Veritas™ Cluster Server Agent for EMC MirrorView Configuration Guide

ESX

5.1

Veritas Cluster Server Agent for EMC MirrorView Configuration Guide

Agent version: 5.1

Legal Notice

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.

Technical support

For technical assistance, visit

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

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 4 Setting up and running a fire drill

Chapter 1

Introducing the EMC MirrorView VCS agent

This chapter contains the following topics:

About the MirrorView agent

The VCS enterprise agent for EMC MirrorView provides application failover support and recovery in environments employing MirrorView to replicate data between EMC CLARiiON arrays. It monitors and manages the state of replicated CLARiiON LUNs attached to VCS nodes. The agent ensures that the system where the MirrorView resource runs has safe and exclusive access to the configured devices.

You can use the agent in global clusters. The agent supports MirrorView in the synchronous mode and supports individual mirrors or consistent groups in asynchronous mode.

Configure EMC MirrorView for use with synchronous or asynchronous mode for replication. In asynchronous mode, you can replicate either individual LUNs or consistency groups. MirrorView can also replicate LUNs or metaLUNs. In synchronous mode, you cannot replicate consistency groups.

See the following Technical Support TechNote for the latest updates or software issues for this agent:

Supported software and hardware

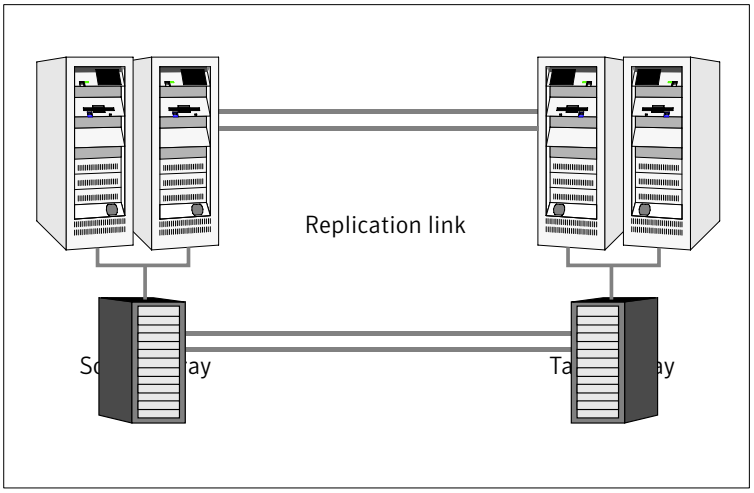
The EMC MirrorView agent supports Veritas Cluster Server 5.1 for ESX.

To determine the supported versions of NaviCLI and FLARE code that are on CLARiiON arrays, consult the EMC hardware compatibility list.

Typical MirrorView setup in a VCS cluster

Figure 1-1 displays a typical cluster setup in a MirrorView environment.

Figure 1-1 Clustering in a MirrorView environment



Clustering in a MirrorView environment typically consists of the following hardware infrastructure:

- The source array consists of one or more hosts with direct connection to a CLARiiON array. The array contains the mirror that is the primary image and the direct connection uses either SCSI or Fibre Channel.
- The target array consists of one or more hosts with a direct connection to another CLARiiON array. The array contains the mirror that is the secondary image and the connection uses either SCSI or Fibre Channel. The secondary image LUNs pairs with the mirrored LUNs in the source array. The target hosts and the array must be at a significant distance from the source side to survive a source-side disaster
- Network heartbeats using LLT or TCP/IP between the two data centers to determine their health.
  See “About cluster heartbeats” on page 18.
- In a global cluster environment, you must attach all hosts in a cluster to the same CLARiiON array.

MirrorView agent functions

The VCS enterprise agent for EMC MirrorView monitors and manages the state of replicated CLARiiON LUNs attached to VCS nodes. Agent functions bring resources online, take them offline, and perform different monitoring actions. Agent functions are also known as entry points.

online	Creates a lock file on the local host. This lock indicates that the resource is online and makes the mirrors available for the application to use. The agent performs specific actions depending on the state of the mirrors. See "About the MirrorView agent's online function" on page 10.
monitor	Verifies that the lock file exists. If the lock file exists, the monitor entry point reports the status of the resource as online. If the lock file does not exist, the monitor entry point reports the status of the resource as offline.
offline	Removes the lock file on the host where the entry point is called.
open	Prevents potential concurrency violation if the service group fails over to another node by checking for the existence of the lock file. If Open detects the lock file, it waits until at least one of its parent resources is probed. If the parent resource is ONLINE, then the agent is calling Open in response to being restarted after it was killed or after HAD was killed. In either case, the agent does not remove the lock file. If the parent resource is OFFLINE, then the agent is being restarted in response to HAD being started, in which case Open removes the lock file and the agent reports OFFLINE. **Note:** The agent does not remove the lock file if the agent was started after a `hastop -force` command.
clean	Removes the lock file on the host where the entry point is called.
info	The info function gives the information about the mirrors (in case of synchronous mode of replication). It also gives information about the mirrors/group in case of asynchronous mode of replication. It uses the `-sync listsyncprogress` and `-async -list` or `-async listgroups` commands to get this information.
resynch	Performs a resynchronization action.

