Veritas Cluster Server Agents for Veritas Volume Replicator Configuration Guide

Solaris

5.0

N18523F

symantec™

Veritas Cluster Server Agents for Veritas Volume Replicator Configuration Guide

Documentation version 5.0

PN: N18523F

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.

Legal Notice

OR USE OF THIS DOCUMENTATION. THE INFORMATION CONTAINED IN THIS DOCUMENTATION IS SUBJECT TO CHANGE WITHOUT NOTICE.

The Licensed Software and Documentation are deemed to be "commercial computer software" and "commercial computer software documentation" as defined in FAR Sections 12.212 and DFARS Section 227.7202.

Symantec Corporation 20330 Stevens Creek Blvd. Cupertino, CA 95014 USA

http://www.symantec.com

Contents

Chapter 1

Overview of the VCS Agents for VVR

This chapter includes the following topics:

Introducing the VCS agents for VVR

Agents are processes that manage predefined resource types. When an agent is started, it obtains configuration information from the Veritas Cluster Server (VCS). It then periodically monitors the resources and updates VCS with the resource status.

Typically agents do the following:

- Bring resources online
- Take resources offline
- Monitor resources and report any state changes to VCS

The VCS Agents for VVR monitor and manage Replicated Volume Groups (RVGs). Each agent includes VCS-type declarations and agent executables, which represent a resource type. The VCS Agents for VVR include:

Agents for Failover Applications

- RVG agent
- RVGPrimary agent
- RVGSnapshot agent

See "How the agents for failover applications work" on page 9.

Agents for Parallel Applications

- RVGShared agent
- RVGSharedPri agent
- RVGLogowner Agent

See "How the agents for parallel applications work" on page 18.

VCS cluster concepts

Resources, attributes, and service groups are components integral to cluster functionality.

For more information, see the *Veritas Cluster Server User's Guide*.

Resources	Resources are hardware or software entities, such as disks, volumes, file system mount points, network interface cards (NICs), IP addresses, applications, and databases. Resources work together to provide a service to clients in a client/server environment. The bundled agents resource types are defined in the types.cf file by a collection of attributes. The VCS configuration file, main.cf, contains the values for the attributes of the resources. The main.cf file incorporates the resources listed in the types.cf by way of an `include` directive. The main.cf file also incorporates the VVR resource types, which are defined in the file VVRTypes.cf by way of an `include` directive.

Attributes	Attributes contain data regarding the cluster, nodes, service groups, resources, resource types, and agents. A specified value for a given attribute configures the resource to function in a specific way. By modifying the value of an attribute of a resource, you change the way the VCS agent manages the resource. Each attribute has a definition and a value. You define an attribute by specifying its data type and dimension. Attributes also have default values that are assigned when a value is not specified.
Service groups	Service groups are comprised of related resources. When a service group is brought online, all the resources within the group are brought online.

You can dynamically configure or modify the VCS agents and their resources from the command line or from the VCS Java and Web consoles. You can also edit the main.cf file directly, however you must stop VCS before editing the main.cf file. Example main.cf files for the VCS agents for VVR are located in the /etc/VRTSvcs/conf/sample_vvr directory.

For instructions, see the chapters on administering VCS in the *Veritas Cluster Server User's Guide*.

How the agents for failover applications work

This section describes how each agent works, summarizes the entry points, state definitions, and attributes for each agent, and explains the dependency graphs for each agent.

The VCS Agents for VVR include the following:

- RVG agent
- RVGPrimary agent
- RVGSnapshot agent

RVG agent

The RVG agent enables replication between clusters by managing the Primary VVR node in one cluster and the Secondary VVR node in another cluster, each of which can be failed over in its respective cluster. In this way, replication is made highly available.

Note: The RVG works with the RVGPrimary agent to provide failover of the Primary VVR node to the Secondary VVR node. If a disaster occurs on the Primary VVR node and all the nodes in the Primary cluster are unavailable, the RVG agent does not fail over the Primary role from the Primary VVR node to the Secondary VVR node. Using a VCS global cluster enables you to fail over the Primary role from a Primary VVR node to a Secondary VVR node.

The RVG agent includes the following key features:

- Removes potential single points of failure by enabling Primary and Secondary VVR nodes to be clustered.
- Makes the process of starting VCS-managed applications that use VVR, as easy as bringing a VCS service group online.
- Continues replication after a node in a cluster fails without losing updates.
- Ensures that VVR can be added to any VCS cluster by including the RVG resource type definitions.

An example configuration file for this agent that can be used as a guide when creating your configuration is located at:

/etc/VRTSvcs/conf/sample_vvr/RVG

Note: This release does not support the attributes Primary, SRL, and RLinks of the RVG agent. If you have a configuration from a previous release, you must remove these attributes during the upgrade or the configuration will fail.

The function of the RVG agent, its entry points, and its state definitions are as follows:

Description	Brings the RVG online, monitors read/write access to the RVG, and takes the RVG offline; this is a failover resource.

Entry Points

- online: Verifies whether the DiskGroup agent has recovered the RVG. If not, recovers and starts the data volumes and the Storage Replicator Log (SRL), recovers the RVG, recovers all RLINKs in the RVG, and then starts the RVG.
- offline: Stops the RVG.
- clean: Stops the RVG.
- info: Gives the information about the replication status for the Replicated Data Set (RDS).
- monitor: Monitors the state of the RVG using the vxprint command.

Note: The RVG resource monitors an RVG for local access only; it does not monitor replication.

Detecting Failure

The RVG resource fails if the RVG is not in the ENABLED/ACTIVE state.

State Definitions

`ONLINE`—Indicates that the RVG is in `ENABLED/ACTIVE` state.

`OFFLINE`—Indicates that the RVG is in `DISABLED/CLEAN` state.

The attributes of the RVG agent are as follows:

Table 1-1 Attributes of the RVG agent

Attribute	Type and Dimension	Definition
RVG	string-scalar	The name of the RVG being monitored.
DiskGroup	string-scalar	The disk group with which this RVG is associated.
StorageDG	string-scalar	The name of the bunker disk group.
StorageRVG	string-scalar	The name of the bunker RVG.
StorageHostIds	string-keylist	A space-separated list of the hostids of each node in the bunker cluster.

Type definition for the RVG agent

```
type RVG (
  static str ArgList[] = { RVG, DiskGroup, StorageDG, StorageRVG, StorageHo
  str RVG
```

```
  str DiskGroup
  str StorageDG
  str StorageRVG
  str StorageHostIds[]
  static int NumThreads = 1
 )
```

Using the info entry point

The info entry point displays information about the replication status of an RDS. By default, the info interval is set to zero. To change the default info interval, use the following command:

```
# hatype -modify resourcetype_name InfoInterval interval
```

For example, to set the info interval to 60 seconds for the RVG resource type, enter:

```
# hatype -modify RVG InfoInterval 60
```

The info interval indicates how frequently VCS executes the info entry point to update the the replication status. In the above example, the info interval is set to 60, so VCS updates the replication status every 60 seconds. To display the output of the info entry point, use the following command:

```
# hares -value resource_name ResourceInfo
```

The output of the info entry point is also logged in the file /var/VRTSvcs/log/engine_A.log.

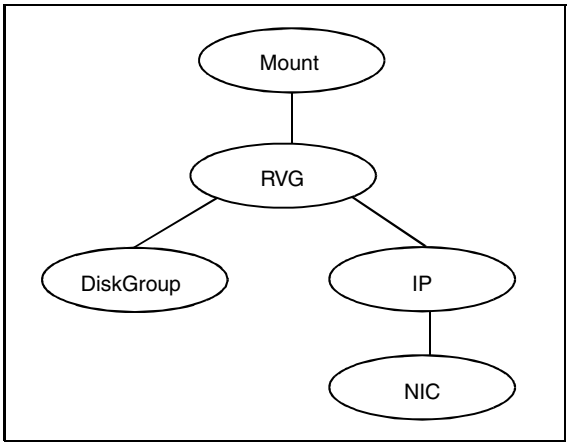
Dependency graph for the RVG agent

The RVG resource represents the RVG (Replicated Volume Group) in the RDS (Replicated Data Set). The RVG resource is dependent on the DiskGroup resource. The RVG resource is also dependent on the IP resources that it uses for replication.

In a VVR environment, higher-level application resources, such as Mount, that would typically depend on a Volume resource must depend on the associated RVG resource.

Refer to the *Veritas Cluster Server User's Guide* for more information on dependencies.

Figure 1-1 Dependency graph for the RVG agent



RVGPrimary agent

The RVGPrimary agent enables migration and takeover of a VVR replicated data set in a VCS environment. Bringing a resource of type RVGPrimary online causes the RVG on the local host to become a primary if it is not already. The agent is useful when hosts in both the primary and secondary side are clustered, in particular a VCS replicated data cluster or a VCS global cluster, to completely automate the availability of writable replicated disks to an application managed by VCS.

