

Serviceguard NFS Toolkit Support for Cluster File System

June 2007



Contents

1. ABSTRACT	3
2. INTENDED AUDIENCE	3
3. INTRODUCTION	3
<i>Non-CFS Implementation with Package Failover</i>	3
<i>Integrating Support for CFS into Serviceguard NFS Toolkit</i>	3
4. OVERVIEW	4
<i>Current SG-NFS over VxFS Support</i>	4
<i>CFS vs. Non-CFS Implementation</i>	4
5. PREREQUISITES	7
5.1 PRODUCT REQUIREMENTS	7
5.2 PATCH REQUIREMENTS.....	7
6. CONFIGURING SERVICEGUARD NFS OVER CFS PACKAGES	7
6.1 SERVICEGUARD NFS OVER CFS PACKAGES WITHOUT FILE LOCKING.....	8
<i>Configuring a Serviceguard NFS export package</i>	9
<i>Starting a Serviceguard NFS export package</i>	11
<i>Configuring a Serviceguard NFS failover package</i>	12
<i>Starting a Serviceguard NFS failover package</i>	13
6.2 SERVICEGUARD NFS OVER CFS PACKAGES WITH FILE LOCKING	14
<i>Configuring a Serviceguard NFS failover package</i>	15
<i>Starting a Serviceguard NFS failover package</i>	18

1. Abstract

This white paper describes how the Serviceguard NFS Toolkit now supports the Veritas Storage Foundation Cluster File System (VxCFS, referred to as CFS throughout this document) in a Highly Available NFS environment. In a Serviceguard NFS Toolkit implementation with VxFS file systems (i.e. a non-CFS implementation), access to specific files and file systems in a Serviceguard environment is limited to a single NFS server. By configuring Serviceguard NFS Toolkit to utilize CFS, simultaneous access to files and file systems is expanded to multiple NFS servers. This paper describes how to configure Serviceguard NFS Toolkit to work with CFS and compares the CFS and non-CFS implementations. This document can be considered as an addendum to the Serviceguard NFS Toolkit Administrator's Guide.

2. Intended Audience

This white paper is intended for administrators who configure Serviceguard NFS Toolkit for CFS support.

3. Introduction

Non-CFS Implementation with Package Failover

Serviceguard NFS Toolkit allows you to use Serviceguard to set up highly available NFS servers. An NFS server is a host that shares its local directories (i.e. makes them available for client hosts to mount using NFS). On the NFS client, these mounted directories look to users like part of the client's local filesystem. With Serviceguard NFS, the NFS server package (also referred to as an NFS failover package) containing the exported filesystems can move to a different node in the cluster in the event of failure. After Serviceguard starts the NFS package on the adoptive node, the NFS filesystems are re-exported from the adoptive node with minimum disruption of service to users. The client system hangs until the NFS server package comes up on the adoptive node. When the service returns, the application can continue access to the file. You do not need to restart the client system or the client applications accessing the NFS filesystem. This implementation will be referred to throughout this document as a non-CFS implementation. Non-CFS implementations mostly use VxFS filesystems, so the term VxFS will also be used to refer to the non-CFS implementation.

Integrating Support for CFS into Serviceguard NFS Toolkit

In a non-CFS implementation, access to a specific file must go through a single NFS server which can limit scalability and cause a performance bottleneck. With CFS, files and filesystems may be accessed concurrently from multiple servers. By integrating CFS into a Serviceguard NFS environment, files and filesystems can be accessed concurrently from multiple highly available NFS servers to improve performance, scalability, and availability.

Initial Serviceguard NFS Toolkit support for CFS will be limited in scope, specifically for applications that require file locking across NFS mounts during a failover. For applications that do not require file locking during a failover, a DNS round-robin scheme or other load balancing tools may be used to distribute the load evenly between the servers. Additional support will be phased into future releases.

4. Overview

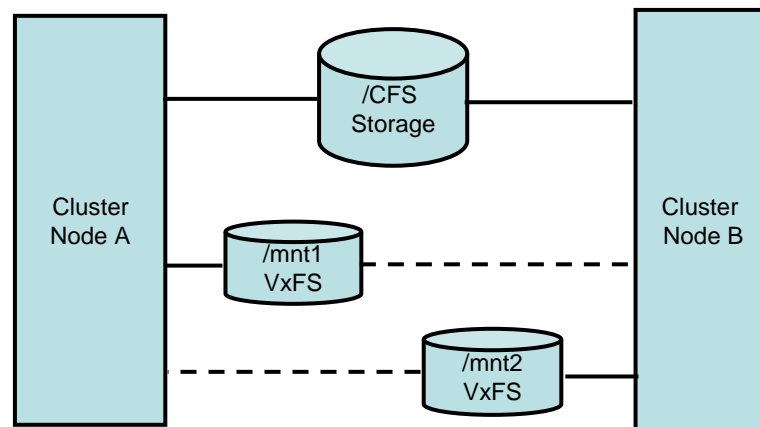
Current SG-NFS over VxFS Support

In a Serviceguard environment, a Serviceguard NFS over VxFS (failover) package is defined as a collection of resources including relocatable IP addresses and logical volume groups (data disks) associated with NFS services. If the server on which a package is running fails, then Serviceguard will automatically start the NFS package on the adoptive node. The adoptive node takes control of the IP addresses and disks and starts the NFS services.

