Apache configuration file and the existence of the directory that contains the httpd binary that is used during start up. For a local installation, if the config file or HttpdDir is not found, make sure that it exists on the failover node.</p>

State definitions

ONLINE	<p>Indicates that the Apache server is running.</p>
OFFLINE	<p>Indicates that the Apache server is not running.</p> <p>Can also indicate that the administrator has stopped the Web server gracefully. Note that the agent uses the PidFile attribute for intentional offline detection.</p>
UNKNOWN	<p>Indicates that a problem exists with the configuration.</p>

Attributes

Table 5-1 Required attributes

Required attribute	Description
ConfigFile	<p>Full path and file name of the main configuration file for the Apache server.</p> <p>Type and dimension: string-scalar</p> <p>Example: <code>"/apache/server1/conf/httpd.conf"</code></p>
httpdDir	<p>Full path of the directory to the httpd binary file</p> <p>Type and dimension: string-scalar</p> <p>Example: <code>"/apache/server1/bin"</code></p>
SecondLevelMonitor	<p>Enables second-level monitoring for the resource. Second-level monitoring is a deeper, more thorough state check of the Apache HTTP server. An HTTP GET request on the Web server's root directory performs the monitoring. Valid attribute values are 1 (true) and 0 (false). Specifying this attribute is required.</p> <p>Type and dimension: boolean-scalar</p> <p>Default: 0</p> <p>Example: <code>"1"</code></p>
ResLogLevel	<p>Controls the agent's logging detail for a specific instance of a resource. Values are:</p> <ul style="list-style-type: none"> ■ ERROR: Logs error messages. ■ WARN: Logs error and warning messages. ■ INFO: Logs error, warning, and informational messages. ■ TRACE: Logs error, warning, informational, and trace messages. Trace logging is verbose. Use for initial configuration or troubleshooting. <p>Type and dimension: string-scalar</p> <p>Default: INFO</p> <p>Example: <code>"TRACE"</code></p>

Table 5-1 Required attributes

Required attribute	Description
PidFile	This attribute is required when you want to enable the detection of a graceful shutdown outside of VCS control. See " PidFile " on page 141.

Table 5-2 Optional attributes

Optional attribute	Description
DirectiveAfter	A list of directives that httpd processes after reading the configuration file. Type and dimension: string-association Example: DirectiveAfter{} = { KeepAlive=On }
DirectiveBefore	A list of directives that httpd processes before it reads the configuration file. Type and dimension: string-association Example: DirectiveBefore{} = { User=nobody, Group=nobody }
User	Account name the agent uses to execute the httpd program. If you do not specify this value, the agent executes httpd as the root user. Type and dimension: string-scalar Example: "apache1"

Table 5-2 Optional attributes

Optional attribute	Description
EnableSSL	<p>Set to 1 (true) to have the online agent function add support for SSL by including the option <code>-DSSL</code> in the start command. For example: <code>/usr/sbin/httpd -f path_to_httpd.conf -k start -DSSL</code></p> <p>Where <code>path_to_httpd.conf</code> file is the path to the <code>httpd.conf</code> file.</p> <p>Set to 0 (false) it excludes the <code>-DSSL</code> option from the command.</p> <p>Type and dimension: boolean-scalar</p> <p>Default: 0</p> <p>Example: "1"</p>
HostName	<p>The virtual host name that is assigned to the Apache server instance. The host name is used in second-level monitoring to establish a socket connection with the Apache HTTP server.</p> <p>Note: The <code>HostName</code> attribute is only required when the value of <code>SecondLevelMonitor</code> is 1 (true).</p> <p>Type and dimension: string-scalar</p> <p>Example: "web1.symantec.com"</p>
Port	<p>Port number where the Apache HTTP server instance listens. The port number is used in second-level monitoring to establish a socket connection with the server. Specify this attribute only if <code>SecondLevelMonitor</code> is set to 1 (true).</p> <p>Type and dimension: integer-scalar</p> <p>Default: 80</p> <p>Example: "80"</p>
EnvFile	<p>Full path and file name of the file that is sourced before executing <code>httpdDir/httpd</code>. With Apache 2.0, the file <code>ServerRoot/bin/envvars</code>, which is supplied in most Apache 2.0 distributions, is commonly used to set the environment before executing <code>httpd</code>. Specifying this attribute is optional. If <code>EnvFile</code> is specified, the shell for user root must be Bourne, Korn, or C shell.</p> <p>Type and dimension: string-scalar</p> <p>Example: "/apache/server1/bin/envvars"</p>

Table 5-2 Optional attributes

Optional attribute	Description
PidFile	<p>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.</p> <p>This attribute is required when you want the agent to detect the graceful shutdown of the Web server. For the agent to detect the graceful shutdown of the Web server, the value of the IntentionalOffline resource type attribute must be 1 (true).</p> <p>Type and dimension: string-scalar</p> <p>Example: <code>/var/run/httpd.pid</code></p>
SharedObjDir	<p>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 <code>-R</code> option when executing the <code>httpd</code> program. Refer to the <code>httpd</code> man pages for more information about the <code>-R</code> option.</p> <p>Type and dimension: boolean-scalar</p> <p>Example: <code>"/apache/server1/libexec"</code></p>
SecondLevelTimeout	<p>The number of seconds that the monitor agent function waits on the execution of second-level monitor. If the second-level monitor program does not return to calling the monitor agent function before the SecondLevelTimeout window expires, the monitor agent function no longer blocks on the program sub-process. It does, however, report that the resource is offline. The value should be high enough to allow the second level monitor enough time to complete. The value should be less than the value of the agent's MonitorTimeout.</p> <p>Type and dimension: integer-scalar</p> <p>Default: 30</p>

Resource type definition

```
type Apache (
  static keylist SupportedActions = { "checkconffile.vfd" }
  static str ArgList[] = { ResLogLevel, State, IState, httpdDir,
  SharedObjDir, EnvFile, PidFile, HostName, Port, User,
  SecondLevelMonitor, SecondLevelTimeout, ConfigFile, EnableSSL,
  DirectiveAfter, DirectiveBefore }
  str ResLogLevel = INFO
  str httpdDir
  str SharedObjDir
  str EnvFile
  str PidFile
  str HostName
  int Port = 80
  str User
  boolean SecondLevelMonitor
  int SecondLevelTimeout = 30
  str ConfigFile
  boolean EnableSSL
  str DirectiveAfter{}
  str DirectiveBefore{}
  static int IntentionalOffline = 0
)
```

Apache Web server notes

The Apache Web server has the following notes:

- [“Tasks to perform before you use the Apache Web server agent”](#) on page 142
- [“Detecting application failure”](#) on page 143
- [“About bringing an Apache Web server online outside of VCS control”](#) on page 144
- [“About the ACC Library”](#) on page 144
- [“High Availability fire drill”](#) on page 145

Tasks to perform before you use the Apache Web 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 Web 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 Web server online outside of VCS control”](#) on page 144.

- 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.
- Install the ACC Library 5.0.30.00 (VRTSacclib-5.0.30.00-MP3_GENERIC) if it is not already installed. If the ACC Library needs to be installed or updated, the library and its documentation can be obtained from the agent software media.
- Remove previous versions of this agent.
- The service group has disk and network resources to support the Apache server resource.
- Assign virtual host name and port to Apache Server.

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 `SecondLevelMonitor` is set to `true`, then a socket connection is established with the Apache HTTP server using the values specified by the `Host` and `Port` agent attributes. When connected, the agent issues an HTTP request to the server to test its ability to respond. If the HTTP Server responds with a return code between 0 and 408, the agent considers the server online. If the server fails to respond or returns any other code, the agent considers the server offline.