The RVGPrimary agent includes the following key features:

- Removes manual steps of migrating a VVR primary and secondary roles when failing over applications across a wide area.
- Minimizes the need for resynchronizing replicated volumes by attempting a migration before attempting a hard takeover.
- Waits for the two sides of a replicated data set to become completely synchronized before migrating roles.
- Supports an automatic fast failback resynchronization of a downed primary if it later returns after a takeover.

A sample configuration file for this agent that can be used as a guide when creating your configuration is located at /etc/VRTSvcs/conf/sample_vvr/RVGPrimary.

The function of the RVGPrimary agent and its entry points are as follows:

Description	Attempts to migrate or takeover a Secondary to a Primary upon an application failover.
Entry Points	Online—Determines the current role of the RVG; if Secondary, attempt a migrate, waiting for any outstanding writes from the original Primary; if the original Primary is down attempt a takeover; if the RVG is a Primary, perform no actions and go online Offline—Perform no actions. Clean—Perform no actions. Monitor—Perform no actions; monitoring of the actual RVG is done by the RVG agent.
Detecting Failure	Monitoring of the actual RVG is done by the RVG agent; accidental migration of a VVR Primary outside of VCS would cause other resources to fault immediately, such as Mount, so no special monitoring by this agent is necessary.

The attributes of the RVGPrimary agent are as follows:

Table 1-2 Attributes of the RVGPrimary agent

Attributes	Type and Dimension	Definition
RvgResourceName	string-scalar	The name of the RVG resource type that this agent will promote, that is, the name RVG resource type which has been configured using the RVG agent.
AutoTakeover	integer-scalar	A flag to indicate whether the agent should perform a takeover on online if the original Primary is down.
AutoResync	integer-scalar	A flag to indicate whether the agent should attempt to automatically perform a fast-failback resynchronization of the original Primary after a takeover and after the original Primary returns.

Type definition for the RVGPrimary agent

```
type RVGPrimary (
 static keylist SupportedActions = { fbsync }
 static int InfoTimeout = 0
 static int NumThreads = 1
```

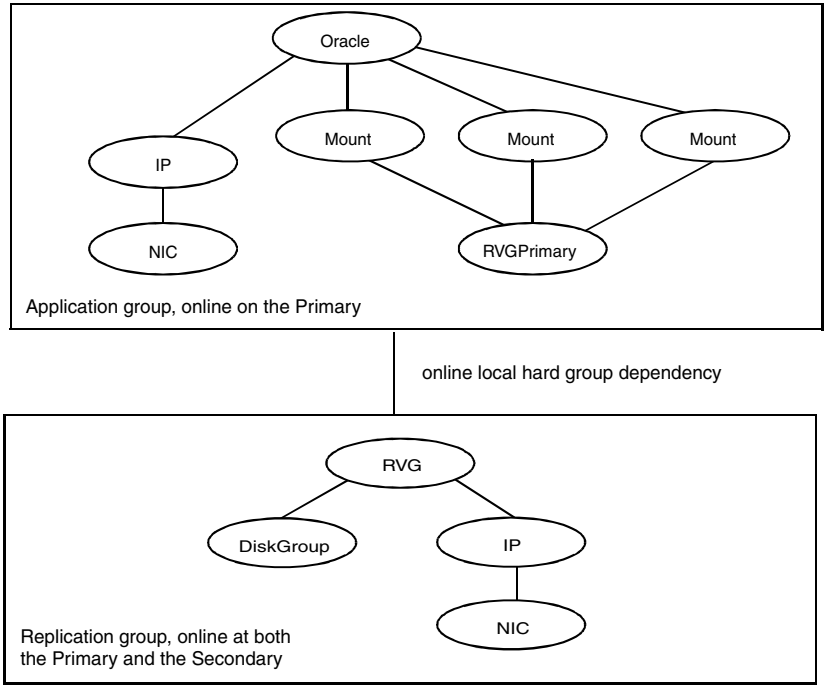
```
 static int OnlineRetryLimit = 1
 static str ArgList[] = { RvgResourceName, AutoTakeover, AutoResync }
 str RvgResourceName
 int AutoTakeover = 1
 int AutoResync = 0
)
```

Dependency graph for the RVGPrimary agent

The RVGPrimary agent is customarily used in conjunction with the RVG agent in two groups with an online local firm group dependency; the parent group contains the resources managing the actual application and file systems as well as the RVGPrimary resource, and the child group contains the resources managing the storage infrastructure, including the RVG and DiskGroup type resources.

Refer to the *Veritas Cluster Server User's Guide* for more information on detailed setup of a VVR environment using the RVGPrimary agent.

Figure 1-2 Dependency graph for the RVGPrimary agent



RVGSnapshot agent

The RVGSnapshot agent automates the taking of space-optimized snapshots on a secondary RVG; since these snapshots can be mounted and written to without affecting the actual replicated data, a space-optimized snapshot can be an effective tool for scheduling a “fire drill” to confirm that a wide-area failover is possible. By combining this agent with VCS Mount agents and VCS agents that manage the application being replicated, a special fire drill service group can be created that can be onlined and offlined at regularly scheduled intervals to confirm the robustness of a disaster recovery environment.

In addition to the agent itself, a text-based wizard /opt/VRTSvcs/bin/fdsetup that prepares the VVR and VCS infrastructure for a fire drill and a script /opt/VRTSvcs/bin/fdsched that runs the fire drill and consolidates the results are included with this package.

Complete details are in the *Veritas Cluster Server User's Guide*.

The RVGSnapshot agent includes the following key features:

- Automates the process of creating a space-optimized snapshot on a VVR secondary that can be mounted to simulate a wide-area failover without affecting the production application.
- Includes a wizard to effectively set up and schedule fire drills that are completely managed by VCS.

While the fdsetup wizard configures the appropriate resources for a fire drill group, the following table summarizes the function of the RVGSnapshot agent, its entry points, and its state definitions:

Description	Creates and destroys a transactionally consistent space-optimized snapshot of all volumes in a VVR secondary replicated data set.
Entry Points	online—Creates a transactionally consistent snapshot of all volumes in the RDS. offline—Destroys the snapshot. clean—Cleans up any failed snapshot creation or deletion. monitor—No operation; failure of the snapshot will be indicated by the failure of the Mount resource of any filesystems mounted on it.
Detecting Failure	The RVGSnapshot resource faults on timeout if a snapshot creation did not succeed during an online.
State Definitions	`ONLINE`—Indicates that a snapshot was created. `OFFLINE`—Indicates that a snapshot was destroyed.

The attributes of the RVGSnapshot agent are as follows:

Table 1-3 Attributes of the RVGSnapshot agent

Required Attributes	Type and Dimension	Definition
RvgResourceName	string-scalar	The name of the VCS RVG-type resource that manages the RVG that will be snapshot by this agent.
CacheObj	string-scalar	Name of the cache object that is required for a space-optimized snapshot; the fdsetup wizard will create one if it does not exist
Prefix	string-scalar	Token prepended to the name of the actual volume when creating the snapshotted volumes.
Optional Attributes	Type and Dimension	Definition
DestroyOnOffline	int-scalar	A flag to indicate whether to destroy the snapshot upon offlining the resources. For a fire drill, the snapshot should be deleted to reduce any performance impact of leaving the snapshot for a long period of time; however, if there is interest in keeping the data, then this value should be set to 0. The default is 1 (true).
FDFile	temporary string-scalar	The fire drill schedule updates this attribute with the system name and the path to a file containing the output of the last complete fire drill for the group containing an RVGSnapshot resource.

Type definition for the RVGSnapshot agent

```
type RVGSnapshot (
 static keylist RegList = { Prefix }
 static int InfoTimeout = 0
 static int NumThreads = 1
 static str ArgList[] = { RvgResourceName, CacheObj, Prefix, DestroyOnOffli
 str RvgResourceName
 str CacheObj
 str Prefix
```

```
 boolean DestroyOnOffline = 1
 temp str FDFile
)
```

How the agents for parallel applications work

The agents for parallel applications include the following:

- RVGShared agent
- RVGLogowner Agent
- RVGSharedPri agent

RVGShared agent

The RVGShared agent enables you to configure parallel applications to use an RVG in a cluster. The RVGShared agent monitors the RVG in a shared disk group environment. The RVGShared agent must be configured as a parallel group in VCS. Typically, the RVGShared resource is online or offline at the same time on all the nodes in the VCS cluster. An example configuration file for this agent that can be used as a guide when creating your configuration is located at /etc/VRTSvcs/conf/sample_vvr/RVGLogowner.