CFS vs. Non-CFS Implementation

The Serviceguard A.11.18 release supports CFS which provides data and lock coherency and allows files and filesystems to be shared and accessed concurrently by multiple servers.

[Figure 1](#) shows how files and filesystems are accessed differently in a non-CFS environment versus a CFS environment. In a non-CFS environment, the highly available filesystems move from one node to another when there is a failover. The solid lines show which primary node provides access to which filesystem and the dotted lines show which adoptive node provides access in the event of a failover. The volumes are exclusively activated only on the server that currently runs the package, which prevents files and filesystems from being altered concurrently by multiple nodes. The Serviceguard NFS package control scripts ensure that upon package failure or shutdown the storage is made inaccessible on the node where the package failed or was halted.



**A cluster file system is concurrently accessed by all cluster nodes.
A non-CFS file system (VxFS) is exclusively activated by one node and can transition between nodes.**

Figure 1. CFS Vs. Non-CFS (VxFS) Implementation

In a Serviceguard CFS environment, files and filesystems are concurrently accessible on multiple nodes. When a package fails over, the adoptive systems do not have to mount the disks from the failed system because they are already mounted. There is a new multi-node package that runs on each server in the cluster and exports all the cluster filesystems. The exported filesystems do not have to be re-exported when a package fails over. These factors may reduce failover time.

The files and filesystems can be shared and accessed concurrently within the cluster. However, file sharing and access from outside the cluster will still require NFS - client systems which are not members of the CFS cluster will use NFS to access the shared filesystems.

Cross mounting (with the `nfs_xmmt` script) is not needed since you can use CFS to share files and filesystems within the cluster.

[Figures 2](#) and [3](#) also show how files and filesystems are accessed differently in a CFS environment versus a non-CFS environment. In a non-CFS environment, clients must access the server which exports a specific filesystem. In a CFS environment, clients can access the cluster via a load balancer or another mechanism such as a DNS round-robin scheme (represented by the cloud in Figure 3). Each NFS client can be directed to the server which currently has the most capacity available. This approach has a limitation that clients are bound to a particular NFS server when they issue the mount command.

Note: The implementation of a load balancer or DNS round-robin scheme is optional and is beyond the scope of this white paper. For more information about DNS round-robin addressing refer to the BIND Name Service Overview section in the [HP-UX IP Address and Client Administrator's Guide](#).

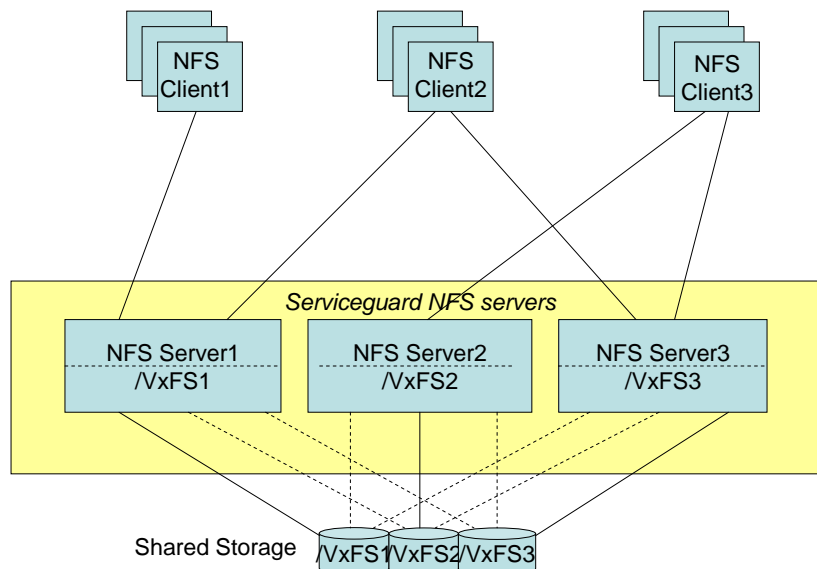


Figure 2. SG NFS Servers over VxFS – High Availability

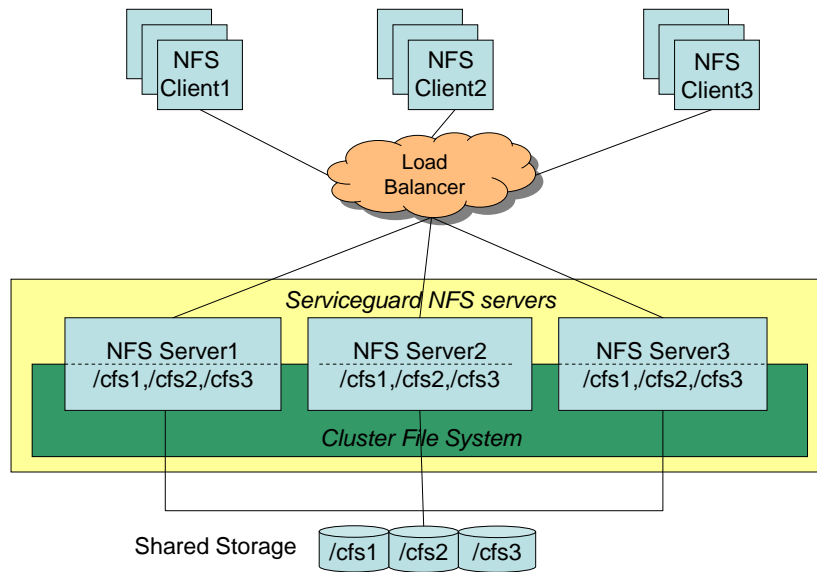


Figure 3. SG NFS Servers over CFS – High Availability, Scalability, Load Balancing

