

Service Guard Cluster

Ressources

- go/ha (<http://www.hp.com/go/ha>)

Packaging

```
Package Group
└-> Package
    -> Disk resource
    -> Network resources
    -> Service process
```

Check if Servers are Identical

1. Check system model

```
# model
```

2. Check OS revision

```
# uname -r
# swlist | grep OE
```

3. Memory

```
# machinfo | grep Mem
```

4. Date / Time / Timezone

```
..
```

```
# date
# cat /etc/default/tz
```

5. Service Guard revision

```
# swlist -l product | grep -i guard
```

Prepare network

Hint: All lan interfaces must be full duplex switched on.

1. Configure the network interfaces (on both nodes)

```
# vi /etc/rc.config.d/netconf
INTERFACE_NAME[2]="lan2"
IP_ADDRESS[2]="10.10.2.xx"
SUBNET_MASK[2]="255.255.255.0"
BROADCAST_ADDRESS[2]="10.21.2.255"
INTERFACE_STATE[2]="up"
DHCP_ENABLE[2]="0"
INTERFACE_MODULES[2]=""
```

```
INTERFACE_NAME[3]="lan3"
IP_ADDRESS[3]="10.21.2.xx"
SUBNET_MASK[3]="255.255.255.0"
BROADCAST_ADDRESS[3]="10.21.2.255"
INTERFACE_STATE[3]="up"
DHCP_ENABLE[3]="0"
INTERFACE_MODULES[3]=""
```

2. Activate the configuration

```
# /sbin/init.d/net stop
# /sbin/init.d/net start
```

3. Verify the configuration

```
# netstat -in
Name      Mtu  Network      Address          Ipkts           Ierrs Opkts          Oerrs C
lan3      1500 10.21.2.0    10.21.2.xx      3               0      3             0     0
lan2      1500 10.10.2.0    10.10.2.xx      2               0      2             0     0
lan1      1500 10.99.185.0  10.99.185.xx   9473            0      64            0     0
lan0      1500 192.168.0.0  192.168.185.xx 110641          0     110368        0     0
lo0       32808 127.0.0.0    127.0.0.1      20600           0     20600         0     0
```

4. Put each other hostname to the roots .rhosts file

```
# vi ~/.rhosts
other-host-name
```

5. Put each other hostname to the /etc/hosts file

```
# vi /etc/hosts
own-ip-address      own-host-name
other-ip-address    other-host-name
```

6. Verify SG is installed

```
# swlist -l product | grep -i guard
ASGV7                A.02.00          HP Serviceguard Cluster File System for RAC
PHSS_38861           1.0              Serviceguard A.11.18.00
SG-Apache-Tool       B.04.01          Serviceguard Apache Script Templates
SG-NFS-Tool          A.11.31.02      Serviceguard NFS Script Templates
SG-NMAPI             A.11.18.00      SG-NMAPI Serviceguard Extension for RAC SD product
SG-Oracle-Tool       B.04.01          Serviceguard Oracle Script Templates
SG-RAC-Tool          A.11.18.00      Serviceguard Extension for RAC Toolkit SD Product
SG-Samba-Tool        B.04.01          Serviceguard Samba Script Templates
SG-Tomcat-Tool       B.04.01          Serviceguard Tomcat Script Templates
SGManagerPI          B.01.01          HP Serviceguard Manager
SGWBEMProviders      A.02.00.00      HP Serviceguard WBEM Providers SD Product
ServiceGuard         A.11.18.00      Serviceguard SD Product
h6487s               N.02             Serviceguard I
```

7. Check if all LAN cards are physically attached to the same bridged-subnet

```
# linkloop -i N mac-address_of_1st_lan_card_own_server
# linkloop -i N mac-address_of_2nd_lan_card_own_server
# linkloop -i N mac-address_of_3rd_lan_card_own_server
# linkloop -i N mac-address_of_4th_lan_card_own_server
```

```
# linkloop -i N mac-address_of_1st_lan_card_other_server
# linkloop -i N mac-address_of_2nd_lan_card_other_server
# linkloop -i N mac-address_of_3rd_lan_card_other_server
# linkloop -i N mac-address_of_4th_lan_card_other_server
```

8. Identify the sharable SCSI cards or shareable bus adapters

```
# ioscan -fC ctl
# ioscan -fC ba
```

9. HW Path of SCSI cards or bus adapters

```
# ioscan -fC ext_bus
# ioscan -fC ba
```

10. HW address of SCSI cards or bus adapters

```
# ioscan -fC ctl
# ioscan -fC ext_bus
```

11. HW Path of disk

```
# ioscan -fC disk
```

12. Device file name of sharable disks

```
# ioscan -fnC disk
```

13. What volume groups currently exists and what disks are currently included in existing volume groups

```
# strings /etc/lvmtab
# vgdisplay -v
```

14. What disks are used by VXVM ore LVM

```
# vxdisk -o alldgs list
```

If you get an error here please run the command:

```
# vxinstall
Are you prepared to enter a license key [y,n,q,?] (default: n) n
Do you want to use enclosure based names for all disks ? [y,n,q,?] (default: n) n
Do you want to setup a system wide default disk group? [y,n,q,?] (default: y) n
# ps ef |grep vx
```

Mirror the integrity boot disk

1. Determine the current boot disk

```
# lvinboot -b
```

2. Identify a large enough disk to become a mirror of the boot disk

\$

```
# ioscan -funC disk
```

3. Find out idf of current boot disk

```
--> TODO
```

4. Create a 500MB system partition, an OS partition, and a 400 MB service partition on the new boot disk

```
# vi /tmp/idf
3
EFI 500MB
HPUX 100%
HPSP 400MB
# idisk -wf /tmp/idf /dev/rdisk/disk53
idisk version: 1.44
***** WARNING *****
If you continue you may destroy all data on this disk.
Do you wish to continue(yes/no)? yes
...
Legacy MBR (MBR Signatures in little endian):
MBR Signature = 0x8cf6d3ed
```

```
Protective MBR
```