The function of the RVGShared agent, its entry points, and its state definitions are as follows:

Description	Monitors the RVG in a shared environment; this is a parallel resource.
Entry Points	online—Verifies whether the RVG is started. If the RVG is not started, recovers and starts the RVG. offline—No action. clean—No action. info—Gives the information about the replication status for the Replicated Data Set (RDS). See "Using the info entry point" on page 12. monitor—Displays the state as `ONLINE` if the RVG is started. Displays the state as `OFFLINE` if the RVG is not started.
State Definitions	`ONLINE`—Indicates that the RVG is in the `ENABLED/ACTIVE` state. `OFFLINE`—Indicates that the RVG is not in the `ENABLED/ACTIVE` state or that the administrator has invoked the offline entry point.

The attributes of the RVGShared agent are as follows:

Table 1-4 Attributes of the RVGShared agent

Attributes	Type and Dimension	Definition
RVG	string-scalar	The name of the RVG being monitored.
DiskGroup	string-scalar	The shared-disk group with which this RVG is associated.

Type definition for the RVGShared agent

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

Dependency graph for the RVGShared agent

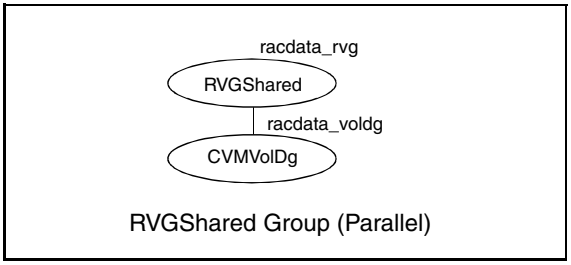
The RVGShared resource represents the RVG of the RDS. The RVGShared resource is dependent on the CVMVolDg resource.

The RVGShared resource must be configured in a parallel group.

See "Example configuration for a parallel application" on page 35.

Refer to the *Veritas Cluster Server User's Guide* for more information on dependencies.

Figure 1-3 Dependency graph for the RVGShared agent



Note: Do not add any volumes that are part of the RVG in the CVMVolume attribute of the CVMVolDg resource. The volumes in the RVG are managed by the RVGShared resource.

RVGLogowner Agent

The RVGLogowner agent assigns or unassigns a node as a logowner in the cluster. To replicate data, VVR requires network connectivity between the Primary and the Secondary. In a shared disk group environment, only one node, that is, the logowner, can replicate data to the Secondary.

For replication to be highly available, the logowner must be highly available. To make the logowner highly available, the RVGLogowner resource must be configured as a resource in a failover group. Also, a virtual IP must be set up on the logowner to enable replication and failover of the logowner from one node to another in a cluster. The virtual IP must be configured as an IP resource.

See "Dependency graph for the RVGLogowner agent" on page 21.

For more information about the logowner, see the *Veritas Volume Replicator Administrator's Guide*. An example configuration file for this agent that can be used as a guide when creating your configuration is located at /etc/VRTSvcs/conf/sample_vvr/RVGLogowner.

The function of the RVGLogowner agent, its entry points, and its state definitions are as follows:

Description	Assigns and unassigns a node as the logowner in the CVM cluster; this is a failover resource.
Operations	online—Assigns the logowner on the node. offline—Unassigns the logowner on the node. monitor—Returns `ONLINE` if the node is the logowner and the RVG is in `ENABLED/ACTIVE` state. Returns `OFFLINE` if the node is the logowner and the state is not `ENABLED/ACTIVE`, or if the node is not the logowner (regardless of the state). **Note:** The RVG for which the logowner is monitored must be configured as the RVGShared resource type. clean—Unassigns the logowner on the node.
State Definitions	`ONLINE`—Indicates that the node is the logowner for the RVG in the cluster. `OFFLINE`—Indicates that the node is not the logowner for the RVG in the cluster.

The attributes of the RVGLogowner agent are as follows:

Table 1-5 Attributes of the RVGLogowner agent

Attributes	Type and Dimension	Definition
RVG	string-scalar	The name of the RVG being monitored.
DiskGroup	string-scalar	The disk group with which this RVG is associated.
Bunker Attributes	Type and Dimension	Definition
StorageDG	string-scalar	The name of the bunker disk group.
StorageRVG	string-scalar	The name of the bunker RVG.
StorageHostIds	string-keylist	A space-separated list of the hostids of each node in the bunker cluster.

Type definition for the RVGLogowner agent

```
type RVGLogowner (
  static str ArgList[] = { RVG, DiskGroup, StorageDG, StorageRVG,
StorageHostIds}
  str RVG
  str DiskGroup
  str StorageDG
  str StorageRVG
  str StorageHostIds
  static int NumThreads = 1
)
```

Dependency graph for the RVGLogowner agent

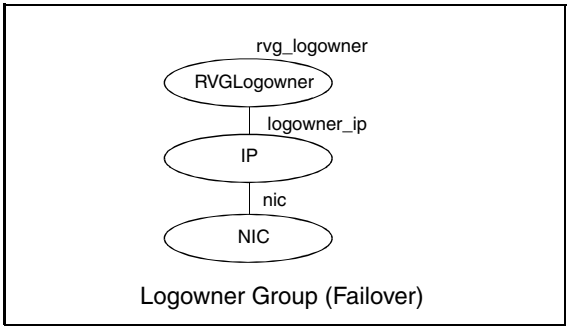
The RVGLogowner resource represents the logowner for RVG in the cluster. The RVGLogowner resource is dependent on the IP resource that it uses for replication.

The RVGLogowner resource must be configured in a failover group. The RVGLogowner group is used in conjunction with the RVGSharedPri and RVGShared agents in separate groups, with the appropriate service group dependencies.

See "Example configuration for a parallel application" on page 35.

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

Figure 1-4 Dependency graph for the RVGLogowner agent



RVGSharedPri agent

The RVGSharedPri agent enables migration and takeover of a VVR replicated data set in parallel groups in a VCS environment. Bringing a resource of type RVGSharedPri online causes the RVG on the local host to become a primary if it is not already. The agent is useful when hosts in both the primary and secondary side are clustered using a VCS global cluster, to completely automate the availability of writable replicated disks to an application managed by VCS.

The RVGSharedPri agent includes the following key features:

- Removes manual steps of migrating a VVR primary and secondary roles when failing over applications across a wide area.
- Minimizes the need for resynchronizing replicated volumes by attempting a migration before attempting a hard takeover.
- Waits for the two sides of a replicated data set to become completely synchronized before migrating roles.
- Supports an automatic fast failback resynchronization of a downed primary if it later returns after a takeover.

Sample configuration files are located in the /etc/VRTSvcs/conf/sample_rac/ directory and include CVR in the filename. These sample files are installed as part of the VRTSdbac package, and can be used as a guide when creating your configuration.

The function of the RVGSharedPri agent and its entry points are as follows:

Description	Attempts to migrate or takeover a Secondary to a Primary when a parallel service group fails over
Entry Points	Online—Determines the current role of the RVG; if Secondary, attempt a migrate, waiting for any outstanding writes from the original Primary; if the original Primary is down attempt a takeover; if the RVG is a Primary, perform no actions and go online Offline—Perform no actions. Clean—Perform no actions. Monitor—Perform no actions; monitoring of the actual RVG is done by the RVGShared agent.
Detecting Failure	Monitoring of the actual RVG is done by the RVGShared agent; accidental migration of a VVR Primary outside of VCS would cause other resources to fault immediately, such as Mount, so no special monitoring by this agent is necessary.

Table 1-6 Attributes of the RVGSharedPri agent

Attributes	Type and Dimension	Definition
RvgResourceName	string-scalar	The name of the RVGShared resource type that this agent will promote, that is, the name RVG resource type which has been configured using the RVGShared agent. The required VVR object names, such as the name of the RVG, Disk Group, RLINKs, SRL are discovered by this agent by querying VCS directly.
AutoTakeover	integer-scalar	A flag to indicate whether the agent should perform a takeover on online if the original Primary is down.
AutoResync	integer-scalar	A flag to indicate whether the agent should attempt to automatically perform a fast-failback resynchronization of the original Primary after a takeover and after the original Primary returns.
VCSResLock	string-scalar	This attribute is reserved for internal use by VCS.