Issues and Limitations with the Current CFS Implementation

The main limitation with the current CFS implementation is that during package failover, an NFS client may lose a file lock in the following situation. If Client1 locks a CFS file on Server1, and Client2 attempts to lock the same CFS file on Server2, Client2 will wait for the lock to become available. If a failover occurs on Server1, Client1 will lose the file lock and Client2 will be granted the lock. If file locking is required for this situation, NFS failover packages must be used to export the CFS filesystems instead of multi-node export packages. Each cluster filesystem must only be exported by one node in the cluster. This configuration is similar to Serviceguard NFS over VxFS, as shown in [Figure 4](#), in that the file would just be accessible on one server and when that package fails over the file locks would be restored properly after the package restarts. One advantage this configuration has compared to the Serviceguard NFS over VxFS configuration is that the filesystems do not need to be unmounted and re-mounted during package failover, which should reduce failover time.

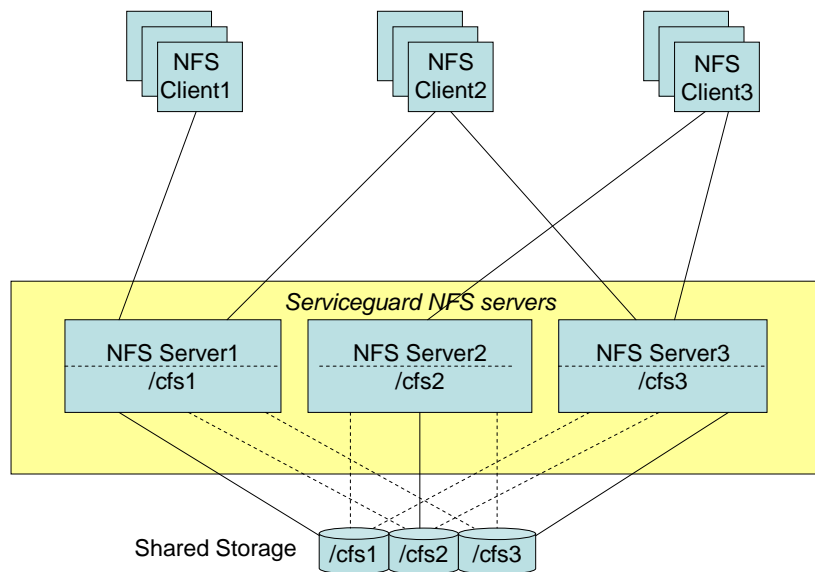


Figure 4. SG NFS Servers over CFS – High Availability, File Locking

5. Prerequisites

5.1 Product Requirements

Currently, Serviceguard NFS Toolkit support for CFS is only available on HP-UX 11i v2 (11.23). You must install Serviceguard A.11.18 (or above), CFS5.0, and Serviceguard NFS Toolkit B.11.23.06 (or above).

5.2 Patch Requirements

You must have the September 2004 11i v2 release installed which includes base patch PHKL_31500. It is recommended that you install the latest NFS (PHNE_35960), Kernel RPC (PHNE_35117), and Lock Manager (PHNE_35512) patches.

6. Configuring Serviceguard NFS over CFS Packages

This section describes how to configure and start Serviceguard NFS Toolkit packages in a Serviceguard over CFS environment. It is assumed that you have already setup your Serviceguard cluster, performed the steps listed in the “Before Creating a Serviceguard NFS Package” section of the [Serviceguard NFS Toolkit A.11.11.06 and A.11.23.05 Administrator's Guide](#) to configure the `/etc/rc.config.d/nfsconf` file, start the NFS server, configure the disk hardware, and setup volume groups, logical volumes, and file systems. It is assumed that you have setup CFS as documented in the [Veritas Storage Foundation Cluster File System Installation and Administration Guide](#), and that you have already started the Serviceguard CFS multi-node packages.