5. Create device files for the new partitions

```
# insf -eC disk
```

6. Populate the /efi/hpux directory in the new EFI partition

```
# mkboot -e -l /dev/rdisk/disk53
```

7. Change the boot string on both boot disks so you can boot w/o quorum

```
# vi /tmp/auto
boot vmunix -lq
# efi_cp -d /dev/rdisk/disk52_p1 /tmp/auto /efi/hpux/auto
# efi_cp -d /dev/rdisk/disk53_p1 /tmp/auto /efi/hpux/auto
```

8. Initialize and add the new OS partition to vg00

```
# pvcreate -fB /dev/rdisk/disk53_p2
# vgextend vg00 /dev/disk/disk53_p2
# lvolboot -v
```

9. Mirror all logical volumes in vg00

```
# for i in /dev/vg00/lvol*
> do
> lvextend -m 1 $i
> done
```

10. Add a new line to the /stand/bootconf file so that the SDUX knows which disks are boot disks

```
# vi /stand/bootconf
l /dev/disk/disk52_p2
l /dev/disk/disk53_p2
```

11. Set the mirror disk as the ha boot device

```
# setboot -h /dev/rdisk/disk53
```

12. Test the new mirror

```
# shutdown -ry 0
```

Select HP-UX HA Alternate Boot

```
# grep "Boot device's HP-UX HW path" /var/adm/syslog/syslog.log
vmunix: Boot device's HP-UX HW path is: 0.0.0.0.1.0
```

Create Volume Group for Service Guard

1. Make a directory for the new volume group called vg01

```
# mkdir /dev/vg01
```

2. Create a control file for the new volume group

```
# mknod /dev/vg01/group c 64 0x010000
```

3. Prepare the disks to be used within the new volume group

```
# pvcreate /dev/rdisk/disk54
Physical volume "/dev/rdisk/disk54" has been successfully created.
# pvcreate /dev/rdisk/disk55
Physical volume "/dev/rdisk/disk55" has been successfully created.
```

or

```
# vxdisksetup -i disk54
# vxdisksetup -i disk55
```

4. Create the new volume group

```
# vgcreate vg01 /dev/disk/disk54 /dev/disk/disk55
Increased the number of physical extents per physical volume to 2559.
Volume group "/dev/vg01" has been successfully created.
Volume Group configuration for /dev/vg01 has been saved in /etc/lvmconf/vg01.conf
```

or

```
# vxdg init datadg datadg01=disk54 datadg02=disk55
```

5. Create a 200MB logical volume named sglvol1

```
# lvcreate -L 200 -n sglvol1 vg01
Logical volume "/dev/vg01/sglvol1" has been successfully created with
character device "/dev/vg01/rsglvol1".
Logical volume "/dev/vg01/sglvol1" has been successfully extended.
Volume Group configuration for /dev/vg01 has been saved in /etc/lvmconf/vg01.conf
```

or

```
# vxassist -g datadg make datavol 200m
```

6. Mirror the new sglvol1 logical volume

```
# lvextend -m 1 /dev/vg01/sglvoll
The newly allocated mirrors are now being synchronized. This operation will
take some time. Please wait ...
Logical volume "/dev/vg01/sglvoll" has been successfully extended.
Volume Group configuration for /dev/vg01 has been saved in /etc/lvmconf/vg01.conf
```

7. Create a journaled file system

```
# newfs -F vxfs /dev/vg01/rsglvoll
version 7 layout
204800 sectors, 204800 blocks of size 1024, log size 1024 blocks
largefiles supported
```

or

```
# newfs -F vxfs /dev/vx/rdisk/datadg/datavol
```

8. Create a mount point and mount the logical volume

```
# mkdir /sgdata1
# mount -F vxfs /dev/vg01/sglvoll /sgdata1
```

or

```
# mount -F vxfs /dev/vx/rdisk/datadg/datavol /sgdata1
```

9. Export the LVM configuration

```
# vgexport -v -p -s -m /tmp/vg01.map vg01
```

or

```
# umount /sgdata1
# vxdg deport datadg
```

10. Copy the configuration (map-file) to the node2

```
# rcp /tmp/vg01.map node2:/tmp/.
```

On **node2**:

11. Make a directory for the new volume group called vg01

```
# mkdir /dev/vg01
```

12. Create a control file for the new volume group

```
# mknod /dev/vg01/group c 64 0x010000
```

13. Import the volume group

```
# vgimport -N -v -s -m /tmp/vg01.map vg01
Beginning the import process on Volume Group "vg01".
Logical volume "/dev/vg01/sglvoll" has been successfully created
with lv number 1.
vgimport: Volume group "/dev/vg01" has been successfully created.
Warning: A backup of this volume group may not exist on this machine.
Please remember to take a backup using the vgcfgbackup command after activating the volume group.
```

or

```
# vxdg -tfc import datadg
```

14. Create a mount point, activate the volume group, mount it read-only

```
# mkdir /sgdata1
# vgchange -a y vg01
Activated volume group
Volume group "vg01" has been successfully changed.
# vgcfgbackup vg01
# mount -F vxfs -o ro /dev/vg01/sglvoll /sgdata1
# umount /sgdata1/
```

or

```
# vxrecover -g datadg -s
# mount -F vxfs -o ro /dev/vx/dsk/datadg/datavol /sgdata1
```

