Veritas™ Cluster Server Application Note: SunFire 12K/15K Dynamic Reconfiguration

Solaris

N18538F

symantec.

Veritas Cluster Server Application Note: SunFire 12K/15K Dynamic Reconfiguration

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. All third-party copyrights associated with this product are listed in the accompanying release notes.

Solaris is a trademark of Sun Microsystems, Inc.

Technical support

For technical assistance, visit http://support.veritas.com and select phone or email support. Use the Knowledge Base search feature to access resources such as TechNotes, product alerts, software downloads, hardware compatibility lists, and our customer email notification service.

Veritas Cluster Server Application Note: SunFire 12K/15K Dynamic Reconfiguration

Introduction

This application note describes how to perform dynamic reconfiguration (DR) operations on VCS clustered system domains of the Sun FireTM 12K and 15K servers. The DR operations typically include configuring and unconfiguring CPU/memory boards to and from domains and configuring and unconfiguring I/O cards to and from I/O boards in a domain. I/O boards cannot be dynamically reconfigured, but the PCI cards on I/O boards can be dynamically reconfigured.

These operations allow switching boards from one domain to another or permit removing a board or card to upgrade or replace it. DR operations can be performed while the operating environment continues to run. However, a DR operation performed on a CPU/memory board that has permanent memory requires that the system domain be temporarily suspended and, in this case, VCS must be stopped. This document describes the procedures for shutting down and restarting VCS.

Note: Currently, VCS does not support using DR in clusters where I/O controllers and storage use Sun's Alternate Pathing (AP).

Do not use the following procedures to dynamically reconfigure a network interface card used for a VCS private heartbeat link. If you need to do so, you must stop VCS before proceeding.

Note: The Sun documentation for dynamic reconfiguration on the Sun Fire F12K/F15K contains comprehensive descriptions of procedures and commands. To avoid damaging system boards and components, you should be familiar with the procedures for their removal and replacement.

Supported software

- Solaris 8 and Solaris 9
- VERITAS Cluster Server, releases 2.0, 3.5 (any patch level) or later
- VERITAS Volume Manager, as supported by the VCS version
- VERITAS File System, as supported by the VCS version

Note: Please check that you are using the latest version of this document.

Dynamic Reconfiguration in VCS environment - Overview

The boards in an F12K/15K domain may contain I/O controllers, CPUs, or memory. Typically, boards within a domain have their functions duplicated on other boards. For example, you can remove a board with CPU or memory dynamically because another board in the domain can perform the equivalent functions.

In a VCS cluster of domains, dynamic reconfiguration operations in one domain may cause VCS to detect that resources are unavailable and initiate failover to another domain. Therefore, it is advisable to freeze the service groups running in the domain and stop VCS before running DR operations. See "When must you stop VCS when performing DR?" on page 10.

For users of VERITAS DBE/AC *for Oracle9i RAC*, it is necessary to stop the Oracle RAC instance within the domain being reconfigured if VCS must be stopped. This permits communications among other RAC instances to occur while the instance in the one domain is temporarily stopped.

Planning to reconfigure devices

To be dynamically reconfigured, the boards must satisfy the following conditions:

Critical resources on boards must be redundant. For example, boards for which CPUs and memory are redundant can be reconfigured after their function has been replaced and their activity stopped. A CPU board that contains the only CPU in a domain cannot be moved.

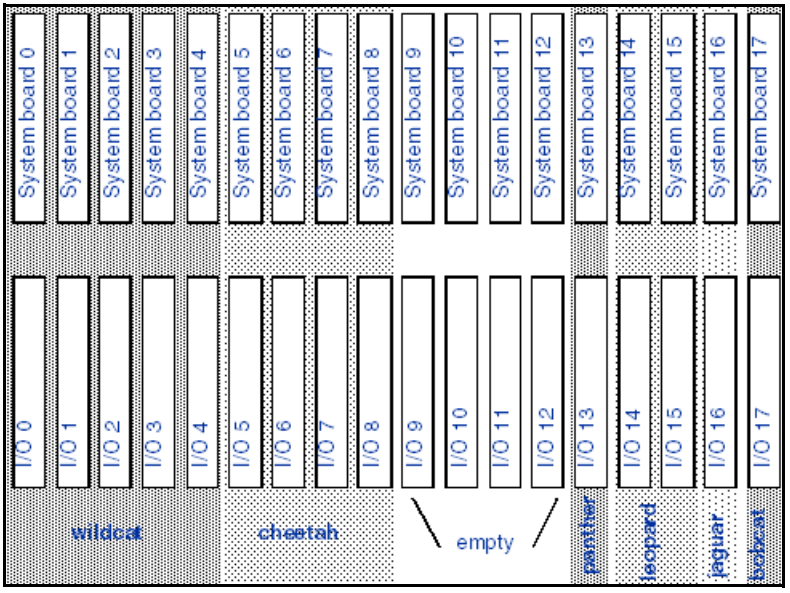
A memory board containing permanent memory, such as the OpenBootTM PROM or kernel memory, can be moved after the memory has been moved to another board. DR on boards with permanent memory requires VCS be shut down.

Disk drives must be accessible via alternate pathways. The Dynamic Multipathing (DMP) feature can provide alternate paths. Before moving a host bus adapter, switch all the card's functions to an alternate card. An HBA that controls sole access to an active drive cannot be moved.

Activity on a PCI card must be stopped before the card is removed.

Example F15K configuration

The following example configuration serves as a reference for some of the procedures described in this docment.



On Sun Fire 15K systems, system boards and I/O boards are numbered 0-17. On Sun Fire 12K systems, system boards and I/O boards are numbered 0-8. In the Sun Fire 15K example shown above, six domains have been configured, and there are additional empty slots.