About the MirrorView agent's online function

The agent online function performs specific actions depending on the state of the mirrors.

Synchronous state

If the local mirror is the primary image, the agent creates a lock file on the local host. This lock indicates that the resource is online and makes the mirrors available for the application to use.

If one or more mirrors are not in the MIRRORED state, the agent promotes them into the MIRRORED state, which enables the application to use them.

- For secondary images in the synchronized state, the agent runs the `mirror -sync -promoteimage` command to promote the remote mirror. This command also converts the current primary to secondary.
- For secondary images in the CONSISTENT state, the agent waits to check if the image has transitioned to the SYNCHRONIZED state.

  If the images have transitioned to the SYNCHRONIZED state, the agent then runs the `mirror -sync -promoteimage` command to promote the remote mirror. This command also converts the current primary to secondary.

  If the image has not transitioned to the SYNCHRONIZED state, the agent checks if the remote array is accessible. If the remote array is accessible, then this condition indicates link failure—the image would be in a fractured condition.

In case of fracture:

- If the SplitTakeover attribute is set to 1, the agent forcibly promotes the secondary image.
- If the SplitTakeover attribute is set to 0, the agent does not try to promote the secondary image forcibly, and becomes the administrator's decision.

Asynchronous state

You can configure the online function for either consistency groups or mirrors in asynchronous mode.

Consistency groups

If the mirrors in the consistency group on the local array are primary images, the agent creates a lock file on the local host to indicate that the resource is online. This lock makes the LUNs available for the application to use.

If one or more mirrors are not in the MIRRORED state, the agent checks to see if the remote array is accessible.

- If the remote array is not accessible, then the agent checks the value of the attribute SplitTakeover before proceeding with any further actions.
- If the SplitTakeover attribute is set to 1, the agent forcibly promotes the secondary image.

- If the SplitTakeover attribute is set to 0, the agent does not try to promote the secondary image forcibly, and becomes the administrator's decision.
- If the remote array is accessible, then the agent runs the `mirror -async -promotegroup` command to promote the remote group.
- In case of a successful promotegroup operation, the operation also converts the current primary to secondary.
- If the promotegroup operation is not successful, then the agent initiates a synchronization.

  The agent periodically checks if the group is SYNCHRONIZED. After a successful synchronization, the agent promotes the group using the `mirror -async -promotegroup` command. If the synchronization is not successful, the agent times out.

Mirrors

If the local image is the primary image, the agent creates a lock file on the local host. The lock indicates that the resource is online and makes the mirrors available for the application to use.

If one or more mirrors are not in the MIRRORED state, the agent checks to see if the remote array is accessible.

- If the remote array is not accessible, then the agent checks the value of the attribute SplitTakeover before proceeding with any further actions.
- If the SplitTakeover attribute is set to 1, the agent forcibly promotes the secondary image.
- If the SplitTakeover attribute is set to 0, the agent does not try to promote the secondary image forcibly, and becomes the administrator's decision.
- If the remote array is accessible, then the agent runs the `mirror -async -promoteimage` command to promote the remote mirrors.
- A successful promoteimage operation converts the current primary to secondary.

If the promoteimage operation is not successful, then the agent initiates a synchronization.

The agent periodically checks if the group is SYNCHRONIZED. After a successful synchronization, the agent promotes the secondary mirror using the `mirror -async -promoteimage` command. If the synchronization is not successful, the agent times out.

Chapter 2

Configuring the MirrorView agent

This chapter contains the following topics:

Configuration concepts

Review the resource type definition and the attribute definitions for the agent.

MirrorView resource type definition

The resource type defines the MirrorView agent.

```
MirrorView
```

Attribute definitions for the MirrorView agent

This section describes the MirrorView agent attributes.

Required attributes

You must assign values to required attributes

NaviCliHome	Specifies the NaviCLI installation directory Type and dimension: string-scalar
LocalArraySPNames	Specifies the list of storage processors within the array to which the local hosts are connected. Can be names or IP addresses. Type and dimension: string-vector
RemoteArraySPNames	Specifies the list of storage processors within the array to which the remote hosts are connected. Can be names or IP addresses. Type and dimension: string-vector
Mode	Specifies the replication mode, which is either: `sync` or `async`. Type and dimension: string-scalar
GrpName	Specifies the name of the consistency group to which the mirrors belong. This function applies only if the mode is `async`. Type and dimension: string-scalar
MirNames	Specifies the list of individual mirrors that are a part of the replication relationship and managed by VCS. This attribute is ignored if you specify the GrpName attribute. Type and dimension: string-vector

SplitTakeover

Indicates whether VCS should forcefully promote a secondary to a primary.