Type definition for the RVGSharedPri agent

```
type RVGSharedPri (
 static keylist SupportedActions = { fbsync, resync }
 static int NumThreads = 1
 static int OnlineRetryLimit = 1
 static str ArgList[] = { RvgResourceName, "RvgResourceName:RVG",
 "RvgResourceName:DiskGroup", AutoTakeover, AutoResync }
 str RvgResourceName
 int AutoTakeover = 1
 int AutoResync = 0
 temp str VCSResLock
)
```

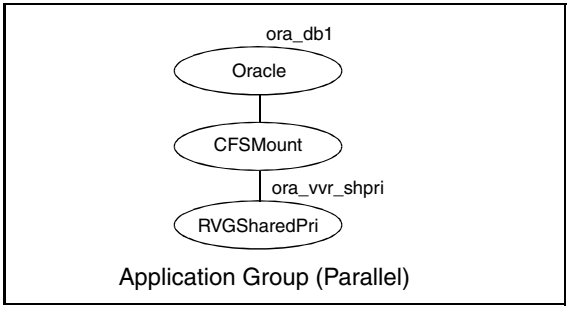
Dependency graph for the RVGSharedPri agent

The RVGSharedPri agent is used in conjunction with the RVGShared and RVGLogowner agents in separate groups.

See “Example configuration for a parallel application” on page 35.

for the service group dependencies. The RVGSharedPri agent must be configured in a parallel service group. The application service group contains the resources managing the actual application and file systems as well as the RVGSharedPri agent.

Figure 1-5 Dependency Graph for the RVGSharedPri Agent



How the agents for hybrid applications work

The agents for hybrid applications include the following:

- RVG agent
- RVGPrimary agent

A hybrid configuration is for Replicated Data Clusters (RDCs) and is a combination of the failover and parallel service groups. A hybrid service group behaves like a failover group within a system zone and like a parallel group across system zones. It cannot fail over across system zones. A switch operation on a hybrid service group is allowed only between systems within the same system zone.

For more information about the RVG agent and RVGPrimary agent, see RVG agent and RVGPrimary agent respectively. These section give information about the entry points, state definitions, and attributes for the RVG agent and the RVGPrimary agent. In addition, the following attribute must be set for the RVG agent and the RVGPrimary agent while configuring RDCs:

Table 1-7 Attribute for RDC s

Optional attributes	Type and dimension	Definition
SystemZones	integer-association	Indicates failover zone.

An RDC uses VVR as opposed to shared storage to provide access to data at the Secondary. An RDC exists within a single VCS cluster. The application group, which is configured as a failover group, can be online only on the Primary host. In the case of the failure of the Primary site, the Secondary is promoted to a Primary and the application is brought online on the new Primary host.

An RDC configuration is appropriate in configurations lacking shared storage or SAN interconnection between the Primary site and Secondary site, but where dual dedicated LLT links are available between the Primary site and the Secondary site.

For more information about RDCs, refer to the *Veritas Cluster Server User's Guide*.

Overview of how to configure VVR in a VCS environment

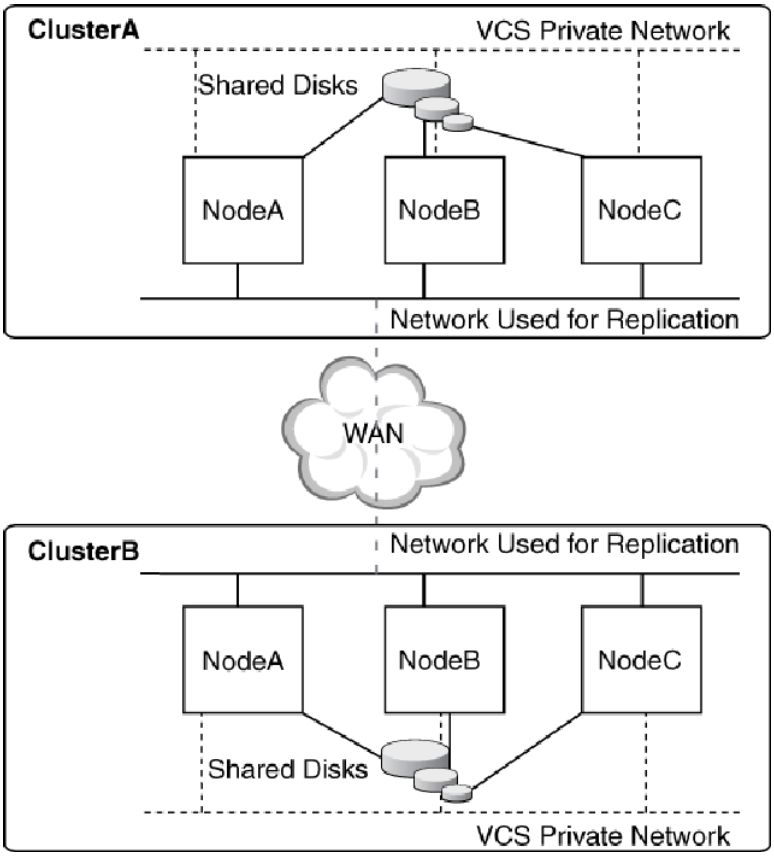
This section gives an overview of how to configure VVR in a VCS environment for high availability of the application that is involved in replication.

To configure VVR in a VCS environment, you must perform the following tasks in the order in which they are listed:

- Setting up a VVR configuration, which involves creating a Replicated Data Set (RDS).
- Creating service groups for the VVR agents and adding the resource and group dependencies appropriately.

Generic VVR setup in a VCS environment

The following illustration shows how VVR replicates in a VCS environment given a two-cluster environment.



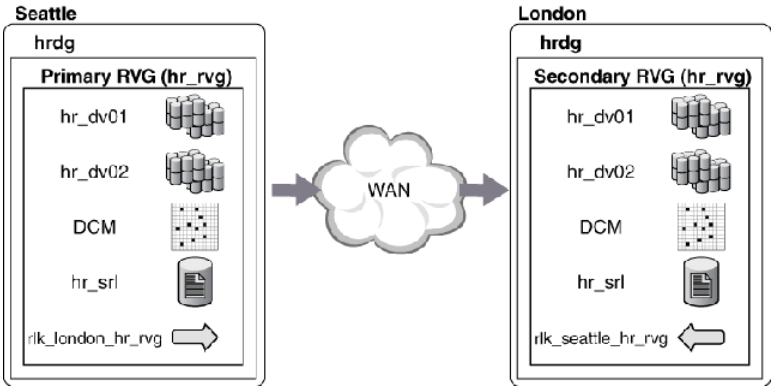
Example VVR configuration in a VCS environment

In the following example, two clusters are located at separate sites. VVR replicates data between the sites using a WAN.

The first cluster is located in Seattle and is named Seattle. The cluster Seattle consists of two nodes: seattle1 and seattle2. The second cluster is located in London and is named London.The cluster London also consists of two nodes: london1 and london2. The nodes located in the cluster Seattle contain the Primary RVG. The nodes located in the cluster London contain the Secondary RVG. Note that the

following illustration shows the names of the VVR components used by the RVG agent.

Figure 1-6 Example—VVR configuration in a VCS environment



Chapter 2

Configuring the agents for high availability

This chapter includes the following topics:

Requirements for configuring VVR in a VCS environment

The requirements for configuring VVR in a VCS environment are as follows:

- Follow the best practices for setting up replication with VVR.
  For information about setting up replication, refer to the *Veritas Volume Replicator Administrator's Guide*.
- Each node that is part of a particular VCS service group involved in replication must use the same port number for replication. You may need to change this number on some nodes before configuring VVR.
- If a node has more than one network interface card on the same physical network being used for replication, each network interface card must have a

different MAC address. This is true for all the nodes at the Primary and Secondary sites.

- This requirement is specific to the RVG Agent. VCS requires the noautoimport attribute of the disk group to be set.
  Refer to the *Veritas Cluster Server Bundled Agents Reference Guide* for more information about setting the noautoimport attribute.

Best practices for setting up the agents

The following list gives the best practices for setting up the agents:

- Only one DiskGroup and one RVG resource must be present in a service group.
- If a disk group is configured as a DiskGroup resource, then all the RVGs in this disk group must be configured as RVG resources. If a disk group is configured as a CVMVolDG resource, then all the RVGs must be configured as RVGShared resources.
- When configuring failover applications, use the RVG, RVGPrimary, and RVGSnapshot agents.
- When configuring parallel applications, use the RVGShared, RVGSharedPri, and RVGLogowner agents. If the configuration has multiple RVGLogowner resources, we recommend that you alternate the order of hosts in the AutoStartList attributes for the service groups containing the RVGLogowner resources. VCS then onlines the RVGLogowner resources on different nodes in the cluster, which facilitates load-balancing. For example, the first service group containing an RVGLogowner resource would appear as:
  `AutoStartList = { seattle1, seattle2 }`
  whereas the next service group would have:
  `AutoStartList = { seattle2, seattle1 }` and so on.
- Do not configure the RVGShared resource in the cvm group. Configure the RVGShared resource in a separate group which contains the RVGShared resource and the CVMVolDg resource.
- If a volume set is fully associated to an RVG, that is, if all its component volumes are associated to the RVG, you can add the volume set to the agent configuration in the same way that a volume is added. Specify the volume set in the Mount resource instead of the component volume names.
  See "Example—Setting up VVR in a VCS environment" on page 36.

