**[HP-UX Service Guard Cluster Implementation](http://www.sysadminshare.com/2013/01/hp-ux-service-guard-cluster.html)**

[0 comments](http://www.sysadminshare.com/2013/01/hp-ux-service-guard-cluster.html#comment-form)

**HP-UX Serviceguard Cluster : Implementation**

**QUESTION:**

What are the steps to creating a functional Serviceguard cluster?

NOTE: "Serviceguard does not support communication across routers between nodes in the same cluster."

Steps:

**1.Install the Serviceguard product.**

2.**NETWORKING Preparation**

3.**LVM Preparation**

**4. Building the cluster**

**5.Creating Package files**

**6. Package global node switching setup.**

**1.Install the Serviceguard product.**

Serviceguard is a licensed product and requires a codeword to install from the Application media.

**2.NETWORKING Preparation**

/etc/rc.config.d/netconf   
Define LANs that will have "stationary" IP addresses. Do NOT define a 0.0.0.0 IP for LAN adapters that will act as a standby LAN for Serviceguard! If this has been done, "unplumb" and then "plumb" the LAN adapter and then remove any reference to the standby LANs from the netconf file:

  # ifconfig lanN unplumb  (where lanN is the standby LAN)   
  # ifconfig lanN plumb    (to make it available to standby)   
  
  
**/etc/services and /etc/inetd.conf**

Serviceguard commands use network ports for commands that work on the local server or in conjunction with other servers in the cluster (or to be in the cluster). After Serviceguard is loaded, /etc/services should have 9 hacl lines:

hacl-hb     5300/tcp          # High Availability (HA) Cluster heartbeat   
  hacl-gs     5301/tcp          # HA Cluster General Services   
  hacl-cfg    5302/tcp          # HA Cluster TCP configuration   
  hacl-cfg    5302/udp          # HA Cluster UDP configuration   
  hacl-probe  5303/tcp          # HA Cluster TCP probe   
  hacl-probe  5303/udp          # HA Cluster UDP probe   
  hacl-local  5304/tcp          # HA Cluster Commands   
  hacl-test   5305/tcp          # HA Cluster Test   
  hacl-dlm    5408/tcp          # HA Cluster distributed lock manager

/etc/inetd.conf should have 2:

  hacl-cfg    dgram   udp    wait    root  /usr/lbin/cmclconfd cmclconfd -p   
  hacl-cfg    stream  tcp    nowait  root  /usr/lbin/cmclconfd cmclconfd -c   
  
  
Serviceguard Manager uses this line in /etc/inetd.conf:  
  
  hacl-probe  stream  tcp    nowait  root  /opt/cmom/lbin/cmomd cmomd -f /var/opt/cmom/cmomd.log

Insure these hacl ports are functioning:

  # netstat -a | grep hacl   
  tcp        0      0  \*.hacl-probe           \*.\*                     LISTEN   
  tcp        0      0  \*.hacl-cfg             \*.\*                     LISTEN   
  udp        0      0  \*.hacl-cfg             \*.\*

If they are not observed, cause inetd to re-read /etc/inetd.conf:

# **inetd -c**

It may be necessary to 'inetd -k' and restart 'inetd'.

**/etc/rc.config.d/netconf**

Use the simple hostname of the server when configuring the hostname.

**/etc/cmcluster/cmclnodelist**   
Serviceguard only requires this file before the cluster is built.

Thereafter, internal security mechanisms authenticate cluster commands.

Requirement: Create the file on every node and list ALL NODES in the file, giving them root access, much link a .rhosts file.

Example:

  node\_1     root    
  node\_2     root   
  
  
OBSOLETE:.rhosts - though used in past versions of Serviceguard, .rhosts is no longer consulted during Serviceguard operations.

**/etc/nsswitch.conf**   
Hostname lookup begins with this file. Current versions of Serviceguard require hostname lookup look in /etc/hosts before looking to DNS. Depending on the version of HPUX, /etc/nsswitch.conf should be configured thus:

  11iv1:     
   hosts:    files   dns   
  
  11iv2 and greater:   
   hosts:    files   dns   
   ipnodes:  files      
  
  
**identd**

As of October 2004, Serviceguard began using more stringent security when validating the source of Serviceguard commands. All current versions of Serviceguard use the 'identd' daemon delivered with sendmail to perform a Caller-ID-like service that matches hostnames to IPs of any node running Serviceguard commands. 'identd' in sendmail version 8.9.3 works with Serviceguard. The following lines must exist in order for Serviceguard to work with identd.