About bringing an Apache Web server online outside of VCS control

When you bring an Apache Web 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 Web server outside of VCS control

- 1 Source the environment file if required.
- 2 Start the Apache Web server. You must use the `-f` option so that the agent can distinguish different instances of the server.

```
httpdDir/httpd -f ConfigFile -k start
```

Where *httpdDir* is `/apache/v2.2/bin` *ConfigFile* is `/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
```

About the ACC Library

The agent functions for the Apache HTTP server depend on a set of Perl modules that are known as the ACC Library. The ACC Library contains the common, reusable functions that perform tasks such as process identification, logging, and system calls.

When you install the ACC library in a VCS environment, you must install the ACC library package before you install the agent.

To install or update the ACC library package, locate the library and related documentation on the agent disc and in the compressed agent tar file.

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 Web 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 *Veritas Cluster Server User's Guide*.

Sample configurations

Running two versions of httpd

This example shows how two versions of `httpd` can run from different locations. In group `Apache_1`, `httpd` runs from Port 80, the default location. The configuration file in `/usr/local/apache/conf/httpd.conf` should indicate `DocumentRoot`, address, port, and other parameters. In group `Apache_2`, `httpd` runs from `/home/web/apache`. The PID file for this is created in `/home/web/apache/logs/httpd.pid`. The configuration file in `/home/web/apache/conf/httpd.conf` should define parameters for running this version of `httpd`.

Each Apache resource requires an online IP resource. In this example, each Apache resource requires an online mount resource to mount block devices from disks reserved by the Disk Reservation agent.

```
system sysa

system sysb

group Apache_1 (
    SystemList = { sysa ,sysb}
    AutoStartList = { sysa}
)

Apache myapacheWeb
    httpdDir = "/mnt/apache/bin"
    SecondLevelMonitor = 1
    ConfigFile = "/mnt/apache/conf/httpd.conf"
    HostName = "server1.mydomain.com"
    Port = 80
)

IP myapacheIP(
    Device = "eth0"
```

```
        Address="192.168.50.50"
        NetMask="255.255.255.0"
    )

    NIC myapacheNIC(
        Device="eth0"
        NetworkHosts={"172.29.9.178", "172.29.9.179"}
    )

    Mount myapacheMnt(
        MountPoint="/mnt/apache/"
        BlockDevice="/dev/sdd2"
    )

    DiskReservation myapacheDiskRes(
        Disks ="/dev/sdd"
    )

    myapacheMnt requires myapacheDiskRes
    myapacheIP requires myapacheNIC
    myapacheWeb requires myapacheIP
    myapacheWeb requires myapacheMnt

    group Apache_2 (
        SystemList = { sysa,sysb}
        AutoStartList = { sysa}
    )

    Apache myapacheWeb2(
        httpdDir = "/mnt/apache1/bin"
        SecondLevelMonitor = 1
        ConfigFile = "/mnt/apache1/conf/httpd.conf"
        HostName = "server2.mydomain.com"
        Port = 8080
    )

    IP myapacheIP2(
        Device = "eth1"
        Address="192.168.60.50"
        NetMask="255.255.255.0"
    )

    NIC myapacheNIC2(
        Device="eth1"
    )

    Mount myapacheMnt2(
        MountPoint="/mnt/apache1/"
        BlockDevice="/dev/sdc3"
    )
```

```
DiskReservation myapacheDiskRes2 (  
    Disks = "/dev/sdc"  
)  
  
myapacheMnt2 requires myapacheDiskRes2  
myapacheIP2 requires myapacheNIC2  
myapacheWeb2 requires myapacheIP2  
myapacheWeb2 requires myapacheMnt2
```

Sample main.cf file

```
include "types.cf"  
  
cluster Cluster1 (  
    UserNames = { admin = xxxxxx }  
)  
  
system SystemA (  
)  
system SystemB (  
)  
  
group Web1 (  
    SystemList = { SystemA = 0, SystemB = 1 }  
)  
  
    DiskGroup Web1_dg (  
        DiskGroup = web1  
    )  
  
    Volume Web1_vol (  
        DiskGroup = web1  
        Volume = volweb1  
    )  
  
    IP Web1_ip (  
        Device = eth0  
        Address = "10.212.88.220"  
        NetMask = "255.255.254.0"  
    )  
  
    Mount Web1_mnt (  
        MountPoint = "/apache/srvr01"  
        BlockDevice = "/dev/vx/dsk/web1/volweb1"  
        FSType = vxfs  
        FsckOpt = "-y"  
    )  
  
    NIC Web1_nic (  
        Device = eth0  
    )
```

```
Apache Web1_http (  
  HostName = spartan  
  Port = 80  
  SecondLevelMonitor = 1  
  SecondLevelTimeout = 25  
  httpdDir = "/apache/srvr01/bin"  
  EnvFile = "/apache/srvr01/bin/envvars"  
  PidFile = /apache/srvr01/log/httpd.pid"  
  ConfigFile = "/apache/srvr01/conf/httpd.conf"  
  IntentionalOffline = 1  
)
```

```
Web1_ip requires Web1_nic  
Web1_mnt requires Web1_vol  
Web1_vol requires Web1_dg  
Web1_http requires Web1_ip  
Web1_http requires Web1_mnt
```

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 must exist locally on each node. You can use this agent to provide high availability for applications that do not have 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

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 Application resources, the high availability fire drill checks for:

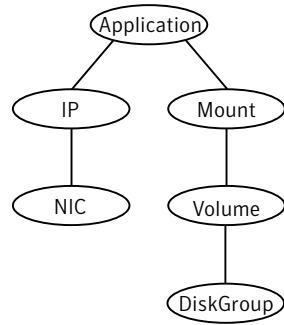
- The availability of the specified program
- Execution permissions for the specified program
- The existence of the specified user on the host
- The existence of the same binary on all nodes

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

Dependencies

Depending on how you plan to use it, this type of resource can depend on IP, IPMultiNIC, and Mount resources.

Figure 5-2 Sample service group for an Application resource



Agent functions

Online	Runs the StartProgram attribute with the specified parameters in the context of the specified user.
Offline	Runs the StopProgram attribute with the specified parameters in the context of the specified user.
Monitor	<p>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 you specify.</p> <p>Use any combination among these attributes (MonitorProgram, PidFiles, or MonitorProcesses) to monitor the application.</p> <p>If any one 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.</p>
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.

State definitions

ONLINE	Indicates that all processes that are specified in the <code>PidFiles</code> and the <code>MonitorProcesses</code> attribute are running and that the <code>MonitorProgram</code> returns <code>ONLINE</code> .
OFFLINE	Indicates that at least one process that are specified in the <code>PidFiles</code> attribute or <code>MonitorProcesses</code> is not running, or that the <code>MonitorProgram</code> returns <code>OFFLINE</code> .
UNKNOWN	Indicates an indeterminable application state or invalid configuration.

Attributes

Table 5-3 Required attributes

Required attribute	Description
StartProgram	<p>The executable, created locally on each node, 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.</p> <p>Note: Do not use the opening and closing ({}) brace symbols in this string.</p> <p>Type and dimension: string-scalar</p> <p>Example: "/usr/sbin/samba start"</p>
StopProgram	<p>The executable, created locally on each node, 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.</p> <p>Note: Do not use the opening and closing ({}) brace symbols in this string.</p> <p>Type and dimension: string-scalar</p> <p>Example: "/usr/sbin/sample_app stop"</p>
<p>At least one of the following attributes:</p> <ul style="list-style-type: none"> ■ MonitorProcesses ■ MonitorProgram ■ PidFiles 	<p>See “Optional attributes” on page 153.</p>

Table 5-4 Optional attributes

Optional attribute	Description
CleanProgram	<p>The executable, created locally on each node, 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.</p> <p>Type and dimension: string-scalar</p>
MonitorProcesses	<p>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.</p> <p>The process name must be the name that the <code>ps -ef</code> command displays for the process.</p> <p>Type and dimension: string-vector</p> <p>Example: "nmbd"</p>
MonitorProgram	<p>The executable, created locally on each node, 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.</p> <p>MonitorProgram can return the following VCSAgResState values: OFFLINE value is 100; ONLINE values range from 101 to 110 (depending on the confidence level); 110 equals confidence level of 100%. Any other value = UNKNOWN.</p> <p>Note: Do not use the opening and closing ({}) brace symbols in this string.</p> <p>Type and dimension: string-scalar</p>

Table 5-4 Optional attributes

Optional attribute	Description
PidFiles	<p>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.</p> <p>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.</p> <p>Type and dimension: string-vector</p>
User	<p>The user ID for running StartProgram, StopProgram, MonitorProgram, and CleanProgram. The processes that is 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.</p> <p>Type and dimension: string-scalar</p> <p>Default: root</p>

Resource type definition