In case of a link-failure between the two arrays, the state of the mirror remains consistent or out-of-sync. Under such circumstances, if the application has to failover—due to disaster or user-driven action—mirrors are not in a SYNCHRONIZED state.

If the value of the SplitTakeOver attribute is 1:

- the agent fails over when it discovers link failures
- the agent determines that mirrors are out of sync

If the value of the attribute is 0, agent does not fail over and the administrator must to determine what to do.

Type and dimension: integer-scalar

Internal attribute

Do not modify internal attributes.

VCSResLock

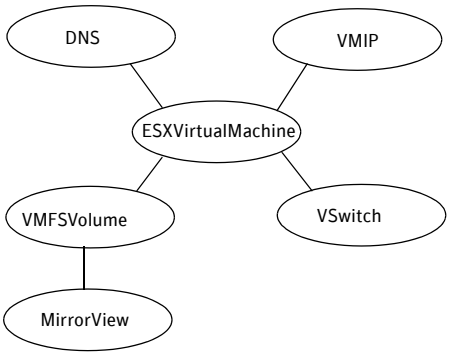
The agent uses this attribute to guarantee serialized management in case of a parallel application. Do not modify this value.

Type and dimension: temporary-string

Sample configuration

Figure 2-1 shows a VCS service group that has a resource of type MirrorView. The VMFSVolume resource depends on the MirrorView resource.

Figure 2-1 Dependency tree



Asynchronous mode

You can configure a resource of type MirrorView in the `main.cf` file. In this example, the resource is configured for asynchronous mode and consistency groups.

```
MirrorView mir (
    NaviCliHome = "/opt/Navisphere/bin"
    LocalArraySPNames @sys1= = { "Local_SP1_Name", "Local_SP1_IP" }
    LocalArraySPNames @sys2 = { "Local_SP2_Name", "Local_SP2_IP" }
    RemoteArraySPNames @sys1 = { "Local_SP2_IP", "Remote_SP2_Name" }
    RemoteArraySPNames @sys2= { "Local_SP1_IP", "Remote_SP1_Name" }

    Mode = async
    GrpName = consistency_grp1

    SplitTakeover = 0
    )
```

If you want set up asynchronous replication with individual mirrors—no consistency groups—replace the lines beginning with Mode and GrpName with:

```
    Mode = async
    MirNames = { "async_mir1", "async_mir2" }
    GrpName = ""
```

If you want to configure the resource for synchronous mode and specify the mirror names, replace the lines beginning with Mode and GrpName with:

```
Mode = sync
MirNames = { "sync_mir1", "sync_mir2" }
GrpName = ""
```

Before you configure the MirrorView agent

Before you configure the agent:

- Review the configuration concepts, which describe the agent's type definition and attributes.
  See "Configuration concepts" on page 14.
- Verify that the agent is installed on all systems in the cluster.
- Verify the hardware setup for the agent.
  See "Typical MirrorView setup in a VCS cluster" on page 9.
- Make sure the cluster has an effective heartbeat mechanism in place.
  See "About cluster heartbeats" on page 18.
- Set up an effective heartbeat mechanism to prevent split-brain.
  See "About preventing split-brain" on page 18

About cluster heartbeats

Additionally, you can configure a low-priority heartbeat across public networks.

In a global cluster, VCS sends ICMP pings over the public network between the two sites for network heartbeating. To minimize the risk of split-brain, VCS sends ICMP pings to highly available IP addresses. VCS global clusters also notify the administrators when the sites cannot communicate.

Heartbeat loss may occur due to the failure of all hosts in the primary cluster. In such a scenario, a failover may be required even if the array is alive. In any case, a host-only crash and a complete site failure must be distinguished. In a host-only crash, only the ICMP heartbeat signals a failure by an SNMP trap. No cluster failure notification occurs because a surviving heartbeat exists. This trap is the only notification to fail over an application.

About preventing split-brain

Split-brain occurs when all heartbeat links between the primary and secondary hosts are cut. In this situation, each side mistakenly assumes that the other side is down. Minimize the effects of split-brain by ensuring the cluster heartbeat

links pass through similar physical infrastructure as the replication links so that if one breaks, so does the other.