15. Deactivate the volume group

```
# vgchange -a n vg01
Volume group "vg01" has been successfully changed.
```

or

```
# vxdg deport datadg
```

Configure Cluster

1. Deactivate AUTO_VG_ACTIVATE

```
# vi /etc/lvmrc
AUTO_VG_ACTIVATE=0
```

2. Edit the node list

```
# vi /etc/cmcluster/cmclnodelist
node1 root
node2 root
# chmod 444 /etc/cmcluster/cmclnodelist
# rcp /etc/cmcluster/cmclnodelist node2:/etc/cmcluster/.
```

3. Create a clusterwide ASCII configuration file

```
# cd /etc/cmcluster
# cmquerycl -v -C cmclconf.ascii -n node1 -n node2
```

4. Edit the configuration file and edit the following parameters

```

# vi cmclconf.ascii
CLUSTER_NAME          cluster1

# QS_HOST qs_host
# QS_ADDR qs_addr
# QS_POLLING_INTERVAL 12000000
# QS_TIMEOUT_EXTENSION 2000000
FIRST_CLUSTER_LOCK_VG      /dev/vglock

NODE_NAME              rx16-144
NETWORK_INTERFACE      lan0
  STATIONARY_IP        192.168.185.49
NETWORK_INTERFACE      lan1
  STATIONARY_IP        10.99.185.49
NETWORK_INTERFACE      lan2
  HEARTBEAT_IP        10.10.2.49
NETWORK_INTERFACE      lan3
# CLUSTER_LOCK_LUN
FIRST_CLUSTER_LOCK_PV  /dev/disk/disk59

NODE_NAME              rx16-145
NETWORK_INTERFACE      lan0
  STATIONARY_IP        192.168.185.51
NETWORK_INTERFACE      lan1
  STATIONARY_IP        10.99.185.51
NETWORK_INTERFACE      lan2
  HEARTBEAT_IP        10.10.2.51
NETWORK_INTERFACE      lan3
# CLUSTER_LOCK_LUN
FIRST_CLUSTER_LOCK_PV  /dev/disk/disk59

HEARTBEAT_INTERVAL     1000000
NODE_TIMEOUT           2000000

AUTO_START_TIMEOUT     600000000
NETWORK_POLLING_INTERVAL 2000000

NETWORK_FAILURE_DETECTION  INOUT

MAX_CONFIGURED_PACKAGES 150

# USER_NAME  john
# USER_HOST  noir
# USER_ROLE  FULL_ADMIN

# VOLUME_GROUP      /dev/vgdatabase
# VOLUME_GROUP      /dev/vg02
VOLUME_GROUP        /dev/vglock

# OPS_VOLUME_GROUP  /dev/vgdatabase
# OPS_VOLUME_GROUP  /dev/vg02

OPS_VOLUME_GROUP    /dev/vg01

```

5. Add a non-root user who can execute certain Serviceguard commands

```

# vi cmclconf.ascii
USER_NAME      oracle
USER_HOST      CLUSTER_MEMBER_NODE
USER_ROLE      MONITOR

```

6. Check and apply the configuration

```

# cmcheckconf -v -C cmclconf.ascii
# cmapplyconf -v -C cmclconf.ascii

```

7. Start the cluster

```

# cmruncl -v

```

8. Verify the cluster

```

# cmviewcl -v [-f line]

```

9. Mark the volumegroup as **cluster-aware**

```
# vgchange -a n vg01 (on both nodes!)  
# vgchange -c y vg01
```

10. Configure both nodes to automatically join the cluster

```
# cd /etc/rc.config.d  
# vi cmcluster  
AUTOSTART_CMCLD=1
```

Cluster Access Control Policies

```
USER_NAME      ANY_USER or a list of specific users  
USER_HOST      ANY_SERVICEGUARD_NODE or CLUSTER_MEMBER_NODE or a specific node name  
USER_ROLE      MONITOR or FULL_ADMIN or PACKAGE_ADMIN (Package admin for all packages!)
```

Package configuration procedure

1. Create directory for package specific files

```
# mkdir /etc/cmcluster/package_name
```

2. Change to the directory

```
# cd /etc/cmcluster/package_name
```

3. Create package configuration template

```
# cmmakepkg -v -p pkg.conf
```

4. Modify package parameters

```
# vi pkg.conf
```

5. Create package control file template

```
# cmmakepkg -v -s pkg.cntl
```

6. Modify script variables

```
# vi pkg.cntl
```

7. Distribute the control script to all nodes

```
# rcp pkg.cntl nodex:/etc/cmcluster/package_name/.
```

8. Verify the package configuration file

```
# cmcheckconf -v -P /etc/cmcluster/package_name/pkg.conf
```

9. Compile package into cluster binary file

```
# cmapplyconf -v -P /etc/cmcluster/package_name/pkg.conf
```

Package Access Control Policies

```
USER_NAME    ANY_USER or a list of specific users
USER_HOST    ANY_SERVICEGUARD_NODE or CLUSTER_MEMBER_NODE or a specific node name
USER_ROLE    PACKAGE_ADMIN (Package admin for this package only!)
```

Create example package

1. As Example use the logprog application on both nodes

```
# cc logprog.c -o /usr/sbin/logprog
# rcp /usr/sbin/logprog node2:/usr/sbin/logprog
```