Note: The agents do not support mounting a volume set that is partially associated to an RVG, that is, if one or more of its component volumes are not associated to the RVG.

For more information about using volume sets in an RVG, refer to the *Veritas Volume Replicator Administrator's Guide*.

Adding the VVR agents to the VCS configuration

You can add the VVR agents to the VCS configuration in the following cases:

- When VCS is running
- When VCS is stopped

To add the agents without stopping the applications on a system, perform the following steps:

To add the agents when VCS is running

1. Log in as root on one node in the cluster.
2. Set the VCS configuration mode to read/write by typing the following command on any system in the cluster:

   ```
   # haconf -makerw
   ```

3. Update the VCS configuration by running the following script:

   ```
   # /etc/VRTSvcs/conf/sample_vvr/RVG/addVVRTypes.sh
   ```

4. If version 1.0 or 1.1 of the agent is installed:
   - Unlink the dependencies for the RVolume resource by using the following command for each dependency:

     ```
     # hares -unlink parent_resource child_resource
     ```

   - Delete occurrences of the RVolume resource from the main.cf file by using the following command for each resource:

     ```
     # hares -delete resource
     ```

   - Create dependencies for the RVG resource.
     See "Dependency graph for the RVG agent" on page 12.
     Type the following command:

```
# hares -link parent_resource child_resource
```

5 Ensure that all changes to the existing configuration are saved and that further changes are prevented.

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

For a new installation of the agents, the configuration is complete. If you are upgrading the agents, continue with steps 6 and 7.

6 If you stopped the agent before installing the new agent, start the agent on the system by entering:

```
# haagent -start agent_name -sys system_name
```

When you get the message Please look for messages in the log file, check the file /var/VRTSvcs/log/engine_A.log for a message confirming that each agent has started.

You can also use the ps command to confirm that the agent is started.

7 If you brought the RVG service group offline before doing the installation, bring it online by using the following command:

```
# hagrp -online service_group -sys system_name
```

You can add the agents by editing the main.cf file. You must stop VCS before editing the main.cf file. Perform the following steps:

To add the agents when VCS is stopped

1 Log in as root on one node in the cluster.

2 Ensure that all changes to the existing configuration have been saved and that further changes are prevented while you modify main.cf located in the /etc/VRTSvcs/conf/config directory.

If the VCS cluster is currently writeable, run the following command:

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

If the VCS cluster is already read only, run the following command:

```
# haconf -dump
```

3 Do not edit the configuration files while VCS is running. The following command stops the had daemon on all systems and leaves resources available:

```
# hastop -all -force
```

4 Copy the VVRTypes.cf file from /etc/VRTSvcs/conf to the /etc/VRTSvcs/conf/config directory.

5 Add the VVRTypes to the main.cf file, located in /etc/VRTSvcs/conf/config directory.

If version 3.5 or earlier of the agent is installed, make the following change:

If the main.cf file contains the following line:

```
include "SRVMTypes.cf"
```

change the line to:

```
include "VVRTypes.cf"
```

For a new agent installation, add the following line to the main.cf file:

```
include "VVRTypes.cf"
```

6 If version 1.0 or 1.1 of the agent is installed, do the following steps:

- Remove the RVolume resources and its dependencies for the RVolume resource type.
- Create new dependencies for the RVG resource.
  See "Dependency graph for the RVG agent" on page 12.

7 This version of the agent does not support the Primary, SRL, and RLINK attributes of the RVG resource. If existing RVG resources use or define these attributes, you must remove the attributes.

8 Verify the syntax of the file /etc/VRTSvcs/conf/config/main.cf:

```
# hacf -verify /etc/VRTSvcs/conf/config
```

9 Start the VCS engine on all systems in both clusters.

See "To start VCS on all systems in both clusters" on page 34.

Starting VCS on all systems

To start VCS on all systems in both clusters

1. In the primary cluster, start the VCS engine on the system on which the main.cf was modified:

   ```
   # hastart
   ```

2. Type the command hastatus:

   ```
   # hastatus
   ```

3. When "LOCAL_BUILD" or "RUNNING" is listed in the message column, start VCS on the other system:

   ```
   # hastart
   ```

4. Verify that all service group resources are brought online. On any system, enter:

   ```
   # hagrp -display
   ```

5. On the secondary cluster, start VCS from the system on which the main.cf was modified:

   ```
   # hastart
   ```

6. Type the command hastatus:

   ```
   # hastatus
   ```

7. When "LOCAL_BUILD" or "RUNNING" is listed in the message column, start VCS on the other system:

   ```
   # hastart
   ```

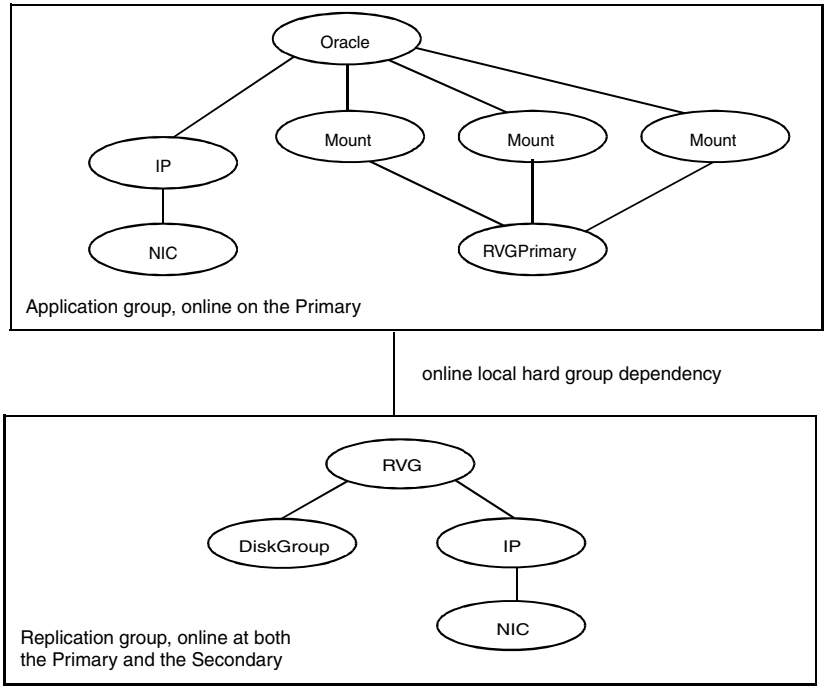
8. Verify the service groups and their resources that are brought online. On any system, enter:

   ```
   # hagrp -display
   ```

Example configuration for a failover application

In the following example, a failover application that uses an RVG is made highly available across two clusters. The application service group contains the following resources: application, Mount, NIC, IP, and RVGPrimary. The replication group contains the RVG, IP, NIC, and DiskGroup resources. The application group has an online local hard dependency on the replication group.

Figure 2-1 RVG and RVGPrimary Agents—Service Groups and Resource Dependencies

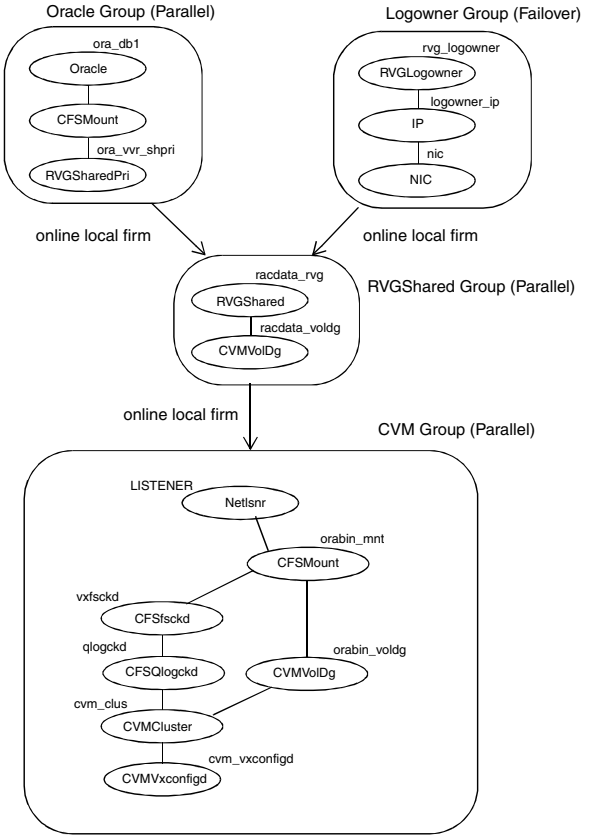


Example configuration for a parallel application

In the following example, a parallel application that uses an RVG is made highly available across two clusters. The Oracle service group is the application group and contains the CFSMount resource. The Logowner service group is a failover group, which manages the logowner. The service group RVGShared manages the RVG used by the application. The Oracle and CVM groups are configured as parallel groups.