In the example that follows, the CFS file systems are `/cfs1` and `/cfs2`, and they correspond to Serviceguard CFS multi-node packages `SG-CFS-MP-1` and `SG-CFS-MP-2`. The cluster ASCII configuration file is assumed to be `/etc/cmcluster/cluster.conf`. The cluster name is `cluster1` and there are two nodes, `thyme` and `basil`. If you run the `cmviewcl` command after setting up the Serviceguard CFS packages, the output should show the following:

```
# cmviewcl
```

```
CLUSTER    STATUS  
cluster1   up
```

```
NODE STATUS    STATE  
thyme up        running
```

```
NODE STATUS    STATE  
basil  up        running
```

```
MULTI_NODE_PACKAGES
```

PACKAGE	STATUS	STATE	AUTO_RUN	SYSTEM
SG-CFS-pkg	up	running	enabled	yes
SG-CFS-DG-1	up	running	enabled	no
SG-CFS-MP-1	up	running	enabled	no
SG-CFS-MP-2	up	running	enabled	no

Create a directory on each server in the cluster to hold all of the configuration files (if this directory already exists you should save the contents before continuing):

```
# mkdir /etc/cmcluster/nfs
```

The rest of the configuration is dependent upon whether or not the cluster requires file locking (as described in the [Issues and Limitations with the current CFS implementation](#) section. If file locking is not required, follow the instructions in the next section "Serviceguard NFS over CFS Packages without File Locking". If file locking is required, follow the instructions in the section "[Serviceguard NFS over CFS Packages with File Locking](#)".

Note: Serviceguard A.11.18 introduces the concept of modular packages which allows packages to be created using building blocks that comprise only the functions that the package needs. It also includes some changes to the variable names in the package control and configuration files created with the `cmmakepkg` command. Creation of modular packages is not supported in this version of Serviceguard NFS Toolkit, but will be supported in a future version. The variable name changes in the package control and configuration files will be reflected in a future version of this white paper.

6.1 Serviceguard NFS over CFS Packages without File Locking

Each active server in the cluster needs to run an export multi-node package and an NFS failover package. An export multi-node package is a package that runs on each server in the cluster and exports all the cluster file systems. Each standby server (i.e. a server that is just an adoptive node for NFS failover packages) needs to have an export multi-node package running to be able to become active in the event of a failover.

[Figure 5](#) shows an example Serviceguard over CFS configuration with two servers, each with an NFS failover package and a multi-node export package.

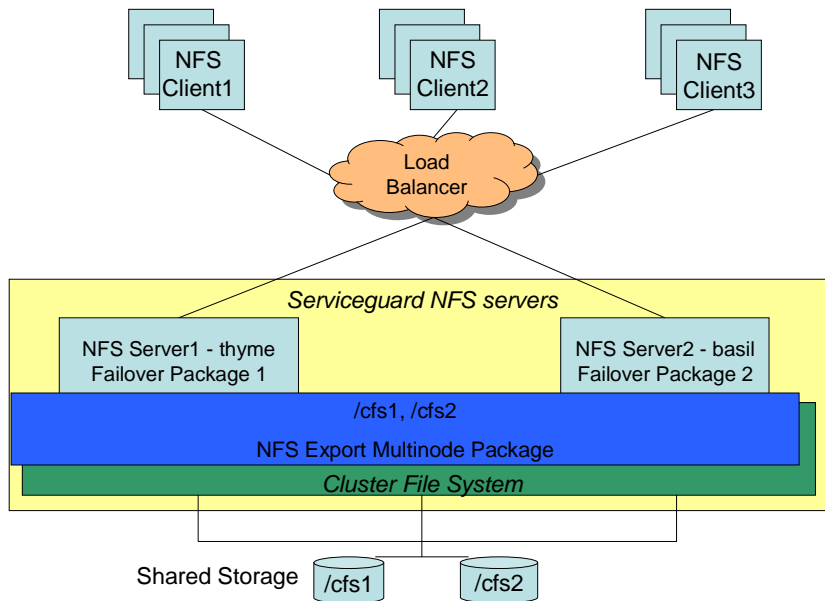


Figure 5. SG NFS over CFS without file locking

Configuring a Serviceguard NFS export package

There should only be one NFS export multi-node package per cluster. This package will run on each server in the cluster.

Note: All steps in this section should be done on a single server.

1. Create the `nfs-export.conf` and `nfs-export.cntl` files with the `cmmakepkg` command. Use the `-p` option to create the `nfs-export.conf` file and the `-s` option to create the `nfs-export.cntl` file.

```
# cd /etc/cmcluster/nfs
# cmmakepkg -p /etc/cmcluster/nfs/nfs-export.conf
# cmmakepkg -s /etc/cmcluster/nfs/nfs-export.cntl
```

2. Copy the `hanfs.sh` script:

```
# cp /opt/cmcluster/nfs/hanfs.sh hanfs.export.sh
```

3. Edit the `nfs-export.cntl` script and set the `HA_NFS_SCRIPT_EXTENSION` to `"export.sh"`. Note that when you edit the configuration scripts referred to throughout this document, you may have to uncomment the lines as you edit them.

```
HA_NFS_SCRIPT_EXTENSION = "export.sh"
```

This will set the NFS specific control script to be run by the package to `hanfs.export.sh` as we have named it in the copy command above. No other changes are needed in this script.

4. Edit the `hanfs.export.sh` script to specify each NFS directory to be exported. In this example these are the same as the CFS file systems that are mounted (you could have CFS file systems mounted and not export them, or export only a subdirectory of a mounted file system). Create a separate `XFS[n]` variable for

each directory to be exported. Specify the directory name and any export options.

These directories must not be included in the /etc/exports file:

```
XFS[0]="/cfs1"  
XFS[1]="/cfs2"
```

Caution:

Do not modify other variables or content in this script since doing so is not supported.