  11.11:   
  /etc/services:   
ident        113/tcp  authentication # RFC1413   
  
  /etc/inetd.conf:   
ident        stream tcp wait   bin  /usr/lbin/identd   identd   
  
  11.23 and up:   
  /etc/services:   
auth         113/tcp  authentication # Authentication Service   
  
  /etc/inetd.conf:   
auth         stream tcp6 wait   bin  /usr/lbin/identd   identd

NOTE: sendmail need not be activated to enable 'identd'.

**/etc/hosts**   
In conjunction with identd, /etc/hosts file on each node must list every NIC on every server which will need to communicate with a cluster node.

Example:

        15.145.162.131          [**gryf.uksr.hp.com**](http://gryf.uksr.hp.com/)            gryf   
        10.8.0.131              [**gryf-hb.uksr.hp.com**](http://gryf-hb.uksr.hp.com/)         gryf   
        10.8.1.131              [**gryf-bkup.uksr.hp.com**](http://gryf-bkup.uksr.hp.com/)       gryf   
        10.8.2.131              [**gryf-data.uksr.hp.com**](http://gryf-data.uksr.hp.com/)       gryf   
        15.145.162.132          [**sly.uksr.hp.com**](http://sly.uksr.hp.com/)             sly   
        10.8.0.132              [**sly-hb.uksr.hp.com**](http://sly-hb.uksr.hp.com/)          sly   
        10.8.1.132              [**sly-bkup.uksr.hp.com**](http://sly-bkup.uksr.hp.com/)        sly   
        10.8.2.132              [**sly-data.uksr.hp.com**](http://sly-data.uksr.hp.com/)        sly   
        10.30.8.7               [**bot.uksr.hp.com**](http://bot.uksr.hp.com/)             bot # quorum server

Notes:

1. Every fixed IP must be identified in the /etc/hosts file.  
   Serviceguard can use any NIC for internode communication.
2. Use the fully-qualified domain name first, to avoid warning messages from Serviceguard, "Warning, Unable to determine local domain name..."
3. The simply hostname must be applied to every line in order to match the IP to the hostname. This does not interfere with normal hostname look operations.

**/var/adm/inetd.sec**

This security file filters messages considered by inetd. It is not normally configured but if the Admin wishes to do so, insure Serviceguard ports are allowed to pass messages through specific IPs.

Example inetd.sec content:

  ident     allow   127.0.0.1    
  hacl-cfg  allow   127.0.0.1     
      - allows a range of servers to contact the hacl-cfg utility 

Examples:

   hacl-cfg  allow 127.0.0.1 alvin simon theodore (all cluster nodes)   
   hacl-cfg  allow 127.0.0.1 15.\*

NOTES:

1. ident refers to identd, integrated into Serviceguard since Oct 2004.
2. 127.0.0.1 is required for Serviceguard configuration.
3. When used with no options, cmquerycl probes port 5302 on all nodes on the heartbeat network to learn of cluster capability.

3.LVM Preparation

On one server, create the volume groups, logical volumes and mount point directories for the application that will be "packaged".

This example assumes a 4-disk VG, each having two channels connecting to each node. Example commands use the fictitious volume group name vg\_data.

Note that the paths to the shared disks may have different dsk identifiers on each server.

                         Storage Array   
           c0     +--------------------------+     c2   
          --------+ (disk t1d0)  (disk t2d0) +---------   
    Node\_1        |                          |         Node\_2   
          --------+ (disk t3d0)  (disk t4d0) +---------         
           c1     +--------------------------+     c3

The intent of this example is to create Physical Volume Groups in order to force mirroring across PVlinks. The disks at cXt1d0 and cXt2d0 will be one PVG, while t3 and t4 the other. Where possible, load balancing will also be incorporated.