If it is physically impossible to place the heartbeats alongside the replication links, there is a possibility that the cluster heartbeats are disabled, but the replication link is not. A failover transitions the original source to source volumes and vice-versa. In this case, the application faults because its underlying volumes become write-disabled, causing the service group to fault. VCS tries to fail it over to another host, causing the same consequence in the reverse direction. This phenomenon continues until the group comes online on the final node. You can avoid this situation by setting up your infrastructure such that loss of heartbeat links also mean the loss of replication links.

Configuring the agent

You can adapt most of the applications that you configure in VCS to a disaster recovery environment by:

- Converting their LUNs to CLARiiON LUNs
- Synchronizing the mirrors
- Adding the EMC MirrorView agent to the service group

After configuration, the application service group must follow the diagram in Figure 2-1 on page 17.

Configuring the agent manually in a global cluster

Configuring the agent manually in a global cluster involves the following tasks.

To configure the agent in a global cluster

1. Start Cluster Manager and log on to the cluster.
2. If the agent resource type (MirrorView) is not added to your configuration, add it. From the Cluster Explorer **File** menu, choose **Import Types** and select /etc/VRTSvcs/conf/MirrorViewTypes.cf.
3. Click **Import**.
4. Save the configuration.
5. Add a resource of type MirrorView at the bottom of the service group.
6. Configure the attributes of the MirrorView resource.
7. If the service group is not configured as a global group, configure the service group using the Global Group Configuration Wizard. See the *Veritas Cluster Server User's Guide* for more information.

8 Change the ClusterFailOverPolicy from the default, if necessary. Symantec recommends keeping the default, which is Manual, to minimize the chance of failing over on a split-brain.

Repeat step 5 through step 8 for each service group in each cluster that uses replicated data.

Chapter 3

Managing and testing clustering support for EMC MirrorView

This chapter contains the following topics:

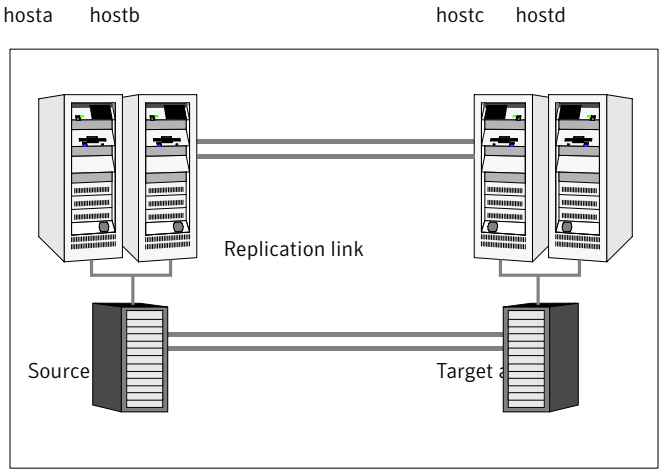
Typical test setup

A typical test environment includes:

- Two hosts (hosta and hostb) attached to the source CLARiiON array.
- Two hosts (hostc and hostd) attached to the target CLARiiON array.
- The application runs on hosta and devices in the local array are read-write enabled in the SYNCHRONIZED state.
- A global cluster has one network heartbeat.

Figure 3-1 depicts a typical test environment.

Figure 3-1 Typical test setup



Testing service group migration

Verify the service group can migrate to different hosts in the cluster.

To perform the service group migration test

1. Migrate the service group to a host that is attached to the same array. In the Service Groups tab of the Cluster Explorer configuration tree, right-click the service group.
2. Click **Switch To**, and click the system that is attached to the same array (hostb) from the menu.

The service group comes online on hostb and local image remains in the MIRRORED state.

3 Migrate the service group to a host that is attached to a different array. In the Service Groups tab of the Cluster Explorer configuration tree, right-click the service group.

4 Click **Switch To**, and click the system that is attached to the another array (hostc) from the menu.
The service group comes online on hostc and the role of the images there transition to primary.

5 Accumulate dirty tracks on the new source-side and update them back on the target:
hares -action *mirrorview_res_name* resync -sys hostc
The variable *mirrorview_res_name* represents the name of the MirrorView resource.

6 After the devices transition to a source SYNCHRONIZED state, migrate the service group back to its original host. In the Service Groups tab of the Cluster Explorer configuration tree, right-click the service group.

7 Click **Switch To**, and click the system on which the group was initially online (hosta).
The group comes online on hosta. The devices return to the RW/SYNCINPROG state at the array that is attached to hosta and hostb, and then eventually transition to the SYNCHRONIZED state.

Testing host failure

In this scenario, the host where the application runs is lost. Eventually all the hosts in the system zone or cluster are lost.