5. Edit the nfs-export.conf file as follows:

- a) Set the PACKAGE_NAME variable to SG-NFS-XP-1 (by default this variable is set to FAILOVER):

```
PACKAGE_NAME      SG-NFS-XP-1
```

- b) Change the PACKAGE_TYPE from FAILOVER to MULTI_NODE:

```
PACKAGE_TYPE      MULTI_NODE
```

- c) Comment out the FAILOVER_POLICY and FAILBACK_POLICY since this package will run on each server and will not failover if a server fails:

```
# FAILOVER_POLICY      CONFIGURED_NODE  
# FAILBACK_POLICY      MANUAL
```

- d) Add NODE_NAME variables for each server in the cluster that will run the export package (the order of the servers does not matter):

```
NODE_NAME thyme  
NODE_NAME basil
```

- e) Set the RUN_SCRIPT and HALT_SCRIPT variables to the full path of the control script, with no timeout:

```
RUN_SCRIPT /etc/cmcluster/nfs/nfs-export.cntl  
RUN_SCRIPT_TIMEOUT NO_TIMEOUT  
HALT_SCRIPT /etc/cmcluster/nfs/nfs-export.cntl  
HALT_SCRIPT_TIMEOUT NO_TIMEOUT
```

- f) Set the DEPENDENCY_NAME, DEPENDENCY_CONDITION, and DEPENDENCY_LOCATION variables so that the export package will only run if the Serviceguard CFS multi-node packages are already running. Since there are two CFS file systems in this example, there are also two CFS multi-node packages:

```
DEPENDENCY_NAME      SG-CFS-MP-1-dep  
DEPENDENCY_CONDITION SG-CFS-MP-1=UP  
DEPENDENCY_LOCATION  SAME_NODE
```

```
DEPENDENCY_NAME      SG-CFS-MP-2-dep  
DEPENDENCY_CONDITION SG-CFS-MP-2=UP  
DEPENDENCY_LOCATION  SAME_NODE
```

Note: The NFS export multi-node package does not monitor exported file systems (for example if a file system becomes unexported or inaccessible). It checks that the CFS dependent packages have access to the file systems. If the NFS failover package loses access and cannot read or write to the disk, it will fail (the exportfs multi-node package will not fail).

Starting a Serviceguard NFS export package

1. Copy the configuration files to each server in the cluster:

```
# rcp -p /etc/cmcluster/nfs/* {hostname}:/etc/cmcluster/nfs
```

2. Verify the cluster and package configuration files on each server:

```
# cmcheckconf -k -v -C /etc/cmcluster/cluster.conf -P /etc/cmcluster/nfs/nfs-export.conf
```

3. Verify and apply the cluster package configuration files on a single server:

```
# cmapplyconf -v -C /etc/cmcluster/cluster.conf -P /etc/cmcluster/nfs/nfs-export.conf
```

4. Run the export package on a single server with the command:

```
# cmrunpkg -v SG-NFS-XP-1
```

5. You can verify the export package is running with the cmviewcl command. The output should show the following:

```
# cmviewcl
```

```
CLUSTER    STATUS  
cluster1   up
```

```
NODE STATUS    STATE  
thyme up        running
```

```
NODE STATUS    STATE  
basil up        running
```

```
MULTI_NODE_PACKAGES
```

PACKAGE	STATUS	STATE	AUTO_RUN	SYSTEM
SG-CFS-pkg	up	running	enabled	yes
SG-CFS-DG-1	up	running	enabled	no
SG-CFS-MP-1	up	running	enabled	no
SG-CFS-MP-2	up	running	enabled	no
SG-NFS-XP-1	up	running	enabled	no

Configuring a Serviceguard NFS failover package

Configuring a Serviceguard NFS failover package for a CFS environment is similar to configuring the package for a non-CFS environment. The main difference is that you must configure one failover package for each server that exports CFS.

1. Copy the following scripts and make a separate copy for each package (one package for each server):

```
# cd /etc/cmcluster/nfs

# cmmakepkg -p /etc/cmcluster/nfs/nfs1.conf
# cmmakepkg -s /etc/cmcluster/nfs/nfs1.cntl
# cp /opt/cmcluster/nfs/hanfs.sh hanfs.1.sh
# cp /opt/cmcluster/nfs/nfs.mon nfs1.mon

# cmmakepkg -p /etc/cmcluster/nfs/nfs2.conf
# cmmakepkg -s /etc/cmcluster/nfs/nfs2.cntl
# cp /opt/cmcluster/nfs/hanfs.sh hanfs.2.sh
# cp /opt/cmcluster/nfs/nfs.mon nfs2.mon
```

2. Edit the nfs.cntl scripts (nfs1.cntl, nfs2.cntl). Specify the IP address for the package and the subnet to which the IP address belongs:

```
IP[0]=15.13.114.243
SUBNET[0]=15.13.112.0
```

This IP address is the relocatable IP address for the package. NFS clients that mount the file systems in the package will use this IP address to identify the server. You should configure a name for this address in the DNS, NIS, or LDAP database, or in the /etc/hosts file.

