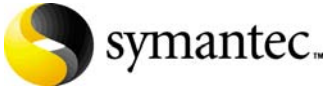
Veritas™ Cluster Server Bundled Agents Reference Guide

AIX

5.0

N18453J



Veritas Cluster Server
Bundled Agents Reference Guide

Veritas Cluster Server 5.0

Symantec Corporation
20330 Stevens Creek Blvd.
Cupertino, CA 95014
www.symantec.com

Third-party legal notices

Third-party software may be recommended, distributed, embedded, or bundled with this Symantec product. Such third-party software is licensed separately by its copyright holder. All third-party copyrights associated with this product are listed in the accompanying release notes.

AIX is a registered trademark of IBM Corporation.

Technical support

For technical assistance, visit http://support.veritas.com and select phone or email support. Use the Knowledge Base search feature to access resources such as TechNotes, product alerts, software downloads, hardware compatibility lists, and our customer email notification service.

Contents

Chapter 5 Service and application agents

Chapter 6 Infrastructure and support agents

Chapter 7 Testing agents

Chapter 1

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), Cluster Manager (Web 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 SnmpConsoles{}.

Chapter 2

Storage agents

This chapter contains:

About the storage agents

Use storage agents to Monitor shared storage.

DiskGroup agent

Brings online, takes offline, and monitors a Veritas Volume Manager (VxVM) disk group. This agent uses VxVM commands.

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 using volume sets, set StartVolumes and StopVolumes attributes of the DiskGroup resource that contains 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 being written to a volume in the disk group.

Virtual fire drill

The virtual 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 DiskGroup resources, the virtual fire drill checks for:

- The Veritas Volume Manager license
- Visibility from host for all disks in the diskgroup

For more information about using the virtual fire drill see the *VCS User's Guide*.

Agent functions

- Online
  Imports the disk group using the `vxdg` command.
- Offline
  Deports the disk group using the `vxdg` command.
- Monitor
  Determines if the disk group is online or offline using the `vxdg` command.
- Clean
  Terminates all ongoing resource actions and takes the resource offline—forcibly when necessary.
- Info
  The DiskGroup info agent function gets information from the Volume Manager and displays the type and free size for the DiskGroup resource.
  Initiate the info agent function by setting the InfoInterval timing to a value greater than 0.

In this example, the info agent function executes every 60 seconds:

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

The command to retrieve information about the DiskType and FreeSize of the DiskGroup resource is:

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

Output includes:

```
DiskType sliced
FreeSize 35354136
```

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.
- 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 configured with Veritas Volume Manager. Type and dimension: string-scalar Example: "diskgroup1"

Table 2-2 Optional attributes

Optional attributes	Description
StartVolumes	If value is 1, the DiskGroup online script starts all volumes belonging to that disk group after importing the group. Type and dimension: string-scalar Default: 1
StopVolumes	If value is 1, the DiskGroup offline script stops all volumes belonging to that disk group before deporting the group. Type and dimension: string-scalar Default: 1
TempUseFence	Do not use. For internal use only.
MonitorReservation	If the value is 1, and SCSI-3 fencing is utilized, 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
NumThreads	Number of threads used within the agent process for managing resources. This number does not include threads used for other internal purposes. Symantec strongly recommends that you set the value of the NumThreads attribute to 1. Setting this attribute to a higher value may result in agent function timeouts due to serialization of underlying commands. Default: 1
PanicSystemOnDGLoss	If the DiskGroup resource is in a DISABLED state, uses I/O fencing, and has PanicSystemOnDGLoss set to 1, the system panics in the first monitor cycle. If the DiskGroup resource is in an ENABLED state, uses I/O fencing, has PanicSystemOnDGLoss set to 1, and fulfills the FaultOnMonitorTimeout attribute's time out number, the system panics. **Note:** System administrators may want to set a high value for FaultOnMonitorTimeout to increase system tolerance. Type and dimension: boolean-scalar Default: 1

Resource type definition

```
type DiskGroup (
    static keylist SupportedActions = { "license.vfd", "disk.vfd",
    numdisks }
    static int OnlineRetryLimit = 1
    static str ArgList[] = { DiskGroup, StartVolumes, StopVolumes,
    MonitorOnly, MonitorReservation, tempUseFence }
    str DiskGroup
    str StartVolumes = 1
    str StopVolumes = 1
    static int NumThreads = 1
    boolean MonitorReservation = 0
    temp str tempUseFence = "INVALID"
    boolean PanicSystemOnDGLoss = 1
)
```

Setting the noautoimport flag for a disk group

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

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

- `# vxprint -l disk_group | grep noautoimport`
  If the output from this command is blank, the noautoimport flag is set to false and VCS lacks the necessary control.

For VxVM version 5.0 on AIX

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

The following command changes the autoimport flag to false:

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

For VxVM version 4.0

Be aware that when you enable a disk group configured as a DiskGroup resource that does not have the noautoimport flag set to true, VCS forcibly deports the disk group. This may disrupt applications running on the disk group.

To explicitly set the noautoimport flag to true, deport the disk group and import it with the -t option as follows:

To deport the disk group, enter:

```
# vxdg deport disk_group
```

To import the disk group, specifying the noautoimport flag be set to true to ensure the disk group is not automatically imported, enter:

```
# vxdg -t import disk_group
```

Configuring the Fiber Channel adapter

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

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
)
```

Example of a disk group resource in the Volume Serving mode.

```
SANVolume vNFS_SANVolume (
    Domain = testdom1
    SANDiskGroup = vsdg
    SANVolume = vsvol
    VolumeServer = "sysA.veritas.com"
)
```

Volume agent

Brings online, takes offline, and monitors a Veritas Volume Manager (VxVM) volume.

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

Dependency

Volume resources depend on DiskGroup resources.

Agent functions

- Online
  Starts the volume using the `vxrecover` command.
- Offline
  Stops the volume using the `vxvol` 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 2-3 Required attributes

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

Table 2-4 Optional attributes

NumThreads	Number of threads used within the agent process for managing resources. This number does not include threads used for other internal purposes. Symantec strongly recommends that you retain the default value of the NumThreads attribute of 1. Setting this attribute to a higher value may result in agent function timeouts due to serialization of underlying commands. Default: 1

Resource type definition

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

Sample configurations

Configuration

```
Volume v0 (
    Volume = vol0
    DiskGroup = testdg_1
)
```

LVMVG agent

Activates, deactivates, and monitors a Logical Volume Manager (LVM) volume group. The LVMVG agent supports JFS or JFS2. It does not support VxFS. The LVMVG agent ensures the ODM is in sync with any changes to the volume group since it was last imported on the system. Refer to “LVMVG agent notes” on page 30 for important information on this agent.

Note: The LVMVG agent is not supported in a VIO server environment.

Agent functions

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

State definitions

- ONLINE
  Indicates that the volume group is activated.
- OFFLINE
  Indicates that the volume group is deactivated.

Attributes

Table 2-5 Required attributes

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

Table 2-6 Optional attributes

Optional attribute	Description
GroupName	Attribute used to specify the volume's group. If set, the groups's name is applied to the volume group and all of its logical volumes. Type and dimension: string-scalar Default: system

Table 2-6 Optional attributes

Optional attribute	Description
ImportvgOpt	Attribute used to specify options for the importvg command. The default option, "n", indicates the volume group is not automatically activated when imported. Type and dimension: string-scalar Default: n
Mode	Attribute used to specify permissions for a volume group and its logical volumes. If set, these permissions are applied to the volume group and all of its logical volumes. Type and dimension: string-scalar Default: 640
OwnerName	Attribute used to specify the volume owner's name. If set, the owner's name is applied to the volume group and all of its logical volumes. Type and dimension: string-scalar Default: root
SyncODM	Integer that specifies whether or not the agent ensures the ODM is in sync with any changes to the volume group. If set to 1, the agent ensures the ODM is in sync with the changes to the volume group (if the volume group was modified on another system in the cluster). The sync operation occurs on the system where the agent brings the volume group online. If set to 0, the changes to the volume group are independent of the ODM. **Note:** The SyncODM attribute is not supported for scalable volume groups. Type and dimension: integer-scalar Default: 1

Table 2-6 Optional attributes

Optional attribute	Description
VaryonvgOpt	Attribute used to specify options for the varyonvg command. Type and dimension: string-scalar

Resource type definition