```

type Application (
    static keylist SupportedActions = { "program.vfd", "user.vfd",
    "cksum.vfd", getcksum }
    static str ArgList[] = { User, StartProgram, StopProgram,
    CleanProgram, MonitorProgram, PidFiles, MonitorProcesses }
    str User
    str StartProgram
    str StopProgram
    str CleanProgram
    str MonitorProgram
    str PidFiles[]
    str MonitorProcesses[]
)

```

Sample configurations

Configuration 1

In this example, you configure the executable samba as StartProgram and StopProgram, with start and stop specified as command line arguments respectively. Configure the agent to monitor two processes: a process that the `smbd.pid` specifies and the process `nmbd`.

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

Configuration 2

In this example, since no user is specified, it uses the root user. The executable `samba` starts and stops the application using `start` and `stop` as the command line arguments. The executable `sambaMonitor` monitors the application and uses `all` as its command line argument. The agent also monitors the `smbd` and `nmbd` processes.

```
Application samba_app2 (  
    StartProgram = "/usr/sbin/samba start"  
    StopProgram = "/usr/sbin/samba stop"  
    CleanProgram = "/usr/sbin/samba force stop"  
    MonitorProgram = "/usr/local/bin/sambaMonitor all"  
    MonitorProcesses = { "smbd", "nmbd" }  
)
```

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 or to monitor it.

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 Process resources, the high availability fire drill checks for:

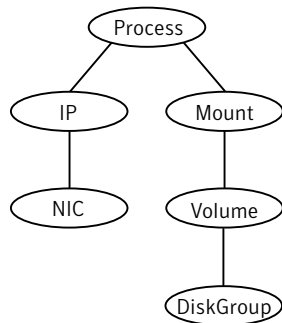
- The existence of the specified process
- Execution permissions for the specified process
- The existence of a binary executable for the specified process
- The existence of the same binary on all nodes

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

Dependencies

Depending on the context, this type of resource can depend on IP, IPMultiNIC, and Mount resources.

Figure 5-3 Sample service group for a Process resource



Agent functions

Online	Starts a process in the background with optional arguments and priority in the specified user context.
Offline	Terminates the process with a <code>SIGTERM</code> . If the process does not exit, a <code>SIGKILL</code> 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.

State definitions

ONLINE	Indicates that the specified process is running in the specified user context.
OFFLINE	Indicates that the specified process is not running in the specified user context.
FAULTED	Indicates that the process has terminated unexpectedly.
UNKNOWN	Indicates that the agent can not determine the state of the process.

Attributes

Table 5-5 Required attribute

Required attribute	Description
PathName	<p>Complete pathname 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.</p> <p>This attribute must not exceed 256 characters.</p> <p>Type and dimension: string-scalar</p> <p>Example: <code>"/usr/sbin/proc1"</code></p>

Table 5-6 Optional attributes

Optional attribute	Description
Arguments	<p>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.</p> <p>Type and dimension: string-scalar</p>
PidFile	<p>The file that contains the process ID for the monitoring process. Specify the PidFile attribute for the monitoring process to use the Pid. Otherwise, to complete the monitoring process the agent uses the ps output.</p> <p>Note that when you use scripts, or other indirect mechanisms, to start processes, you must set the PidFile attribute if the ps output is different from the configured values for the PathName or Arguments attributes.</p> <p>Type and dimension: string-scalar</p> <p>Example: <code>"/var/lock/sendmail.pid"</code></p>

Table 5-6 Optional attributes

Optional attribute	Description
Priority	Priority that the process runs. Priority values range between -20 (highest) to +19 (lowest). Type and dimension: string-scalar Default: 10
UserName	This attribute is the owner of the process. The process runs with the user ID. Type and dimension: string-scalar Default: root

Resource type definition

```

type Process (
    static keylist SupportedActions = { "program.vfd", getcksum }
    static str ArgList[] = { PathName, Arguments, UserName,
        Priority, PidFile }
    str PathName
    str Arguments
    str UserName = root
    str Priority = 10
    str PidFile
)

```

Sample configurations

Configuration

In this example, the Process agent starts, stops, and monitors sendmail. This process is started with two arguments as determined in the Arguments attribute. The pid stored in the PidFile attribute is used to monitor the sendmail process.

```

Process sendmail (
    PathName = "/usr/sbin/sendmail"
    Arguments = "-bd -q30m"
    PidFile = "/var/run/sendmail.pid"
)

```

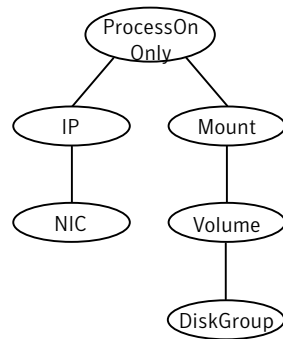
ProcessOnOnly agent

The ProcessOnOnly agent starts, stops, and monitors a process that you specify. You can use the agent to make a process highly available or to monitor it.

Dependencies

Depending on the context, this type of resource can depend on IP, IPMultiNIC, and Mount resources.

Figure 5-4 Sample service group for a ProcessOnOnly 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.
FAULTED	Indicates that the process has unexpectedly terminated.
UNKNOWN	Indicates that the agent can not determine the state of the process.

Attributes

Table 5-7 Required attributes

Required attribute	Description
PathName	<p>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. The PathName attribute must not exceed 256 characters.</p> <p>Type and dimension: string-scalar</p>

Table 5-8 Optional attributes

Optional attribute	Description
Arguments	<p>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.</p> <p>Type and dimension: string-scalar</p> <p>Example: "-bd -q30m"</p>
IgnoreArgs	<p>A flag that indicates whether monitor ignores the argument list.</p> <ul style="list-style-type: none">■ If the value is 0, it checks the process pathname and argument list.■ If the value is 1, it only checks for the executable pathname and ignores the rest of the argument list. <p>Type and dimension: boolean-scalar</p> <p>Default: 0</p>

Table 5-8 Optional attributes

Optional attribute	Description
PidFile	<p>The file that contains the process ID for the monitoring process. Specify the PidFile attribute for the monitoring process to use the Pid. Otherwise, to complete the monitoring process the agent uses the ps output.</p> <p>Note that when you use scripts, or other indirect mechanisms, to start processes, you must set the PidFile attribute when the ps output is different from the configured values for the PathName or Arguments attributes.</p> <p>Type and dimension: string-scalar Example: "/var/lock/sendmail.pid"</p>
Priority	<p>Priority with which the process will run. Priority values range between -20 (highest) to +19 (lowest).</p> <p>Type and dimension: string-scalar Default: 10</p>
UserName	<p>Owner of the process. The process runs with the user ID.</p> <p>Type and dimension: string-scalar Default: root</p>

Resource type definition