The service groups Logowner and Oracle are dependent on the service group RVGShared. The RVGShared manages the RVG in a shared environment; therefore, it is dependent on the cvm service group.

Figure 2-2 RVGShared, RVGLogowner, and RVGSharedPri agents—Service Groups and Resource Dependencies



Example—Setting up VVR in a VCS environment

Configuring VVR with VCS requires the completion of several tasks, each of which must be performed in the following order:

- Setting up the VVR configuration
- Verifying the VVR replication state
- Configuring the agents for failover applications

- Configuring the agents for parallel applications

Before setting up the VVR configuration, verify whether all the nodes in the cluster that have VVR installed use the same port number for replication. To verify and change the port numbers, use the vrport command. If the port number is the same on all nodes, add the VVR agents to the VCS configuration.

For instructions on using the vrport command, see the *Veritas Volume Replicator Administrator's Guide*.

Setting up the VVR configuration

This section provides the steps to set up a sample VVRconfiguration. The VVR configuration that is being set up in this example applies to the RVG Agent, that is, it uses the names that are used in the sample configuration file of the RVG agent. The procedure to configure VVR is the same for all the VVR agents. Use the sample configuration files located in /etc/VRTSvcs/conf/sample_vvr directory to configure the other agents.

For more information on configuring VVR, refer to the *Veritas Volume Replicator Administrator's Guide*

The example uses the names listed in the following table.

Name of Cluster: Seattle

Disk group	hrdg
Primary RVG	hr_rvg
Primary RLINK to london1	rlk_london_hr_rvg
Primary data volume #1	hr_dv01
Primary data volume #2	hr_dv02
Primary volume set (with data volumes hr_dv03, hr_dv04)	hr_vset01
Primary SRL for hr_rvg	hr_srl
Cluster IP	10.216.144.160

Name of Cluster: London

Disk group	hrdg
Secondary RVG	hr_rvg

Secondary RLINK to seattle	rlk_seattle_hr_rvg
Secondary data volume #1	hr_dv01
Secondary data volume #2	hr_dv02
Secondary volume set (with data volumes hr_dv03, hr_dv04)	hr_vset01
Secondary SRL for hr_rvg	hr_srl
Cluster IP	10.216.144.162

This example assumes that each of the hosts seattle1 and london1 has a disk group named hrdg with enough free space to create the VVR objects mentioned in the example. Set up the VVR configuration on seattle1 and london1 to include the objects used in the sample configuration files, main.cf.seattle and main.cf.london, located in the /etc/VRTSvcs/conf/sample_vvr/RVG directory.

See "Example VVR configuration in a VCS environment" on page 26.

To set up the VVR configuration

1 On london1:

- Create the Secondary data volumes.

```
# vxassist -g hrdg make hr_dv01 100M \
    layout=mirror logtype=dcm mirror=2
# vxassist -g hrdg make hr_dv02 100M \
    layout=mirror logtype=dcm mirror=2
```

- Create the data volumes for the volume set on the Secondary and create the volume set.

```
# vxassist -g hrdg make hr_dv03 100M \
    layout=mirror logtype=dcm mirror=2
# vxassist -g hrdg make hr_dv04 100M \
    layout=mirror logtype=dcm mirror=2
# vxmake -g hrdg vset hr_vset01 \
    appvols=hr_dv03,hr_dv04
```

- Create the Secondary SRL.

```
# vxassist -g hrdg make hr_srl 200M mirror=2
```

2 On seattle1:

- Create the Primary data volumes.

```
# vxassist -g hrdg make hr_dv01 100M \
    layout=mirror logtype=dcm mirror=2
# vxassist -g hrdg make hr_dv02 100M \
    layout=mirror logtype=dcm mirror=2
```

- Create the data volumes for the volume set on the Primary and create the volume set.

```
# vxassist -g hrdg make hr_dv03 100M \
    layout=mirror logtype=dcm mirror=2
# vxassist -g hrdg make hr_dv04 100M \
    layout=mirror logtype=dcm mirror=2
# vxmake -g hrdg vset hr_vset01 \
    appvols=hr_dv03,hr_dv04
```

- Create the Primary SRL.

```
# vxassist -g hrdg make hr_srl 200M mirror=2
```

- Create the Primary RVG.

```
# vradmin -g hrdg createpri hr_rvg \
    hr_dv01,hr_dv02,hr_vset01 hr_srl
```

- Determine the virtual IP address to be used for replication, and then verify that the device interface for this IP is plumbed. If the device interface for this IP is not plumbed, then plumb the device. Get the IP up using the OS-specific command. This IP address that is to be used for replication must be configured as the IP resource for this RVG service group.
- Create the Secondary RVG.

```
# vradmin -g hrdg addsec hr_rvg 10.216.144.160 \
    10.216.144.162 prlink=rlk_london_hr_rvg \
    srlink=rlk_seattle_hr_rvg
```

Note: The RLINKs must point to the virtual IP address for failovers to succeed. The virtual IP address 10.216.144.160 must be able to ping virtual IP address 10.216.144.162 and vice versa.

- Start replication.

```
# vradmin -g hrdg -f startrep hr_rvg
```

3 Create the following directories on seattle1 and seattle2. These directories will be used as mount points for volumes hr_dv01 and hr_dv02 and the volume set hr_vset01 on the seattle site.

```
# mkdir /hr_mount01
# mkdir /hr_mount02
# mkdir /hr_mount03
```

4 On seattle1, create file systems on the volumes hr_dv01 and hr_dv02 and on the volume set hr_vset01.

Verifying the VVR replication state

Test the replication state between seattle1 and london1 to verify that VVR is configured correctly.

To verify the replication state

1 Type the following command on each node:

```
# vxprint -g hrdg hr_rvg
```

2 In the output, verify the following:

- State of the RVG is `ENABLED/ACTIVE`.
- State of the RLINK is `CONNECT/ACTIVE`.

Configuring the agents for failover applications

This section explains how to configure the VVR agents for failover applications.

See "Configuring the agents for parallel applications" on page 47.

You can configure the RVG agent and RVGPrimary agent when VCS is stopped or when VCS is running. Sample configuration files, main.cf.seattle and main.cf.london, are located in the /etc/VRTSvcs/conf/sample_vvr/RVG and /etc/VRTSvcs/conf/sample_vvr/RVGPrimary directories respectively, and can be used for reference.

You can add the RVG resource to your existing VCS configuration using any one of the following procedures:

- Configuring the agents when VCS is running
- Configuring the agents when VCS is stopped

Configuring the agents when VCS is running

The example in this section explains how to configure the RVG and RVGPrimary agents when VCS is running.

See "Example configuration for a failover application" on page 35.

Note: Use this example as a reference when creating or changing your resources and attributes.

To add the agent resources to your existing VCS configuration when VCS is running, perform the following procedures:

- Create the replication service group
- Create the application service group

Perform the following steps on the system seattle1 in the Primary cluster Seattle, and then repeat the steps (with minor changes as noted) on the system london1 in Secondary cluster London:

To create the replication service group

1. Log in as root.
2. Set the VCS configuration mode to read/write by issuing the following command:

   ```
   # haconf -makerw
   ```

3. Add the replication service group, `VVRGrp`, to the cluster. This group will contain all the storage and replication resources. Modify the attributes `SystemList` and `AutoStartList` of the service group to populate `SystemList` and `AutoStartList`:

   ```
   # hagrp -add VVRGrp
   # hagrp -modify VVRGrp SystemList seattle1 0 seattle2 1
   # hagrp -modify VVRGrp AutoStartList seattle1 seattle2
   ```

   On the Secondary cluster, replace seattle1 and seattle2 with london1 and london2

4. Add the DiskGroup resource `Hr_Dg` to the service group `VVRGrp` and modify the attributes of the resource:

   ```
   # hares -add Hr_Dg DiskGroup VVRGrp
   # hares -modify Hr_Dg DiskGroup hrdg
   ```

5 Add a NIC resource `vvrnic` to the service group `VVRGrp` and modify the attributes of the resource:

```
# hares -add vvrnic NIC VVRGrp
# hares -modify vvrnic Device qfe3
```

6 Add the IP resource `vvrip` to the service group `VVRGrp` and modify the attributes of the resource:

```
# hares -add vvrip IP VVRGrp
# hares -modify vvrip Device qfe3
# hares -modify vvrip Address 192.2.40.20
# hares -modify vvrip NetMask "255.255.248.0"
```

On the Secondary cluster, use the appropriate IP for the Address. For example:

```
# hares -modify vvrip Address 192.2.40.21
```

