Chapter 12

I/O Addressing

HP-UX Handbook
Revision 13.00

TERMS OF USE AND LEGAL RESTRICTIONS FOR THE HP-UX RECOVERY HANDBOOK

FEEDBACK or QUESTIONS: please email essam.ackleh@hp.com
(please use subject syntax: *HP-UX Handbook v13.00 Chapter <YY> - <Feedback Title>*

TABLE OF CONTENTS

Introduction

The HP-UX addressing scheme assigns fixed and hierarchical **hardware paths** to the devices based on the SCSI-II specification. The access to the devices is done through associated **device (special) files**. During the initialization of the kernel at boot-up time the system searches for connected hardware and builds an **IO-tree**.

I/O Architectures – SIO and WSIO

SIO (Server Input/Output) is only relevant for systems based on **HP-PB** (HP Precision Bus). These are legacy system like the E/F/G/H/I-Class, K-Class, T5xx-Class, and D-Class servers.

WSIO (Workstation Input/Output) does not mean it is restricted to workstations. WSIO is relevant for all systems that use the **HSC bus** (Highspeed System Connect) like the K-Class or the **EISA bus** like the D-Class. All the current systems like N-/L-/A-Class servers, SuperDome, Keystone, Matterhorn, IA-64 systems and modern workstations use the **PCI bus** (Peripheral Component Interconnect) and are also subjected to the WSIO architecture.

NOTE: SIO and WSIO architecture may coexist on a system (e.g. K-Class). SIO is sometimes also called NIO (Native Input/Output). HSC is sometimes also called GSC (General System Connect).

The table below shows a list of common device tapes and their drivers:

Device Type	**Device Class**	**SIO** (HP-PB)	**WSIO** (HSC/GSC/ PCI/EISA)
SCSI bus SE	`ext_bus`	`scsi1`	`c700`
SCSI bus F/W, Ultra wide SE	`ext_bus`	`scsi3`	`c720`
SCSI C895 Ultra Wide LVD	`ext_bus`		`c8xx`
SCSI Ultra320	`ext_bus`		`Mpt`
Serial Attached SCSI (SAS)	`ext_bus`		`sasd_vbus`
SAS MPT	`escsi_ctlr`		`Sasd`
Disk device	`disk`	`disc3`	`Sdisk`
Tape device	`tape`	`tape2`	`Stape`
Parallel Interface (Centronics)	`ext_bus`	`lpr2`	`CentIf/SCentIf`
MO Autochanger	`autoch`	`autox0`	`Schgr`
MO disks (Surface)	`surface`	`ssrfc`	`Ssrfc`
Target	`target`	`target`	`Tgt`
Pass-through devices (e.g. picker, scanner, ...)	`ctl`	`spt`	`Sctl`

SCSI Ultra320: mpt-driver

The mpt-driver offers the possibility of managing and troubleshooting the Ultra320 SCSI HBA (Host Bus Adapter) with HP-UX. There are two commands available:

```
/opt/mpt/bin/mptutil [options] <device_file>
```

for general information about the HBA, Protocol Statistics and Firmware download.

```
/opt/mpt/bin/mptconfig [options] <device_file>
```

shows the SCSI configuration and changes Initiator ID, SCSI speed, and SCSI width.

Example:

```
# mptconfig /dev/mpt2

 Scan For Devices ...

---- ADAPTER INFORMATION -----------------------------------------------------

 Device File                  : /dev/mpt2
 Hardware Path                : 0/1/1/0

---- BUS PARAMETERS ----------------------------------------------------------

 Initiator SCSI ID            : 7
 SCSI Bus Rate                : Ultra160
 SCSI Bus Width               : Wide

---- CHANNEL CAPABILITIES ----------------------------------------------------

 Req/Ack Offset               : 127
 Bus Mode                     : LVD
 Quick Arbitration Selection  : Enabled
 DT Clocking                  : Enabled
 Packetized                   : Enabled

---- TARGET PARAMETERS -------------------------------------------------------
 Target   Description          Firmware   In Use      In Use
   Id                          Version    Rate        Width
------------------------------------------------------------------------------
    0     ST336706LC           HP05       Ultra160    Wide
```

Serial Attached SCSI (SAS)

With the newest generation of ZX2 based servers (rx6600, rx3600, rx2660), HP introduced a new type of internal SCSI disks called SAS disks. SAS is a full-duplex serial point-to-point connection between a HBA and a device (also called *PHY*). Like other protocols as Fibre

Channel or iSCSI, it incorporates SCSI commands. The normal SCSI drivers can be used to access the devices, but a special driver is needed for the HBA. Currently, the following SAS cards exist:

- PCI-X 8-Port Internal SAS Controller Card for HP-UX, OVMS, Linux used as Core I/O for internal disks of rx6600, rx3600, rx2660
- Direct attached disks or RAID 1

Information about SAS Core I/O card for rx2660, rx3600 and rx6600 (HP-UX configuration)

The SAS Core I/O card for HP-UX provides access to up to 8 internal SAS disks (rx6600 has two of them). The controller provides simple RAID 1 mirroring capabilities (on a hardware level, no *mirror disk UX* needed), but the disks can also be accessed individually.