```
type LVMVG (
    static int NumThreads = 1
    static str ArgList[] = { VolumeGroup, MajorNumber, OwnerName,
    GroupName, Mode, ImportvgOpt, VaryonvgOpt, SyncODM }
    str VolumeGroup
    int MajorNumber
    str OwnerName
    str GroupName
    str Mode
    str ImportvgOpt = n
    str VaryonvgOpt
    int SyncODM = 1
)
```

LVMVG agent notes

The LVMVG agent for AIX has the following notes:

- "Deactivation failure using the varyoffvg command on losing storage connectivity" on page 31
- "LVMVG Agent Supports JFS or JFS2" on page 31
- "Volume group needs to be imported" on page 31
- "Varyonvg options" on page 32
- "SyncODM Attribute" on page 32
- "Major Numbers" on page 32
- "Autoactivate Options" on page 33
- "LVMVG agent support for the Subsystem Device Driver (SDD)" on page 34
- "The hadevice utility" on page 34

Deactivation failure using the varyoffvg command on losing storage connectivity

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

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

When the volume group loses its storage connectivity, the clean script executes the varyoffvg command. Deactivation using the varyoffvg command can fail, however, if the volume group is busy. For example, when the volume group has pending I/O operations or when an application or an upper-level resources in the resource dependency tree uses the volume group.

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

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

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

LVMVG Agent Supports JFS or JFS2

The LVMVG Agents supports these file systems: JFS or JFS2. It does not support VxFS.

Volume group needs to be imported

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

For example, in the "LVMVG agent notes" on page 30, the volume groups (vg1 and vg2) must be imported on the specified systems (sysA and sysB).

Varyonvg options

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

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

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

SyncODM Attribute

The LVMVG agent ensures the ODM is in sync with any changes to the volume group since it was last imported on the system. This sync happens only if this attribute is set to 1. The agent maintains a timestamp file, /var/VRTSvcs/log/tmp/volume_group_name.ts, which records the time when the volume group was last imported on the system. When the agent initially brings a volume group online, the agent exports and reimports the group while initializing the timestamp file for that group. During the export and re-import processes, the agent preserves the ownership and mode information for the volume group and all its logical volumes.

The sync operation occurs when the timestamp value in the volume group's timestamp file is older than the timestamp value in the volume group's descriptor area. The timestamp value in the VGDA area of a volume group is updated after creating or deleting logical volumes, and adding or removing physical volumes.

Major Numbers

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

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

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

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

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

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

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

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

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

Autoactivate Options

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

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

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

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

For example:

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

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

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

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

   ```
   # varyonvg vg00
   ```

2. Run the `chvg` command:

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

3. Verify the changes:

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

LVMVG agent support for the Subsystem Device Driver (SDD)

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

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

The hadevice utility

The LVMVG agent provides the hadevice utility to check the status of a disk device, reset a disk device to an available state, and break any SCSI reservations on a disk device. Its syntax is:

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

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

To check the state of a disk device, enter:

```
# hadevice -c device_name
```

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

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

```
# hadevice -b device_name
```

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

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

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

```
# hadevice -r device_name
```

Removing a ghost disk from VxVM control

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

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

In this example, hdisk4 is a ghost disk.

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

sysA# vxdisk rm HDS9500-ALUA0_0
```

Sample configuration

```
system sysA

system sysB

system sysC

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

LVMVG lvmvg_vg1 (
    VolumeGroup = vg1
    MajorNumber = 50
    )

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

Mount agent

Use this agent to bring online, take offline, and monitor a file system or NFS client mount point.

Virtual fire drill

The virtual 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 virtual fire drill checks for:

- The existence of the mount directory
- The correct filesystem mounted at the specified mount directory

For more information about using the virtual fire drill see the *VCS User's Guide*.

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 `fsck` 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 BlockDevice attribute.
- Offline
  Unmounts the mounted file system gracefully.
- Monitor
  Determines if the file system is mounted.
- Clean
  Unmounts the mounted file system forcefully.
- Info
  The Mount info agent function executes the command:
  ```
  df -k mount_point
  ```
  The output displays Mount resource information:
  ```
  Size Used Avail Use%
  ```
  To initiate the info agent function, set the InfoInterval timing to a value greater than 0. In this example, the info agent function executes every 60 seconds:
  ```
  haconf -makerw
  hatype -modify Mount InfoInterval 60
  ```

The command to retrieve information about the Mount resource is:

```
hares -value mountres ResourceInfo
```

Output includes:

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

State definitions

- ONLINE

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

  For the NFS client, indicates that the NFS remote client is mounted in the specified mount directory.

- OFFLINE

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

  For the NFS client, indicates that the NFS remote client is not mounted in the specified mount directory.

- FAULTED

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

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

- UNKNOWN

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

Attributes

Table 2-7 Required attributes

Required attribute	Description
BlockDevice	Block device for mount point. Type and dimension: string-scalar Example: "/dev/vx/dsk/myngc_dg/myvol"
FsckOpt	Mandatory for non-NFS mounts. See “FsckOpt” on page 39
FSType	Type of file system. Supports jfs, jfs2, nfs, or vxfs. Type and dimension: string-scalar Example: "vxfs"
MountPoint	Directory for mount point Type and dimension: string-scalar Example: "/tmp/mnt"
SecondLevelMonitor	This attribute is only applicable to NFS client mounts. It executes the `df -k` command for the NFS mounted file system and detects network outage. If set to 1, this attribute enables detailed monitoring of a NFS mounted file system. Type and dimension: boolean-scalar Default: 0

Table 2-7 Required attributes

Required attribute	Description
SecondLevelTimeout	This attribute is only applicable for a NFS client mount. This is the timeout (in seconds) for the SecondLevelMonitor attribute that you try to request. The actual timeout value can be much smaller. This setting depends on how much time remains before exceeding the MonitorTimeout interval. Type and dimension: integer-scalar Default: 30

Table 2-8 Optional attributes

Optional attribute	Description
FsckOpt	Options for `fsck` command. For non-NFS mounts, the FsckOpt attribute is mandatory. If the mount process fails, `fsck` is executed with the specified options before attempting to remount the block device. Its value must include either -y or -n. Refer to the fsck manual page for more information. Type and dimension: string-scalar
MountOpt	Options for the `mount` command. Refer to the `mount` manual page for more information. Do not specify `-o` in the MountOpt field. The agent uses this option only when bringing a Mount resource online. Type and dimension: string-scalar Example: "rw"

Table 2-8 Optional attributes

Optional attribute	Description
SnapUmount	If set to 1, this attribute automatically unmounts VxFS snapshots when the file system is unmounted. If set to 0 and snapshots are mounted, then failover does not occur. Type and dimension: integer-scalar Default: 0
CkptUmount	If set to 1, this attribute automatically unmounts VxFS checkpoints when file system is unmounted. If set to 0 and checkpoints are mounted, then failover does not occur. Type and dimension: integer-scalar Default: 1

Resource type definition

```
type Mount (
    static keylist SupportedActions = { "mountpoint.vfd",
    "mounted.vfd", "vxfslic.vfd" }
    static str ArgList[] = { MountPoint, BlockDevice, FSType,
    MountOpt, FsckOpt, SnapUmount, CkptUmount, SecondLevelMonitor,
    SecondLevelTimeout }
    str MountPoint
    str BlockDevice
    str FSType
    str MountOpt
    str FsckOpt
    int SnapUmount = 0
    int CkptUmount = 1
    boolean SecondLevelMonitor = 0
    int SecondLevelTimeout = 30
)
```

Mount agent notes

The Mount agent for AIX has the following notes:

- “Listing file systems in /etc/filesystems on AIX 5.1c” on page 41
- “Example 1” on page 41

- "Example 2" on page 42
- "Example 3" on page 42

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

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

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

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

See the *VCS User's Guide* for more information about the OfflineTimeout attribute.

Listing file systems in /etc/filesystems on AIX 5.1c

The Mount agent uses the `fsck` command to repair a corrupted file system. In a cluster running AIX 5.1c, the `fsck` command requires the /etc/filesystems file on each system to contain entries for all file systems referenced by the BlockDevice attribute of the Mount agent. The `fsck-V vfstype filesystemname` command also does not work on AIX 5.1c systems without a corresponding entry for the file system in /etc/filesystems.

The `crfs` command automatically adds an entry for a new file system to /etc/filesystems on the system it was created on. You must add entries to /etc/filesystems on all other systems in the cluster. The `mkfs` command does not add an entry for a new file system to /etc/filesystems. You must add entries to /etc/filesystems on all systems in the cluster.

Example 1

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

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

Example 2

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

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

Example 3

Use the `crfs` and `mkfs` commands to create file systems. VCS supports the following configurations for the Mount agent:

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

Sample configurations

Configuration 1

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

```
LVMVG lvmvg_vg00 (
    VolumeGroup = vg00
    Disks = { "hdisk3" }
    Options = "u"
)

Mount mnt (
    MountPoint = "/lvm_testmnt"
    BlockDevice = "/dev/lv00"
    FSType = jfs
)
mnt requires vg00
```

Configuration 2

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

```
DiskGroup  dg1 (
    DiskGroup = testdg_1
)

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

m0 requires dg1
```

Configuration 3

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

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

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

---

ChapterX 3

Network agents

This chapter contains the following:

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
- Support EtherChannel

IPMultiNIC and MultiNICA agents

The IPMultiNIC and MultiNICA agents:

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

IPMultiNICB and MultiNICB agents

The IPMultiNICB and MultiNICB agents:

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

802.1Q trunking

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

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

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

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

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.

EtherChannel support

EtherChannel aggregates multiple network interfaces so that they appear as a single interface. For example you can combine en0 and en1 into an EtherChannel and call the combined interface en2. You then use the NIC agent to monitor this en2 interface and use the IP agent to configure and monitor an IP address on the en2 interface. Note that you use the en2 interface configured through EtherChannel for the Device attribute.

The IP and NIC bundled agents support EtherChannel use with VCS. EtherChannel is responsible for providing local adapter swapping, which is outside of VCS control. Only EtherChannel Backup mode is supported at this time.

Virtual fire drill

The virtual 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 IP resources, the virtual fire drill checks for the existence of a route to the IP from the specified NIC.

For more information about using the virtual fire drill see the *VCS User's Guide*.

Dependency

IP resources depend on NIC resources.

Agent functions

- Online
  For AIX, uses the `ifconfig` command to set the IP address as an alias on the interface.
- Offline
  Brings down the IP address 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 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	A virtual IP address that is different from the base IP address, and that is associated with the interface. Type and dimension: string-scalar Example: "192.203.47.61"
Device	The name of the NIC device associated with the IP address. Requires the device name without an alias. Type and dimension: string-scalar Example: "en0"
NetMask	The subnet mask associated with the IP address. Example: "255.255.255.0"

Table 3-2 Optional attributes

Optional attribute	Description
Options	Options for the `ifconfig` command. Type and dimension: string-scalar Example: "metric 4 mtu 1400"

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

NetMask in decimal (base 10)