7 Specify resource dependencies for the resources you added in the previous steps:

```
# hares -link Hr_Rvg vvrip
# hares -link Hr_Rvg Hr_Dg
# hares -link vvrip vvrnic
```

8 Enable all resources in `VVRGrp`

```
# hagrp -enableresources VVRGrp
```

9 Save and close the VCS configuration

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

Perform the following steps on the system seattle1 in the Primary cluster Seattle, and then repeat the steps (with minor changes as noted) on the system london1 in Secondary cluster London:

To create the application service group

1 Log in as root.

2 Set the VCS configuration mode to read/write by issuing the following command:

```
# haconf -makerw
```

3 Add a service group, `ORAGrp`, to the cluster `Seattle`. This group will contain all the application specific resources. Populate the attributes `SystemList`, `AutoStartList` and `ClusterList` of the service group

```
# hagrp -add ORAGrp
# hagrp -modify ORAGrp SystemList seattle1 0 seattle2 1
# hagrp -modify ORAGrp AutoStartList seattle1 seattle2
# hagrp -modify ORAGrp ClusterList  Seattle 0 London 1
```

On the Secondary , replace seattle1 and seattle2 with london1 and london2, as follows:

```
# hagrp -add ORAGrp
# hagrp -modify ORAGrp SystemList london1 0 london2 1
# hagrp -modify ORAGrp AutoStartList london1 london2
# hagrp -modify ORAGrp ClusterList  Seattle 0 London 1
```

4 Add a NIC resource `oranic` to the service group `ORAGrp` and modify the attributes of the resource:

```
# hares -add oranic NIC ORAGrp
# hares -modify oranic Device hme0
```

5 Add an IP resource `oraip` to the service group `ORAGrp` and modify the attributes of the resource:

```
# hares -add oraip IP ORAGrp
# hares -modify oraip Device hme0
# hares -modify oraip Address 192.2.40.1
# hares -modify oraip NetMask "255.255.248.0"
```

On the Secondary, modify the Address attribute for the IP resource appropriately.

6 Add the Mount resource `Hr_Mount01` to mount the volume `hr_dv01` in the RVG resource `Hr_Rvg`:

```
# hares -add Hr_Mount01 Mount ORAGrp
# hares -modify Hr_Mount01 MountPoint /hr_mount01
# hares -modify Hr_Mount01 BlockDevice /dev/vx/dsk/Hr_Dg/hr_dv01
# hares -modify Hr_Mount01 FSType vxfs
# hares -modify Hr_Mount01 FsckOpt %-n
# hares -modify Hr_Mount01 MountOpt rw
```

7 Add the Mount resource `Hr_Mount02` to mount the volume `hr_dv02` in the RVG resource `Hr_Rvg`:

```
# hares -add Hr_Mount02 Mount ORAGrp
# hares -modify Hr_Mount02 MountPoint /hr_mount02
# hares -modify Hr_Mount02 BlockDevice /dev/vx/dsk/Hr_Dg/hr_dv02
# hares -modify Hr_Mount02 FSType vxfs
# hares -modify Hr_Mount02 FsckOpt %-n
# hares -modify Hr_Mount02 MountOpt rw
```

8 Add the Mount resource Hr_Mount03 to mount the volume set hr_vset01 in the RVG resource Hr_Rvg:

```
# hares -add Hr_Mount03 Mount ORAGrp
# hares -modify Hr_Mount03 MountPoint /hr_mount03
# hares -modify Hr_Mount03 BlockDevice /dev/vx/dsk/ Hr_Dg/hr_vset01
# hares -modify Hr_Mount03 FSType vxfs
# hares -modify Hr_Mount03 FsckOpt %-n
# hares -modify Hr_Mount03 MountOpt rw
```

9 Add the Oracle resource `Hr_Oracle`

```
# hares -add Hr_Oracle Oracle ORAGrp
# hares -modify Hr_Oracle Sid hr1
# hares -modify Hr_Oracle Owner oracle
# hares -modify Hr_Oracle Home "/hr_mount01/OraHome1"
# hares -modify Hr_Oracle Pfile "inithr1.ora"
# hares -modify Hr_Oracle User dbtest
# hares -modify Hr_Oracle Pword dbtest
# hares -modify Hr_Oracle Table oratest
# hares -modify Hr_Oracle MonScript "./bin/Oracle/SqlTest.pl"
# hares -modify Hr_Oracle StartUpOpt STARTUP
# hares -modify Hr_Oracle ShutDownOpt IMMEDIATE
# hares -modify Hr_Oracle AutoEndBkup 1
```

10 Add the Oracle listener resource `LISTENER`

```
# hares -add LISTENER Netlsnr ORAGrp
# hares -modify LISTENER Owner oracle
# hares -modify LISTENER Home "/hr_mount01/OraHome1"
# hares -modify LISTENER Listener LISTENER
# hares -modify LISTENER EnvFile "/oracle/.profile"
# hares -modify LISTENER MonScript "./bin/Netlsnr/LsnrTest.pl"
```

11 Add the RVGPrimary resource `Hr_RvgPri`

```
# hares -add Hr_RvgPri RVGPrimary ORAGrp
# hares -modify Hr_RvgPri RvgResourceName Hr_Rvg
```

12 Specify resource dependencies for the resources you added in the previous steps:

```
# hares -link LISTENER Hr_Oracle
# hares -link LISTENER oraip
# hares -link Hr_Oracle Hr_Mount01
# hares -link Hr_Oracle Hr_Mount02
# hares -link Hr_Mount01 rvg-pri
# hares -link Hr_Mount02 rvg-pri
# hares -link Hr_Mount03 rvg-pri
# hares -link oraip oranic
```

13 The application service group and the replication service group must both exist before doing this step. If you have not yet created the replication service group, do so now.

See "To create the replication service group " on page 41.

After you have created the application service group and the replication service group, specify an online local hard group dependency between `ORAGrp` and `VVRGrp`.

```
# hagrp -link ORAGrp VVRGrp online local hard
```

14 Enable all resources in `ORAGrp`

```
# hagrp -enableresources ORAGrp
```

15 Save and close the VCS configuration

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

16 Bring the service groups online, if not already online.

```
# hagrp -online VVRGrp -sys seattle1
# hagrp -online ORAGrp -sys seattle1
```

17 Verify that the service group `ORAGrp` is ONLINE on the system `seattle1` by issuing the following command:

```
# hagrp -state ORAGrp
```

Configuring the agents when VCS is stopped

Perform the following steps to configure the RVG agent using the sample configuration file on the first node in the Primary cluster and Secondary cluster. In the example in this guide, seattle1 is the first Primary node and london1 is the first Secondary node.

To configure the agents when VCS is stopped

1 Log in as root.

2 Ensure that all changes to the existing configuration have been saved and that further changes are prevented while you modify main.cf:

If the VCS cluster is currently writeable, run the following command:

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

If the VCS cluster is already read only, run the following command:

```
# haconf -dump
```

3 Do not edit the configuration files while VCS is started. The following command will stop the had daemon on all systems and leave resources available:

```
# hastop -all -force
```

4 Make a backup copy of the main.cf file:

```
# cd /etc/VRTSvcs/conf/config
# cp main.cf main.cf.orig
```

5 Edit the main.cf files for the Primary and Secondary clusters. The files main.cf.seattle and main.cf.london located in the /etc/VRTSvcs/conf/sample_vvr/RVGPrimary directory can be used for reference for the primary cluster and the secondary cluster respectively.

6 Save and close the file.

7 Verify the syntax of the file /etc/VRTSvcs/conf/config/main.cf:

```
# hacf -verify /etc/VRTSvcs/conf/config
```

8 Start VCS on all systems in both clusters.

See "To start VCS on all systems in both clusters" on page 34.

9 Administer the service groups.

See "Administering the service groups" on page 53.

Configuring the agents for parallel applications

Use the RVGShared, RVGSharedPri, and the RVGLogowner agents to manage and monitor RVGs used by parallel applications in a shared environment.

Note: Determine the node that is performing the most writes by running the vxstat command on each node for a suitable period of time; after you set up replication, specify this node as the logowner.

The prerequisites for configuring the agents are as follows:

- You must have replication set up between the Primary and Secondary sites.
  For more information about replicating in a shared environment, see the *Veritas Volume Replicator Administrator's Guide*.
- The sites must be configured in a global cluster and the application service must be configured as a global service group.
  For more information about configuring global clusters, see the *Veritas Cluster Server User's Guide*.

Sample configuration files are located in the /etc/VRTSvcs/conf/sample_rac/ directory and include CVR in the filename. These sample files are installed as part of the VRTSdbac package, and can be used as a guide when creating your configuration. You can configure agents from the command line or from the VCS Java and Web consoles.