What are the differences to what we know about SCSI disks?

- Disks are known by the SAS Address (like WWN in fibre channel)
- The hardware pathes are assigned dynamically by the sasd. Physical location is not related to hardware path.
- disks can be directly used or combined in RAID 1 mirrors

Output of `ioscan`

```
# ioscan -fH 0/4/1/0
escsi_ctlr   0  0/4/1/0           sasd        CLAIMED    INTERFACE    HP   PCI/PCI-X SAS
MPT Adapter
ext_bus      1  0/4/1/0.0.0       sasd_vbus   CLAIMED    INTERFACE    SAS  Device
Interface
target       2  0/4/1/0.0.0.2     tgt         CLAIMED    DEVICE
disk         3  0/4/1/0.0.0.2.0   sdisk       CLAIMED    DEVICE       HP   DG072A9BB7
target       3  0/4/1/0.0.0.3     tgt         CLAIMED    DEVICE
disk         4  0/4/1/0.0.0.3.0   sdisk       CLAIMED    DEVICE       HP   DG072A9BB7
target       4  0/4/1/0.0.0.4     tgt         CLAIMED    DEVICE
disk         5  0/4/1/0.0.0.4.0   sdisk       CLAIMED    DEVICE       HP   DG036A8B5B
target       5  0/4/1/0.0.0.7     tgt         CLAIMED    DEVICE
disk         8  0/4/1/0.0.0.7.0   sdisk       CLAIMED    DEVICE       HP   DG036A9BB6
target       6  0/4/1/0.0.0.10    tgt         CLAIMED    DEVICE
disk        11  0/4/1/0.0.0.10.0  sdisk       CLAIMED    DEVICE       HP    IR Volume
```

Note that the disks are driven by the sdisk driver as normal SCSI disk, but the controller is driven by the `sasd` driver. The directly attached disk shows up with the model string, the last device is a RAID 1 volume (*IR Volume*) of two individual disks.

sasmgr

sasmgr is the HPUX tool to view status and modify the configuration of the SAS controller.

```
sasmgr get_info -D /dev/sasd0 -q vpd
Vital Product Data Information
------------------------------
Product Description                 : PCI-X SERIAL ATTACHED SCSI HBA
Part Number                         : 399490-001
Engineering Date Code               : A-4606
Serial Number                       : NA
Misc. Information                   : PW=15W PCI 66MHZ PCI-X 133MHZ CORE IO
Manufacturing Date                  : 4616
Manufacturing ID                    : N/A
Checksum                            : 0xdd
EFI Version                         : 02.00.14.00
HBA Firmware Version                : 01.16.00.00
Asset Tag                           : NA

sasmgr get_info -D /dev/sasd0 -q raid

Wed Dec 13 12:21:19 2006

---------- PHYSICAL DRIVES ----------
LUN dsf               SAS Address          Enclosure    Bay      Size(MB)

/dev/rdsk/c1t2d0     0x500000e012691cb2     1             3      70007
/dev/rdsk/c1t3d0     0x500000e0126926d2     1             4      70007
/dev/rdsk/c1t4d0     0x500000e01122d7d2     1             5      34732
/dev/rdsk/c1t7d0     0x500000e01263fcc2     1             8      34732

---------- LOGICAL DRIVE 4 ----------

Raid Level                                                  : RAID 1
Volume sas address                                          : 0x611d224fa01c82
Device Special File                                         : /dev/rdsk/c1t10d0
Raid State                                                  : OPTIMAL
Raid Status Flag                                            : ENABLED
Raid Size                                                   : 34000
Rebuild Rate                                                : 20.00 %
Rebuild Progress                                            : 100.00 %

Participating Physical Drive(s) :

SAS Address          Enc    Bay    Size(MB)       Type            State

0x500000e01268e312   1      1      70007          SECONDARY       ONLINE
0x500000e012691ee2   1      2      70007          PRIMARY         ONLINE

sasmgr get_info -D /dev/sasd0 -q lun=all -q lun_locate
/dev/rdsk/c1t2d0          0/4/1/0.0.0.2.0          1      3      OFF
/dev/rdsk/c1t3d0          0/4/1/0.0.0.3.0          1      4      OFF
/dev/rdsk/c1t4d0          0/4/1/0.0.0.4.0          1      5      OFF
/dev/rdsk/c1t7d0          0/4/1/0.0.0.7.0          1      8      OFF

RAID VOL ID is 4 :
/dev/rdsk/c1t10d0         0/4/1/0.0.0.10.0

Physical disks in volume are :
       1      1      OFF           HP            DG072A9BB7         HPD0
       1      2      OFF           HP            DG072A9BB7         HPD0
```