2. Make a directory for the logprog package

```
# cd /etc/cmcluster
# mkdir logprog
```

3. Create the package configuration template

```
# cd logprog
# cmmakepkg -v -p logprog.conf
```

4. Edit the package configuration file

```
# vi logprog.conf
PACKAGE_NAME    logprog
#PACKAGE_TYPE    FAILOVER
NODE_NAME       rx16-144
NODE_NAME       rx16-145
#AUTO_RUN       YES
#NODE_FAIL_FAST_ENABLED    NO
RUN_SCRIPT      /etc/cmcluster/logprog/logprog.cnt1
HALT_SCRIPT     /etc/cmcluster/logprog/logprog.cnt1
RUN_SCRIPT_TIMEOUT    NO_TIMEOUT
HALT_SCRIPT_TIMEOUT    NO_TIMEOUT
#SUCCESSOR_HALT_TIMEOUT    NO_TIMEOUT
#SCRIPT_LOG_FILE
#FAILOVER_POLICY    CONFIGURED_NODE
#FAILBACK_POLICY    MANUAL
#PRIORITY           NO_PRIORITY
#DEPENDENCY_NAME
#DEPENDENCY_CONDITION
#DEPENDENCY_LOCATION
#LOCAL_LAN_FAILOVER_ALLOWED    YES
#MONITORED_SUBNET
#MONITORED_SUBNET_ACCESS
#CLUSTER_INTERCONNECT_SUBNET
SERVICE_NAME    logprog_service
#SERVICE_FAIL_FAST_ENABLED
#SERVICE_HALT_TIMEOUT
#RESOURCE_NAME
#RESOURCE_POLLING_INTERVAL
#RESOURCE_START
#RESOURCE_UP_VALUE
#STORAGE_GROUP
#USER_NAME
#USER_HOST
#USER_ROLE
SUBNET           10.10.2.0
```

5. Create the package control template

```
# cmmakepkg -v -s logprog.cntl
```

6. Edit the package control template

```
# vi logprog.cntl
. ${SGCONFFILE:=/etc/cmcluster.conf}
PATH=$SGSBIN:/usr/bin:/usr/sbin:/etc:/bin
VGCHANGE="vgchange -a e" # Default
CVM_ACTIVATION_CMD="vxvg -g \${DiskGroup} set activation=exclusivewrite"
#VG[0]=""
#CVM_DG[0]=""
#VXVM_DG[0]=""
VXVM_DG_RETRY="NO"
DEACTIVATION_RETRY_COUNT=2
KILL_PROCESSES_ACCESSING_RAW_DEVICES="NO"
#LV[0]=""; FS[0]=""; FS_MOUNT_OPT[0]=""; FS_UMOUNT_OPT[0]=""; FS_FSCK_OPT[0]=""
#FS_TYPE[0]=""
VXVOL="vxvol -g \${DiskGroup} startall" # Default
FS_UMOUNT_COUNT=1
FS_MOUNT_RETRY_COUNT=0
CONCURRENT_VGCHANGE_OPERATIONS=1
ENABLE_THREADED_VGCHANGE=0
CONCURRENT_FSCK_OPERATIONS=1
CONCURRENT_MOUNT_AND_UMOUNT_OPERATIONS=1
#IP[0]=""
#SUBNET[0]=""
SERVICE_NAME[0]="logprog_service"
SERVICE_CMD[0]="/usr/sbin/logprog"
SERVICE_RESTART[0]="-r 2"
#DEFERRED_RESOURCE_NAME[0]=""
#DTC_NAME[0]=
#HA_NFS_SCRIPT_EXTENSION=""
log_file=${SG_SCRIPT_LOG_FILE:-$0.log}
...
```

7. Adjust file mode

```
# chmod 755 logprog.cntl
```

8. On node2 make the package directory

```
# mkdir /etc/cmcluster/logprog
```

9. Copy the file to the other node

```
# rcp logprog.cntl node2:/etc/cmcluster/logprog/.
```

10. Check the package configuration

```
# cmcheckconf -v -P logprog.conf
Begin package verification...
Checking existing configuration ... Done
Parsing package file: logprog.conf.
logprog.conf: A legacy package is being used.
Attempting to add package logprog.
logprog.conf:0: SERVICE_HALT_TIMEOUT value of 0 is equivalent to 1 sec.
WARNING: Incorrect permissions for /etc/cmcluster/logprog (40777). Directory must be executable for
Maximum configured packages parameter is 150.
Configuring 1 package(s).
149 package(s) can be added to this cluster.
200 access policies can be added to this cluster.
Adding the package configuration for package logprog.
cmcheckconf: Verification completed with no errors found.
Use the cmapplyconf command to apply the configuration.
```

11. Apply the new configuration

```
# cmapplyconf -v -P logprog.conf
Begin package verification...
Checking existing configuration ... Done
Parsing package file: logprog.conf.
logprog.conf: A legacy package is being used.
Attempting to add package logprog.
logprog.conf:0: SERVICE_HALT_TIMEOUT value of 0 is equivalent to 1 sec.
WARNING: Incorrect permissions for /etc/cmcluster/logprog (40777). Directory must be executable fo
Maximum configured packages parameter is 150.
Configuring 1 package(s).
149 package(s) can be added to this cluster.
200 access policies can be added to this cluster.
Adding the package configuration for package logprog.

Modify the package configuration ([y]/n)? y
Completed the cluster update
```

12. View the package status

```
# cmviewcl -v
# cmviewcl -vp logprog
```

13. Start the package

```
# cmmodpkg -e logprog
```

14. Test the package

```
# cmhaltpkg -v logprog
# cmviewcl -vp logprog
# cmrunpkg -v -n nodex logprog
# cmmodpkg -v -e logprog
# cmviewcl -vp logprog
```

15. Failover test

```
# cmviewcl -vp logprog
PACKAGE      STATUS      STATE      AUTO_RUN      NODE
logprog      up          running    enabled        node1
...
# pkill logprog (about 3 times)
# cmmodpkg -v -e -n node2 logprog
# cmviewcl -vp logprog
PACKAGE      STATUS      STATE      AUTO_RUN      NODE
logprog      up          running    enabled        node2
...
```