```
IP          IP_192_203_47_61 (
    Device = en0
    Address = "192.203.47.61"
    NetMask = "255.255.248.0"
    )
```

NetMask in hexadecimal (base 16)

```
IP          IP_192_203_47_61 (
    Device = en0
    Address = "192.203.47.61"
    NetMask = "0xfffff800"
    )
```

NIC agent

Monitors the configured NIC. If a network link fails, or if a problem arises with the NIC, the resource is marked FAULTED.

Virtual fire drill

The virtual 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 NIC resources, the virtual fire drill checks for the existence of the NIC on the host.

For more information about using the virtual fire drill see the *VCS User's Guide*.

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

Before using this agent:

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

EtherChannel support

EtherChannel aggregates multiple network interfaces so that they appear as a single interface. For example you can combine en0 and en1 into an EtherChannel and call the combined interface en2. You then use the NIC agent to monitor this en2 interface and use the IP agent to configure and monitor an IP address on the en2 interface. Note that you use the en2 interface configured through EtherChannel for the Device attribute.

The IP and NIC agents support EtherChannel use with VCS. EtherChannel is responsible for providing local adapter swapping, which is outside of VCS control. Only EtherChannel Backup mode is supported at this time.

Agent functions

- Monitor

  Tests the network card and network link. Pings the network hosts or broadcast address of the interface to generate traffic on the network. Counts the number of packets passing through the device before and after

the address is pinged. If the count decreases or remains the same, the resource is marked FAULTED.
Systems do not respond to broadcast pings by default. Run the `no -o bcastping=1` command to enable response to broadcast pings.

State definitions

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

Attributes

Table 3-3 Required attributes

Required attribute	Description
Device	Name of the NIC that you want to monitor. Use the `lsdev` command to check for all available network adapters. Type and dimension: string-scalar Example: "en0"

Table 3-4 Optional attributes

Optional attribute	Description
NetworkHosts	List of hosts on the network that are pinged to determine if the network connection is alive. Enter the IP address of the host, instead of the host name, to prevent the monitor from timing out. DNS causes the ping to hang. If more than one network host is listed, the monitor returns ONLINE if at least one of the hosts is alive. If you do not specify network hosts, the monitor tests the NIC by sending pings to the broadcast address on the NIC. Type and dimension: string-vector Example: { "166.96.15.22", "166.97.1.2" }
NetworkType	Type of network Type and Dimension: string-scalar Example: "ether"
PingOptimize	Determines whether to ping every monitor cycle. A value of 1 means that the agent pings either the network host or the broadcast address every monitor cycle to determine the state of the network interface. A value of 0 means that the agent uses the device statistics from the netstat output to determine the state of the interface. If no activity exists on the interface, the agent then pings the broadcast address to double-check the state of the network interface. Type and dimension: integer-scalar Default: 1

Resource type definition

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

```
    str NetworkType
    int PingOptimize = 1
    str NetworkHosts[]
)
```

Sample configurations

Configuration without network hosts (using default ping mechanism)

```
NIC groupx_en0 (
    Device = en0
    PingOptimize = 1
    )
```

Configuration with network hosts

```
NIC groupx_en0 (
    Device = en0
    NetworkHosts = { "10.182.1.1", "10.182.1.2" }
    )
```

IPMultiNIC agent

Manages the virtual IP address 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.

Dependency

IPMultiNIC resources depend on MultiNICA resources.

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.
- 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 may be due to an incorrect configuration.

Attributes

Table 3-5 Required attributes

Required attribute	Description
Address	Virtual IP address that is assigned to the active NIC. Type and dimension: string-scalar Example: "10.128.10.14"
MultiNICResName	Name of associated MultiNICA resource that determines the active NIC. Type and dimension: string-scalar Example: "MultiNICA_grp1"
NetMask	Netmask for the virtual IP address. Type and dimension: string-scalar Example: "255.255.240.0"

Table 3-6 Optional attributes

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

Resource type definition

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

Sample configuration: IPMultiNIC and MultiNICA

```
group grp1 (
    SystemList = { sysa, sysb }
    AutoStartList = { sysa }
    )
    MultiNICA mnic (
        Device@sysa = { en0 = "10.128.8.42", en1 = "10.128.8.42" }
        Device@sysb = { en0 = "10.128.8.43", en1 = "10.128.8.43" }
        NetMask = "255.255.255.0"
        Gateway = "10.128.1.1"
        BroadcastAddr = "10.128.8.255"
        )

    IPMultiNIC ip1 (
        Address = "10.128.10.14"
        NetMask = "255.255.255.0"
        MultiNICAResName = mnic
        )

ip1 requires mnic

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

    IPMultiNIC ip2 (
        Address = "10.128.9.4"
        NetMask = "255.255.255.0"
        MultiNICAResName = mnic
        Options = "mtu m"
        )

Proxy proxy (
        TargetResName = mnic
        )

ip2 requires proxy
```

MultiNICA agent

Represents a set of network interfaces and provides failover capabilities between them. Each interface in a MultiNICA resource has a base IP address. You can use one base IP address for all NICs, or you can specify a different IP address for use with each NIC. The MultiNICA agent configures one interface at a time. If it does not detect activity on the configured interface, it configures a new interface and migrates IP aliases to it.

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

Agent function

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

Note: For AIX, systems do not respond to broadcast pings by default. You must run `"no -o bcastping=1"` in order to enable response to broadcast pings.

State definitions

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

Attributes

Table 3-7 Required attributes

Required attribute	Description
BroadcastAddr	Broadcast address Type and dimension: string-scalar Example: "10.192.15.255"
Device	List of interfaces and their base IP addresses. Type and dimension: string-association Example: en0 = { "10.128.8.42", en1 = "10.128.8.42" }
Gateway	IP address for the default gateway. Type and dimension: string-scalar Example: "10.192.1.7"
NetMask	Netmask for the base IP address. Type and dimension: string-scalar

Table 3-8 Optional attributes

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

Table 3-8 Optional attributes

Optional attribute	Description
PingOptimize	Determines whether to ping every monitor cycle. A value of 1 means that the agent pings either the network host or the broadcast address every monitor cycle to determine the state of the network interface. A value of 0 means that the agent uses the device statistics from the netstat output to determine the state of the interface. If no activity exists on the interface, the agent then pings the broadcast address to double-check the state of the network interface. Type and dimension: integer-scalar Default: 1
RouteOptions	String to add a route when configuring an interface. Use only when configuring the local host as the default gateway. The string contains the destination gateway metric. No routes are added if the value of this string is null. Type and dimension: string-scalar Example: "-net 192.100.201.0 192.100.13.7"
FailoverInProgress	For internal use only.

Resource type definition

```
type MultiNICA (
    static int OfflineMonitorInterval = 60
    static int MonitorTimeout = 300
    static str ArgList[] = { Device, NetMask, Gateway,
    BroadcastAddr, Options, RouteOptions, PingOptimize,
    MonitorOnly, HandshakeInterval, NetworkHosts }
    static str Operations = None
    str Device{}
    str NetMask
    str Gateway
    str BroadcastAddr
    str Options
    str RouteOptions
    int PingOptimize = 1
    int HandshakeInterval = 1
```

```
    str NetworkHosts[]
    temp boolean FailoverInProgress = 0
)
```

MultiNICA notes

- If all NICs configured in the Device attribute are down, the MultiNICA agent faults the resource after a two to three minute interval. This delay occurs because the MultiNICA agent tests the failed NIC several times before marking the resource OFFLINE. Messages recorded in the log during failover provide a detailed description of the events that take place.
- The MultiNICA agent supports only one active NIC on one IP subnet; the agent does not work with multiple active NICs on the same subnet.
  On AIX for example, you have two active NICs, en0 (10.128.2.5) and en1 (10.128.2.8), and you configure a third NIC, en2, as the backup NIC to en1. The agent does not fail over from en1 to en2 because some ping tests are redirected through en0 on the same subnet, making the MultiNICA monitor return an online status.

Sample configurations

MultiNICA and IPMultiNIC

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

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

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

If you have more than one group using the MultiNICA resource, the second group can use a Proxy resource to point to the MultiNICA resource in the first group. This 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 56.