```

type ProcessOnOnly (
    static str ArgList[] = { PathName, Arguments, UserName,
    Priority, PidFile, IgnoreArgs }
    static str Operations = OnOnly
    str PathName
    str Arguments
    str UserName = root
    str Priority = 10
    str PidFile
    boolean IgnoreArgs = 0
)
    
```

Sample configurations

Configuration 1

```
ProcessOnOnly nfs_daemon(  
    PathName = "/usr/lib/nfs/nfsd"  
    Arguments = "-a 8"  
)
```

Configuration 2

```
include "types.cf"  
  
cluster ProcessCluster (  
.  
.  
.  
group ProcessOnOnlyGroup (  
    SystemList = { sysa, sysb }  
    AutoStartList = { sysa }  
)  
  
    ProcessOnOnly Process1 (  
        PathName = "/usr/local/bin/myprog"  
        Arguments = "arg1 arg2"  
    )  
  
    ProcessOnOnly Process2 (  
        PathName = "/bin/csh"  
        Arguments = "/tmp/funscript/myscript"  
    )  
  
    // resource dependency tree  
    //  
    //     group ProcessOnOnlyGroup  
    //     {  
    //     ProcessOnOnly Process1  
    //     ProcessOnOnly Process2  
    //     }
```


Infrastructure and support agents

This chapter contains the following agents:

- [“NotifierMngr agent”](#) on page 166
- [“VRTSWebApp agent”](#) on page 173
- [“Proxy agent”](#) on page 175
- [“Phantom agent”](#) on page 179
- [“RemoteGroup agent”](#) on page 181

About the infrastructure and support agents

Use the infrastructure and support agents to monitor Veritas 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. See the *Veritas Cluster Server User's 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 effective after restarting the notifier.

Other applications with the name `notifier` can interfere with the NotifierMngr agent. If `notifier` is started outside VCS control, VCS can only monitor the notifier process if its started with the absolute path. For example, use:

```
# /opt/VRTSvcs/bin/notifier -s m=xyz &
```

Dependency

The NotifierMngr resource depends on the NIC resource.

Agent functions

Online	Starts the notifier process with its required arguments.
Offline	VCS sends a <code>SIGABORT</code> . If the process does not exit within one second, VCS sends a <code>SIGKILL</code> .
Monitor	Monitors the notifier process.
Clean	Sends <code>SIGKILL</code> .

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

Required attribute	Description
SnmpConsoles	<p>Specifies the machine name of the SNMP manager and the severity level of the messages to be delivered to the SNMP manager. The severity levels of messages are <i>Information</i>, <i>Warning</i>, <i>Error</i>, and <i>SevereError</i>. Specifying a given severity level for messages generates delivery of all messages of equal or higher severity.</p> <p>SnmpConsoles is a required attribute if SntpServer is not specified; otherwise, SnmpConsoles is an optional attribute. Specify both SnmpConsoles and SntpServer if desired.</p> <p>Type and dimension: string-association</p> <p>Example: "172.29.10.89" = Error, "172.29.10.56" = Information</p>
SntpServer	<p>Specifies the machine name of the SMTP server.</p> <p>SntpServer is a required attribute if SnmpConsoles is not specified; otherwise, SntpServer is an optional attribute. You can specify both SntpServer and SnmpConsoles if desired.</p> <p>Type and dimension: string-scalar</p> <p>Example: "smtp.example.com"</p>

Table 6-2 Optional attributes

Optional attribute	Description
EngineListeningPort	<p>Change this attribute if the VCS engine is listening on a port other than its default port.</p> <p>Type and dimension: integer-scalar</p> <p>Default: 14141</p>

Table 6-2 Optional attributes

Optional attribute	Description
MessagesQueue	<p>Size of the VCS engine's message queue. Minimum value is 30.</p> <p>Type and dimension: integer-scalar</p> <p>Default: 30</p>
NotifierListeningPort	<p>Any valid, unused TCP/IP port numbers.</p> <p>Type and dimension: integer-scalar</p> <p>Default: 14144</p>
SmtpFromPath	<p>Set to a valid email address, if you want the notifier to use a custom email address in the FROM: field.</p> <p>Type and dimension: string-scalar</p> <p>Example: "usera@example.com"</p>
SmtpRecipients	<p>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.</p> <p>Note: SmtpRecipients is a required attribute if you specify SmtpServer.</p> <p>Type and dimension: string-association</p> <p>Example:</p> <p>"james@symantec.com" = SevereError, "admin@symantec.com" = Warning</p>

Table 6-2 Optional attributes

Optional attribute	Description
SmtpReturnPath	<p>Set to a valid email address, if you want the notifier to use a custom email address in the Return-Path: <> field.</p> <p>If the mail server specified in SmtpServer does not support VRFY, then you need to set the SmtpVrfyOff to 1 in order for the SmtpReturnPath value to take effect.</p> <p>Type and dimension: string-scalar</p> <p>Example: "usera@example.com"</p>
SmtpServerTimeout	<p>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.</p> <p>Type and dimension: integer-scalar</p> <p>Default: 10</p>
SmtpServerVrfyOff	<p>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.</p> <p>Type and dimension: boolean-scalar</p> <p>Default: 0</p>
SnmCommunity	<p>Specifies the community ID for the SNMP manager.</p> <p>Type and dimension: string-scalar</p> <p>Default: public</p>

Table 6-2 Optional attributes

Optional attribute	Description
SnmpdTrapPort	<p>Port on the SNMP console machine where SNMP traps are sent.</p> <p>If you specify more than one SNMP console, all consoles use this value.</p> <p>Type and dimension: integer-scalar</p> <p>Default: 162</p>

Resource type definition

```

type NotifierMngr (
  static int RestartLimit = 3
  static str ArgList[] = { EngineListeningPort, MessagesQueue,
  NotifierListeningPort, SnmpdTrapPort, SnmpCommunity,
  SnmpConsoles, SntpServer, SntpServerVrfyOff, SntpServerTimeout,
  SntpReturnPath, SntpFromPath, SntpRecipients }
  int EngineListeningPort = 14141
  int MessagesQueue = 30
  int NotifierListeningPort = 14144
  int SnmpdTrapPort = 162
  str SnmpCommunity = public
  str SnmpConsoles{}
  str SntpServer
  boolean SntpServerVrfyOff = 0
  int SntpServerTimeout = 10
  str SntpReturnPath
  str SntpFromPath
  str SntpRecipients{}
)

```

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 `SnmpConsole: snmpserv`. In this example, only messages of `SevereError` level are sent to the `SmtpServer (smtp.example.com)`, and the recipient (`vcadmin@example.com`).

Configuration

```
system north

system south

group NicGrp (
    SystemList = { north, south }
    AutoStartList = { north }
    Parallel = 1
)

Phantom my_phantom (
)

NIC    NicGrp_eth0 (
    Enabled = 1
    Device = eth0
)

group Grp1 (
    SystemList = { north, south }
    AutoStartList = { north }
)

Proxy nicproxy(
    TargetResName = "NicGrp_eth0"
```

```
)