Cluster and package online configuration

Add a node while a cluster is running

1. Update `cmclnodelist` on all nodes in the cluster
2. Get the current cluster configuration and save it to a temp ascii file

```
# cmgetconf -v -c cluster1 temp.ascii
```

3. Create a new config file including all nodes

```
# cmquerycl -v -n n1 -n n2 -n n3 -n n4 -C cmclconfig.ascii
```

4. Combine the two configuration files into one.
5. Check and apply the configuration

```
# cmcheckconf -v -C cmclconfig.ascii
# cmapplyconf -v -C cmclconfig.ascii
```

6. Run the new node

```
# cmrunnode -v n4
```

7. Modify package files to reflect new node

8. If using a quorum server, update the `qs_authfile`

Remove a node while a cluster is running

1. Halt all packages on the node

```
# cmhaltpkg -v pkg_name
```

2. Remove any references to the node from all pkg config files

3. Halt the node

```
# cmhaltnode -v node4
```

4. Edit the cluster config file and remove node4's definition

5. Check and apply configuration for the node

```
# cmcheckconf -v -C cmclconfig.ascii
# cmapplyconf -v -C cmclconfig.ascii
```

6. Check and apply configuration for all packages

```
# cmcheckconf -v -P pkg_name.conf
# cmapplyconf -v -P pkg_name.conf
```

Add a package while a cluster is running

```
# cmmake -v -p pkgB.conf and edit as usual
# cmmake -v -s pkgB.cntl and edit as usual
# chmod +x pkgB.cntl
# rcp pkgB.cntl node2:/etc/cmcluster/pkgB/pkgB.cntl
# cmcheckconf -v -P pkgB.conf
# cmapplyconf -v -P pkgB.conf
# cmmodpkg -v -e pkgB
```

Remove a package while a cluster is running

```
# cmhaltpkg -v pkgB
# cmdeleteconf -v -p pkgB
(optional) # rm pkgB.conf
(optional) # rm pkgB.cntl
```

Storage maintenance

Increasing the size of a logical volume / file system

```
# lvextend -L xxx /dev/vg01/sglvoll
# fsadm -F vxfs -b xxxM /sgdata1
```

or

```
# vxassist -g datadg growto datavol xxm
# fsadm -F vxfs -b xxxM /sgdata1
```

→ xxx is in MB

Add a disk to a volume group

1. Halt the package that contains the volume group

```
# cmhaltpkg -v pkg_name
```

for SG version 11.19 no longer necessary!

2. Make a note of a unused disk

```
# ioscan -funC disk
# strings /etc/lvmtab
```

3. Make a note of the disks currently included in the volume group

```
# vgdisplay -v vg01
```

or

```
# vxdisk -o alldgs list
```

4. Set the volume group to exclusive mode

```
# vgchange -a e vg01
```

for SG version 11.19 no longer necessary!

5. Add the unused disk to the volume group

```
# pvcreate -f /dev/rdisk/diskXX
# vgextend vg01 /dev/disk/diskXX
```

or

```
# vxdisksetup -i diskXX
# vxdg -g datadg adddisk datadg04=diskXX
```

6. Re-vgimport the vg01 on the node2

```

node1# vgexport -p -s -m /tmp/vg01.map vg01
node1# vgchange -a n vg01
node2# vgexport vg01
node2# rcp node1:/tmp/vg01.map /tmp/vg01.map
node2# mkdir /dev/vg01
node2# mknod /dev/vg01/group c 64 0x010000      (0x010000 MUST be the same as on node1!!)
node2# vgimport -N -v -s -m /tmp/vg01.map /dev/vg01
node2# vgchange -a r vg01
node2# vgcfgbackup vg01
node2# vgchange -a n vg01

```

7. Reenable the package that contains the volume group

```

# cmmmodpkg -e -n node1 pkg_name
# cmmmodpkg -e pkg_name

```

Add a logical volume / filesystem to volume group

```

node1# cmhaltpkg -v pkg_name
node1# vgchange -a e vg01
node1# lvcreate -L size -n lvol_name /dev/vg01
node1# newfs -F vxfs /dev/vg01/rlvol_name
node1# mkdir /new_mount_point
node1# vgexport -v -s -p -m /tmp/vg01.map vg01
node1# vgchange -an vg01
node2# rcp node1:/tmp/vg01.map /tmp/.
node2# mkdir /new_mount_point
node2# vgexport vg01
node2# mkdir /dev/vg01
node2# mknod /dev/vg01/group c 64 0x010000
node2# vgimport -N -v -s -m /tmp/vg01.map vg01
node2# vgchange -a r vg01
node2# vgcfgbackup vg01
node2# vgchnge -a n vg01

```

Add a volume group to a package

```

node1# pvcreate [ -f ] /dev/disk/new_disk1
node1# pvcreate [ -f ] /dev/disk/new_disk2
node1# mkdir /dev/vg_new
node1# mknod /dev/vg_new/group c 64 0x140000
node1# vgcreate vg_new /dev/disk/new_disk1 /dev/disk/new_disk2
node1# vgexport -v -p -s -m /tmp/vg_new.map vg_new
node2# rcp node1:/tmp/vg_new.map /tmp/.
node2# mkdir /dev/vg_new
node2# mknod /dev/vg_new/group c 64 0x140000
node2# vgimport -N -v -s -m /tmp/vg_new.map vg_new
node1# vgchange -c y vg_new
node1# vgchange -a n
node1# cmhaltpkg pkg_name

```

Modify the control scripts on both systems to insure this new vg is activated when the package starts and to insure all new filesystems in this nre volume group are mounted.

```

node1# cmrunpkg -v pkg_name
node1# cmmmodpkg -v -e pkg_name

```

Move a physical volume in a cluster volume group

- No need to halt the package
- Do it on the node on which the vg is activated, exclusive