```
group grp1 (
```

```
    SystemList = { sysa, sysb }
    AutoStartList = { sysa }
    )
    MultiNICA mnic (
        Device@sysa = { en0 = "10.128.8.42", en1 = "10.128.8.42" }
        Device@sysb = { en0 = "10.128.8.43", en1 = "10.128.8.43" }
        NetMask = "255.255.255.0"
        Gateway = "10.128.1.1"
        BroadcastAddr = "10.128.25.255"
        Options = "mtu m"
        )

    IPMultiNIC ip1 (
        Address = "10.128.10.14"
        NetMask = "255.255.255.0"
        MultiNICAResName = mnic
        Options = "mtu m"
        )

ip1 requires mnic

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

        IPMultiNIC ip2 (
            Address = "10.128.9.4"
            NetMask = "255.255.255.0"
            MultiNICAResName = mnic
            Options = "mtu m"
            )
        Proxy proxy (
            TargetResName = mnic
            )

ip2 requires proxy
```

About the IPMultiNICB and MultiNICB agents

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

Checklist to ensure the proper operation of MultiNICB

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

- Each interface must have a unique MAC address.
- At boot time, you must configure and connect all the interfaces that are under the MultiNICB resource and give them test IP addresses.
- All test IP addresses for the MultiNICB resource must belong to the same subnet as the virtual IP address.
- If you specify the NetworkHosts attribute, then that host must be on the same subnet as the other IP addresses for the MultiNICB resource.
- If any network host is meant to respond to a broadcast ping, run `no -o bcastping = 1` on the network host.
- You must use the AIX SMIT configuration tool to configure the test IP addresses and to make them persistent across reboots. If you do not use SMIT is to configure the IP addresses the agent may failover incorrectly.
- Ensure that media speed settings are the same for both the interface and the corresponding switch port. Symantec recommends setting the media speed to 100 Mbps full duplex.

IPMultiNICB agent

Works with the MultiNICB agent, Configures and manages virtual IP addresses (IP aliases) on an active network device specified by the MultiNICB resource. When the MultiNICB agent reports a particular interface as failed, the IPMultiNICB agent moves the IP address to the next active interface.

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

Dependencies

IPMultiNICB resources depend on MultiNICB resources.

Requirements for IPMultiNICB

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

- The MultiNICB agent must be running to inform the IPMultiNICB agent of the available interfaces.
- Only one IPMultiNICB agent can control each logical IP address.

Minimal configuration

The minimal configuration for this agent consists of the failover IP address, the subnet mask, and the name of the MultiNICB resource that it depends on.

See “Sample configurations” on page 69.

The haipswitch utility

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

Agent functions

- Online
  Finds a working interface with the appropriate interface alias or interface name, and configures the logical IP address on it.
- Offline
  Removes the logical IP address.

- Clean
  Removes the logical IP address.
- Monitor
  If the logical IP address is not configured as an alias on one of the working interfaces under a corresponding MultiNICB resource, monitor returns OFFLINE. If the current interface fails, the agent fails over the logical IP address to the next available working interface within the MultiNICB resource on the same node. If no working interfaces are available then monitor returns OFFLINE.
- Open
  Data structures necessary for monitoring the network interfaces are created.
- Close
  Data structures used by the monitor agent function are freed.
- Attr_Changed
  Updates the data structures used for monitoring the NICs.

State definitions

- ONLINE
  Indicates that the IP address specified in the Address attribute is up on one of the working network interfaces of the resource specified in the MultiNICBResName attribute.
- OFFLINE
  Indicates that the IP address specified in the Address attribute is not up on any of the working network interfaces of the resource specified in the MultiNICBResName attribute.
- UNKNOWN
  Indicates that the agent cannot determine the status of the virtual IP address that is specified in the Address attribute.
- FAULTED
  The IP address could not be brought online, usually due to all NICs in the MultiNICB resource faulting.

Attributes

Table 3-9 Required attributes

Required attribute	Description
Address	Defines the dotted decimal failover IP address. This IP address must be different than the base or test IP addresses in the MultiNICB resource. The IPMultiNICB agent automatically assigns the failover IP address. Do not configure this IP address before the IPMultiNICB agent goes online. If the IP address is already configured, the agent returns an error. Type and dimension: string-scalar Example: "10.118.10.15"
MultiNICBResName	Contains the name of the MultiNICB resource that the IPMultiNICB resource depends on. Type and dimension: string-scalar Example: "MultiNICA_grp1"
NetMask	Netmask associated with the logical IP address. Type and dimension: string-scalar "255.255.255.0"

Resource type definition

```
type IPMultiNICB (
    static int MonitorTimeout = 120
    static int OfflineMonitorInterval = 60
    static int MonitorInterval = 10
    static str ArgList[] = { Address, NetMask, MultiNICBResName,
    "MultiNICBResName:Probed"}
    str Address
    str NetMask
    str MultiNICBResName
)
```

Sample configurations

IPMultiNICB and MultiNICB

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

    MultiNICB MNICB_grp1 (
        Device@sysa = { en0 = "10.128.8.42", en1 = "10.128.8.43" }
        Device@sysb = { en0 = "10.128.8.44", en1 = "10.128.8.45" }
        NetworkHosts = "10.128.8.10"
        )

    IPMultiNICB ip1 (
        Address = "10.128.10.14"
        Netmask = "255.255.255.0"
        MultiNICBResName = MNICB_grp1
        )
    ip1 requires MNICB_grp1

group grp2 (
    SystemList = { sysa, sysb }
    AutoStartList = { sysa }
    )
    IPMultiNICB ip2 (
        Address = "10.128.10.15"
        Netmask = "255.255.255.0"
        MultiNICBResName = MNICB_grp1
        )
    Proxy MNICB_proxy (
        TargetResName = MNICB_grp1
        )
    ip2 requires MNICB_proxy
```

Other sample configurations for IPMultiNICB and MultiNICB

See “IPMultiNICB and MultiNICB configuration” on page 75.

MultiNICB agent

Works with the IPMultiNICB agent. Allows IP addresses to fail over to multiple NICs on the same system before VCS attempts to fail over to another system.

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

AIX haping utility

The administrator can use the haping utility (/opt/VRTSvcs/bin/ MultiNICB/ haping) to test each NIC prior to configuring the MultiNICB resource. This utility takes the NIC interface as an argument, and can be used to perform a link test, broadcast ping, or ping a specific remote host. It is highly recommended that the administrator perform a test ping with the remote host prior to adding it to the NetworkHosts parameter. Some examples of the command syntax are as follows:

Link test only on interface en0:

```
haping -l en0
```

Ping a remote host 10.10.10.10 from interface en0:

```
haping -g 10.10.10.10 en0
```

Agent functions

- Open
  Allocates an internal structure to store information about the resource.
- Close
  Frees the internal structure used to store information about the resource.
- Monitor
  Checks the status of each physical interface. Writes the status information to the export information file for IPMultiNICB resources to read it.

State definitions

- ONLINE
  Indicates that one or more of the network interfaces listed in the Device attribute of the resource is in working condition.
- UNKNOWN
  Indicates that the MultiNICB resource is not configured correctly.

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

Attributes

Table 3-10 Required attributes

Required attribute	Description
Device	Lists the interfaces that you want the agent to monitor. A unique test IP address must be assigned to each interface. You must use the AIX SMIT configuration tool to configure the test IP addresses and to make them persistent across reboots. **Note:** You also must manually configure the default IP route on each NIC in the MultiNICB resource. Type and dimension: string-association Example: en1= { "10.182.9.34", "en2=10.182.10.34" }
Gateway	IP address for the default gateway on the local network. Type and dimension: string-scalar Example: "136.22.1.1"

Table 3-11 Optional attributes

Optional attribute	Description
LinkTestRatio	Controls the frequency of the ping test in relation to the link test. The ping test may be run at a lesser frequency to reduce network traffic. If this attribute is set to 1, packets are sent during every monitor cycle. If this attribute is set to 0, packets are never sent during a monitor cycle. Symantec does not recommend setting the value to zero. The agent determines link status without transmitting any ping packets. For other values greater than 1, packets are sent at a lower frequency. For example, if LinkTestRatio=2, then ping packets are sent out during every other monitor cycle. In other words, packets are sent out half as often than if LinkTestRatio were equal to one. Type and dimension: integer-scalar Default: 1
NetworkHosts	The NetworkHosts attribute is a list of hosts on the local network that are pinged to determine if the network connection is available. These must be IP addresses, and not host names. If you do not specify this attribute, the agent monitors the NIC by pinging the broadcast address on the NIC. If you specify one or more network hosts, and at least one host responds to a ping, the agent reports the MultiNICB resource online. The IP addresses for the NetworkHosts attribute must be on the same subnet as the other IP addresses for the MultiNICB resource. Type and dimension: string-vector Default: 0.0.0.0 Example: "10.128.8.10, 10.128.8.45"

Table 3-11 Optional attributes

Optional attribute	Description
NoBroadcast	If set to 1, NoBroadcast prevents the agent from sending broadcast pings. ARP requests may still be generated. **Note:** If no NetworkHosts are specified and NoBroadcast is set to 1, the agent cannot function properly. Symantec does not recommend setting NoBroadcast to 1. Type and dimension: integer-scalar Default: 0
OfflineTestRepeatCount	Number of times the test is repeated if the interface status changes from up to down. For every repetition of the test, the next NetworkHosts attribute is selected in round-robin manner. At the end of this process, broadcast is performed if NoBroadcast is set to 0. A greater value prevents spurious changes, but increases the response time. Type and dimension: integer-scalar Default: 3
OnlineTestRepeatCount	Number of times the test is repeated if the interface changes from down to up. This helps prevent oscillations in the status of the interface. Type and dimension: integer-scalar Default: 3
NetworkTimeout	Timeout for ARP and ICMP packets in milliseconds. MultiNICB waits for the response to ICMP and ARP packets only during this time period. Assign NetworkTimeout a value in the order of tens of milliseconds, given the ICMP and ARP destinations must be on the local network. Increasing this value increases the time for failover. Type and dimension: integer-scalar Default: 100

Resource type definition

```
type MultiNICB (
    static int OfflineMonitorInterval = 60
    static int MonitorInterval = 10
    static str ArgList[] = { Device, NetworkHosts, Gateway,
    LinkTestRatio, NoBroadcast, NetworkTimeout,
    OnlineTestRepeatCount, OfflineTestRepeatCount }
    static str Operations = None
    str Device{}
    str NetworkHosts[] = { "0.0.0.0" }
    str Gateway
    int LinkTestRatio = 1
    int NoBroadcast
    int NetworkTimeout = 100
    int OnlineTestRepeatCount = 3
    int OfflineTestRepeatCount = 3
)
```

Trigger script

MultiNICB monitor agent function calls a VCS trigger in case of an interface going up or down. The agent passes the following arguments to the script:

- MultiNICB resource name
- The device whose status changed, for example:
  - AIX: en0
- The device's previous status (0 for down, 1 for up)
- The device's current status and monitor heartbeat

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

The sample script does the following:

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

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

Manually reattach the MultiNICB agent interfaces after failover

In order to prevent AIX from attempting to send packets through a broken interface (cable unplugged), the MultiNICB agent detaches the interface when it detects that the interface is down. When the interface is brought up (cable plugged in), you need to re-attach the interface to enable the agent to monitor it.