3. Set the HA_NFS_SCRIPT_EXTENSION to "1.sh" in the nfs1.cntl file:

```
HA_NFS_SCRIPT_EXTENSION = "1.sh"
```

This will set the NFS specific control script to be run by the package to hanfs.1.sh as we have named it in the copy command above. Set this to "2.sh" in nfs2.cntl.

4. Edit the hanfs.sh scripts (hanfs.1.sh and hanfs.2.sh) if you want to monitor NFS services (by running the NFS monitor script). To monitor NFS services, set the NFS_SERVICE_NAME and NFS_SERVICE_CMD variables:

```
NFS_SERVICE_NAME[0]=nfs1.monitor
NFS_SERVICE_CMD[0]=/etc/cmcluster/nfs/nfs1.mon
```

In hanfs.2.sh, set NFS_SERVICE_NAME[0] to nfs2.monitor and set NFS_SERVICE_CMD[0] to /etc/cmcluster/nfs/nfs2.mon. If you do not want to monitor NFS services, leave these variables commented out.

5. Edit the nfs.conf scripts (nfs1.conf and nfs2.conf) as follows:

- a) Specify the package name:

```
PACKAGE_NAME SG-NFS1
```

- b) In `nfs2.conf` set the `PACKAGE_NAME` to `SG-NFS2`.
- c) The `PACKAGE_TYPE` should be set to the default value (`FAILOVER`).
- d) Set the `NODE_NAME` variables for each node that can run the package. The first `NODE_NAME` should specify the primary node, followed by adoptive node(s) in the order in which they will be tried:

```
NODE_NAME thyme
NODE_NAME basil
```

- e) Set the `RUN_SCRIPT` and `HALT_SCRIPT` variables to the full path of the control script, with no timeout:

```
RUN_SCRIPT /etc/cmcluster/nfs/nfs1.cntl
RUN_SCRIPT_TIMEOUT NO_TIMEOUT
HALT_SCRIPT /etc/cmcluster/nfs/nfs1.cntl
HALT_SCRIPT_TIMEOUT NO_TIMEOUT
```

Similarly, set the `RUN_SCRIPT` and `HALT_SCRIPT` to `/etc/cmcluster/nfs/nfs2.cntl` in `nfs2.conf`.

- f) Set the `DEPENDENCY_NAME`, `DEPENDENCY_CONDITION`, and `DEPENDENCY_LOCATION` variables so that the failover package will only run if the export package is already running:

```
DEPENDENCY_NAME          SG-NFS-XP-1-dep
DEPENDENCY_CONDITION     SG-NFS-XP-1=UP
DEPENDENCY_LOCATION     SAME_NODE
```

- g) To monitor NFS services, set the `SERVICE_NAME` variable:

```
SERVICE_NAME nfs1.monitor
```

- h) In `nfs2.conf`, set `SERVICE_NAME` to `nfs2.monitor`. The `SERVICE_NAME` variable in the `nfs.conf` scripts must match the `NFS_SERVICE_NAME[0]` variable in the `hanfs.sh` scripts. If you do not want to monitor NFS services, leave this variable commented out.

- i) Set the `SUBNET` variable in both `nfs1.conf` and `nfs2.conf`:

```
SUBNET 15.13.112.0
```

Starting a Serviceguard NFS failover package

1. Copy the configuration files to each server in the cluster:

```
# rcp -p /etc/cmcluster/nfs/* {hostname}:/etc/cmcluster/nfs
```

2. Verify the cluster and package configuration files on each server:

```
# cmcheckconf -k -v -C /etc/cmcluster/cluster.conf -P
/etc/cmcluster/nfs/nfs1.conf -P /etc/cmcluster/nfs/nfs2.conf
```

3. Verify and apply the cluster package configuration files on a single server:

```
# cmapplyconf -v -C /etc/cmcluster/cluster.conf -P
/etc/cmcluster/nfs/nfs1.conf -P /etc/cmcluster/nfs/nfs2.conf
```

4. Run the failover packages:

```
# cmrunpkg -n thyme -v SG-NFS1
# cmrunpkg -n basil -v SG-NFS2
```

If you run the `cmviewcl` command, the output should show the following:

```
# cmviewcl
```

```
CLUSTER    STATUS
cluster1   up
```

```
NODE STATUS    STATE
thyme up        running
```

```
PACKAGE    STATUS    STATE    AUTO_RUN    NODE
SG-NFS1    up        running  enabled     thyme
```

```
NODE STATUS    STATE
basil      up        running
```

```
PACKAGE    STATUS    STATE    AUTO_RUN    NODE
SG-NFS2    up        running  enabled     basil
```

```
MULTI_NODE_PACKAGES
```

```
PACKAGE    STATUS    STATE    AUTO_RUN    SYSTEM
SG-CFS-pkg    up        running  enabled     yes
SG-CFS-DG-1    up        running  enabled     no
SG-CFS-MP-1    up        running  enabled     no
SG-CFS-MP-2    up        running  enabled     no
SG-NFS-XP-1    up        running  enabled     no
```

6.2 Serviceguard NFS over CFS Packages with File Locking

Serviceguard NFS over CFS packages requiring file locking are similar to Serviceguard NFS over VxFS packages. The main differences are that there will be no filesystems to mount in the `nfs.cntl` script (since the CFS multi-node packages already mount the CFS file systems), and there will be package dependencies added on the CFS packages.