To perform the host failure test

1 Halt or shut down the host where the application runs (hosta).
The service group fails over to hostb and devices are in the SYNCHRONIZING state.

2 Halt or shut down hostb.
In a global cluster, a cluster down alert appears and gives you the opportunity to fail over the service group manually.
In both environments, the role of the devices changes from secondary to primary and starts on the target host.

3 Reboot the two hosts that were shut down.

4 Switch the service group to its original host when VCS starts. In the Service Groups tab of the Cluster Explorer configuration tree, right-click the service group.
Click **Switch To**, and click the system where the service group was initially online (hosta).The service group comes online on hosta and devices transition to the SYNCHRONIZING state and then to the SYNCHRONIZED state.

Performing a disaster test

Test how robust your cluster is in case of a disaster.

To perform a disaster test

1 Shut down all hosts on the source side and shut down the source array.
If you cannot shut down the source array, change the value of the RemoteArraySPNames in the target side to non-existent names and IP addresses. This action mimics a disaster scenario from the target's point of view.

2 In a global cluster, the administrator is notified of the failure. The administrator can then initiate the failover.

Performing the failback test

You can set up your cluster for a failback test.

To perform the failback test for asynchronous mode with Consistency groups

1 Remove all the mirrors form the consistency group on the old primary.

2 Destroy the consistency group on the old primary.

3 Forcefully destroy the remote mirrors on the old primary.

4 Remove the LUNs from the storage group on the old primary.

5 Remove the mirrors from the consistency group on the new primary.

6 Add secondary images to each of the remote mirrors on the new primary.

7 Add the mirrors into the consistency group on the new primary.

8 Add the LUNs, where the secondary image resides, into the appropriate storage group on the old primary

Between step 5 and step 7, the LUNs become vulnerable to data corruption. For example, if one of the LUNs has sustained hardware damage and failed.

During this window, the mirrors are not a part of the consistency group. The writes to other mirrors that were a part of the consistency group are not stopped. This situation could result in data corruption.

To perform the failback test for synchronous and asynchronous mode with Individual mirrors

1. Forcefully destroy the remote mirrors on the old primary.
2. Remove the LUNs from the storage group on the old primary.
3. Add secondary images to each of the remote mirrors on the new primary.
4. Add the LUNs, where the secondary image resides, into the appropriate storage group on the old primary.

In either of the modes, the original contents of the old primary are lost.

Failure scenarios

Review the failure scenarios and agent behavior in response to failure.

All host or all application failure

Even if both arrays are operational, the service group fails over in the following conditions:

- All hosts on the source side are disabled.
- The application cannot start successfully on any source host.

In global cluster environments, failover requires user confirmation by default. Multiple service groups can fail over in parallel.

Site disaster

In a total site failure, all hosts and the array are completely disabled, either temporarily or permanently. In a global cluster, VCS detects site failure by the loss of all configured heartbeats.

A total disaster renders the devices on the surviving array in the FRACTURED state. If the SplitTakeover attribute is set to its default value of 1, the online entry point runs the 'promote' operation. If the attribute is set to 0, no takeover occurs and the online entry point times out and faults.

The online entry point detects whether any synchronization was in progress when the source array was lost. Since the target devices are inconsistent until the synchronization completes, the agent does not write-enable the devices, but it times out and faults. You must restore consistent data from a snapshot or tape backup.

Replication link failure

Before the MirrorView takes any action, it waits for the synchronization to complete in the following situations:

- The two arrays are healthy and the link that failed is restored.
- A failover is initiated while synchronization is in progress.

After the synchronization completes, the MirrorView runs the promote operation.

If the agent times out before the synchronization completes, the resource faults.

If the SplitTakeover attribute is set to 0, the agent does not attempt a promote operation, but it times out and faults. If you write-enable the devices manually, the agent can come online after it is cleared.

Chapter 4

Setting up and running a fire drill

This chapter contains the following topics:

About fire drills

A fire drill procedure verifies the readiness of a disaster recovery configuration. This procedure is performed without stopping the application at the primary site and disrupting user access.

A fire drill is performed at the secondary site using a special service group for fire drills. The fire drill service group is identical to the application service group, but uses a fire drill resource in place of the replication agent resource. The fire drill service group uses a copy of the data used by the application service group.

In clusters employing EMC MirrorView, the MirrorView resource manages the replication relationship during a fire drill. Bringing the fire drill service group online demonstrates the ability of the application service group to come online at the remote site when a failover occurs.

For VMware, a fire drill can also be performed locally to generate a candidate for a “last-known good copy” of your application data. If the replicated data produced by the fire drill tests successfully, that copy is your last-known good copy.