Listing all boards in all domains

You can display information about all boards in all domains in one F12K or F15K server using the `showboards` command when you are logged in as superuser to the platform shell. For example:

```
# showboards
Retrieving board information. Please wait
Location    Pwr    Type of Board   Board Status  Test Status   Domain
--------    ---    -------------   ------------  -----------   ------
SB0         On     CPU             Active        Passed        wildcat
SB1         On     CPU             Active        Passed        wildcat
SB2         On     CPU             Active        Passed        wildcat
SB3         On     CPU             Active        Passed        wildcat
SB4         On     CPU             Active        Passed        wildcat
SB5         On     CPU             Active        Passed        cheetah
SB6         On     CPU             Active        Passed        cheetah
SB7         On     CPU             Active        Passed        cheetah
SB8         On     CPU             Active        Passed        cheetah
SB9         -      Empty Slot      Available      -           Isolated
SB10        -      Empty Slot      Available      -           Isolated
SB11        -      Empty Slot      Available      -           Isolated
SB12        -      Empty Slot      Available      -           Isolated
SB13        On     CPU             Active        Passed        panther
SB14        On     CPU             Active        Passed        leopard
SB15        On     CPU             Active        Passed        leopard
SB16        Off    CPU             Assigned      Unknown        jaguar
SB17        Off    CPU             Assigned      Unknown        bobcat
IO0         On     HPCI            Active        Passed        wildcat
IO1         On     HPCI            Active        Passed        wildcat
IO2         On     HPCI            Active        Passed        wildcat
IO3         On     HPCI            Active        Passed        wildcat
IO4         On     HPCI            Active        Passed        wildcat
IO5         On     HPCI            Active        Passed        cheetah
IO6         On     HPCI            Active        Passed        cheetah
IO7         On     HPCI            Active        Passed        cheetah
IO8         On     HPCI            Active        Passed        cheetah
IO9          -     Empty Slot      Available        -          Isolated
IO10         -     Empty Slot      Available        -          Isolated
IO11         -     Empty Slot      Available        -          Isolated
IO12         -     Empty Slot      Available        -          Isolated
IO13        On     HPCI            Active        Passed        panther
```

```
IO14        On     HPCI           Active       Passed       leopard
IO15        On     HPCI           Active       Passed       leopard
IO16        Off    HPCI           Assigned     Unknown       jaguar
IO17        Off    HPCI           Assigned     Unknown       bobcat
```

Listing boards in a domain

You can list the boards in a domain using the `cfgadm` command. For example, if you are logged into the `leopard` domain (see "Example F15K configuration" on page 8), enter:

```
# cfgadm
```

The output resembles:

```
Ap_Id                Type         Receptacle  Occupant      Condition
IO14                  HPCI          connected   configured    ok
IO15                  HPCI          connected   configured    ok
SB14                  CPU           connected   configured    ok
SB15                  CPU           connected   configured    ok
c0                   scsi-bus      connected   configured    unknown
c1                   scsi-bus     connected    configured    unknown
c12                  scsi-bus     connected    unconfigured  unknown
c13                  scsi-bus     connected    unconfigured  unknown
c2                   scsi-bus     connected    unconfigured  unknown
c3                   scsi-bus     connected    unconfigured  unknown
c8                   scsi-bus     connected    configured    unknown
c9                   scsi-bus     connected    configured    unknown
pcisch0:e15b1slot1    pci-pci/hp   connected    configured    ok
pcisch1:e15b1slot0    mult/hp      connected    configured    ok
pcisch2:e15b1slot3    pci-pci/hp   connected    configured    ok
pcisch3:e15b1slot2    ethernet/hp  connected    configured    ok
pcisch4:e14b1slot1    pci-pci/hp   connected    configured    ok
pcisch5:e14b1slot0    mult/hp      connected    configured    ok
pcisch6:e14b1slot3    pci-pci/hp   connected    configured    ok
pcisch7:e14b1slot2    ethernet/hp  connected    configured    ok
```

In the example output shown above, the boards `IO14` and `IO15` each contain four slots, all of which are occupied by PCI cards, listed at the bottom of the output.

When must you stop VCS when performing DR?

It is necessary to stop VCS and unconfigure GAB and LLT in certain circumstances as described in the following paragraphs.

CPU/Memory Boards - Stopping VCS

If the CPU/memory board to be removed contains permanent memory, the operating system's function must be suspended to permit dynamic reconfiguration to occur. In such a case, VCS must be stopped.

However, you do not need to stop VCS when:

- You are performing DR on a board that does not contain permanent memory.
  Typically, in a domain with multiple CPU/memory boards, one board has permanent memory, while the others do not.
- When you are performing DR to add a new board to the domain.
  The existing functions in the domain are not affected by the dynamic addition of a new CPU/memory board.

Note: If you must reconfigure multiple boards and a board with permanent memory is among them, reconfigure the board with permanent memory last. This sequence ensures minimum VCS downtime.

To determine if the CPU/memory board has permanent memory

1. Log into the domain as domain administrator.
2. List the boards with permanent memory in the domain by entering:

```
# cfgadm -av | grep permanent
SB15::memory connected    configured   ok         base
address 0x1e000000000, 16777216 KBytes total, 2001200 KBytes
permanent
```

The output in the example shows SB15 to contain permanent memory. Before this board can be dynamically reconfigured, VCS must be stopped. The procedures are described in "Stopping VCS in a standard environment" on page 12 and "Stopping VCS in an Oracle9i RAC environment" on page 14. Other CPU/memory boards in the domain do not have permanent memory and may be dynamically reconfigured without stopping VCS.

I/O boards - stopping VCS

You must stop VCS when you reconfigure an I/O board in the following circumstances:

- When the I/O board requiring reconfiguration contains all the private network links used by the domain.
- When the I/O board contains the only public network links used by the domain.

- When the I/O board contains all of the paths to a storage device.

Stopping and starting VCS

When you dynamically reconfigure CPU/Memory boards and I/O boards, it may be necessary, in some circumstances, to stop VCS in the domain. See "When must you stop VCS when performing DR?" on page 10.

Applications running on clusters of three or more domains remain highly available on two or more domains if VCS operation must be stopped on one domain. In a cluster of two domains, the applications running during reconfiguration are not highly available when VCS must be stopped on one of the domains.