To attach the interface, type:

```
# chdev -l interface -a state='up'
```

For example, if the fixed interface is `en4`, enter:

```
# chdev -l en4 -a state='up'
```

Standby interfaces are the ones where the virtual IP is not configured. If a standby interface fails, the current agent does not detach it. Hence, if there are two interfaces within the MultiNICB resource, this can result in a 50% packet loss when the virtual IP address is pinged from within the subnet. No packet loss occurs if the virtual IP address is pinged from outside the subnet.

A workaround for this issue is to manually detach the broken interface and re attach it again after it is fixed.

The command to detach an interface is:

```
# chdev -l interface -a state='detach'
```

Sample configurations

IPMultiNICB and MultiNICB configuration

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

    MultiNICB MNICB_grp1 (
        Device@sysa = { en0 = "10.128.8.42", en1 = "10.128.8.43" }
        Device@sysb = { en0 = "10.128.8.44", en1 = "10.128.8.45" }
        NetworkHosts = "10.128.8.10 10.128.8.45"
        LinkTestRatio = 1
        )

    IPMultiNICB ip1 (
        Address = "10.128.10.14"
        Netmask = "255.255.255.0"
        MultiNICBResName = MNICB_grp1
        )
    ip1 requires MNICB_grp1

group grp2 (
    SystemList = { sysa, sysb }
```

```
AutoStartList = { sysa }
)

IPMultiNICB ip2 (
    Address = "10.128.10.15"
    Netmask = "255.255.255.0"
    MultiNICBResName = MNICB_grp1
    )
Proxy MNICB_proxy (
    TargetResName = MNICB_grp1
    )
ip2 requires MNICB_proxy
```

DNS agent

The DNS agent updates and monitors the canonical name (CNAME) mapping in the domain name server when failing over applications across subnets (performing a wide-area failover.)

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.

Agent functions

- Online
  Queries the authoritative name server of the domain for CNAME records and updates the CNAME record on the name server with the specified alias to canonical name mapping. Adds a new CNAME record if a related record is not found. Creates an Online lock file if the Online function was successful.
- Offline
  Removes the Online lock file, which the Online agent function created.
- Monitor
  If the Online lock file exists, the Monitor function queries the name servers for the CNAME record for the alias. It reports back ONLINE if the response from at least one of the name servers contains the same canonical name associated with the alias in the Hostname attribute. If no servers return the appropriate name, the monitor reports the resource as OFFLINE.
- Clean
  Removes the Online lock file, if it exists.
- Open
  Removes the Online lock file if the Online lock file exists, and the CNAME record on the name server does not contain the expected alias or canonical name mapping.

State definitions

- ONLINE
  An Online lock exists and the CNAME RR is as expected.
- OFFLINE
  Either the Online lock does not exist, or the expected record is not found.
- UNKNOWN
  Problem exists with the configuration.

Attributes

Table 3-12 Required attributes

Required attribute	Description
Alias	A string representing the alias to the canonical name. Type and dimension: string-scalar Example: "www" Where www is the alias to the canonical name mtv.veritas.com.
Domain	A string representing the domain name. Type and dimension: string-scalar Example: "veritas.com"
Hostname	A string representing canonical name of a system. Type and dimension: string-scalar Example: "mtv.veritas.com"
TTL	A non-zero integer representing the “Time To Live” value, in seconds, for the DNS entries in the zone you are updating. A lower value means more hits on your DNS server, while a higher value means more time for your clients to learn about changes. Type and dimension: integer-scalar Default: 86400 Example: "3600"

Table 3-13 Optional attributes

Optional attribute	Description
StealthMasters	The list of primary master name servers in the domain. This attribute is optional if the zone's name server record lists the primary master name server. 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 is not listed in that zone's name server records. Type and dimension: string-keylist Example: { "10.190.112.23" }

Resource type definition

```
type DNS (
    static str ArgList[] = { Domain, Alias, Hostname, TTL,
    StealthMasters }
    str Domain
    str Alias
    str Hostname
    int TTL = 86400
    str StealthMasters[]
)
```

Online query

If the canonical name in the response CNAME record does not match the one specified for the resource, the Online function tries to update the CNAME record on all authoritative master name servers in its domain (those master name servers that it can reach and where it has update permission). If the DNS update was successful, or was not necessary on at least one of the name servers, the Online function creates an online lock file. The monitor agent function checks for the existence of this file. The Online agent function does not create the online lock file if it is unable to update at least one domain name server.

A stealth server is a name server that is authoritative for a zone, but is not listed in that zone's NS records. If you specify the StealthMasters attribute, the Online agent function tries to update the name servers specified in the StealthMasters attribute.

In BIND 8 and above, the primary master name server on receiving an update sends notification (NOTIFY) to all its slave servers asking them to pick up the update.

Monitor scenarios

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

Table 3-14 Monitor scenarios for the Online lock file

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

Note: The DNS agent supports BIND version 8 and above.

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 the web server for www.veritas.com is failed over from Mountain View to Heathrow, the domain name servers need to be updated with the new canonical name mapping so that the web browsers are directed to Heathrow instead of Mountain View. In this case, 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.

Sample DNS configuration

```
DNS www (
        Domain = "example.com"
        Alias = www
        Hostname = virtual1
        )
```

Bringing the www resource online updates the authoritative nameservers for domain example.com with the following CNAME record:

All DNS lookups for www.example.com resolve to www.virtual1.example.com.

Chapter 4

File share agents

This chapter contains the following:

About the file service agents

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

NFS agent

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

The srcmstr daemon is the System Resource Controller (SRC). This agent sends requests to the SRC to start and monitor these daemons. You need to start the srcmstr daemon before using this agent.

Note: NFSv4root and NFSSecurity require AIX 5.3 ML03.

Agent functions

- Online
  - Checks if nfsd and mountd are running. If they are not running, the agent starts the daemons and exits.
  - The nfsrgyd daemon is started if NFSv4Root is specified.
  - The gssd daemon is started if NFSSecurity is set to 1.
- Offline
  Not applicable.
- Monitor
  - Monitors nfsd and mountd by checking whether or not the daemons are active.
  - The nfsrgyd daemon is monitored if NFSv4Root is specified.
  - The gssd daemon monitored if NFSSecurity is set to 1.
- Clean
  Terminates the resource and takes it offline—forcibly if necessary.

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 attribute	Description
Nservers	Specifies the number of concurrent NFS requests the server can handle. Type and dimension: integer-scalar Default: 10
NFSv4Root	Root directory of the NFSv4 pseudo file system to be exported. All exports should have a path relative to the path specified by this attribute. You can explicitly create the NFSv4 pseudo file system by specifying the exname option of the `exportfs` command in the Options attribute of the Share resource. If you want to export file systems with NFSv4 protocols and do not want to explicitly create NFSv4 pseudo file system by using the exname option, then set NFSv4Root to "/". Required for filesystems to be exported with v4 protocol. Type and dimension: string-scalar
NFSSecurity	If the value of this attribute is 1,the gssd daemon starts. You must configure the type of security that NFS supports, for example: Kerberos. Type and dimension: boolean-scalar Default: 0
GracePeriod	Specifies the grace period for which the server allows lock recovery. Required for NFS lock recovery. Type and dimension: integer-scalar

Table 4-1 Optional attributes

Optional attribute	Description
LockFileTimeout	Specify the amount of time required, in seconds, for the service group to go online. The agent uses this attribute to synchronize the starting and stopping of daemons between multiple service groups. Type and dimension: integer-scalar Default: 180 Example: "240""

Notes on using NFSv4

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

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

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

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

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

Caveats

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

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

Resource type definition

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

Sample configurations

Configuration 1

```
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 = en0
                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 = vxfs
                MountOpt = rw
                FsckOpt = "-y"
                )
```

```
        NFS test_nfs (
                Nservers = 20
        )

        NFSRestart test_nfsrestart (
        )

        Share test_share (
                PathName = "/test_mnt"
                Options = "-o rw,root=vcsaix3"
                )

        Volume test_vol (
                Volume = test_vol
                DiskGroup = test_dg
                )


        test_nfsrestart requires test_ip
        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
```

Configuration 2 This is a sample VCS configuration for using the NFSv4 feature by explicitly creating the NFSv4 pseudo file system. Here, NFSv4Root is set to "/export", which is different than the root of the local file system. If you want to use NFSv4 features and the NFSv4Root attribute is not equal to "/", you must configure the Share resources with exname in the Options attribute relative to the NFSv4Root. In this example it is: exname="/export/export1".