Considerations for using MirrorView and SnapView together

MirrorView is an EMC software application that maintains a copy or image of a logical unit (LUN) at a separate location as a provision for disaster recovery. You can use this image in the event that a disaster disables the production image. The production image (the image that is mirrored) is called the primary image; the copy image is called the secondary image.

SnapView is a storage-system-based software application that enables you to create a copy of a LUN by using either clones or snapshots. A clone is an actual copy of a LUN and takes time to create, depending on the size of the source LUN. A clone is a full copy. A snapshot is a virtual point-in-time copy of a LUN and takes only seconds to create. A snapshot uses a copy-on-write principle.

Fire drills use SnapView in conjunction with MirrorView. The following are important considerations regarding the joint use of these applications:

- If a LUN is a MirrorView primary or secondary image, you cannot create a clone group for that image. Similarly, if a LUN is a member of a clone group as the source or clone, it cannot serve as a MirrorView primary or secondary image.
- If the MirrorView Synchronous option is installed, you can create a snapshot of the primary or secondary image. However, Symantec recommends that

you take a snapshot of a secondary image only if the state of the image is either SYNCHRONIZED or CONSISTENT. If the image is SYNCHRONIZING or OUT-OF-SYNC, the snapshot data is not useful.

- If the MirrorView Asynchronous option is installed, you can create a snapshot of the primary or secondary image. However, Symantec recommends that you take a snapshot of a secondary image only if the last update started has completed successfully. If the update did not complete successfully because the image is fractured or the update is still in progress, the snapshot data is not useful.

The VCS MirrorView fire drill agent, MirrorViewSnap, does not support clones because of these considerations.

About the MirrorViewSnap agent

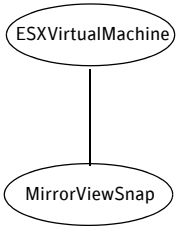
The MirrorViewSnap agent uses SnapView to take a snapshot of a source LUN and then makes the snapshot available to the secondary server. The MirrorViewSnap resource is configured in the fire drill service group.

Agent fire drill functionality

The MirrorViewSnap agent makes a local copy of the LUNs in the mirrors that are specified as part of the fire drill configuration. The agent resides at the bottom of the dependency tree because a snapshot of data must be made available before the application can use it.

Note: The MirrorViewSnap agent creates SnapView snapshots, not SnapView clones.

Figure 4-1 MirrorViewSnap agent dependency tree



A snapshot is a virtual LUN. When a secondary server views a snapshot, it is viewing a point-in-time copy of a source LUN. You determine the point in time when you start a SnapView session. The session keeps track of the source LUN's data at a particular point in time. During a session, the production server can still write to the source LUN and modify data.

Agent prerequisites

Before initiating a fire drill, you must complete the initial setup for the MirrorViewSnap agent. After the initial setup is completed, VCS can:

- Take a snapshot
- Make the snapshot read-writable
- Start the application, which is already running on the production cluster, to verify the production data

To set up the MirrorViewSnap agent

1. Establish the MirrorView relationship.
2. Reserve sufficient unallocated LUNs in the reserved LUN pool.
3. Install and enable the SnapView licence.
4. Install the Navisphere CLI.
5. Before running a fire drill on a cluster at the secondary site , ensure that no VCS global service groups are online on the secondary site. Do this by ensuring that any virtual machines that are configured for disaster recovery through VCS are not in the powered-on state.

Agent operations

The MirrorViewSnap agent performs the following operations.

Online

Destroys any existing snapshot, takes a new snapshot of the LUNs in the mirror/consistency group, and then make it available for use.

The Online entry point performs the following steps:

- Destroys any existing snapshot. (If a snapshot already exists, it was taken as a part of a previous Online operation for the current MirrorViewSnap type of resource.) If the agent is performing an Online operation for the first time or if an existing snapshot was manually destroyed, you receive error messages in the log file that pertain to the inability to destroy the snapshot. You may safely ignore these messages.
- Retrieves the state of the mirror/consistency group. If the mirror/consistency group is not in either SYNCHRONIZED/CONSISTENT state, the fire drill cannot continue. The agent logs an error and exits.
- Creates the MirrorViewSnap object.
- Takes the snapshot.
  In case of any error, the agent rolls back the changes for all the other mirrors.
- Creates a lock file and exits.

Offline

Removes the lock file.

Monitor

Checks for the existence of the lock file. If the lock file exists, Monitor reports the state of the MirrorViewSnap resource as ONLINE. Otherwise, it reports the state as OFFLINE.

Open

Checks for the existence of the lock file. If Open detects the lock file, it waits until at least one of its parent resources is probed. If the parent resource is ONLINE, then the agent is calling Open in response to being restarted after it was killed or after HAD was killed. In either case, the agent does not remove the lock file. If the parent resource is OFFLINE, then the agent is being restarted in response to HAD being started, in which case Open removes the lock file and the agent reports OFFLINE.