This section contains:

- The procedures for stopping VCS if required for dynamic reconfiguration
- The procedures for starting VCS if it has been stopped for dynamic reconfiguration

Stopping VCS in a standard environment

If you are running VERITAS DBE/AC *for Oracle9i RAC*, see "Stopping VCS in an Oracle9i RAC environment" on page 14.

To stop VCS in a standard environment

1. Log in as administrator to the domain (`wildcat`, for example) you are reconfiguring.
2. List the VCS service groups to determine which are online on the domain:
   ```
   # hagrp -list
   ```
3. If you can switch the service groups running on the domain to another domain (`cheetah`, for example), do the following:
   a. Switch the service groups:
      ```
      # hagrp -switch service_grp_name -to cheetah
      ```
   b. Verify the service groups are offline on `wildcat`:
      ```
      # hastatus
      ```
   c. Stop VCS on `wildcat`:
      ```
      # hastop -local
      ```

4 If you cannot switch the online service groups to another system, freeze each of them for the duration of dynamic reconfiguration as follows:

   a Make the VCS configuration writable:
   ```
   # haconf -makerw
   ```

   b Freeze each of the service groups persistently:
   ```
   # hagrp -freeze service_grp_name -persistent
   ```

   c Verify the groups are frozen:
   ```
   # hagrp display | grep Frozen
   ```

   d Make the configuration read-only:
   ```
   # haconf -dump -makero
   ```

   e Stop VCS:
   ```
   # hastop -local -force
   ```

5 Unconfigure GAB:
```
# /sbin/gabconfig -U
```

6 Unconfigure LLT:
```
# /sbin/lltconfig -U
```
When you are prompted, answer "**y**" to confirm that you want to stop LLT.

7 Remove the GAB and LLT modules from the kernel.

   a Determine the IDs of the GAB and LLT modules:
   ```
   # modinfo | egrep "gab|llt"
   305 78531900  30e 305   1  gab
   292 78493850  30e 292   1  llt
   ```

   b Unload the GAB and LLT modules based on their module IDs:
   ```
   # modunload -i 305
   # modunload -i 292
   ```

8 You can begin performing dynamic reconfiguration.

Restarting VCS in a standard environment

If you are ready to restart VCS in the domain where you are performing dynamic reconfiguration, use the following procedure. If you are running VERITAS DBE/AC *for Oracle9i RAC*, and are ready to restart VCS, see "Restarting VCS in an Oracle9i RAC environment" on page 16.

To restart LLT, GAB, and VCS

1 Restart LLT:

```
# /etc/rc2.d/S70llt start
```

2 Restart GAB:

```
# /etc/rc2.d/S92gab start
```

3 Start VCS:

```
# hastart
```

4 Verify GAB and VCS are started:

```
# /sbin/gabconfig -a
GAB Port Memberships
===============================================
Port a gen 4a1c0001 membership 012
Port h gen g8ty0002 membership 012
```

To bring service groups online

1 Determine which service groups are frozen (see step 4 on page 13):

```
# hagrp -display | grep Frozen
```

2 Make the configuration writable:

```
# haconf -makerw
```

3 Unfreeze the frozen service groups:

```
# hagrp -unfreeze service_grp_name -persistent
```

4 Make the configuration read-only.

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

Stopping VCS in an Oracle9*i* RAC environment

If VCS must be stopped on a domain where VERITAS DBE/AC *for Oracle9i RAC* is running, the Oracle RAC application on the domain being reconfigured must be offlined. In addition, the GAB, LLT, LMX, and VXFEN modules must be unconfigured. Performing these steps ensures that other instances do not attempt communication with the stopped instance, which could cause the application to hang when the instance does not respond.

To stop VCS in a VERITAS DBE/AC for Oracle9*i* RAC environment

1 Log in as administrator to the domain being reconfigured (`wildcat`, for example).

2 List the configured VCS service groups and see which are online in the domain:

```
# hagrp -list
```

3 Based on the output of step 2, offline each service group that is online in the domain wildcat. Use the following command:

```
# hagrp -offline service_grp_name -sys wildcat
```

4 Stop VCS:

```
# hastop -local
```

In addition to port h, this command stops the CVM drivers using ports v and w.

5 If any CFS file systems outside of VCS control are mounted, unmount them.

6 Stop and unconfigure the drivers required by DBE/AC:

```
# cd /opt/VRTSvcs/rac
# ./uload_drv
Unloading qlog
Unloading odm
Unloading fdd
Unloading vxportal
Unloading vxfs
```

7 Unconfigure the VCSMM and I/O fencing drivers, which use ports b and o, respectively:

```
# /sbin/vxfenconfig -U
# /sbin/vcsmmconfig -U
```

8 Unconfigure the LMX driver:

```
# /sbin/lmxconfig -U
```

9 Verify that the drivers h, v, w, f, q, d, b, and o are stopped. They should not show memberships when you use the `gabconfig -a` command:

```
# gabconfig -a
GAB Port Memberships
===============================================================
Port a gen 4a1c0001 membership 01
```

10 Unload the VCSMM, I/O fencing, and LMX modules.

a Determine the module IDs for VCSMM, I/O fencing, and LMX:

```
# modinfo | egrep "lmx|vxfen|vcsmm"
237 783e4000  25497 237   1  vcsmm (VERITAS Membership
Manager)
238 78440000  263df 238   1  vxfen (VERITAS I/O
Fencing)
239 7845a000  12b1e 239   1  lmx (LLT Mux 3.5B2)
```

b Unload the VCSMM, I/O fencing, and LMX modules based on their module IDs:

```
# modunload -i 237
# modunload -i 238
# modunload -i 239
```

11 Unconfigure GAB

```
# /sbin/gabconfig -U
```

12 Unconfigure LLT

```
# /sbin/lltconfig -U
```

13 Remove the GAB and LLT modules from the kernel.

a Determine the IDs of the GAB and LLT modules:

```
# modinfo | egrep "gab|llt"
305 78531900  30e 305   1  gab
292 78493850  30e 292   1  llt
```

b Unload the GAB and LLT modules based on their module IDs:

```
# modunload -i 305
# modunload -i 292
```

14 You can begin performing dynamic reconfiguration.

Restarting VCS in an Oracle9i RAC environment

If you used the procedure described in "Stopping VCS in a standard environment" on page 12 before dynamically reconfiguring a CPU/memory board, use the following procedures to restart VCS and online the service groups on the domain.