1. Extend the vg with the new physical volume

```
# vgextend vg02 /dev/disk/disk21
```

2. `pvmmove` the old physical volume to the new one

```
# pvmmove /dev/disk/disk20 /dev/disk/disk21
```

3. Remove the old physical volume from the volume group

```
# vgreduce vg02 /dev/disk/disk20
```

4. Export / Import the vg

```
# vgexport -v -p -s -m /tmp/vg02.map vg02  
# scp /tmp/vg02.map other_node:/tmp/.  
# ssh other_node "vgexport vg02"  
# ssh other_node "vgimport -N -v -s -m /tmp/vg02.map vg02"
```

5. Create a config backup

```
# ssh other_node "vgchange -a r vg02"  
# ssh other_node "vgcfgbackup vg02"  
# ssh other_node "vgchange -a n vg02"
```

Move a physical volume in a cluster volume group over a mirror

- Considering the example of VG01 having few existing LUN's with few lv's on it.

1. Assign the new LUN's
2. Add the LUN's to existing vg VG01

```
# pvcreate <new pv paths>  
# vgextend vg01 <new pv's>
```

3. Extend the existing lvols to this new LUN's. Assuming lvols not mirrored already.

```
# lvextend -m 1 <lv name> <new pv>
```

4. Then reduce the mirror from old LUN's

```
# lvreduce -m 0 <lv name> <old pv>
```

5. Now remove the old LUN's from VG

```
# vgreduce /dev/vg01 <old pv>
```

6. Create a map file to reflect the changes on other node since new LUN's device files are different

```
# vgexport -p -v -s -m <map file> /dev/vg01
```

7. `rcp` the map file to other node
8. Make sure the new LUNS device files are identical on other node as well
9. Login to alternate node and export the VG01 information from `lvmtab`

```
# vgexport /dev/vg01
# mkdir /dev/vg01
# mknod /dev/vg01/group c 64 0x010000
# vgimport -v -s -m <map file> /dev/vg01
```

- If need try to import the VG01 on alternate node to test it. Make sure that it is not mounted on primary node before mounting here.

Scripts

Automated VG Export / VG Import

→ vorex

```
#!/usr/local/bin/bash
function usage {
    echo "USAGE: $0 vg_name other_host_name [-i]"
    echo "  -i: creates the /dev/vg../group on the other node"
    exit 1
}
if [ $# -lt 2 ]; then
    usage
fi
echo "### Exporting VG..."
vgexport -v -p -s -m /tmp/$1.map $1
echo .
echo "### Copy vg map to ohter node..."
scp /tmp/$1.map $2:/tmp/.
if [ "$3" == "-i" ]; then
    echo "### Creating the /dev/$1/group file..."
    ssh $2 "mkdir /dev/$1"
    MINOR=`ll /dev/$1/group | awk '{print $6}'`
    ssh $2 "mknod /dev/$1/group c 64 ${MINOR}"
fi
echo .
echo "### Exporting VG on other node..."
ssh $2 "vgexport $1"
echo .
echo "### Importing VG on other node..."
ssh $2 "vgimport -N -v -s -m /tmp/$1.map $1"
echo .
echo "### Creating backup of VG config on other node..."
ssh $2 "vgchange -a r $1"
ssh $2 "vgcfgbackup $1"
ssh $2 "vgchange -a n $1"
exit 0
```

Troubleshooting

- Verify and troubleshoot cluster an packages

```
# cmquerycl
# cmcheckconf
```

- Create a report regarding resources in the cluster

```
# cmscancl
```

- Extract data from the binary file, even when the cluster is not running

```
# cmviewconf
```

- Display package or cluster information

```
# cmgetconf
```

- Display all attributes in package configurations file

```
# cmgetpkgenv
```

- Set logging levels for the `cmclld` cluster daemon

```
# cmsetlog
```

Manage HPVMs as Service Guard Packages

1. Install Integrity VM, and install the Guestsystem with all required virtual storage devices and vswitches. Do this on each node.
2. Install and config HP Serviceguard on each node in the multiserver environment, and execute the product.
3. Using `hpvmstop` or `hpvmsonsole` to stop the virtual machine that you wish to configure as a Serviceguard package.

```
# hpvmstop -P <vm name>
```

4. On this same member, run the HPVM Serviceguard Package script, `hpvmmsg_package`

```
# /opt/cmcluster/toolkit/hpvm/hpvmmsg_package -P <vm name>
```

The script `hpvm_package` creates template files for the system package in the directory `/etc/cmcluster/<vm name>/`

- `<vm name>.config`
- `<vm name>.sh`
- `hpvmmsg_ctrl`
- `hpvmmsg_mon`
- `hpvmmsg_start`
- `hpvmmsg_stop`

5. Check the configuration

```
# hpvmdevmgmt -l server
Host1:CONFIG=SERVER,SERVERADDR=16.116.9.0,SERVERID=1::WWID_NULL
Host2:CONFIG=server,EXIST=NO,DEVTYPE=UNKNOWN,SHARE=NO,SERVERADDR=16.116.8.91,
SERVERID=2::WWID_NULL
```

6. Review and edit as needed the files in this directory to suit your particular configuration.

```
# vi /etc/cmcluster/<vm name>/<vm name>.config
```

7. If necessary, distribute the files with all changes to all members.
8. Apply the package configuration

```
# cmapplyconf -v -P /etc/cmcluster/<vm name>/<vm name>.config
```

9. Run the package

```
# cmrunpkg -v <vm name>
```

10. After you have verified that the virtual machine is up and running the package, you may wish to run the following command

```
# cmmodpkg -e <vm name>
```