See the *Veritas Cluster Server User's Guide* for more information.

To modify the VCS configuration on the Primary cluster

1. Define two new service groups: A logowner group that includes the RVGLogowner resource, and an RVG group that includes the RVGShared resource replication objects.
2. In the logowner group, define IP and NIC resources, used by the RLINKs for the RVG, and the RVGLogowner resource, for which the RVG and its associated disk group are defined as attributes.
3. In the RVG service group, set up the RVGShared agent to monitor the RVG resource. Because it is shared, the RVG must be configured to depend on the CVMVolDg resource, which defines the shared disk group and its activation mode.

   Define the RVGShared and CVMVolDg resources within a parallel service group so that the service group may be online at the same time on all cluster nodes.
4. Add the RVGSharedPri resource to the existing application service group and define the service group to be a global group.

   See the *Veritas Cluster Server User's Guide* for instructions on how to create global groups.
5. Move the CVMVolDg resource from the existing application service group to the newly created RVGShared service group.
6. Set the following service group dependencies:
   - The RVG logowner service group has an “online local firm” dependency on the service group containing the RVG.
   - The RVG service group has an “online local firm” dependency on the CVM service group.
   - The application service group has an “online local firm” dependency on the RVG service group.

To modify the VCS configuration on the Secondary cluster

1. Log on to a node in the secondary cluster as root.
2. Ensure that all changes to the existing configuration have been saved and that further changes are prevented while you modify main.cf:

   If the VCS cluster is currently writeable, run the following command:

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

   If the VCS cluster is already read only, run the following command:

   ```
   # haconf -dump
   ```

3. Ensure VCS is not running while you edit main.cf by using the `hastop` command to stop the VCS engine on all systems and leave the resources available:

   ```
   # hastop -all -force
   ```

4. Make a backup copy of the main.cf file:

   ```
   # cd /etc/VRTSvcs/conf/config
   # cp main.cf main.orig
   ```

5. Use vi or another text editor to edit the main.cf file, making the following changes:
   - Edit the CVM group on the secondary cluster. Use the CVM group on the primary as your guide.
   - Add the logowner group and the RVG service groups.
   - Add an application service group. Use the application service group on the primary cluster as a pattern for the service group on the secondary cluster.
   - Since the service group is a global group, assign it the same name as the group on the primary cluster.
   - Define the ClusterList and ClusterFailOverPolicy cluster attributes.
   - Include the RVGSharedPri resource.
6. Save and close the main.cf file.

7 Verify the syntax of the file /etc/VRTSvcs/conf/config/main.cf:

```
# hacf -verify /etc/VRTSvcs/conf/config
```

8 Start VCS on all systems in both clusters.

See "To start VCS on all systems in both clusters" on page 34.

The application group should be online on both systems of the primary cluster.

The application service group should not be online on the secondary cluster, but the CVM, RVG logowner, and RVG groups should be online.

Configuring the agents for a bunker replication configuration

This section describes how to set up the VCS agents for a bunker replication configuration, that is, an RDS that includes a bunker site. A bunker can be set up using the STORAGE protocol, or using IP.

Refer to one of the following sections to configure the VCS agents:

- VCS configuration for a bunker using the STORAGE protocol
- VCS configuration for a bunker using IP

VCS configuration for a bunker using the STORAGE protocol

When a bunker is set up using the STORAGE protocol, the disk group containing the bunker RVG is imported on the Primary node. If the Primary RVG is in a VCS cluster, the bunker RVG must remain online on the same node on which the Primary RVG is online.

In a shared disk group environment, the bunker RVG must be online on the logowner node.

This section describes how to configure the agents to automate the failover of the bunker RVG.

In a private disk group environment, the RVG resource handles the failover process. If the node on which the RVG resource is online fails, the RVG resource fails over to another node within the cluster. The RVG resource ensures that the bunker RVG also fails over, so that the bunker RVG continues to be on the same node with the Primary RVG.

In a shared disk group environment, the RVGLogowner agent handles the failover of the bunker RVG. If the logowner fails over, the bunker RVG must be deported from the original logowner node and imported on the new logowner node.

To set up automated failover of the bunker RVG, specify the bunker RVG, the bunker disk group, and the bunker node using the following attributes of the RVG resource in the application service group or the RVGLogowner agent:

Table 2-1 Attributes for configuring bunker failover

Attribute	Description
StorageDG	The name of the bunker disk group.
StorageRVG	The name of the bunker RVG.
StorageHostIds	Hostid of the bunker node or, if the bunker is clustered, a space-separated list of the hostids of each node in the bunker cluster.

The bunker failover attributes described in this section are the only specific attributes that differ for an RDS containing a bunker. The rest of the configuration for the VCSAgent is the same as for any other RDS.

See "Example—Setting up VVR in a VCS environment" on page 36.

Sample configuration files for VCS agents in a bunker replication environment

The following examples show sample configuration files when the bunker Secondary is connected to the Primary using the STORAGE protocol.

This example uses the following names:

- seattle: primary cluster node
- london: bunker node
- bdg : bunker disk group name
- brvg: bunker RVG name

Sample configuration file (failover application)

The following sample file shows the configuration for the VCS agent on the Primary. The RVG agent includes attributes for a STORAGE bunker, to enable the bunker diskgroup to failover together with the parent RVG.

In this example, the disk group on the Primary is not a shared disk group.

If the Secondary for the RDS has a bunker associated to it, the RVG agent on the Secondary similarly would include the StorageRVG, StorageDG, and StorageHostIds attributes.

```
group AppSG (
        ClusterList = { cluster_london = 0 }
        SystemList = { seattle = 0, london = 1 }
        Authority = 1
        AutoStartList = { seattle }
        ClusterFailOverPolicy = Manual
        )
        RVG RVG-1 (
     RVG = vcsrvg
     DiskGroup = pdg
     Primary = true
     StorageRVG = brvg
     StorageDG = bdg
     StorageHostIds = "portland"
                    )
...
```

Sample configuration file (parallel application)

The following sample file shows the configuration for the VCS agent on the Primary. The RVGLogowner agent includes attributes for a STORAGE bunker, to enable the bunker diskgroup to failover together with the logowner. In this example, the disk group on the Primary is a shared disk group. If the Secondary for the RDS has a bunker associated to it, the RVGLogowner resource on the Secondary similarly would include the StorageRVG, StorageDG, and StorageHostIds attributes.

```
group RVGLogownerGrp (
        SystemList = { seattle = 0, london = 1 }
        AutoStartList = { seattle, london }
        )
        IP vvr_ip (
                Device = bge0
                Address = "192.168.3.13"
                )
        NIC vvr_nic (
                Device = bge0
                )
        RVGLogowner vvr_rvglogowner (
    RVG = rvg
    DiskGroup = vvrdg
    StorageRVG = brvg
    StorageDG = bdg
    StorageHostIds = "portland"
```

```
    )
requires group RVGSharedGrp online local firm
      vvr_ip requires vvr_nic
```

VCS configuration for a bunker using IP

The configuration for the VCS agents for a bunker over IP is the same as for any other Secondary.

To set up a bunker configuration

1. The Primary and Secondary configurations are the same as for any other RDS using VCS agents.

   See "Example—Setting up VVR in a VCS environment" on page 36.

2. Add the bunker to the RDS with the `vradmin addbunker` command.

   For a detailed procedure, see the *Veritas Volume Replicator Adminstrator's Guide*.

3. Configure the VCS agent on the bunker in the same way as the configuration for any other Secondary. There is no special configuration that needs to be done for a bunker over IP.

Administering the service groups

This section explains how to administer a VCS service group for cluster Seattle from the command line. Note that you can also use the VCS Java and Web consoles to administer service groups.

To administer a VCS service group

1. Start the VCS engine on seattle1:

   ```
   # hastart
   ```

2. Verify that all the service groups that contain RVG resource type are brought online:

   ```
   # hagrp -display
   ```

3. Take the service group offline and verify that all resources are stopped:

   ```
   # hagrp -offline hr_grp -sys seattle1
   # hagrp -display
   ```

4 Bring the service group online again and verify that all resources are available:

```
# hagrp -online hr_grp -sys seattle1
# hagrp -display
```

5 Start the VCS engine on seattle2:

```
# hastart
```

6 Switch the VVR service group to seattle2:

```
# hagrp -switch hr_grp -to seattle2
```

7 Verify that all the service groups that contain RVG resource type are brought online on seattle2:

```
# hagrp -display
```

8 Repeat step 1 through step 7 for the cluster London.

9 If required, check the following log files on any system for the status or any errors:

/var/VRTSvcs/log/engine_A.log

/var/VRTSvcs/log/RVG_A.log

Index

L

M

N

O

P

R

S

T

V