How to create a LUN

With `sasmgr`, the command would be:

```
sasmgr [-h][-f] add -D device_file -q raid -q level=raid_level -q
enc_bay=enc:bay[,enc:bay] [-q size=size] [-q rebuild_rate=rate]
```

(see manual page for details)

How to replace a directly attached disk

Example:

```
sasmgr set_attr -D /dev/sasd0 -q lun=/dev/rdsk/c3t1d0 -q locate_led=off
locate LED set to ON.
```

When a disk is unplugged and a new one is inserted into the same slot (or in another slot; this does not matter for SAS), then some commands have to be issued, because the new disk will get a hardware path as the SAS address is new to the controller.

```
disk  2  0/4/1/0.0.0.1.0  sdisk     NO_HW      DEVICE     HP   4 -> Original path
                              /dev/dsk/c3t1d0   /dev/rdsk/c3t1d0
disk  3  0/4/1/0.0.0.2.0  sdisk     CLAIMED    DEVICE     HP   4 -> New path for the
replacement disk
```

In this case the ***replace_target*** option of the `sasmgr` command has to be used to reassign the original hardware path (and device file) to the replacement disk. **NOTE:** this command cannot be used if there are outstanding I/O requests on the old physical volume. `lvreduce` has to be used.

```
# sasmgr replace_tgt -D /dev/sasd0 -q old_tgt=/dev/dsk/c3t2d0 -q
new_tgt_hwpath=0/4/1/0.0.0.1.0

WARNING: This is a DESTRUCTIVE operation.
This might result in failure of current I/O requests.
Do you want to continue ?(y/n) [n]...
LUN has been replaced with new Target.
```

Device Special Files

Devices can be classified in two categories, **raw** (or **character**) and **block**.

A **character special file** is a special file associated with I/O devices that transfer data byte-by-byte. Other byte-mode I/O devices include printers, nine-track magnetic tape drives, and disk drives when accessed in ``raw" mode. A character special file has no predefined structure.

A **block special file** is a special file associated with a mass storage device (such as a hard disk or tape cartridge drive) that transfers data in multiple-byte blocks, rather than a series of individual bytes. Block special files can be mounted. A block special file provides access to the device where hardware characteristics of the device are not visible.

NOTE: Accessing a disk in block mode, i.e. by the block special file, means using the **file system buffer cache**. Accessing a disk in raw mode (i.e. character special file) bypasses the buffer cache. Always use the character special to test a disk.

Major and Minor Number

A device special file is uniquely identified by its major and minor number:

```
   brw-r-----   1 bin     sys       31 0x09f000 Jul  7 15:55 c9t15d0
block device                     major #  minor #          file name
```

the major number of a certain driver can be determined using the `lsdev(1M)` command, e.g.:

```
# lsdev -b 31
    Character      Block       Driver          Class
      188           31         sdisk           disk

# lsdev | grep lvm
       64           64         lv              lvm
```

File Name

A device special file becomes associated with a device when the file is created (usually in the `/dev` directory), using the `mksf(1M)`, `insf(1M)`, or `mknod(1M)` commands. When creating device special files, it is recommended to use the following standard naming convention:

```
/dev/prefix/devspec[options]
      |        |       |
/dev/dsk/   c1t6d0               (a disk special)
/dev/rmt/   c0t3d0   BESTn       (a tape special)
```

`dev` device special files are located in `/dev`.

`prefix` indicates the subdirectory for the device class (for example, `rdsk` for raw device special files for disks, `dsk` for block device special files for disks, `rmt` for raw tape devices).

`devspec` indicates hardware path information and is typically in the format `c#t#d#` as follows:

- `c#` Instance number assigned by the operating system to the interface card (ext_bus). There is no direct correlation between instance number and physical slot number. The instance numbers are stored in the file `/etc/ioconfig` to guarantee that they do not change across reboots.
- `t#` Target address on a remote bus (for example, SCSI address).
- `d#` Device unit number at the target address (for example, SCSI LUN).

ptions Further qualifiers, such as disk section s# (for backward compatibility), tape density selection for a tape device, or surface specification for magneto-optical media.

Hardware path information can be derived from ioscan(1M) output.

The following is an example of a **disk** device special file name:

```
/dev/dsk/c1t6d0
```

where dsk indicates block disk access and c1t6d0 indicates disk access at interface card instance 1 (see ioscan output for the instance number of ext_bus, where the disk is connected to) target address 6, and unit 0. Absence of s# indicates access to the entire disk (see disk(7) for details).