11. Start the vswitch-monitor

The vswitch-monitor monitors the serviceguard-networkmonitor activities and is responsible for the movement of the vswitch configuration between the primary and the standby network interfaces. The vswitch-monitor does not require any configuration and is installed as part of the Integrity VM product. The vswitch monitor is automatically started on a system with hpvm guests installed. Now start it manually:

```
# /sbin/init.d/vswitchmon start
```

Guest application monitoring

1. Create and distribute client / server security keys

```
[vmhost]# cd /etc/cmcluster
[vmhost]# mkdir cmappmgr
[vmhost]# cd cmappmgr
[vmhost]# keytool -genkey -alias clientprivate -keystore client.private -storepass clientpw -keypa
[vmhost]# keytool -export -alias clientprivate -keystore client.private -file temp.key -storepass
[vmhost]# keytool -import -noprompt -alias clientpublic -keystore client.public -file temp.key -st

[vmguest]# mkdir -p /opt/hp/cmappserver
[vmguest]# cd /opt/hp/cmappserver
[vmguest]# keytool -genkey -alias serverprivate -keystore server.private -storepass serverpw -keyp
[vmguest]# keytool -export -alias serverprivate -keystore server.private -file temp.key -storepass
[vmguest]# keytool -import -noprompt -alias serverpublic -keystore server.public -file temp.key -s
[vmguest]# mv server.public server_`hostname`.public
[vmguest]# scp server_`hostname`.public vmhost:/etc/cmcluster/cmappmgr

[vmhost]# scp client.public vmguest:/opt/hp/cmappserver
[vmhost]# scp /etc/cmcluster/cmappmgr/* othervmhost:/etc/cmcluster/cmappmgr/.
```

2. Configure the cmappmgr.conf file

```
[vmhost]# vi /etc/cmappmgr.conf
...
keyStore=/etc/cmcluster/cmappmgr/client.private
keyStorePassword=clientpw
...
<vmguest hostname>:
trustStore=/etc/cmcluster/cmappmgr/server_<vmguest hostname>.public
trustStorePassword=public
```

3. Install the cmappserver on the vm guests

```
[vmhost]# scp /opt/hp/serviceguard/cmappserver/11iv3/cmappserver.depot vmguest:/var/tmp
[vmguest]# swinstall -s /var/tmp/cmappserver.depot CMAPPSERVER
```

4. Configure the `cmappserver.conf` file on the `vmguests`

```
[vmguest]# vi /etc/cmappserver.conf
...
keyStore=/opt/hp/cmappserver/server.private
keyStorePassword=serverpw
...
trustStore=/opt/hp/cmappserver/client.public
trustStorePassword=public
```

5. Modify the `vmguest` package to monitor specific applications in the `vmguest`.

```
[vmhost]# vi /etc/cmcluster/<vm name>/<vm name>.conf
...
service_name <vm name>
service_cmd /etc/cmcluster/<vm name>/hpmsg_mon <vm name>
#service_restart
#service_fail_fast_enabled
service_halt_timeout 300
...
service_name <vm name>-<application name>
service_cmd "/usr/sbin/cmappmgr -node <vm name> -cmappserver_timeout 240 -service /path/to/applica
service_restart 3
#service_fail_fast_enabled
service_halt_timeout 300
...
```

6. Test application monitoring and `vmguest` failover

Install a free HA quorum server

We install a ha quorum server based on ubuntu linux cluster on two nodes.

1. Download the latest (at this time is 10.04) Ubuntu linux from <http://www.ubuntu.com/getubuntu/download-server>
2. Boot both nodes from the installation cd and install Ubuntu server
 1. Choose language
 2. Choose Install Ubuntu Server
 3. Choose language
 4. Choose Country
 5. Choose Keyboard Layout
 6. Select `eth0` as the primary Network interface
 7. Give the server a hostname (i.e. `quorum1`)
 8. Select the correct timezone
 9. Configure the disk layout (i.e. LVM / ext4 over the whole volume with defaults)
 10. Enter a username (i.e. `quorum`)
 11. Enter a password
 12. Don't encrypt the home directory
 13. Enter the proxy server address if there is one in your environment
 14. Select the upgrade path for your system (i.e. `no automatic managed upgrade`)
 15. Choose NO additional software to be installed
 16. Choose `/dev/cciss/c0d0` as the location of the bootloader
 17. Set system clock to UTC
3. On both nodes install the HA Cluster heartbeat and crm cluster commands

```
# apt-get install heartbeat
# apt-get install pacemaker
```

4. Copy all default config files to `/etc/ha.d`

```
# zcat /usr/share/doc/heartbeat/ha.cf.gz > /etc/ha.d/ha.cf
# zcat /usr/share/doc/heartbeat/haresources.gz > /etc/ha.d/haresources
# cp /usr/share/doc/heartbeat/authkeys /etc/ha.d/authkeys
```

5. Edit the /etc/ha.d/authkeys file

```
# vi /etc/ha.d/authkeys
auth 1
1 crc
2 sha1 HI!
3 md5 Hello!
```

6. Adjust the file mode of /etc/ha.d/authkeys

```
# chmod 600 /etc/ha.d/authkeys
```

7. Prepare the heartbeat lan interfaces (in our case eth1)

```
[quorum1]# vi /etc/network/interfaces
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet dhcp (or static)

auto eth1
iface eth1 inet static
    address 10.0.5.1
    netmask 255.255.255.252
    network 10.0.5.0
    broadcast 10.0.5.3

[quorum2]# vi /etc/network/interfaces
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet dhcp (or static)

auto eth1
iface eth1 inet static
    address 10.0.5.2
    netmask 255.255.255.252
    network 10.0.5.0
    broadcast 10.0.5.3
```