  # pvcreate -f /dev/rdsk/c0t1d0   (repeat for other disks)   
  # mkdir /dev/vgdata    
  # mknod /dev/vgdata/group c 64 0xNN0000  - where NN = unique number  
  # vgcreate -g PVG0 vg\_data /dev/dsk/c0t1d0 /dev/dsk/c0t2d0    
  # vgextend -g PVG1 vg\_data /dev/dsk/c1t3d0 /dev/dsk/c1t4d0    
  # vgextend  vg\_data /dev/dsk/c1t1d0 /dev/dsk/c1t2d0    
  # vgextend  vg\_data /dev/dsk/c0t3d0 /dev/dsk/c0t4d0    
  # lvcreate -L 17179 -s g /dev/vg\_data   (PVG-strict allocation)   
  # lvextend -m 1 /dev/vg\_data/lvol1 PVG1

In the example, node\_1's /etc/lvmtab would look like this:

vg\_data   
/dev/dsk/c0t1d0  (pri-path t1 - PVG0)   
/dev/dsk/c0t2d0  (pri-path t2 - PVG0)   
/dev/dsk/c1t3d0  (pri-path t3 - PVG1)   
/dev/dsk/c1t4d0  (pri-path t4 - PVG1)   
/dev/dsk/c1t1d0  (alt-path t1 - PVG0)   
/dev/dsk/c1t2d0  (alt-path t2 - PVG0)   
/dev/dsk/c0t3d0  (alt-path t3 - PVG1)   
/dev/dsk/c0t4d0  (alt-path t4 - PVG1)

And node\_1's /etc/lvmpvg would look like this:

**VG /dev/vg\_data**   
PVG PVG0   
/dev/dsk/c0t1d0   
/dev/dsk/c0t2d0   
PVG PVG1   
/dev/dsk/c1t3d0   
/dev/dsk/c1t4d0   
  
 # vgchange -a y vg\_data   
 # newfs -F vxfs /dev/vg\_data/lvol1   
 # mkdir /data1   
 # mount /dev/vg\_data/lvol1 /data1

Repeat this process for every volume group managed by a Serviceguard package.

It is recommended that all VG manipulation be performed on the server maintaining the /etc/lvmpvg file. If that is not desired, create the /etc/lvmpvg on each node that will be used to manage the LVM VGs for the packages. Insure the correct dsk pathing is specificed.

Insure the application runs and can utilize the file systems that will be governed by a Serviceguard package. Then unmount the logical volumes and deactivate the volume groups in which they reside.

**3.A Importing the VG into the other nodes**

Any server that will adopt a package must reference that packages' LVM volume groups in the /etc/lvmtab file. Use strings /etc/lvmtab to determine the volume groups and their special files.

Import the volume groups that will be managed by Serviceguard using this procedure:

  # vgexport -pvs -m map.vg\_data /dev/vg\_data    
  # rcp map.vg\_data :/etc/lvmconf/ 

Repeat for other volume groups and servers as needed.

Create a listing of /dev/vg\_data/group files that shows the minor number used.

  # ll /dev/vg\*/group | awk '{print $6,$10}' >/tmp/group\_list

On the other servers that will adopt the package and it's volume groups:

  # mkdir /dev/vg\_data ; cd /dev/vg\_data    
  # mknod group c 64 0xNN0000     
      NN = Match the group minor number listed in /tmp/group\_list!   
  # vgimport -vs -m /etc/lvmconf/map.vg\_data /dev/vg\_data   
  
  # mkdir /data1   
  # mount /dev/vg\_data/lvol1 /data1

Repeat as needed for every volume group to be managed by Serviceguard.

Consecutively on each server that will adopt the package, create the mount points, activate the volume group, mount the file systems for the application and run the application on the adoptive server to verify it works before expecting Serviceguard to do the same.

**/etc/lvmrc**

Normally, one would want all VGs to be activated at boot time. In a Serviceguard node, only private VGs should be activated at startup. Attempts to activate "clustered" VGs at boot time will generate error messages to the console. To prevent the attempt and subsequent messages, edit /etc/lvmrc - change **AUTO\_VG\_ACTIVATE=0.**

Under the section titled custom\_vg\_activation() add private volume groups (other than vg00) that Serviceguard will not be managing:

     /sbin/vgchange -a y -s vg06 vg07    
     parallel\_vg\_sync "/dev/vg06 /dev/vg07"

**/etc/fstab**

Do not list the lvol/mount pair of any file system that Serviceguard will manage in the /etc/fstab file. The administrator will load the package control script will this information later.

**4. Building the cluster**

Creating the Cluster configuration file.