To restart LLT, GAB, VCS, and DBE/AC processes

1 Restart LLT:

```
# /etc/rc2.d/S70llt start
```

2 Restart GAB:

```
# /etc/rc2.d/S92gab start
```

3 Restart the LMX driver:

```
# /etc/rc2.d/S71lmx start
```

4 Restart the VCSMM driver:

```
# /etc/rc2.d/S98vcsmm start
```

5 Restart the VXFEN driver:

```
# /etc/rc2.d/S97vxfen start
```

6 Restart the ODM driver:

```
# mount /dev/odm
```

7 Start VCS:

```
# hastart
```

8 Verify that the CVM service group is online:

```
# hagrp -state cvm
```

9 Verify the GAB memberships required for DBE/AC for Oracle9i RAC are configured:

```
# /sbin/gabconfig -a
GAB Port Memberships
===============================================================
Port a gen 4a1c0001 membership 012
Port b gen g8ty0002 membership 012
Port d gen 40100001 membership 012
Port f gen f1990002 membership 012
Port h gen g8ty0002 membership 012
Port o gen f1100002 membership 012
Port q gen 28d10002 membership 012
Port v gen 1fc60002 membership 012
Port w gen 15ba0002 membership 012
```

10 Online the service groups that had been take offline in step 3 on page 11:

```
# hagrp -online service_grp_name -sys wildcat
```

Dynamically reconfiguring CPU/Memory boards

You may want to remove a CPU/memory board that is malfunctioning. Or, you may want to reconfigure a board from one domain to another where it is more needed.

To reassign a board from one domain to another, you must unconfigure it from one domain and reassign it to another domain. This can be done without physically removing the board from its slot. To replace a board, however, you must unconfigure it from one domain, physically remove it, add its replacement board and reconfigure it to the domain.

Performing Dynamic Reconfiguration on a CPU/memory board

Use the following procedure to dynamically reconfigure a CPU/memory board.

Determine the status of the board you are reconfiguring

1 If necessary, log in as the administrator to the domain containing the CPU/memory board.

2 Determine the attachment point of the board you are removing:

```
# cfgadm
Ap_Id          Type       Receptable      Occupant       Cond
.
SB2            CPU        connected       configured     ok
.
```

3 Make sure you have checked whether the board has permanent memory. See "To determine if the CPU/memory board has permanent memory" on page 11 if necessary.

- If the board in the domain you want to dynamically reconfigure contains permanent memory, be sure you have first stopped VCS using the procedures described in "Stopping VCS in a standard environment" on page 12 or described in "Stopping VCS in an Oracle9i RAC environment" on page 14, whichever is appropriate.
- If the board you want to reconfigure does not have permanent memory, you can proceed to dynamically reconfigure it.

To unbind processes bound to CPU on the board

1. To determine if any processes are bound to a CPU, enter:

   # pbind -q

   If a processes is bound to the board, the output indicates the process ID and the ID number of the CPU:

   ```
   process id 650: 0
   ```

   If you see no output or see output showing no processes bound to a CPU on the board you are reconfiguring, perform the steps in "To unconfigure the board" on page 18.

2. Unbind all processes bound to the CPU on the board. For example, enter:

   # pbind -u 650

3. Rebind the processes to a processor on another board, if necessary. For example, bind process 650 to processor with ID 9, which is on another board, using the command:

   # pbind -b 650 9

If you try to unconfigure a board with processes bound to it, you see a message similar to:

```
cfgadm: Hardware specific failure: unconfigure SB15: Failed to
off-line:dr@0:SB15::cpu3
```

To unconfigure the board

1. Unconfigure and disconnect the board:

   # cfgadm -v -c disconnect SB2

2. If a board does not contain permanent memory, the command's output resembles:

   ```
   request delete capacity (4 cpus)
   request delete capacity (2097152 pages)
   request delete capacity SB2 done
   request offline SUNW_cpu/cpu448
   request offline SUNW_cpu/cpu449
   request offline SUNW_cpu/cpu450
   request offline SUNW_cpu/cpu451
   request offline SUNW_cpu/cpu448 done
   request offline SUNW_cpu/cpu449 done
   request offline SUNW_cpu/cpu450 done
   ```

```
request offline SUNW_cpu/cpu451 done
unconfigure SB2
unconfigure SB2 done
notify remove SUNW_cpu/cpu448
notify remove SUNW_cpu/cpu449
notify remove SUNW_cpu/cpu450
notify remove SUNW_cpu/cpu451
notify remove SUNW_cpu/cpu448 done
notify remove SUNW_cpu/cpu449 done
notify remove SUNW_cpu/cpu450 done
notify remove SUNW_cpu/cpu451 done
disconnect SB2
disconnect SB2 done
poweroff SB2
poweroff SB2 done
unassign SB2 skipped
```

Skip to step 4.

3 If the board has permanent memory, the system prompts you to proceed:

```
System may be temporarily suspended; proceed (yes/no)?
```

If you answer "**yes**," the DR proceeds. The system is suspended during reconfiguration. When the system resumes operation on another board, the board you are reconfiguring is disconnected. If the disconnect operation succeeds, the output resembles:

```
request suspend SUNW_OS
request suspend SUNW_OS done
request delete capacity (2097152 pages)
request delete capacity SB15 done
request offline SUNW_cpu/cpu480
request offline SUNW_cpu/cpu481
request offline SUNW_cpu/cpu482
request offline SUNW_cpu/cpu483
request offline SUNW_cpu/cpu480 done
request offline SUNW_cpu/cpu481 done
request offline SUNW_cpu/cpu482 done
request offline SUNW_cpu/cpu483 done
unconfigure SB15
unconfigure SB15 done
notify remove SUNW_cpu/cpu480
notify remove SUNW_cpu/cpu481
notify remove SUNW_cpu/cpu482
notify remove SUNW_cpu/cpu483
notify remove SUNW_cpu/cpu480 done
notify remove SUNW_cpu/cpu481 done
notify remove SUNW_cpu/cpu482 done
notify remove SUNW_cpu/cpu483 done
disconnect SB15
disconnect SB15 done
poweroff SB15
```

```
poweroff SB15 done
unassign SB15 skipped
notify resume SUNW_OS
notify resume SUNW_OS done
```

Skip to step 4.