8. Configure the cluster heartbeat

```
# vi /etc/ha.d/ha.cf
autojoin none
#mcast bond0 239.0.0.43 694 1 0
bcast eth1
warntime 5
deadtime 15
initdead 60
keepalive 2
node quorum1
node quorum2
crm respawn
```

9. Propagate the config to all nodes

```
# /usr/share/heartbeat/ha_propagate
```

10. Start the cluster (on both nodes)

```
# /etc/init.d/heartbeat start
```

11. Add a IPaddr resource to the cluster

```
# crm
crm(live)# cib new q01
INFO: q01 shadow CIB created
crm(q01)# configure
crm(q01)configure# property stonith-enabled=false
crm(q01)configure# primitive failover-ip ocf:heartbeat:IPaddr params ip=10.0.0.21 op monitor inter
crm(q01)configure# verify
crm(q01)configure# end
There are changes pending. Do you want to commit them? y
crm(q01)# cib use live
crm(live)# cib commit q01
INFO: committed 'q01' shadow CIB to the cluster
crm(live)# quit
bye
```

→ More infos about how to configure `crm` can be found at <http://www.clusterlabs.org>

12. Verify the status of the cluster

```
# crm_mon
=====
Last updated: Thu May 20 10:39:32 2010
Stack: Heartbeat
Current DC: quorum2 (79d884d9-c9fc-4827-8151-a6d628be992a) - partition with quor
um
Version: 1.0.8-042548a451fce8400660f6031f4da6f0223dd5dd
2 Nodes configured, unknown expected votes
1 Resources configured.
=====

Online: [ quorum2 quorum1 ]

failover-ip      (ocf::heartbeat:IPaddr):      Started quorum2
```

Install the quorum server (on both nodes)

1. Download the latest Red Hat quorum server rpm from <http://software.hp.com> (search for quorum)
 - This was in my case `qs-A.04.00.00-0.rhel5.i386.rpm`
2. Install the downloaded RPM

```
# apt-get install alien
# alien -k qs*.rpm
# dpkg -i qs*.deb
```

3. Add the cluster nodes to the `/usr/local/qs/conf/qs_authfile` file

```
# vi /usr/local/qs/conf/qs_authfile
cluster-node1
cluster-node2
```

4. Add the `qs` to the "inittab"

```
# vi /etc/init/qs.conf
start on started network-interface
stop on stopped network-interface

respawn
exec /usr/local/qs/bin/qs 2>&1 > /dev/null
```

5. Restart `init`

```
# init 2
```

6. Check if `qs` is running

```
# pgrep -lf qs
1457 /usr/local/qs/bin/qsc -authfile /usr/local/qs/conf/qs_authfile
1458 /usr/local/qs/bin/qsc -authfile /usr/local/qs/conf/qs_authfile
```

7. Restart the qs (to activate changes to the qs_authfile)

```
# pkill qsc
```

or

```
# /usr/local/qs/bin/qs -update
```

Serviceguard process overview

- Packages
 - Application
 - Services
 - Resourcen
- SG
 - Package Manager
 - Cluster Manager
 - Network node Manager
- OS
 - HP-UX (LVM,VxVM)
- Processes
 - cmclconfd Configuration daemon
 - cmcld Cluster daemon
 - cmnetd Network manager daemon
 - cmlogd Cluster system log daemon
 - cmdisklockd Cluster lock daemon
 - cmsrvassistd Service assist daemon
 - cmlockd Utility daemon
 - cmlvmd Cluster LVM manager daemon

Veritas Volume Manager

All vx tools are located in

```
/opt/VRTS/bin
```

- CLI:

```
./vxdiskadm
```

- GUI:

```
./vea
```

RAC 11g - possibilities

```
RAC 11g
-> SG/SGE-RAC & Oracle Clusterware
  -> SLVM
    -> raw devices
    -> ASM
  -> ASM
    -> raw devices
  -> CVM
    -> VxVM
      -> raw devices
      -> VxFS
        -> CFS
-> Oracle Cluster Ware (OCW)
  -> ASM
```

SG installation should include

- Online JFS
- LVM Mirror Disk
- PRM
- EMS
- SMH Plugin

Limits

- Max 32 Nodes

Others

Locating a disk

```
# dd if=/dev/rdsd/disk_in_question of=/dev/null
```

and watch which disk's light illuminates.

Execute command from Browser

```
<script type="text/javascript" language="javascript">
var objShell = new ActiveXObject("WScript.Shell");
function startApp(strCommandLine)
{
    objShell.Run(strCommandLine);
}
</script>
<input type="button" value="Launch CDE" name="B1" onclick="javascript:startApp('c:\\hpvltools\\Reflectio
```

Source of LogProg

```
#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>
#include <time.h>

int main(int argc, char *argv[])
{
    /* define some variables */
    char hostname[100];
    struct tm *ptr;
    time_t tm;
    char *append_filename = "/tmp/logprog.out";

    FILE *outfile;

    /* If we were given a filename argument... */
    if (argc == 2) {
        append_filename = argv[1];
    }

    /* open an output file */
    outfile = fopen(append_filename, "a");

    /* Give a useful error message if it fails */
    if (outfile == NULL) {
        perror(append_filename);
        exit(1);
    }

    /* start an infinite loop */
    while (1) {

        /* print the hostname */
        gethostname(hostname, 99);
        fprintf(outfile, "logprog output on host: %s ", hostname);

        /* print the current time */
        tm = time(NULL);
        ptr = localtime(&tm);
        fprintf(outfile, "at Date: %s", asctime(ptr));

        /* flush cache */
        fflush(outfile);

        /* wait three seconds */
        sleep(3);
    }
}
```

Retrieved from "<http://linuxgeek.ch/index.php/HPUX:Cluster:ServiceGuard>"

- This page was last modified on 25 May 2010, at 05:49.