For the purposes of this example, the cluster configuration file will be called "CONF". Make Serviceguard discover the nodes that will be used in the cluster, and create a cluster configuration file:

# cd /etc/cmcluster   
# cmquerycl -C CONF -n Node\_1 -n Node\_2 ...

If using a quorum server, the syntax is:  
  
# cmquerycl -C CONF -n Node\_1 -n Node\_2 ... -q (qs\_hostname)

The CONF file should be created, listing both nodes, a cluster lock disk VG (automatically selected by Serviceguard), sections describing the LANs on each node, cluster parameters and finally the volume groups each node has in common.

* Set the **NODE\_TIMEOUT** to 8 seconds (8000000).
* Configure **MAX\_CONFIGURED\_PACKAGES** to at least the number of packages that the cluster will run.
* If there are multiple active LANs between nodes, change the**STATIONARY\_IP** name to **HEARTBEAT\_IP** to induce redundant heartbeat transmission.
* Insure all of the volume groups common to the cluster nodes' lvmtab files are listed at the bottom of the file.

If satisfied, check the file:

  # vgchange -a y (vg\_lock)   
  # cmcheckconf -C CONF

If this succeeds, proceed to create the cluster binary file:

  # cmapplyconf -C CONF   
  # vgchange -a n (vg\_lock)

The initial cmapplyconf performs the following 4 tasks:

* cmcheckconf (again)
* Build and distribute the cluster binary file, cmclconfig, to each cluster node.
* Install the cluster lock structure on the designated disk.
* Load the cluster ID into each of the common VG disks to require them to be activated in "exclusive" mode.

Now, with the cluster binary distributed, a packageless cluster can be started:

# cmruncl

NOTES:

1. Even with no packages, a Serviceguard cluster can perform standby LAN failover.
2. "Clustered" VGs will no longer activate using the "vgchange -a y" method.  
   "cmlvmd" must be active to authorize clustered VG activation using the "vgchange -a e" method. cmlvmd must be active to install an 'exclusive' or 'shared' activation mode on cluster-intended a volume group.
3. If a clustered volume group will be used for SGeRAC (Serviceguard extension for Real Applications Clusters (Oracle), the VG's activation mode must be changed from "exclusive" to "shared" in order that multiple nodes can activate it concurrently.

The procedure is:

# vgchange -c n   
# vgchange -c y -S y

Such VGs should only be activated for RAW access. Co-mounted file systems will cause system panics if any file system structure is modified. VGs configured to activate in 'shared' mode cannot be modified by LVM commands. (Note man page warning).

**5. Creating the package files**

Create a directory for each package:

 cmcluster# mkdir orapkg  (example name)   
 cmcluster# cd orapkg     (example name)   
 orapkg#

In the package directory, build the package configuration file, and the package control script:

orapkg# cmmakepkg -p config     (example pkg config file name)   
orapkg# cmmakepkg -s control.sh (example pkg control file name)

Edit the package configuration file, adding or setting the package-specific parameters (which are described in the file).

NOTES:

* The following is an example of what might be used in each line.
* the preferred default settings are listed first.
* - "|" = "or"

PACKAGE\_NAME                  (user-defined name)   
PACKAGE\_TYPE                  FAILOVER | SYSTEM\_MULTI\_NODE   
   NOTE: SYSTEM\_MULTI\_NODE packages are only supported with HP CVM and CFS.  
  
FAILOVER\_POLICY               CONFIGURED\_NODE | MIN\_PACKAGE\_NODE  
FAILBACK\_POLICY               MANUAL | AUTOMATIC   
  
NODE\_NAME                     (primary hostname)   
NODE\_NAME                     (adoptive hostname)   
NODE\_NAME                     (2nd adoptive hostname - if any)   
  
AUTO\_RUN                                  YES | NO   
LOCAL\_LAN\_FAILOVER\_ALLOWED    YES | NO   
NODE\_FAIL\_FAST\_ENABLED            NO  | YES   
  
RUN\_SCRIPT                    /etc/cmcluster/orapkg/control.sh    
RUN\_SCRIPT\_TIMEOUT            600  (in seconds)   
HALT\_SCRIPT                   /etc/cmcluster/orapkg/control.sh   
HALT\_SCRIPT\_TIMEOUT           600  (in seconds)   
  
SERVICE\_NAME                  ora\_mon (referenced in control.sh too)   
SERVICE\_FAIL\_FAST\_ENABLED     NO   
SERVICE\_HALT\_TIMEOUT          100  (in seconds)   
  