Note: If there are real-time processes running on the board you are unconfiguring, the disconnect operation may not succeed. You must stop these processes in the appropriate manner before continuing with DR.

---

a. If the board has real-time processes that must be stopped, the DR operation fails, indicating which processes are running. For example:

```
.
.
notify remove SUNW_cpu/cpu481 done
notify remove SUNW_cpu/cpu482 done
notify remove SUNW_cpu/cpu483 done
cfgadm: Hardware specific failure: unconfigure SB15:
Cannot
quiesce realtime thread: 621
```

To determine the name of the processes, use the command:

```
# ps -ef | grep PID
```

b. Stop the process in the appropriate manner. For example, the processes in our example must be stopped using the kill command:

```
# kill -9 PID
```

Retry the command in step 1.

4. To verify the board is disconnected and unconfigured, use the cfgadm command:

```
# cfgadm
Ap_Id          Type       Receptable     Occupant       Cond
.
SB2            CPU        disconnected   unconfigured   unknown
.
```

Now you can remove the board from the slot, or reassign it to another domain.

Caution: Do not remove the board until you have verified it is disconnected.

---

5. If you are immediately replacing the board, see "To add a board to a domain" on page 21. If you are reconfiguring the board to another domain, see "To reconfigure a board to another domain" on page 22. Otherwise, return the

cluster to operation without replacing the disconnected CPU/memory board using the procedure in the following section.

Adding a CPU/memory board

If you have unconfigured a CPU/memory board from a domain, you can remove it or reassign it to another domain. To add a CPU/memory board to a domain, you need not stop VCS.

To add a board to a domain

1. Log in as administrator to the domain where you plan to add or configure the boards.
2. If you are adding a new or a replacement board to a domain (for example, `wildcat`), verify the state of the slot to contain the board. To be configured with a new board, the slot must have the following states and condition:
   - Receptacle state: empty
   - Occupant state: unconfigured
   - Condition: unknown

   Verify this by using the `cfgadm` command to list the slots, as in the following example. In the `wildcat` domain, slot SB2 is to contain the CPU board:

   ```
   # cfgadm
   Ap_Id          Type      Receptable     Occupant      Cond
   .
   SB2            unknown   empty          unconfigured  unknown
   ```

   After you add the board to the slot, you can use the cfgadm command to verify that the state of the slot changes from "`empty`" to "`disconnected`."
3. Use the `cfgadm` command to connect and configure a CPU or memory board:

   ```
   cfgadm -v -c configure SBx
   ```

   For example:

   ```
   # cfgadm -v -c configure SB2
   assign SB2
   assign SB2 done
   poweron SB2
   poweron SB2 done
   test SB2
   test SB2 done
   connect SB2
   connect SB2 done
   configure SB2
   configure SB2 done
   notify online SUNW_cpu/cpu448
   notify online SUNW_cpu/cpu449
   ```

```
notify online SUNW_cpu/cpu450
notify online SUNW_cpu/cpu451
notify add capacity (4 cpus)
notify add capacity (2097152 pages)
notify add capacity SB2 done
```

4 Verify the new board has been connected and configured using the command `cfgadm`. For example:

```
# cfgadm
Ap_Id          Type       Receptable     Occupant       Cond
.
SB2            CPU        connected      configured     ok
```

To reconfigure a board to another domain

1 If you have unconfigured a board from one domain (for example, `wildcat`) and plan to configure it to another domain (for example, `cheetah`), verify the state of the slot containing the board.

To be configured to another domain, the board in the slot must have the following states and condition:

- Receptacle state: disconnected
- Occupant state: unconfigured
- Condition: unknown

2 Verify this by using the `cfgadm` command to list the boards, as in the following example. In the `cheetah` domain, slot `SB2` contains the CPU board that had been unconfigured from the `wildcat` domain:

```
# cfgadm
Ap_Id          Type       Receptable     Occupant       Cond
.
SB2            unknown    disconnected   unconfigured   unknown
.
.
```

3 Use the `cfgadm` command to connect and configure a CPU or memory board:

```
cfgadm -v -c configure SBx,
```

For example:

```
# cfgadm -v -c configure SB2
```

After the system configures and tests the board, it displays a message in the domain console log indicating the configuration of the components.

4 Verify the reconfiguration of the board using `cfgadm`:

```
# cfgadm
Ap_Id          Type       Receptable     Occupant       Cond
.
SB2            CPU        connected      configured     ok
.
.
```

5 You can log into the platform level and use the `showboards` command to verify that `SB2` is now part of the `cheetah` domain:

```
# showboards
Retrieving board information. Please wait
Location Pwr Type of Board   Board Status  Test Status   Domain
---------- -------------   ------------  -----------   ------
SB0      On     CPU            Active        Passed       wildcat
SB1      On     CPU            Active        Passed       wildcat
SB2      On     CPU            Active        Passed       cheetah
SB3      On     CPU            Active        Passed       wildcat
SB4      On     CPU            Active        Passed       wildcat
SB5      On     CPU            Active        Passed       cheetah
SB6      On     CPU            Active        Passed       cheetah
.
.
```

Dynamically reconfiguring I/O boards and cards

You can dynamically reconfigure I/O boards and PCI cards on I/O boards.

Dynamically reconfiguring PCI cards

A card containing a host bus adapter can be removed and replaced on an I/O board. If a failed HBA has been used with other adapters on separate cards in a dynamic multipathing (DMP) configuration, I/O can proceed through the alternate path and VCS need not be stopped.