NotifierMngr ntfr (
    SnmpConsoles = { snmpserv = Information }
    SmtServer = "smtp.example.com"
    SmtRecipients = { "vcsadmin@example.com" =
        SevereError }
)

ntfr requires nicproxy

// resource dependency tree
//
//     group Grp1
//     {
//     NotifierMngr ntfr
//         {
//             Proxy nicproxy
//         }
//     }
// }
```

VRTSWebApp agent

Brings Web applications online, takes them offline, and monitors their status. This agent is used to monitor the Web consoles of various Symantec products, such as the Cluster Management Console.

Agent functions

Online	Starts the Web application with the specified parameters. If the Web server is not already running, it first starts the server.
Offline	Removes the Web application from the Web server. If no other Web application is running, it shuts down the Web server.
Monitor	Checks if the specified Web application is currently running inside the Web server. If the application is running, monitor reports ONLINE. If the application is not running, monitor reports OFFLINE.
Clean	Removes the Web application from the Web server. If no other Web application is running, it shuts down the Web server.

State definitions

ONLINE	Indicates that the Web application is running.
OFFLINE	Indicates that the Web application is not running.
UNKNOWN	Indicates that the agent could not determine the state of the resource or that the resource attributes are invalid.

Attributes

Table 6-3 Required attributes

Required attribute	Description
AppName	Name of the application as it appears in the Web server. Type and dimension: string-scalar Example: "cmc"

Table 6-3 Required attributes

Required attribute	Description
InstallDir	<p>Path to the Web application installation. You must install the Web application as a <code>.war</code> file with the same name as the <code>AppName</code> parameter. Point this attribute to the directory that contains this <code>.war</code> file.</p> <p>Type and dimension: string-scalar</p> <p>Example: If the <code>AppName</code> is <code>cmc</code> and <code>InstallDir</code> is <code>/opt/VRTSweb/VERITAS</code>, the agent constructs the path for the Web application as: <code>/opt/VRTSweb/VERITAS/cmc.war</code></p>
TimeForOnline	<p>The time the Web application takes to start after it is loaded into the Web server. This parameter is returned as the exit value of the online script, which inform VCS of the time it needs to wait before calling monitor on the Web application resource. This attribute value is typically at least five seconds.</p> <p>Type and dimension: integer-scalar</p>

Resource type definition

```
type VRTSWebApp (
    static int NumThreads = 1
    static str ArgList[] = { AppName, InstallDir, TimeForOnline }
    str AppName
    str InstallDir
    int TimeForOnline
)
```

Sample configuration

```
VRTSWebApp VCSweb (
    AppName = "cmc"
    InstallDir = "/opt/VRTSweb/VERITAS"
    TimeForOnline = 5
)
```

Proxy agent

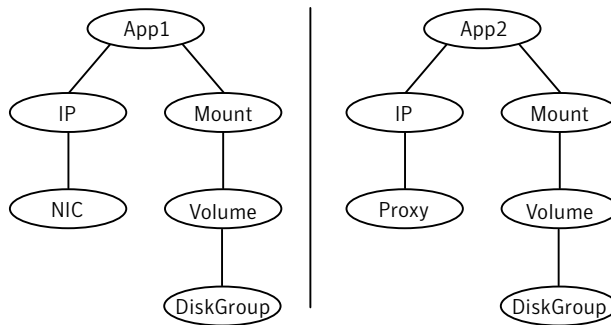
The Proxy agent mirrors the state of another resource on a local or remote system. It provides a means to specify and modify one resource and have its state reflected by its proxies. You can use the agent when you need to replicate the status of a resource.

A Proxy resource can only point to None or OnOnly type of resources, and can reside in a failover/parallel group.

Dependencies

No dependencies exist for the Proxy resource.

Figure 6-1 Sample service group for an Proxy resource



Agent functions

Monitor Determines status based on the target resource status.

Attributes

Table 6-4 Required attribute

Required attribute	Description
TargetResName	<p>Name of the target resource that the Proxy resource mirrors.</p> <p>The target resource must be in a different resource group than the Proxy resource.</p> <p>Type and dimension: string-scalar</p> <p>Example: "tmp_VRTSvcs_file1"</p>

Table 6-5 Optional attribute

Optional attribute	Description
TargetSysName	<p>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.</p> <p>Type and dimension: string-scalar</p> <p>Example: "sysa"</p>

Resource type definition

```
type Proxy (
    static int OfflineMonitorInterval = 60
    static str ArgList[] = { TargetResName, TargetSysName,
        "TargetResName:Probed", "TargetResName:State" }
    static str Operations = None
    str TargetResName
    str TargetSysName
)
```

Sample configurations

Configuration 1

The proxy resource mirrors the state of the resource tmp_VRTSvcs_file1 on the local system.

```
Proxy proxy1 (
    TargetResName = "tmp_VRTSvcs_file1"
)
```

Configuration 2

The proxy resource mirrors the state of the resource tmp_VRTSvcs_file1 on sysa.

```
Proxy proxy1(
    TargetResName = "tmp_VRTSvcs_file1"
    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 @vcslx3 = { eth0 = "192.123.8.42", eth3 =
        "192.123.8.42" }
    Device @vcslx4 = { eth0 = "192.123.8.43", eth3 =
        "192.123.8.43" }
    NetMask = "255.255.248.0"
    NetworkHosts = { "192.123.10.129", "192.123.10.130" }
)
```

```
IPMultiNIC ip1 (  
  Address = "192.123.10.177"  
  MultiNICAResName = mnic  
  NetMask = "255.255.248.0"  
)
```

```
ip1 requires mnic
```

```
group grp2 (  
  SystemList = { sysa, sysb }  
  AutoStartList = { sysa }  
)  
  
IPMultiNIC ip2 (  
  Address = "192.123.10.178"  
  NetMask = "255.255.255.0"  
  MultiNICAResName = mnic  
)  
Proxy proxy (  
  TargetResName = mnic  
)  
ip2 requires proxy
```

Phantom agent

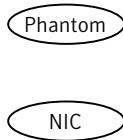
The Phantom agent enables VCS to determine the status of parallel service groups that do not include OnOff resources. Do not use the Phantom agent in failover service groups. You can use the agent to determine the state of service groups having resources of type None only.

Do not attempt manual online or offline operations on the Phantom resource or on the service group containing the Phantom resource. Doing so may result in unpredictable behavior.

Dependencies

No dependencies exist for the Phantom resource.

Figure 6-2 Sample service group for a Phantom resource



Agent functions

Monitor Determines status based on the status of the service group.

Resource type definition

```
type Phantom (  
    static str ArgList[] = { }  
)
```

Sample configurations

Configuration 1