```
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 = en0
        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 = vxfs
        MountOpt = rw
        FsckOpt = "-y"
        )

NFS test_nfs (
        Nservers = 20
        NFSv4Root = "/export"
)

NFSRestart test_nfsrestart (
)

Share test_share (
        PathName = "/test_mnt"
        Options = "-o rw,vers=4,root=vcsaix3,exname=/ \
        export/export1"
        )

Volume test_vol (
        Volume = test_vol
        DiskGroup = test_dg
        )


test_nfsrestart requires test_ip
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
```

NFSRestart agent

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

The NFSRestart agent brings online, takes offline, and monitors the three daemons: smsyncd, statd, and lockd. If you have configured the NFSRestart agent for lock recovery, the NFSRestart agent starts the smsyncd daemon. The daemon copies the NFS information on the location of connections from the shared-storage to the local directory (/var/statmon/sm) and vice-versa.

Note: NFSv4root and NFSSecurity require AIX 5.3 ML03.

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.

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/statmon/sm directory.
  - Copies the locks from the shared storage to the /var/statmon/sm directory if NFSLockFailover is set to 1.
  - Starts the statd and lockd daemons.
  - Starts the smsyncd daemon to copy the contents of the /var/statmon/sm directory to the shared storage (LocksPathName) at regular, two-second intervals if the value of the NFSLockFailover attribute is 1.
- Monitor
  Monitors the statd and lockd daemons and restarts them if they are not running. It also monitors the smsyncd daemon if the value of the NFSLockFailover attribute is 1.

- Offline
  - 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
  - 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.
- nfs_postoffline
  - Restarts nfsd, mountd, lockd and statd after the group goes offline.
  - The nfsrgyd daemon is restarted if NFSv4Root is set.
  - The gssd daemon is restarted if NFSSecurity is set to 1.

State definitions

- ONLINE
  Indicates that the daemons are running properly.
- OFFLINE
  Indicates that one or more daemons are not running.
- UNKNOWN
  Indicates the inability to determine the agent's status.

Attributes

Table 4-2 Optional attributes

Optional attribute	Description
LocksPathName	The path name of the directory to store the NFS lock information. This attribute is required when the value of the NFSLockFailover attribute is 1. The path that you specify for the LocksPathName attribute should be on shared storage. This is to ensure that it is accessible to all the systems where the NFSRestart resource fails over. Type and dimension: string-scalar

Table 4-2 Optional attributes

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

NFSRestart notes

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

For example:

```
hosts = local,bind,nis
```

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

Resource type definition

```
type NFSRestart (
    static str ArgList[] = { NFSLockFailover, LocksPathName,
    "NFSRes:GracePeriod", "NFSRes:LockFileTimeout"  }
    str LocksPathName
    str NFSRes
    boolean NFSLockFailover = 0
)
```

Sample configurations

```
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 = en0
        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 = vxfs
    MountOpt = rw
    FsckOpt = "-y %-o full"
    )

Mount test_lockinfo_mnt (
    MountPoint = "/lockinfo"
    BlockDevice = "/dev/vx/dsk/test_dg/test_lockinfo_vol"
    FSType = vxfs
    MountOpt = rw
    FsckOpt = "-y"
    )

NFS test_nfs (
    Nservers = 20
)

NFSRestart test_nfsrestart (
    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_nfsrestart requires test_ip
test_nfsrestart 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
```

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.

Dependencies

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

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.

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
PathName	Pathname of the file system to be shared. Type and dimension: string-scalar Example: "/share1x"

Table 4-4 Optional attributes

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

Resource type definition

```
type Share (
    static str ArgList[] = { PathName, Options }
    str PathName
    str Options
)
```

Sample configurations

Configuration

The options for the Share agent must be specific. In this configuration, root access is only given to the system sysA. By default, systems do not have root access to the exported directory /mnt1. Refer to the exportfs manual page for more information.

```
Share share1 (
    PathName = "/mnt1"
```

```
Options = "rw=sysA,access=sysA,root=sysA"
)
```

Chapter 5

Service and application agents

This chapter contains the following agents:

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

Brings an Apache Server online and offline, and monitors the processes. The Apache Web server agent consists of resource type declarations and agent scripts.

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.

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

Before you use this agent:

- Install the Apache server on shared disk.
- Verify that the floating IP has the same subnet as that of 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 4.1.04.0 (VRTSacclib) 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 prior 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.

Dependency

This type of resource depends on IP and Mount resources.

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

  To stop the Apache HTTP server, the agent:

  - 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).

  When you specify a file with the EnvFile attribute, the file is sourced before the agent executes the httpd command.

- Monitor

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

- Clean

  Removes 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.

State definitions

- ONLINE

  Indicates that the Apache server is running.

- OFFLINE

  Indicates that the Apache server is not running.

- UNKNOWN

  Indicates that a problem exists with the configuration.

Attributes

Table 5-1 Required attributes

Required attribute	Description
ConfigFile	Full path and file name of the main configuration file for the Apache server. Type and dimension: string-scalar Example: "/apache/server1/conf/httpd.conf"
httpdDir	Full path of the directory to the httpd binary file Type and dimension: string-scalar Example: "/apache/server1/bin"
HostName	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. Specify this attribute only if the SecondLevelMonitor is set to 1 (true). Type and dimension: string-scalar Example: "web1.veritas.com"
Port	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 SecondLevelMonitor is set to 1 (true). Type and dimension: integer-scalar Default: 80 Example: "80"

Table 5-1 Required attributes

Required attribute	Description
ResLogLevel	Controls the agent's logging detail for a specific instance of a resource. Values are: ■ 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. Type and dimension: string-scalar Default: INFO Example: "TRACE"
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
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 }

Table 5-2 Optional attributes

Optional attribute	Description
EnableSSL	Set to 1 (true) to have the online agent function add support for SSL by including the option `-DSSL` in the start command. For example: `/usr/sbin/httpd -k start -DSSL` Set to 0 (false) it excludes the `-DSSL` option from the command. Type and dimension: boolean-scalar Default: 0 Example: "1"
EnvFile	Full path and file name of the file that is sourced prior to executing httpdDir/httpd. With Apache 2.0, the file *ServerRoot*/bin/envvars, which is supplied in most Apache 2.0 distributions, is commonly used to set the environment prior to executing httpd. Specifying this attribute is optional. If EnvFile is specified, the login shell for user root must be Bourne, Korn, or C shell. Type and dimension: string-scalar Example: "/apache/server1/bin/envvars"
SecondLevelMonitor	Enables second-level monitoring for the resource. Second-level monitoring is a deeper, more thorough state check of the Apache HTTP server performed by issuing an HTTP GET request on the web server's root directory. Valid attribute values are **1** (true) and **0** (false). Specifying this attribute is required. Type and dimension: boolean-scalar Default: 0 Example: "1"
SharedObjDir	Full path of the directory in which the Apache HTTP shared object files are located. Specifying this attribute is optional. It is used when the HTTP Server is compiled using the SHARED_CORE rule. If specified, the directory is passed to the -R option when executing the httpd program. Refer to the httpd man pages for more information about the -R option. Type and dimension: boolean-scalar Example: "/apache/server1/libexec"

Table 5-2 Optional attributes

Optional attribute	Description
SecondLevelTimeout	Number of seconds monitor entry point will wait on the execution of second-level monitor. If the second-level monitor program does not return to the calling monitor entry point before the SecondLevelTimeout window expires, the monitor entry point will no longer block on the program sub-process but will report that the resource is offline. The value should be sufficiently high to allow second level monitor enough time to complete, but the value should also be less than the value specified by the agent's MonitorTimeout. Type and dimension: integer-scalar Default: 30

Resource type definition

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

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 by searching for the existence of the parent httpd daemon and 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 agent attributes Host and Port. Once 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 the ACC Library

The agent functions for the Apache HTTP server depend on a set of Perl modules known as the ACC Library. The ACC Library contains 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.

Sample configurations

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

        Apache httpd_server (
                Critical = 0
                httpdDir = "/apache/bin"
                HostName = vcsaix1
                Port = 8888
                User = root
                SecondLevelMonitor = 1
                ConfigFile = "/apache/conf/httpd.conf"
                )

        DiskGroup Apache_dg (
                Critical = 0
                DiskGroup = apc1
                )

        IP Apache_ip (
                Critical = 0
                Device = en0
                Address = "11.123.99.168"
                NetMask = "255.255.254.0"
                )

        Mount Apache_mnt (
                Critical = 0
```

```
        MountPoint = "/apache"
        BlockDevice = "/dev/vx/dsk/apc1/apcvol1"
        FSType = vxfs
        FsckOpt = "-y"
        )

Apache_mnt requires Apache_dg
httpd_server requires Apache_mnt
httpd_server requires Apache_ip
```

Application agent

Brings applications online, takes them offline, and monitors their status. Enables you to specify different executables for the online, offline, and monitor routines, because most applications have executables to start and stop the application. The executables must exist locally on each node.

An application runs in the default context of root. Specify the user name to run an application in a user context.

The agent starts and stops the application with user-specified programs.

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

Virtual fire drill

The virtual 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 Application resources, the virtual 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 virtual fire drill see the *VCS User's Guide*.

Dependencies

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

Agent functions

- Online
  Runs the StartProgram with the specified parameters in the context of the specified user.