[Figure 6](#) shows an example Serviceguard NFS over CFS configuration with two servers, each with an NFS failover package.

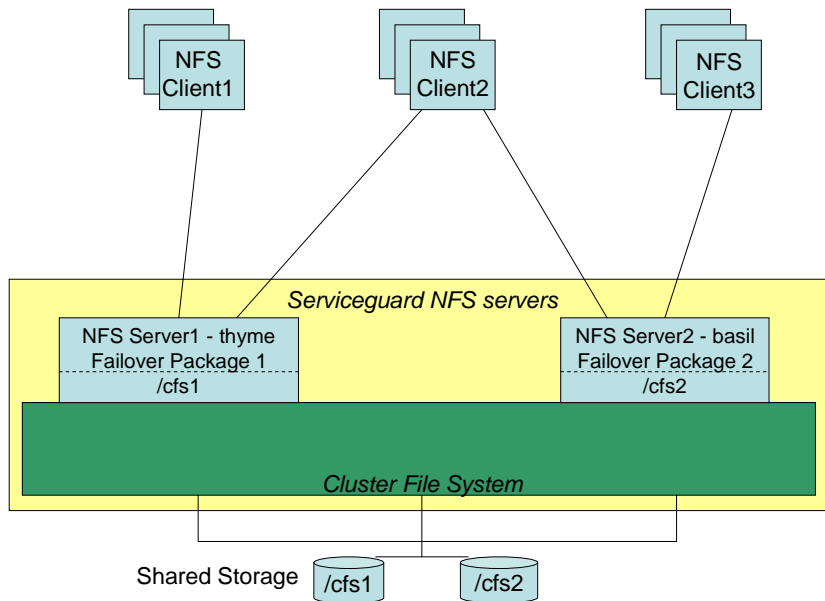


Figure 6. SG NFS over CFS with file locking

Configuring a Serviceguard NFS failover package

Configuring a Serviceguard NFS failover package for a CFS environment is similar to configuring the package for a non-CFS environment. The main difference is that you must configure one failover package for each server that exports CFS.

1. Copy the following scripts and make a separate copy for each package (one package for each server):

```
# cd /etc/cmcluster/nfs
# cmmakepkg -p /etc/cmcluster/nfs/nfs1.conf
# cmmakepkg -s /etc/cmcluster/nfs/nfs1.cntl
# cp /opt/cmcluster/nfs/hanfs.sh hanfs.1.sh
# cp /opt/cmcluster/nfs/nfs.mon nfs1.mon
# cp /opt/cmcluster/nfs/nfs.flm nfs1.flm
```

```
# cmmakepkg -p /etc/cmcluster/nfs/nfs2.conf
# cmmakepkg -s /etc/cmcluster/nfs/nfs2.cntl
# cp /opt/cmcluster/nfs/hanfs.sh hanfs.2.sh
# cp /opt/cmcluster/nfs/nfs.mon nfs2.mon
# cp /opt/cmcluster/nfs/nfs.flm nfs2.flm
```

2. Edit the nfs.cntl scripts (nfs1.cntl, nfs2.cntl). Do not modify the FILESYSTEMS section (leave it commented out), since the file systems are mounted and unmounted by the CFS packages. Specify the IP address for the package and the subnet to which the IP address belongs:

```
IP[0]=15.13.114.243
SUBNET[0]=15.13.112.0
```

This IP address is the relocatable IP address for the package. NFS clients that mount the file systems in the package will use this IP address to identify the

server. You should configure a name for this address in the DNS, NIS, or LDAP database, or in the /etc/hosts file.

3. Set the HA_NFS_SCRIPT_EXTENSION to "1.sh" in the nfs1.cntl file:

```
HA_NFS_SCRIPT_EXTENSION = "1.sh"
```

This will set the NFS specific control script to be run by the package to hanfs.1.sh as we have named it in the copy command above. Set this to "2.sh" in nfs2.cntl.

4. Edit the hanfs.sh scripts (hanfs.1.sh and hanfs.2.sh) as follows:

- a) Set the exported directory in hanfs.1.sh:

```
XFS[0]="/cfs1"
```

- b) Set XFS[0] to "/cfs2" in hanfs.2.sh.

- c) If you want to monitor NFS services (by running the NFS monitor script), set the NFS_SERVICE_NAME and NFS_SERVICE_CMD variables in hanfs.1.sh to:

```
NFS_SERVICE_NAME[0]=nfs1.monitor  
NFS_SERVICE_CMD[0]=/etc/cmcluster/nfs/nfs1.mon
```

- d) In hanfs.2.sh, set NFS_SERVICE_NAME[0] to nfs2.monitor and set NFS_SERVICE_CMD[0] to /etc/cmcluster/nfs/nfs2.mon. If you do not want to monitor NFS services, leave these variables commented out.