```
Phantom (  
)
```

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, sysb }
    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.

Symantec supports the RemoteGroup agent when it points to a global group. The RemoteGroup agent must then map the state of the global group in the local cluster.

For more information on the functionality of this agent see the *Veritas Cluster Server User's Guide*.

Dependency

As a best practice, establish a RemoteGroup resource dependency on a NIC resource. Symantec recommends that the RemoteGroup resource not be by itself in a service group.

Agent functions

Online	Brings the remote service group online. See the “ControlMode” on page 184 for more information.
Offline	Takes the remote service group offline. See the “ControlMode” on page 184 for more information.
Monitor	Monitors the state of the remote service group. The true state of the remote service group is monitored only on the online node in the local cluster. See the “VCSSysName” on page 183.
Clean	If the RemoteGroup resource faults, the Clean function takes the remote service group offline. See the “ControlMode” on page 184 for more information.

State definitions

ONLINE	Indicates that 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.
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 6-6 Required attributes

Required attribute	Description
IpAddress	<p>The IP address or DNS name of a node in the remote cluster. The IP address can be either physical or virtual.</p> <p>When configuring a virtual IP address of a remote cluster, do not configure the IP resource as a part of the remote service group.</p> <p>Type and dimension: string-scalar</p> <p>Examples: "www.example.com" or "11.183.12.214"</p>
Port	<p>The port where the remote engine listens for requests.</p> <p>This is an optional attribute, unless the remote cluster listens on a port other than the default value of 14141.</p> <p>Type and dimension: integer-scalar</p> <p>Default: 14141</p>
GroupName	<p>The name of the service group on the remote cluster that you want the RemoteGroup agent to monitor or manage.</p> <p>Type and dimension: string-scalar</p> <p>Example: "DBGrp"</p>
VCSSysName	<p>You must set this attribute to either the VCS system name or the ANY value.</p> <ul style="list-style-type: none"> ■ ANY The RemoteGroup resource goes online if the remote service group is online on any node in the remote cluster. ■ VCSSysName Use the name of a VCS system in a remote cluster where you want the remote service group to be online when the RemoteGroup resource goes online. Use this to establish a one-to-one mapping between the nodes of the local and remote clusters. <p>Type and dimension: string-scalar</p> <p>Example: "vcssys1" or "ANY"</p>

Table 6-6 Required attributes

Required attribute	Description
ControlMode	<p>Select only one of these values to determine the mode of operation of the RemoteGroup resource: MonitorOnly, OnlineOnly, or OnOff.</p> <ul style="list-style-type: none">■ OnOff The RemoteGroup resource brings the remote service group online or takes it offline. When you set the VCSSysName attribute to ANY, the SysList attribute of the remote service group determines the node where the remote service group onlines.■ MonitorOnly The RemoteGroup resource only monitors the state of the remote service group. The RemoteGroup resource cannot online or offline the remote service group. Make sure that you bring the remote service group online before you online the RemoteGroup resource.■ OnlineOnly The RemoteGroup resource only brings the remote service group online. The RemoteGroup resource cannot take the remote service group offline. When you set the VCSSysName attribute to ANY, the SysList attribute of the remote service group determines the node where the remote service group onlines. <p>Type and dimension: string-scalar</p>

Table 6-6 Required attributes

Required attribute	Description
Username	<p>This is the login user name for the remote cluster.</p> <p>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.</p> <p>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 Symantec Product Authentication Service, you do not need to enter the domain name.</p> <p>For a secure remote cluster:</p> <ul style="list-style-type: none"> ■ Local Unix user user@nodename—where the nodename is the name of the node that is specified in the IPAddress attribute. Do not set the DomainType attribute. ■ NIS or NIS+ user user@domainName—where domainName is the name of the NIS or NIS+ domain for the user. You must set the value of the DomainType attribute to either to nis or nisplus. <p>Type and dimension: string-scalar</p> <p>Example:</p> <ul style="list-style-type: none"> ■ For a cluster without the Symantec Product Authentication Service: "johnsmith" ■ For a secure remote cluster: "foobar@example.com"
Password	<p>This is the password that corresponds to the user that you specify in the Username attribute. You must encrypt the password with the <code>vcscrypt -agent</code> command.</p> <p>Note: Do not use the vcscrypt utility when entering passwords from a configuration wizard or from the Cluster Management Console or the Cluster Manager (Java Console).</p> <p>Type and dimension: string-scalar</p>

Table 6-7 Optional attributes

Optional attribute	Description
DomainType	<p>For a secure remote cluster only, enter the domain type information for the specified user.</p> <p>For users who have the domain type unixpwd, you do not have to set this attribute.</p> <p>Type: string-scalar</p> <p>Example: "nis", "nisplus"</p>
BrokerIp	<p>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.</p> <p>Type: string-scalar</p> <p>Example: "128.11.295.51"</p>
OfflineWaitTime	<p>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.</p> <p>Type and dimension: integer-scalar</p> <p>Default: 0</p>

Table 6-8 Type-level attributes

Type level attributes	Description
<p>OnlineRetryLimit OnlineWaitLimit ToleranceLimit MonitorInterval AutoFailover</p>	<p>In case of remote service groups that take a longer time to Online, Symantec recommends that you modify the default OnlineWaitLimit and OnlineRetryLimit attributes.</p> <p>If you expect the RemoteGroup agent to tolerate sudden offlines of the remote service group, then modify the ToleranceLimit attribute.</p> <p>See the <i>Veritas Cluster Server User's Guide</i> for more information about these attributes.</p>

Resource type definition

```
type RemoteGroup (  
    static int OnlineRetryLimit = 2  
    static int ToleranceLimit = 1  
    static str ArgList[] = { IPAddress, Port, Username, Password,  
        GroupName, VCSSysName, ControlMode, OfflineWaitTime,  
        DomainType, BrokerIp }  
    str IPAddress  
    int Port = 14141  
    str Username  
    str Password  
    str GroupName  
    str VCSSysName  
    str ControlMode  
    int OfflineWaitTime  
    str DomainType  
    str BrokerIp  
)
```

Testing agents

This chapter contains the following agents:

- [“ElifNone agent”](#) on page 190
- [“FileNone agent”](#) on page 192
- [“FileOnOff agent”](#) on page 194
- [“FileOnOnly agent”](#) on page 196

About the program support agents

Use the program support agents to provide high availability for program support resources.

ElifNone agent

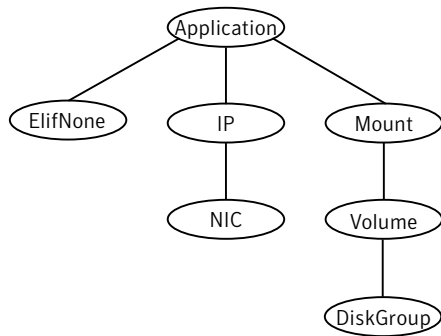
The ElifNone agent monitors a file. It checks for the file's absence.

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

No dependencies exist for the ElifNone resource.

Figure 7-1 Sample service group for an ElifNone resource



Agent function

Monitor	Checks for the specified file. If it exists, the resource faults. If it does not exist, the agent reports as ONLINE.
---------	--

Attributes

Table 7-1 Required attribute

Required attribute	Description
PathName	Specifies the complete pathname. Starts with a slash (/) preceding the file name. Type and dimension: string-scalar Example: "/tmp/file01"

Resource type definition

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

Sample configuration

```
ElifNone tmp_file01 (  
    PathName = "/tmp/file01"  
)
```

FileNone agent

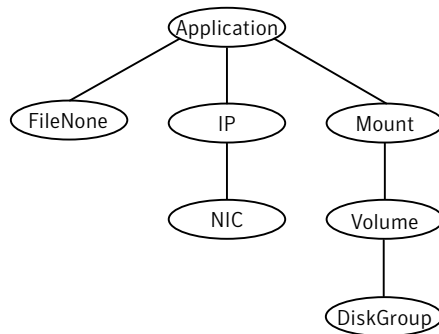
Monitors a file—checks for the file’s existence.

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

No dependencies exist for the FileNone resource.

Figure 7-2 Sample service group for an FileNone resource



Agent functions

Monitor	Checks for the specified file. If it exists, the agent reports as ONLINE. If it does not exist, the resource faults.
---------	--

Attribute

Table 7-2 Required attribute

Required attribute	Description
PathName	Specifies the complete pathname. Starts with a slash (/) preceding the file name. Type and dimension: string-scalar Example: "/tmp/file01"

Resource type definition

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

Sample configuration