To determine the status of the card you are unconfiguring

1 Log into the domain as the administrator. For the following example, the I/O board is in the `leopard` domain.

2 Check the status of the boards. On the `leopard` domain, use the `cfgadm` command:

```
# cfgadm
```

The output resembles:

```
Ap_Id                Type          Receptacle   Occupant
Condition
IO14                 HPCI          connected    configured   ok
IO15                 HPCI          connected    configured   ok
SB14                 CPU           connected    configured   ok
.
pcisch0:e15b1slot1   pci-pci/hp    connected    configured   ok
pcisch1:e15b1slot0   mult/hp       connected    configured
failed
pcisch2:e15b1slot3   pci-pci/hp    connected    configured   ok
pcisch3:e15b1slot2   ethernet/hp   connected    configured   ok
pcisch4:e14b1slot1   pci-pci/hp    connected    configured   ok
```

The failed card, `pcisch1:e15b1slot0`, is to be removed and replaced.

To remove a PCI card

1. Disable the controllers on the I/O system card using the `vxdmpadm` command:

   ```
   vxdmpadm disable ctlr=ctlr
   # vxdmpadm disable ctlr=c3
   ```

   If the card has more than one controller, repeat this command for each controller on the card.

2. Disconnect the card:

   ```
   # cfgadm -v -c disconnect pcisch1:e15b1slot0
   ```

3. Check the states and the condition of the card using the `cfgadm` command:

   ```
   # cfgadm
   ```

   The disconnected card must have the following states and condition:

   - Receptacle state: disconnected
   - Occupant state: unconfigured
   - Condition: unknown

4. Remove the disconnected card only if it is powered off.

To add a card

1. Verify that the slot you selected can accept a device, such as a PCI card. To accept a device, the slot must have the following states and condition:

   - Receptacle state: empty or disconnected
   - Occupant state: unconfigured
   - Condition: unknown

   Verify this by using the `cfgadm` command to list all of the system boards, as in the following example for the `leopard` domain:

   ```
   # cfgadm
   ```

   The output resembles:

   ```
   Ap_Id                Type         Receptacle    Occupant
   Condition
   IO14                 HPCI         connected     configured    ok
   IO15                 HPCI         connected     configured    ok
   SB14                 CPU          connected     configured    ok
   SB15                 CPU          connected     configured    ok
   c0                   scsi-bus     connected     configured
   unknown
   .
   .
   pcisch0:e15b1slot1   pci-pci/hp   connected     configured    ok
   pcisch1:e15b1slot0   unknown      disconnected  unconfigured
   unknown
   ```

```
pcisch2:e15b1slot3  pci-pci/hp   connected     configured    ok
pcisch3:e15b1slot2  ethernet/hp  connected     configured    ok
pcisch4:e14b1slot1  pci-pci/hp   connected     configured    ok
pcisch5:e14b1slot0  mult/hp      connected     configured    ok
pcisch6:e14b1slot3  pci-pci/hp   connected     configured    ok
pcisch7:e14b1slot2  ethernet/hp  connected     configured    ok
```

2. Add the replacement PCI card to the empty card slot.
3. To configure the new card, use the `cfgadm` command. For example:

   `# cfgadm -c configure pcisch1:e15b1slot0`

   After the system configures and tests the board, it displays a message in the domain console log indicating the configuration of the components.
4. Check the states and the condition of the board using the `cfgadm` command; it must be "`connected`," "`configured`," and "`ok`."
5. Enable the controller for the HBA:

   `vxdmpadm enable ctlr=ctlr`

   `# vxdmpadm enable ctlr=c3`

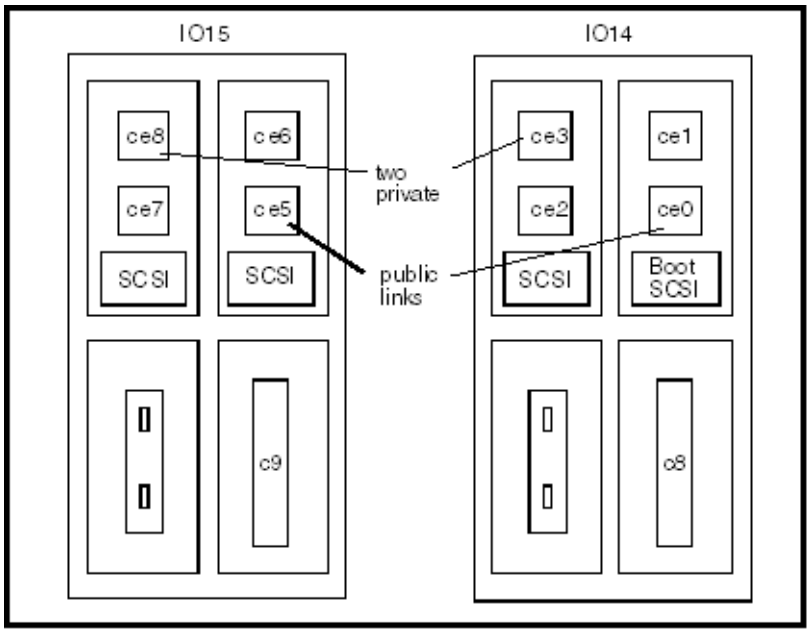
   Note that this command succeeds if the controller is accessible to the domain and I/O can be performed on it.