SUBNET                        15.44.48.0   (if pkg relies on a LAN NIC)

NOTE: The Admin can configure the Serviceguard package to trigger a package failover on thresholds or up/down state of system hardware and software resources:

#RESOURCE\_NAME               /vg/vg00/pv\_summary   
#RESOURCE\_POLLING\_INTERVAL   10   
#RESOURCE\_START              AUTOMATIC   
#RESOURCE\_UP\_VALUE           = UP   
#RESOURCE\_UP\_VALUE           = PVG\_UP

Edit the package control script. Reference the package-specific resources that the package will activate when started. The following are example resource definition lines from a package control script:

VGCHANGE="vgchange -a e"

Note: for SGeRAC 'shared' VGs, use this mode instead:  
  
VGCHANGE="vgchange -a s"

CVM\_ACTIVATION\_CMD="vxdg -g \$DiskGroup set activation=exclusivewrite"   
  
VG[0]=""      Example.  Add more VGs as needed   
  
LV[0]="/dev/vg01/lvol1"; FS[0]="/log1"; FS\_MOUNT\_OPT[0]="-o rw"   
LV[1]="/dev/vg01/lvol2"; FS[1]="/log2"; FS\_MOUNT\_OPT[1]="-o rw"   
LV[2]="/dev/vg01/lvol3"; FS[2]="/log3"; FS\_MOUNT\_OPT[2]="-o rw"   
   +   |-------------|      +   |---|                +    \\\\\   
user-defined values   
  
  
+ The index number must be a continuous sequence - no gaps!  
\\\ The "-o largefiles" mount option is legitimate if the file system has been created thus.

FS\_UMOUNT\_COUNT=1  increment this value of the application needs more time  
                   to release file systems.   
  
FS\_MOUNT\_RETRY\_COUNT=0   
  
IP[0]="15.44.49.77"          (sample relocatable IP)   
SUBNET[0]="15.44.48.0"       (matching subnet for relocatable IP)   
  
SERVICE\_NAME[0]="ora\_mon"   
SERVICE\_CMD[0]="/etc/cmcluster/orapkg/control.sh monitor"   
SERVICE\_RESTART[0]="-r 1"    (restart attempts)

Add custom commands to customer\_defined\_run\_cmds and customer\_defined\_halt\_cmds sections as needed.

With the package configuration and control files customized, copy the control script to the adoptive nodes so that they too can run the package.

 # rcp -pr /etc/cmcluster/orapkg  (node\_2):/etc/cmcluster/

Once complete, perform a check of the files:

orapkg# **cmcheckconf -P config**

If the command succeeds without error, add the package to the cluster binary:

orapkg# **cmapplyconf -P config**

Use cmviewcl to view the state of the package:

# **cmviewcl -v -p orapkg**

Add packages as needed.

**6. Package global and node switching**

Serviceguard packages can be enabled or disabled to run on the cluster or specific nodes in the cluster. Observe this cmviewcl output:

# **cmviewcl -v -p clk**

UNOWNED\_PACKAGES   
  
    PACKAGE      STATUS       STATE        AUTO\_RUN     NODE   
    clk          down         halted       disabled     unowned   
  
      Policy\_Parameters:   
      POLICY\_NAME     CONFIGURED\_VALUE   
      Failover        configured\_node   
      Failback        manual   
  
      Script\_Parameters:   
      ITEM       STATUS   NODE\_NAME    NAME   
      Subnet     up       eon          16.113.0.0   
      Subnet     up       ion          16.113.0.0   
  
      Node\_Switching\_Parameters:   
      NODE\_TYPE    STATUS       SWITCHING    NAME   
        Primary      up           enabled      eon   
        Alternate    up           disabled     ion 

The **cmmodpkg**command enables or disables switching flags.

Compare AUTO\_RUN (global switching) to a master circuit breaker and Node\_switching to room-based circuit breakers.

To enable the master (AUTO\_RUN) breaker such that the package is allowed to run, use this command:

# **cmmodpkg -e**

To enable a specific node to run the specified package, use this form of the cmmodpkg command:

# **cmmodpkg -e -n**

**NOTES:**

1. When the "primary" node is enabled to run a package and that package has been enabled to run, the package auto-start.
2. Though AUTO\_RUN can be initially set in the package configuration file, it is a dynamic flag, changed by the run-ability of the package.