```
FileNone tmp_file01 (  
    PathName = "/tmp/file01"  
)
```

FileOnOff agent

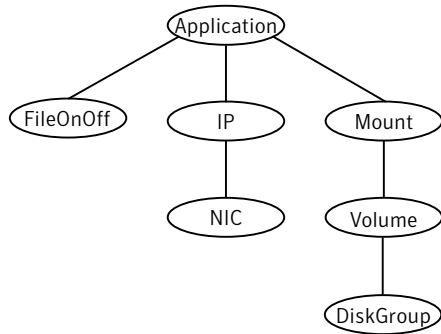
The FileOnOff agent creates, removes, and monitors files.

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

No dependencies exist for the FileOnOff resource.

Figure 7-3 Sample service group for a FileOnOff resource



Agent functions

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

Attribute

Table 7-3 Required attribute

Required attribute	Description
PathName	Specifies the complete pathname. Starts with a slash (/) preceding the file name. Type and dimension: string-scalar Example: "/tmp/file01"

Resource type definition

```
type FileOnOff (  
    static str ArgList[] = { PathName }  
    str PathName  
)
```

Sample configuration

```
FileOnOff tmp_file01 (  
    PathName = "/tmp/file01"  
)
```

FileOnOnly agent

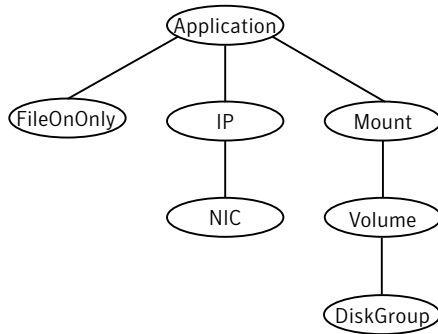
The FileOnOnly agent creates and monitors files.

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

No dependencies exist for the FileOnOnly resource.

Figure 7-4 Sample service group for a FileOnOnly resource



Agent functions

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

Attribute

Table 7-4 Required attributes

Required attribute	Description
PathName	Specifies the complete pathname. Starts with a slash (/) preceding the file name. Type and dimension: string-scalar Example: "/tmp/file02"

Resource type definition

```
type FileOnOnly (  
    static str ArgList[] = { PathName }  
    static str Operations = OnOnly  
    str PathName  
)
```

Sample configuration