- Offline
  Runs the StopProgram with the specified parameters in the context of the specified user.
- Monitor
  If you specify the MonitorProgram, the agent executes the user-defined MonitorProgram in the user-specified context. If you specify PidFiles, the routine verifies that the process ID found in each listed file is running. If you specify MonitorProcesses, the routine verifies that each listed process is running in the context you specify.
  Use any one, two, or three of these attributes to monitor the application.
  If any one process 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.
- Clean
  Terminates processes specified in PidFiles or MonitorProcesses. Ensures that only those processes (specified in MonitorProcesses) 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 specified in PidFiles and MonitorProcesses are running and that the MonitorProgram returns ONLINE.
- OFFLINE
  Indicates that at least one process specified in PidFiles or MonitorProcesses is not running, or that the MonitorProgram returns OFFLINE.
- UNKNOWN
  Indicates an indeterminable application state or invalid configuration.

Attributes

Table 5-3 Required attributes

Required attribute	Description
StartProgram	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 are separated by spaces. Type and dimension: string-scalar Example: "/usr/sbin/samba start"
StopProgram	The executable, created locally on each node, that stops the application. Specify the complete path of the executable. Applicable command line arguments follow the name of the executable and are separated by spaces. Type and dimension: string-scalar Example: "/usr/sbin/samba stop"
At least one of the following attributes: ■ MonitorProcesses ■ MonitorProgram ■ PidFiles	See "Optional attributes" on page 110.

Table 5-4 Optional attributes

Optional attribute	Description
CleanProgram	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 are separated by spaces. Type and dimension: string-scalar

Table 5-4 Optional attributes

Optional attribute	Description
MonitorProcesses	A list of processes that you want monitored and cleaned. Each process name is the name of an executable. Qualify the executable name with its complete path if the path starts the executable. The process name must be the full command line argument displayed by the `ps -u user -eo pid,comm | more` command for the process. Type and dimension: string-vector
MonitorProgram	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 are separated by spaces. 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. Type and dimension: string-scalar Example: "/usr/sbin/sample_app_monitor all"
PidFiles	A list of PID files that contain the process ID (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. The process ID can change when the process restarts. If the application takes time to update the PID file, the agent's monitor script may return an incorrect result. If this occurs, increase the ToleranceLimit in the resource definition. Type and dimension: string-vector

Table 5-4 Optional attributes

Optional attribute	Description
User	The user ID for running StartProgram, StopProgram, MonitorProgram, and CleanProgram. The processes 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. Type and dimension: string-scalar Default: root

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 specified by the pid smbd.pid, 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

Starts, stops, and monitors a user-specified process.

Virtual fire drill

The virtual 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 virtual 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 virtual fire drill see the *VCS User's Guide*.

Dependencies

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

Agent functions

- Online
  Starts the process with optional arguments.
- Offline
  Terminates the process with a `SIGTERM`. If the process does not exit, a `SIGKILL` is sent.
- Monitor
  Checks to see if the process is running by scanning the process table for the name of the executable pathname and argument list.
- Clean
  Terminates all ongoing resource actions and takes the resource offline, forcibly when necessary.

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	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. Type and dimension: string-scalar Example: "/usr/sbin/sendmail"

Table 5-6 Optional attributes

Optional attribute	Description
Arguments	Passes arguments to the process. If a script controls the process, the script is passed as an argument. Separate multiple arguments with a single space. A string cannot accommodate more than one space between arguments, nor allow for leading or trailing whitespace characters. This attribute must not exceed 80 characters. Type and dimension: string-scalar Example: "bd q1h"

Resource type definition

```
type Process (
    static keylist SupportedActions = { "program.vfd", getcksum }
```

```
    static str ArgList[] = { PathName, Arguments }
    str PathName
    str Arguments
)
```

Sample configurations

Configuration 1

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

Configuration 2

```
include "types.cf"
cluster ProcessCluster (
.
.
.
group ProcessGroup (
    SystemList = { sysa, sysb }
    AutoStartList = { sysa }
    )

    Process Process1 (
        PathName = "/usr/local/bin/myprog"
        Arguments = "arg1 arg2"
        )

    Process Process2 (
    PathName = "/bin/csh"
    Arguments = "/tmp/funscript/myscript"
    )

    // resource dependency tree
    //
    //    group ProcessGroup
    //    {
    //    Process Process1
    //    Process Process2
    //    }
```

ProcessOnOnly agent

Starts and monitors a user-specified process.

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
IgnoreArgs	A flag that indicates whether monitor ignores the argument list. ■ If the value is 0, it checks the process pathname and argument list. ■ If the value is 1, it only checks for the executable pathname and ignores the rest of the argument list. Type and dimension: boolean-scalar Default: 0
PathName	Defines complete pathname to access an executable program. This path includes the program name. If a process is controlled by a script, the PathName defines the complete path to the shell. Type and dimension: string-scalar

Table 5-8 Optional attributes

Optional attribute	Description
Arguments	Passes arguments to the process. If a process is controlled by a script, the script is passed as an argument. Multiple arguments must be separated by a single space. A string cannot accommodate more than one space between arguments, nor allow for leading or trailing whitespace characters. Arguments must not exceed 80 characters (total). Type and dimension: string-scalar

Resource type definition

```
type ProcessOnOnly (
    static str ArgList[] = { IgnoreArgs, PathName, Arguments }
    static str Operations = OnOnly
    int IgnoreArgs
    str PathName
```

```
    str Arguments
)
```

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
    //    }
```

Chapter 6

Infrastructure and support agents

This chapter contains the following agents:

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.

Note: 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.

Dependency

The NotifierMngr resource depends on the NIC resource.

Agent functions

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

State definitions

- ONLINE
  Indicates that the Notifier process is running.
- OFFLINE
  Indicates that the Notifier process is not running.
- UNKNOWN
  Indicates that the user did not specify the required attribute for the resource.

Attributes

Table 6-1 Required attributes

Required attribute	Description
SnmpConsoles	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 Information, Warning, Error, and SevereError. Specifying a given severity level for messages generates delivery of all messages of equal or higher severity. **Note:** SnmpConsoles is a required attribute if SmtpServer is not specified; otherwise, SnmpConsoles is an optional attribute. Specify both SnmpConsoles and SmtpServer if desired. Type and dimension: string-association Example: "172.29.10.89" = Error, "172.29.10.56" = Information
SmtpServer	Specifies the machine name of the SMTP server. **Note:** SmtpServer is a required attribute if SnmpConsoles is not specified; otherwise, SmtpServer is an optional attribute. You can specify both SmtpServer and SnmpConsoles if desired. Type and dimension: string-scalar Example: "smtp.your_company.com"

Table 6-2 Optional attributes

Optional attribute	Description
MessagesQueue	Size of the VCS engine's message queue. Minimum value is 30. Type and dimension: integer-scalar Default: 30

Table 6-2 Optional attributes

Optional attribute	Description
NotifierListeningPort	Any valid, unused TCP/IP port numbers. Type and dimension: integer-scalar Default: 14144
SmtpFromPath	Set to a valid email address, if you want the notifier to use a custom email address in the FROM: field. Type and dimension: string-scalar Example: "usera@example.com"
SmtpRecipients	Specifies the email address where SMTP sends information and the severity level of the messages. The severity levels of messages are Information, Warning, Error, and SevereError. Specifying a given severity level for messages indicates that all messages of equal or higher severity are received. **Note:** SmtpRecipients is a required attribute if you specify SmtpServer. Type and dimension: string-association Example: "james@veritas.com" = SevereError, "admin@veritas.com" = Warning
SmtpReturnPath	Set to a valid email address, if you want the notifier to use a custom email address in the Return-Path: <> field. If the mail server specified in SmtpServer does not support VRFY, then you need to set the SmtpVrfyOff to `1` in order for the SmtpReturnPath value to take effect. Type and dimension: string-scalar Example: "usera@example.com"

Table 6-2 Optional attributes

Optional attribute	Description
SmtpServerTimeout	This attribute represents the time in seconds notifier waits for a response from the mail server for the SMTP commands it has sent to the mail server. This value can be increased if you notice that the mail server is taking a longer duration to reply back to the SMTP commands sent by notifier. Type and dimension: integer-scalar Default: 10
SmtpServerVrfyOff	Set this value to `1` if your mail server does not support SMTP VRFY command. If you set this value to `1`, the notifier does not send a SMTP VRFY request to the mail server specified in SmtpServer attribute while sending emails. Type and dimension: boolean-scalar Default: 0
SnmpCommunity	Specifies the community ID for the SNMP manager. Type and dimension: string-scalar Default: public
SnmpdTrapPort	Port on the SNMP console machine where SNMP traps are sent. If you specify more than one SNMP console, all consoles use this value. Type and dimension: integer-scalar Default: 162
EngineListeningPort	Change this attribute if the VCS engine is listening on a port other than its default port. Type and dimension: integer-scalar Default: 14141

Resource type definition