Dynamically reconfiguring an I/O board

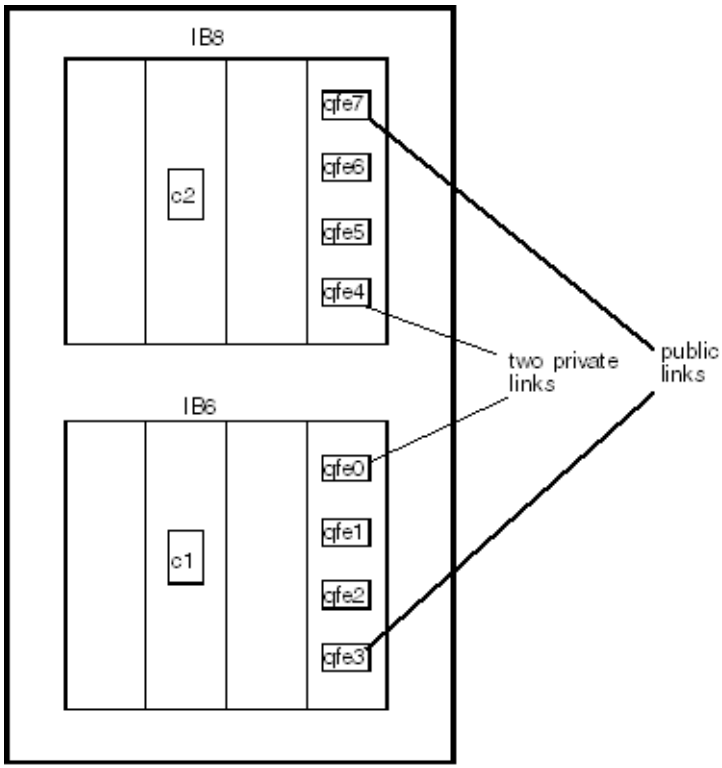
Under certain circumstances, you must stop VCS on the domain where you are reconfiguring a board. See "I/O boards - stopping VCS" on page 11.

In the following scenario, a cluster consists of the leopard and the S6800f0 domains. The cluster is running service groups on the leopard domain, which includes I/O boards IO14 and IO15. IO15 requires dynamic reconfiguration because of a malfunctioning component. The domain S6800f0 includes I/O boards IB8 and IB6. The disk controllers and NICs are labeled in the following diagrams.

Domain: Leopard



Domain: S6800f0



The highlights of the procedure to dynamically reconfigure the IO15 board in the leopard domain include:

- ✔ Disabling all the controllers on the board.
- ✔ Disabling all the NIC devices used for private communications on the board (this step is not necessary if you have stopped VCS)
- ✔ Disabling all the NIC devices used for public communications on the board (this step is not necessary if you have stopped VCS)
- ✔ Disabling the IO board and removing it
- ✔ Adding the replacement IO board
- ✔ Enabling the replacement board
- ✔ Enabling the public NIC devices
- ✔ Enabling the private NIC devices
- ✔ Enabling the controllers

To verify the status of the cluster and domain before DR

1 Use the VCS command `hastatus -sum` to verify the current state of the service groups in the cluster. Use the command before reconfiguring the I/O board and after reconfiguration to verify the cluster's state:

```
-- SYSTEM STATE
-- System               State                Frozen
A  leopard              RUNNING              0
A  s6800f0              RUNNING              0

-- GROUP STATE
-- Group           System       Probed     AutoDisabled  State
B  ServiceGroupA   leopard      Y          N             ONLINE
B  ServiceGroupA   s6800f0      Y          N             OFFLINE
B  cvm             leopard      Y          N             ONLINE
B  cvm             s6800f0      Y          N             ONLINE
```

2 By using the `cfgadm -al` command, you can show the I/O boards and cards in the `leopard` domain. For example:

```
# cfgadm -al
Ap_Id                  Type         Receptacle   Occupant
Condition
IO14                   HPCI         connected    configured    ok
IO14::pci0             io           connected    configured    ok
IO14::pci1             io           connected    configured    ok
IO14::pci2             io           connected    configured    ok
IO14::pci3             io           connected    configured    ok
IO15                   HPCI         connected    configured    ok
IO15::pci0             io           connected    configured    ok
IO15::pci1             io           connected    configured    ok
IO15::pci2             io           connected    configured    ok
IO15::pci3             io           connected    configured    ok
SB14                   CPU          connected    configured    ok
SB14::cpu0             cpu          connected    configured    ok
.
.
.
pcisch1:e14b1slot0     fibre/hp     connected    configured    ok
pcisch2:e14b1slot3     pci-pci/hp   connected    configured    ok
pcisch3:e14b1slot2     ethernet/hp  connected    configured    ok
pcisch4:e15b1slot1     pci-pci/hp   connected    configured    ok
pcisch5:e15b1slot0     fibre/hp     connected    configured    ok
pcisch6:e15b1slot3     pci-pci/hp   connected    configured    ok
pcisch7:e15b1slot2     ethernet/hp  connected    configured    ok
```

To determine the controllers on a board and disable them

1 Use the command `vxdmpadm listctlr all` to determine all controllers in the domain. For example, on the `leopard` domain:

```
# vxdmpadm listctlr all
CTLR-NAME       ENCLR-TYPE      STATE      ENCLR-NAME
=====================================================
c0              Disk            ENABLED    Disk
c9              HDS9960         ENABLED    HDS99600
c8              HDS9960         ENABLED    HDS99600
```

2 To determine which controllers are on a specific board, for example `IO15`, use the following commands to display information about the disks in the domain, their controllers, and the location of the controllers on the IO boards.

a Use the command `cfgadm -lv`, which provides a verbose listing of all boards in the domain. In the output, you can see the device slots listed for the board `IO15`.

```
# cfgadm -lv
```

In the following example (not all output is shown) the listing might contain lines that resemble:

```
.
pcish4:e15b1slot1  . . .
/devices/pci@1fc,700000:e15b1slot1
pcish5:e15b1slot0  . . .
/devices/pci@1fc,600000:e15b1slot0
pcish6:e15b1slot3  . . .
/devices/pci@1fd,700000:e15b1slot3
pcish7:e15b1slot2  . . .
/devices/pci@1fd,600000:e15b1slot2
.
```

The listing indicates that the device labeled `pci@1fc` is used by slots 0 and 1 of board 15, the device labeled `pci@1fd` is used by slots 3 and 2.

b Using the `format` command in the domain, you can list the disk devices. The listing may be lengthy, but in the output, the controller, indicated by "`c#`" in the first two characters of the device name, corresponds to a device that is listed in the previous command (step a). For example:

```
# format
c0t0d0 <SUN18G ..... /pci@1dc,700000/pci@1.. .....
.
c8t0d0 <HITACHI-OPEN ....
/pci@1dc,600000/fibre-channel ...
```

```
.
c9t0d0 <HITACHI-OPEN ....
/pci@1fc,600000/fibre-channel ...
```

c A comparison of the output of the previous two commands shows that board 15 slot 0 contains the controller c9.