```
FileOnOnly tmp_file02 (  
    PathName = "/tmp/file02"  
)
```


Glossary

administrative IP address

The operating system controls these IP addresses and brings them up even before VCS brings applications online. Use them to access a specific system over the network for doing administrative tasks, for example: examining logs to troubleshoot issues, cleaning up temp files to free space, etc. Typically, you have one administrative IP address per node.

agent function

Agent functions start, stop, fault, forcibly stop, and monitor resources using scripts. Sometimes called an entry point.

base IP address

The first logical IP address, can be used as an administrative IP address.

entry point

See [agent function](#).

floating IP address

See [virtual IP address](#).

logical IP address

Any IP address assigned to a NIC.

NIC bonding

Combining two or more NICs to form a single logical NIC, which creates a fatter pipe.

operation

All agents have scripts that turn the resource on and off. Operations determine the action that the agent passes to the resource. See None operation, OnOff operation, and OnOnly operation.

None operation

For example the NIC resource. Also called persistent resource, this resource is always on. This kind of resource has no online and offline scripts, and only monitors a resource.

OnOff operation

For example the IP and Share agents--in fact most agents are OnOff. This resource has online and offline scripts. Often this type of resource does not appear in the types file because by default when a resource does not have this resource type defined, it is OnOff.

OnOnly operation

For example the NFS, FileOnOnly resources. This kind of resource has an online script, but not an offline one.

plumb

Term for enabling an IP address—used across all platforms in this guide.

test IP address

IP addresses to help determine the state of a link by sending out a ping probe to another NIC (on another system.) Requires a return ping to complete the test. Test IP addresses can be the same as base IP addresses.

virtual IP address

IP addresses that can move from one NIC to another or from one node to another. VCS fails over these IP address with your application. Sometimes called a floating IP address.

Index

Numerics

802.1Q trunking 66

A

about

- Network agents 65
- Samba agents 121

ACC library 144

agent

- modifying 18

agent functions

- Apache Web server agent 136
- Application agent 150
- DiskGroup agent 23
- DiskGroupSnap agent 30
- DiskReservation agent 36
- DNS agent 90
- ElifNone agent 190
- FileNone agent 192
- FileOnOff agent 194
- FileOnOnly agent 196
- IP agent 68
- IPMultiNIC agent 77
- LVMLogicalVolume agent 46
- LVMVolumeGroup agent 49
- Mount agent 52
- MultiNICA agent 83
- NetBIOS agent 132
- NFS agent 101
- NFSRestart agent 110
- NIC agent 72
- NotifierMngr agent 166
- Phantom agent 179
- Process agent 157
- ProcessOnOnly agent 160
- Proxy agent 175
- RemoteGroup agent 182
- SambaServer agent 123
- SambaShare agent 128
- SANVolume agent 60
- Share agent 115

Volume agent 42

VRTSWebApp agent 173

agents

- Apache Web server 136
- Application 149
- DiskGroup 22
- DiskGroupSnap 29
- DiskReservation 36
- DNS 89
- ElifNone 190
- FileNone 192
- FileOnOff 194
- FileOnOnly 196
- IP 67
- IPMultiNIC 77
- LVMLogicalVolume 46
- LVMVolumeGroup 49
- Mount 52
- MultiNICA 81
- NetBIOS 131
- NFS 100
- NFSRestart 110
- NIC 71
- NotifierMngr 166
- Phantom 179
- Process 156
- ProcessOnOnly 160
- Proxy 175
- RemoteGroup 181
- SambaServer 123
- SambaShare 128
- SANVolume 60
- Share 115
- Volume 42
- VRTSWebApp 173

agents, typical functions 17

Apache Web server agent

- ACC library 144
- agent functions 136
- attributes 138
- description 136
- detecting application failure 143

- sample configuration 145
- state definitions 137
- Application agent
 - agent functions 150
 - attributes 152
 - description 149
 - high availability fire drill 149
 - resource type definition 154
 - sample configurations 155
 - state definitions 151
- association dimension 19
- attribute data types 18
- attributes
 - Application agent 152
 - DiskGroup agent 25
 - DiskGroupSnap agent 30
 - DiskReservation agent 37
 - DNS agent 92
 - ElifNone agent 191
 - FileNone agent 193
 - FileOnOff agent 195
 - FileOnOnly agent 197
 - IP agent 69
 - IPMultiNIC agent 79
 - LVMLogicalVolume agent 47
 - LVMVolumeGroup agent 50
 - Mount agent 55
 - MultiNICA agent, 84
 - NFS agent 102
 - NFSRestart agent 112
 - NIC agent 73
 - NotifierMngr agent 167
 - Process agent 158
 - ProcessOnOnly 161
 - Proxy agent 176
 - RemoteGroup agent 183
 - SambaServer agent 125
 - SANVolume agent 61
 - Share agent 116
 - Volume agent 43
 - VRTSWebApp agent 173
- attributes, modifying 17, 18

B

- bonded network interfaces 72
- boolean data types 18
- bundled agents 17

C

- Cluster Manager (Java Console), modifying
 - attributes 18
- Cluster Manager (Web Console)
 - modifying attributes 18
- CNAME record 97
- configuration files
 - main.cf 180
 - modifying 18
 - types.cf 17
- configuring, Samba agents 122

D

- data type
 - boolean 18
 - string 18
- data types
 - integer 18
- description, resources 17
- dimensions
 - keylist 19
 - scalar 19
 - vector 19
- DiskGroup agent
 - agent functions 23
 - attributes 25
 - description 22
 - high availability fire drill 28
 - resource type definition 27
 - sample configurations 28
 - state definitions 24
- DiskGroupSnap agent
 - agent functions 30
 - attributes 30
 - description 29
 - resource type definition 33
 - sample configurations 33
 - state definitions 30
- DiskReservation agent
 - agent functions 36
 - attributes 37
 - description 36
 - resource type definition 39
 - sample configurations 40
 - state definitions 36
- DNS agent 91
 - agent functions 90
 - attributes 92

- description 89
- resource type definition 96
- sample web server configuration 97

E

- ElifNone agent
 - agent functions 190
 - attributes 191
 - description 190
 - resource type definition 191
 - sample configuration 191

F

- Fiber Channel adapter 28
- FileNone agent
 - agent functions 192
 - attribute 193
 - description 192
 - resource type definition 193
 - sample configurations 193
- FileOnOff agent
 - agent functions 194
 - attribute 195
 - description 194
- FileOnOnly agent
 - agent functions 196
 - attribute 197
 - description 196
 - resource type definition 197
 - sample configuration 197

H

- high availability fire drill 28, 58, 67, 71, 96, 113, 149, 156

I

- integer data types 18
- IP agent
 - agent functions 68
 - attributes 69
 - description 67
 - high availability fire drill 67
 - resource type definitions 70
 - sample configurations 70
 - state definitions 68
- IPMultiNIC agent
 - agent functions 77

- attributes 79
- description 77
- resource type definitions 79
- sample configuration 80
- state definitions 78

K

- keylist dimension 19

L

- LVMLogicalVolume agent
 - agent functions 46
 - attributes 47
 - description 46
 - resource type definition 47
 - sample configurations 48
 - state definitions 47
- LVMVolumeGroup agent
 - agent functions 49
 - attributes 50
 - description 49
 - resource type definition 50
 - sample configurations 51
 - state definitions 50

M

- main.cf 17, 180
- modifying
 - Cluster Manager (Web Console) 18
 - configuration files 18
- modifying agents 18
- monitor scenarios, DNS agent 97
- monitoring bonded NICs, Linux 75
- Mount agent
 - agent functions 52, 54
 - attributes 55
 - description 52
 - high availability fire drill 58, 96, 113
 - notes 58
 - resource type definition 57
 - sample configurations 59
- MultiNICA agent
 - agent functions 83
 - attributes 84
 - description 81
 - IP Conservation mode 82
 - Performance mode 82

- resource type definitions 86
- resource type definitions, Linux 86
- sample configurations 86

N

NetBIOS agent

- agent functions 132
- description 131
- resource type definition 133
- sample configurations 134
- state definitions 132

NFS agent

- agent functions 101
- attributes 102
- description 100
- resource type definition 103
- sample configurations 103
- state definitions 101

NFS lock recovery 100

NFSRestart agent

- agent functions 110
- attributes 112
- description 110
- resource type definition 112
- sample configuration 114
- state definitions 111

NIC agent

- agent functions 72
- attributes 73
- description 71
- high availability fire drill 71
- resource type definitions 74
- sample configurations 76
- state definitions 72

NotifierMngr agent

- agent functions 166
- attributes 167
- description 166
- resource type definition 170
- sample configurations 171
- state definitions 166

O

- online query 97

P

Phantom agent

- agent functions 179
- description 179
- resource type definition 179
- sample configurations 179

prerequisites

- NFS lock recovery 100
- Samba agents 121

Process agent

- agent functions 157
- attributes 158
- description 156
- high availability fire drill 156
- resource type definition 159
- sample configurations 159
- state definitions 157

ProcessOnOnly agent

- agent functions 160
- attributes 161
- description 160
- resource type definition 162
- sample configurations 163
- state definitions 160

Proxy agent

- agent functions 175
- attributes 176
- description 175
- resource type definition 177
- sample configurations 177

R

RemoteGroup agent

- agent functions 182
- attributes 183
- description 181
- resource type definition 188
- state definitions 182

- resource type definition 43
- SambaShare agent 130

resource type definitions

- Application agent 154
- DiskGroup agent 27
- DiskGroupSnap agent 33
- DiskReservation agent 39
- DNS agent 96
- ElifNone agent 191
- FileNone agent 193
- FileOnOnly agent 197
- IP agent 70
- IPMultiNIC agent 79

- LVMLogicalVolume agent 47
- LVMVolumeGroup agent 50
- Mount agent 57
- MultiNICA agent 86
- MultiNICA agent, Linux 86
- NetBIOS agent 133
- NFS agent 103
- NFSRestart agent 112
- NIC agent 74
- NotifierMngr agent 170
- Phantom agent 179
- Process agent 159
- ProcessOnOnly agent 162
- Proxy agent 177
- RemoteGroup agent 188
- SambaServer agent 127
- SANVolume agent 62
- Share agent 117
- Volume agent 43
- VRTSWebApp agent 174
- resource types 17
- resources
 - description of 17

S

- Samba agents 121
 - overview 121
 - prerequisites 121
- Samba agents configuring 122
- SambaServer agent
 - agent functions 123
 - attributes 125
 - description 123
 - resource type definition 127
 - sample configuration 127
 - state definitions 124
- SambaShare agent 128
 - agent functions 128
 - attributes 129
 - resource type definition 130
 - sample configurations 130
 - state definitions 129
- sample configurations
 - Apache Web server agent 145
 - Application agent 155
 - DiskGroup agent 28
 - DiskGroupSnap agent 33
 - DiskReservation agent 40
 - ElifNone agent 191

- FileNone agent 193
- FileOnOff agent 195
- FileOnOnly agent 197
- IP agent 70
- IPMultiNIC 80
- LVMLogicalVolume agent 48
- LVMVolumeGroup agent 51
- Mount agent 59
- MultiNICA agent 86
- NetBIOS agent 134
- NFS agent 103
- NFSRestart agent 114
- NIC agent 76
- NotifierMngr agent 171
- Phantom agent 179
- Process agent 159
- ProcessOnOnly agent 163
- Proxy agent 177
- SambaServer agent 127
- SambaShare agent 130
- SANVolume agent 63
- Share agent 118
- Volume agent 44
- VRTSWebApp agent 174
- SANVolume agent
 - agent functions 60
 - attributes 61
 - description 60
 - resource type definition 62
 - sample configuration 63
 - state definitions 60
- scalar dimension 19
- secure DNS update 97
- setting Mii and miimon 75
- Share agent
 - agent functions 115
 - attributes 116
 - description 115
 - resource type definitions 117
 - sample configurations 118
 - state definitions 116
- state definitions 91
 - Apache Web server agent 137
 - Application agent 151
 - DiskGroup agent 24
 - DiskGroupSnap agent 30
 - DiskReservation agent 36
 - DNS agent 91
 - IP agent 68

- IPMultiNIC agent 78
- LVMLogicalVolume agent 47
- LVMVolumeGroup agent 50
- Mount agent 54
- NetBIOS agent 132
- NFS agent 101
- NFSRestart agent 111
- NIC agent 72
- NotifierMngr agent 166
- Process agent 157
- ProcessOnOnly agent 160
- RemoteGroup agent 182
- SambaServer agent 124
- SambaShare agent 129
- SANVolume agent 60
- Share agent 116
- Volume agent 43
- VRTSWebApp agent 173
- string data type 18

T

- trunking 66
- types.cf 17

V

- VCS, resource types 17
- vector dimension 19
- Volume agent
 - agent functions 42
 - attributes 43
 - description 42
 - sample configurations 44
 - state definitions 43
- VRTSWebApp agent
 - agent functions 173
 - attributes 173
 - description 173
 - resource type definition 174
 - sample configuration 174
 - state definitions 173