```
type NotifierMngr (
    static int RestartLimit = 3
    static str ArgList[] = { EngineListeningPort, MessagesQueue,
    NotifierListeningPort, SnmpdTrapPort, SnmpCommunity,
    SnmpConsoles, SmtpServer, SmtpServerVrfyOff,
    SmtpServerTimeout, SmtpReturnPath, SmtpFromPath, SmtpRecipients
    }
    int EngineListeningPort = 14141
    int MessagesQueue = 30
    int NotifierListeningPort = 14144
    int SnmpdTrapPort = 162
    str SnmpCommunity = "public"
    str SnmpConsoles{}
    str SmtpServer
    boolean SmtpServerVrfyOff = 0
    int SmtpServerTimeout = 10
    str SmtpReturnPath
    str SmtpFromPath
    str SmtpRecipients{}
)
```

Sample configuration

In the following configuration, the NotifierMngr agent is configured to run with two resource groups: NicGrp and Grp1. NicGrp contains the NIC resource and a Phantom resource that enables VCS to determine the online and offline status of the group. See the Phantom agent for more information on verifying the status of groups that only contain OnOnly or Persistent resources such as the NIC resource. You must enable NicGrp to run as a parallel group on both systems.

Grp1 contains the NotifierMngr resource (ntfr) and a Proxy resource (nicproxy), configured for the NIC resource in the first group.

In this example, NotifierMngr has a dependency on the Proxy resource.

Note: Only one instance of the notifier process can run in a cluster. The process cannot run in a parallel group.

The NotifierMngr resource sets up notification for all events to the `SnmpConsole: snmpserv`. In this example, only messages of SevereError level are sent to the SmptServer (`smtp.example.com`), and the recipient (`vcsadmin@example.com`).

Configuration

```
system north

system south

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

    Phantom my_phantom (
    )

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

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

    Proxy nicproxy(
    TargetResName = "NicGrp_en0"
    )

    NotifierMngr ntfr (
        SnmpConsoles = { snmpserv = Information }
        SmtpServer = "smtp.your_company.com"
        SmtpRecipients = { "vcsadmin@your_company.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"
InstallDir	Path to the Web application installation. You must install the Web application as a .war file with the same name as the AppName parameter. Point this attribute to the directory that contains this .war file. Type and dimension: string-scalar Example: If the AppName is cmc and InstallDir is opt/VRTSweb/VERITAS, the agent constructs the path for the Web application as /opt/VRTSweb/VERITAS/cmc.war.
TimeForOnline	The time the Web application takes to start after loading it 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 is typically at least five seconds. Type and dimension: integer-scalar

Table 6-4 Optional attributes

Optional attribute	Description
NumThreads	Number of threads used within the agent process for managing resources. This number does not include threads used for other internal purposes. Symantec strongly recommends that you retain the default value of the NumThreads attribute of 1. Setting this attribute to a higher value may result in agent function timeouts due to serialization of underlying commands. Default: 1

Resource type definition

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

Sample configuration

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

Proxy agent

Mirrors the state of another resource on a local or remote system. Provides a means to specify and modify one resource and have its state reflected by its proxies.

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

Agent functions

- Monitor
  Determines status based on the target resource status.

Attributes

Table 6-5 Required attribute

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

Table 6-6 Optional attribute

Optional attribute	Description
TargetSysName	Mirrors the status of the TargetResName attribute on systems that the TargetSysName variable specifies. If this attribute is not specified, the Proxy resource assumes the system is local. Type and dimension: string-scalar Example: "sysa"

Resource type definition

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

Sample configurations

Configuration 1

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

The proxy resource mirrors the state of the resource mnic on the local system; note that target resource is in grp1, and the proxy is in grp2; a target resource and its proxy cannot be in the same group.

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

MultiNICA mnic (
        Device@sysa = { en0 = "10.128.8.42", en1 = "10.128.8.42" }
        Device@sysb = { en0 = "10.128.8.43", en1 = "10.128.8.43" }
        NetMask = "255.255.255.0"
        NameServerAddr = "10.130.8.1"
        Gateway = "10.128.1.1"
        Domain = "veritas.com"
        BroadcastAddr = "10.128.25.255"
        Options = "mtu m"
```

```
        )

        IPMultiNIC ip1 (
            Address = "166.98.14.78"
            NetMask = "255.255.255.0"
            MultiNICAResName = mnic
            Options = "mtu m"
            )
    ip1 requires mnic

group grp2 (
        SystemList = { sysa, sysb }
        AutoStartList = { sysa }
        )
        IPMultiNIC ip2 (
            Address = "166.98.14.79"
            NetMask = "255.255.255.0"
            MultiNICAResName = mnic
            Options = "mtu m"
            )
        Proxy proxy (
            TargetResName = mnic
            )
    ip2 requires proxy
```

Phantom agent

Enables VCS to determine the status of service groups that do not include OnOff resources, which are resources that VCS can start and stop. Without the "dummy" resource provided by this agent, VCS cannot assess the status of groups that only contain None (Persistent) and OnOnly resources because the state of these resources is not considered in the process of determining whether a group is online. Refer to the VCS *User's Guide* for information on categories of service groups and resources.

Agent functions

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

Attribute

Table 6-7 Attribute for

Attribute	Description
Dummy	The Dummy attribute is for internal use only.

Resource type definition

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

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 are:

- 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.
- Global groups are not supported as remote service groups.

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 140 for more information.
- Offline
  Takes the remote service group offline.
  See the "ControlMode" on page 140 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 139.
- Clean
  If the RemoteGroup resource faults, the Clean function takes the remote service group offline.
  See the "ControlMode" on page 140 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 6-8 Required attributes

Required attribute	Description
IpAddress	The IP address or DNS name of a node in the remote cluster. The IP address can be either physical or virtual. When configuring a virtual IP address of a remote cluster, do not configure the IP resource as a part of the remote service group. Type and dimension: string-scalar Example: "www.example.com" or "11.183.12.214"
Port	The port on which the remote engine listens for requests. This is an optional attribute, unless the remote cluster listens on a port other than the default value of 14141. Type and dimension: integer-scalar Default: 14141
GroupName	The name of the service group on the remote cluster that you want the RemoteGroup agent to monitor or manage. Type and dimension: string-scalar Example: "DBGrp"
VCSSysName	You must set this attribute to either the VCS system name or the ANY value. ■ ANY The RemoteGroup resource goes online if the remote service group is online on any node in the remote cluster. ■ VCSSysName Use the name of a VCS system in a remote cluster where you want the remote service group to be online when the RemoteGroup resource goes online. Use this to establish a one-to-one mapping between the nodes of the local and remote clusters. Type and dimension: string-scalar Example: "vcssys1" or "ANY"

Table 6-8 Required attributes

Required attribute	Description
ControlMode	Select only one of these values to determine the mode of operation of the RemoteGroup resource: MonitorOnly, OnlineOnly, or OnOff. ■ OnOff The RemoteGroup resource brings the remote service group online or takes it offline. When you set the VCSSysName attribute to ANY, the SysList attribute of the remote service group determines the node where the remote service group onlines. ■ MonitorOnly The RemoteGroup resource only monitors the state of the remote service group. The RemoteGroup resource cannot online or offline the remote service group. Make sure that you bring the remote service group online before you online the RemoteGroup resource. ■ OnlineOnly The RemoteGroup resource only brings the remote service group online. The RemoteGroup resource cannot take the remote service group offline. When you set the VCSSysName attribute to ANY, the SysList attribute of the remote service group determines the node where the remote service group onlines. Type and dimension: string-scalar

Table 6-8 Required attributes

Required attribute	Description
Username	This is the login user name for the remote cluster. When you set the ControlMode attribute to OnOff or OnlineOnly, the Username must have administrative privileges for the remote service group that you specify in the GroupName attribute. When you use the RemoteGroup Wizard to enter your username data, you need to enter your username and the domain name in separate fields. For a cluster that has the Symantec Product Authentication Service, you do not need to enter the domain name. For a secure remote cluster: ■ Local Unix user user@nodename—where the nodename is the name of the node that is specified in the IpAddress attribute. Do not set the DomainType attribute. ■ NIS or NIS+ user user@domainName—where domainName is the name of the NIS or NIS+ domain for the user. You must set the value of the DomainType attribute to either to nis or nisplus. Type and dimension: string-scalar Example: ■ For a cluster without the Symantec Product Authentication Service: "johnsmith" ■ For a secure remote cluster: "foobar@example.com"
Password	This is the password that corresponds to the user that you specify in the Username attribute. You must encrypt the password with the `vcsencrypt -agent` command. **Note:** Do not use the vcsencrypt utility when entering passwords from a configuration wizard or from the Cluster Management Console or the Cluster Manager (Java Console). Type and dimension: string-scalar

Table 6-9 Optional attributes

Optional attribute	Description
DomainType	For a secure remote cluster only, enter the domain type information for the specified user. For users who have the domain type unixpwd, you do not have to set this attribute. Type: string-scalar Example: "nis", "nisplus"
BrokerIp	For a secure remote cluster only, if the user needs the RemoteGroup agent to communicate to a specific authentication broker, then set this attribute. Enter the information for the specific authentication broker in the format "IP:Port". Type: string-scalar Example: "128.11.295.51:1400"
OfflineWaitTime	The maximum expected time in seconds that the remote service group may take to offline. VCS calls the Clean function for the RemoteGroup resource if the remote service group takes a longer time to offline than the time that you have specified for this attribute. Type and dimension: integer-scalar Default: 0

Table 6-10 Type-level attributes

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

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
)
```

Chapter 7

Testing agents

This chapter contains the following agents:

About the program support agents

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

ElifNone agent

Monitors a file—checks for the file's absence.

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—check's for the file's existence.

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 int AutoRestart = 1
    static int OfflineMonitorInterval = 60
    static str ArgList[] = { PathName }
    static str Operations = None
    str PathName
)
```

Sample configuration

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

FileOnOff agent

Creates, removes, and monitors files.

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

Creates and monitors files.

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.

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.

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

B

C

D

E

F

H

I

K

L

M

N

O

P

R

S

T

V