3 Disable the controller `c9` on board 15 using the following command:

```
# vxdmpadm disable ctlr=c9
```

Using the `vxdmpadm` command, verify that controller `c9` is disabled:

```
# vxdmpadm listctlr all
CTLR-NAME       ENCLR-TYPE      STATE      ENCLR-NAME
=====================================================
c0              Disk            ENABLED    Disk
c9              HDS9960         DISABLED   HDS99600
c8              HDS9960         ENABLED    HDS99600
```

If necessary, disable each controller on a board being removed. In this example, the only controller on board IO15 is `c9`.

To list the status of the private network links and to disable them

---

Note: If you have stopped VCS, you may skip the procedures in this section.

---

1 Enter the command `lltstat -nv`:

```
# lltstat -nvs
LLT node information:
    Node                State       Links
    0 s6800f0           OPEN        2
    * 1 leopard           OPEN        2
    2                   CONNWAIT    0
    .
    .
    31                  CONNWAIT    0
```

The output shows that both domains have two links for private communication. Both links are "`OPEN`," that is, operational.

2 Display the `/etc/llttab` file on the `leopard` domain.

```
# cat /etc/llttab
set-node leopard
set-cluster 13
link cd3 /dev/ce:3 - ether - -
link cd8 /dev/ce:8 - ether - -
```

The devices `ce3` and `ce8` are shown as the private network links.

3 Disable the private network link device, `ce8`,on I/O board 15.

```
# /sbin/lltconfig -u ce8
```

4 Check the status of the private network links:

```
# lltstat -nv
LLT node information:
    Node                 State       Links
    0 s6800f0            OPEN        1
    * 1 leopard            OPEN        2
    2                    CONNWAIT    0
    .
    .
    .
    31                   CONNWAIT    0
```

To list the status of the public NICs and to disable them

Note: If you have stopped VCS, you may skip the procedures in this section.

---

1 Use the command `ifconfig -a`. For example, `ce0` (on board IO14) and `ce5` (on board IO15), the NICs used for the public network connections, are operational.

```
# ifconfig -a
lo0: flags=1000849<UP,LOOPBACK,RUNNING,MULTICAST,IPv4> mtu 8232
    index 1
    inet 127.0.0.1 netmask ff000000
ce4: flags=1000843<UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500
    index 2
inet 10.182.66.112 netmask fffff000 broadcast 10.182.79.255
    ether 0:0:be:a8:2c:5e
dman0:flags=1008843<UP,BROADCAST,RUNNING,MULTICAST,PRIVATE,IPv4
> mtu
    1500 index 3
    inet 10.2.1.6 netmask ffffffe0 broadcast 10.2.1.31
    ether 0:0:be:a8:2c:5e
ce0:
flags=9040843<UP,BROADCAST,RUNNING,MULTICAST,DEPRECATED,IPv4,
NOFAILOVER> mtu 1500 index 5
    groupname mn1
    ether 0:0:be:a8:2c:5e
ce0:1: flags=1000843<UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu
1500
    index 5
    inet 10.182.66.140 netmask fffff000 broadcast 10.182.79.255
ce5:flags=9040843<UP,BROADCAST,RUNNING,MULTICAST,DEPRECATED,IPv
4,
    NOFAILOVER> mtu1500 index 6
    inet 0.0.0.0 netmask ff000000
    groupname mn1
    ether 0:0:be:a8:2c:5e
```

2 To disable the device `ce5` on board IO15, use the command:

```
# ifconfig ce5 down
```

3 Use the `ifconfig -a` command to verify that `ce5` is down:

```
# ifconfig -a
```

No information about `ce5` should appear in the output.

To disable and remove the IO Board

1 When the controllers and network interface cards are disabled, disconnect board IO15:

```
# cfgadm -f -c disconnect IO15
```

2 Use the `cfgadm` command to check the status of `IO15`:

```
# cfgadm -al
Ap_Id                Type         Receptacle    Occupant
Condition
IO14                 HPCI         connected     configured    ok
IO14::pci0           io           connected     configured    ok
IO14::pci1           io           connected     configured    ok
IO14::pci2           io           connected     configured    ok
IO14::pci3           io           connected     configured    ok
IO15                 HPCI         disconnected  unconfigured
unknown
SB14                 CPU          connected     configured    ok
SB14::cpu0           cpu          connected     configured    ok
.
.
```

The I/O boasrd, `IO15`, may be physically removed at this time.

To add the new IO Board

Use the following procedure to add the new I/O board, a new `IO15`. Make sure that the output of the `cfgadm` command shows the slot where the new board is to be added has the status `disconnected`, `unconfigured`, and `unknown`.

1 Physically add the board, connecting all necessary cables, and configure it:

```
# cfgadm -c configure IO15
```

2 Run the `cfgadm -al` to verify the board has been configured; the board should be connected, configured and ok.

Note: If you have stopped VCS, you may skip step 3 through step 6.

---

3 Reconfigure the network interface cards on the new board:

```
# ifconfig ce5 plumb
# ifconfig ce5 up
```

4 Run the command `ifconfig -a` to verify `ce5` is up and running.

5 Reconfigure LLT to reestablish the private network links:

```
# /sbin/lltconfig -t ce8 -d /dev/ce:8
```

6 Verify the private network links are restored using the command `lltstat -nv`:

```
# /sbin/lltstat -nv
```

7 Enable the controller `c9` on the `IO15` using `vxdmpadm` command:

```
# vxdmpadm enable ctlr=c9
```

8 Verify the controller is up and running:

```
# vxdmpadm listctlr all
```

9 If you have stopped VCS before reconfiguring the I/O board, restart it. Refer to one of the appropriate sections, “Restarting VCS in a standard environment” on page 13 or “Restarting VCS in an Oracle9i RAC environment” on page 16.