5. Edit the hanfs.sh script (hanfs.1.sh or hanfs.2.sh) if you want to enable file lock migration.

- a) To enable file lock migration in the first NFS package, set the NFS_FILE_LOCK_MIGRATION variable to 1 and specify the NFS_FLM_SCRIPT in hanfs.1.sh:

```
NFS_FILE_LOCK_MIGRATION=1  
NFS_FLM_SCRIPT="${0%/*}/nfs1.flm"
```

- b) If you want to enable file lock migration in the second NFS package, set the NFS_FLM_SCRIPT to "\${0%/*}/nfs2.flm" and the NFS_FILE_LOCK_MIGRATION variable to 1 in hanfs.2.sh.

- c) If you enable file lock migration and monitor NFS services, set the NFS_FILE_LOCK_MIGRATION and NFS_FLM_SCRIPT variables in the nfs.mon script (nfs1.mon or nfs2.mon) as they were set in the previous step:

```
NFS_FILE_LOCK_MIGRATION=1  
NFS_FLM_SCRIPT="${0%/*}/nfs1.flm"
```

- d) Edit the corresponding nfs.flm script(s) (nfs1.flm and/or nfs2.flm) to set the holding directory. For example in nfs1.flm:

```
NFS_FLM_HOLDING_DIR="/cfs1/sm"
```

and in nfs2.flm:

```
NFS_FLM_HOLDING_DIR="/cfs2/sm"
```

6. Edit the nfs.conf scripts (nfs1.conf and nfs2.conf) as follows:

a) Specify the package name:

```
PACKAGE_NAME SG-NFS1
```

b) In nfs2.conf set the PACKAGE_NAME to SG-NFS2.

c) Set the NODE_NAME variables for each node that can run the package. The first NODE_NAME should specify the primary node, followed by adoptive node(s) in the order in which they will be tried:

```
NODE_NAME thyme  
NODE_NAME basil
```

d) Set the RUN_SCRIPT and HALT_SCRIPT variables to the full path of the control script, with no timeout:

```
RUN_SCRIPT /etc/cmcluster/nfs/nfs1.cntl  
RUN_SCRIPT_TIMEOUT NO_TIMEOUT  
HALT_SCRIPT /etc/cmcluster/nfs/nfs1.cntl  
HALT_SCRIPT_TIMEOUT NO_TIMEOUT
```

Similarly, set the RUN_SCRIPT and HALT_SCRIPT to /etc/cmcluster/nfs/nfs2.cntl in nfs2.conf.

e) Set the DEPENDENCY_NAME, DEPENDENCY CONDITION, and DEPENDENCY_LOCATION variables so that the failover package will only run if the corresponding CFS package is already running. In nfs1.conf set these to:

```
DEPENDENCY_NAME          SG-CFS-MP-1-dep  
DEPENDENCY_CONDITION     SG-CFS-MP-1=UP  
DEPENDENCY_LOCATION      SAME_NODE
```

In nfs2.conf set these to:

```
DEPENDENCY_NAME          SG-CFS-MP-2-dep  
DEPENDENCY_CONDITION     SG-CFS-MP-2=UP  
DEPENDENCY_LOCATION      SAME_NODE
```

f) To monitor NFS services, set the SERVICE_NAME variable in nfs1.conf:

```
SERVICE_NAME nfs1.monitor
```

In nfs2.conf, set SERVICE_NAME to nfs2.monitor. The SERVICE_NAME variable in the nfs.conf scripts must match the NFS_SERVICE_NAME[0] variable in the hanfs.sh scripts. If you do not want to monitor NFS services, leave this variable commented out.

g) Set the SUBNET variable in both nfs1.conf and nfs2.conf:

```
SUBNET 15.13.112.0
```

Starting a Serviceguard NFS failover package

1. Copy the configuration files to each server in the cluster:

```
# rcp -p /etc/cmcluster/nfs/* {hostname}:/etc/cmcluster/nfs
```

2. Verify the cluster and package configuration files on each server:

```
# cmcheckconf -k -v -C /etc/cmcluster/cluster.conf -P  
/etc/cmcluster/nfs/nfs1.conf -P /etc/cmcluster/nfs/nfs2.conf
```

3. Verify and apply the cluster package configuration files on a single server:

```
# cmapplyconf -v -C /etc/cmcluster/cluster.conf -P  
/etc/cmcluster/nfs/nfs1.conf -P /etc/cmcluster/nfs/nfs2.conf
```

4. Run the failover packages:

```
# cmrunpkg -n thyme -v SG-NFS1  
# cmrunpkg -n basil -v SG-NFS2
```

If you run the `cmviewcl` command, the output should show the following:

```
# cmviewcl
```

```
CLUSTER    STATUS  
cluster1   up
```

```
NODE STATUS    STATE  
thyme up        running
```

```
PACKAGE    STATUS    STATE    AUTO_RUN    NODE  
SG-NFS1    up        running  enabled     thyme
```

```
NODE STATUS    STATE  
basil      up        running
```

```
PACKAGE    STATUS    STATE    AUTO_RUN    NODE  
SG-NFS2    up        running  enabled     basil
```

```
MULTI_NODE_PACKAGES
```

```
PACKAGE          STATUS    STATE    AUTO_RUN    SYSTEM  
SG-CFS-pkg       up        running  enabled     yes  
SG-CFS-DG-1      up        running  enabled     no  
SG-CFS-MP-1      up        running  enabled     no  
SG-CFS-MP-2      up        running  enabled     no
```