Clean

Removes the lock file.

Agent resource type definition

The MirrorView Snap agent is represented by the MirrorViewSnap resource type in VCS.

```
type MirrorViewSnap (
    static keylist SupportedActions = { resync }
    static int MonitorInterval = 600
    static int NumThreads = 1
    static int OnlineTimeout = 600
    static int MonitorTimeout = 600
    static int ActionTimeout = 300
    static int RestartLimit = 1
    static str ArgList[] = { StorageGrpName, TargetResName,
    NaviCliHome, LocalArraySPNames }
    str StorageGrpName
    str TargetResName
    str NaviCliHome = "/opt/Navisphere/bin"
    str LocalArraySPNames[]
    temp str Responsibility
)
```

Attribute definitions

Configure the following attributes to customize the behavior of the MirrorViewSnap agent.

TargetResName	Name of the resource managing the LUNs to be snapshot. Set this attribute to the name of the MirrorView resource if the data being snapshot is replicated. Set the attribute to the name of the ESXVirtualMachine resource if the data is not replicated. Type-Dimension: string-scalar
StorageGrpName	Name of the storage group to which you want to add the snapshot. The host that runs the fire drill must be a part of this storage group. Otherwise, the fire drill fails. Type-Dimension: string-scalar
Responsibility	For internal use only. Used by the agent to keep track of resynchonizing snapshots. Type-Dimension: temporary string

Sample main.cf file including the fire drill service group

The configuration of a fire drill service group is almost identical to that of an application service group that has a hardware replication resource. The difference is that, in a fire drill service group, the MirrorViewSnap resource replaces the MirrorView resource as shown in the sample main.cf file.

The fire drill groups for running the fire drill on both a secondary site (normal fire drill) and locally (to generate last-known good copy) are denoted by the following declarations:

```
group test_mirrorview_fd
group test_mirrorview_fd_local
```

The vmname attribute is not a mandatory attribute for the ESXVirtualMachine resource. However, to work correctly, both a local fire drill and a remote fire drill require that you configure the vmname attribute. The sample main.cf file provides an example.

Note: In the MirrorViewSnap resource type definition, ensure that the MonitorInterval attribute is set to 600 and that the ActionTimeout attribute is set to 300.

See "Agent resource type definition" on page 32.

The following is the sample main.cfg file:

```
//MirrorView firedrill service group
group test_mirrorview_fd(
        SystemList = { dr_sys0 = 0, dr_sys1 = 1 }
        )

        ESXVirtualMachine vm_res_fd (
                Critical = 0
               vmname = vm_fd_test
                CfgFile = " "
                )

        MirrorViewSnap mvs_snapres_fd (
                Critical = 0
                StorageGrpName = fd_storage_grp_name
                TargetResName = MirrorView_vm_res_fd
                LocalArraySPNames = { "dr_array1.veritas.com",
"dr_array2.veritas.com" }
                )

        vm_res_fd requires mvs_snapres_fd

//Global MirrorView service group
group MirrorView_SG (
        SystemList = { dr_sys0 = 0, dr_sys1 = 1 }
```

```
        ClusterList = { cluster_dr = 0, cluster_pri = 1 }
        )

        MirrorView MirrorView_vm_res_fd (
                LocalArraySPNames = { "dr_array.veritas.com" }
                RemoteArraySPNames = { "prim_array.veritas.com" }
                Mode = sync
                MirNames = { fd_replication_MirName}
                )

//Local MirrorView firedrill service group
group test_mirrorview_fd_local (
        SystemList = { dr_local_sys0 = 0, dr_local_sys1 = 1 }
        )

        ESXVirtualMachine vm_res_fd_local (
                Enabled = 0
               vmname = local_vm_fd
                CfgFile = " "
                )

        MirrorViewSnap mvsnp_res_fd_local (
                Critical = 0
                StorageGrpName = fd_storage_grp_name
                TargetResName = vm_res_fd
               LocalArraySPNames = { "dr_local_array1.veritas.com" }
                )

        vm_res_fd_local requires mvsnp_res_fd_local
```

Before you configure the fire drill service group

Do the following before configuring the service group:

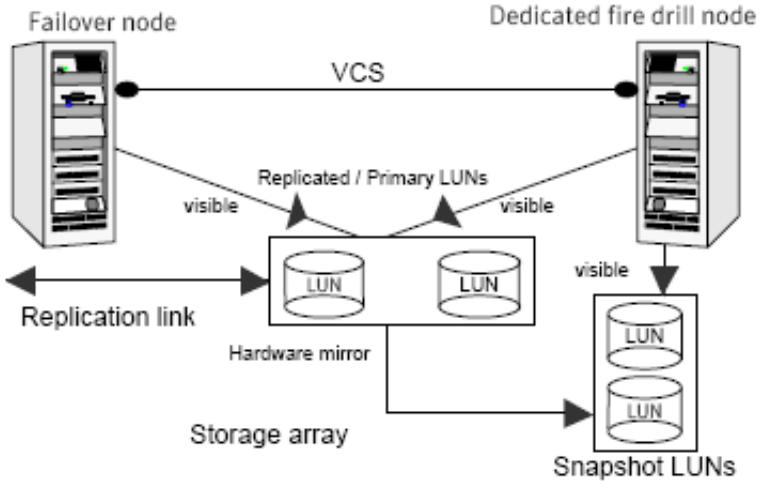
- Ensure that the application service group is configured with a MirrorView resource.
- Ensure that the infrastructure that takes snapshots is properly configured between the source and target arrays.

Additional requirements for running a fire drill in an ESX environment

Follow these guidelines for fire drills in an ESX environment:

- Ensure that each service group contains only one virtual machine resource to be able to support fire drill.
- Symantec recommends dedicating one node in the cluster for fire drills.

- Configure fire drill service groups on this node. Do not configure replication resources on this node.
- Configure your array such that snapshot LUNs (i.e. target LUNs on which hardware snapshots are taken) are visible to this node only; the other nodes must not see snapshot LUNs. The restriction occurs because other nodes in the cluster may have LVM settings LVM.EnableResignature = 0 and LVM.DisAllowSnapshotLUN = 0 set by the replication agents, in which case VMWare recommends that no snapshot LUNs should be exposed to such ESX hosts.



- Install VMware tools to all virtual machines in the fire drill configuration. Otherwise, the missed virtual machine heartbeats cause the virtual machine resource to fault.
- Ensure that no virtual machines are configured with raw device mapping.
- Ensure that the resources configured within the fire drill service group are not marked as critical. Not marking the fire drill service group resources as critical prevents service group failover.

Configuring the fire drill service group

This section describes how to use Cluster Manager (Java Console) to create the fire drill service group.

To create the fire drill service group

1. Open the Veritas Cluster Manager (Java Console).
2. Log on to the cluster and click **OK**.
3. Click the Service Group tab in the left pane and click the Resources tab in the right pane.
4. Right-click the cluster in the left pane and click Add Service Group.
5. In the Add Service Group dialog box, provide information about the new service group
6. In Service Group name, enter a name for the fire drill service group
7. From the Available Systems box, select the node that is dedicated for fire drills.
8. Click the fire drill service group in the left pane and click the Resources tab in the right pane.
9. Add a resource of type MirrorViewSnap and configure its attributes.
10. Add a resource of type ESXVirtualMachine and configure its attributes.
See “Sample main.cf file including the fire drill service group” on page 33.
11. Link resources such that the ESXVirtualMachine resource depends on the MirrorViewSnap resource.

Running the fire drill

Before running a fire drill on the secondary site, ensure that it does not have any global service groups that have failed or switched over to the secondary site in the online state.

Additionally, ensure that you do not run a local fire drill on the secondary site, even if the secondary site has become the new primary site after a failover to the remote cluster. These restrictions occur because of ESX server behavior during datastore rescans with LVM settings: EnableResignature = 1; DisAllowSnapshotLun = 0; which may cause datastores to be resignatured. Refer to the Release Notes for more information.

You are now ready to run the fire drill.

To run the fire drill

1. Bring the fire drill service group online. The configured applications come up using snapshot data.
2. Ensure the validity of your application by using it to perform tasks that are appropriate for the application. Being able to use the application indicates a successful fire drill and ensures that your replicated data is valid. The snapshot is stored in the current state until you run the next fire drill.
3. Take the fire drill service group offline. .
   Failing the take the fire drill service group offline could cause failures in your environment. For example, if the application service group fails over to the node hosting the fire drill service group, there would be resource conflicts, resulting in both service groups faulting.

Rescanning HBAs

The agent usually performs this task. However, if you run rescan HBA operations outside VCS, it is very important that you do the following:

- Ensure that you are aware of the current values of the LVM settings DisAllowSnapshotLun and EnableResignature.
- Use the following LVM setting while you are rescanning snapshot luns for new datastores:
  ```
  EnableResignature=1
  ```
- Ensure that snapshot luns are not presented to the server if both of the following conditions apply:
  - The snapshot LUNs do not contain previously-resignatured datastores
  - You are rescanning with the following settings:
    ```
    DisAllowSnapshotLun=0
    EnableResignature=0
    ```

Index

N

R

S

T

V