The following is an example of a **tape** device special file name:

```
/dev/rmt/c0t3d0BESTn
```

where rmt indicates raw magnetic tape, c0 indicates that the device is connected to interface card instance 0, t3 indicates that target device address is set to 3, d0 indicates that the tape transport resides at unit address 0, and BESTn identifies device specific information. BEST ensures the highest capacity is used and n means “no rewind on close”. See mt(7) for details.

NOTES:

- In the past, other naming conventions have been used for device special files. Using ln(1) to create a link between the old and new standard name is useful as a temporary expedient until all programs using an old naming convention have been converted.
- There is no difference in device naming for workstations (s700) and servers (s800).
- The naming convention follows the SVR4 standard of the UNIX System Laboratories (USL) like e.g. SunOS 5.x also does.

Example ioscan output:

```
# ioscan -fn

Class        I  H/W Path       Driver    S/W State   H/W Type     Description
=============================================================================
root         0                 root      CLAIMED     BUS_NEXUS
ioa          0  0              sba       CLAIMED     BUS_NEXUS    System Bus Adapter
(803)
ba           0  0/0            lba       CLAIMED     BUS_NEXUS    Local PCI Bus Adapter
(782)
lan          0  0/0/0/0        btlan     CLAIMED     INTERFACE    HP PCI 10/100Base-TX
Core
                              /dev/diag/lan0  /dev/ether0     /dev/lan0
ext_bus      0  0/0/1/0        c720      CLAIMED     INTERFACE    SCSI C895 Ultra2 Wide
Single-Ended
target       0  0/0/1/0.1      tgt       CLAIMED     DEVICE
```

```
disk        14  0/0/1/0.1.0    sdisk     CLAIMED     DEVICE       HP  DVD-ROM 304
                              /dev/dsk/c0t1d0   /dev/rdsk/c0t1d0
target       1  0/0/1/0.3      tgt       CLAIMED     DEVICE
tape         0  0/0/1/0.3.0    stape     CLAIMED     DEVICE       HP  C1537A
                              /dev/rmt/0m           /dev/rmt/0mn
/dev/rmt/c0t3d0BEST    /dev/rmt/c0t3d0BESTn    /dev/rmt/c0t3d0DDS
/dev/rmt/c0t3d0DDSn
                              /dev/rmt/0mb          /dev/rmt/0mnb
/dev/rmt/c0t3d0BESTb   /dev/rmt/c0t3d0BESTnb   /dev/rmt/c0t3d0DDSb
/dev/rmt/c0t3d0DDSnb
target       2  0/0/1/0.7      tgt       CLAIMED     DEVICE
ctl          0  0/0/1/0.7.0    sctl      CLAIMED     DEVICE       Initiator
                              /dev/rscsi/c0t7d0
ext_bus      1  0/0/2/0        c720      CLAIMED     INTERFACE    SCSI C87x Ultra Wide
Single-Ended
target       3  0/0/2/0.6      tgt       CLAIMED     DEVICE
disk         0  0/0/2/0.6.0    sdisk     CLAIMED     DEVICE       SEAGATE ST39102LC
                              /dev/dsk/c1t6d0   /dev/rdsk/c1t6d0
target       4  0/0/2/0.7      tgt       CLAIMED     DEVICE
ctl          1  0/0/2/0.7.0    sctl      CLAIMED     DEVICE       Initiator
                              /dev/rscsi/c1t7d0
ext_bus      2  0/0/2/1        c720      CLAIMED     INTERFACE    SCSI C87x Ultra Wide
Single-Ended
target       5  0/0/2/1.6      tgt       CLAIMED     DEVICE
disk         1  0/0/2/1.6.0    sdisk     CLAIMED     DEVICE       SEAGATE ST39102LC
                              /dev/dsk/c2t6d0   /dev/rdsk/c2t6d0
```

See also `intro(7)` man page about device special files.

Administering the I/O Structure

The following commands are used to administer the systems IO structure. To create or delete device files `insf(1M)`, `mksf(1M)`, `rmsf(1M)` and `mknod(1M)` can be used. Instance numbers can be modified using `ioinit(1M)`.

Install Special File (`insf`)

The `insf` command installs special files in the devices directory `/dev`. If required, `insf` creates any subdirectories that are defined for the resulting special file. If no options are specified, special files are created for all new devices in the system. New devices are those devices for which no special files have been previously created. A subset of the new devices can be selected with the `-C`, `-d`, and `-H` options. With the `-e` option, insf **reinstalls** the special files for pseudo-drivers and existing devices. This is useful for restoring special files when one or more have been removed. Usually, `insf` displays a message as the special files are installed for each driver. The `-q` (quiet) option suppresses the installation message. The `-v` (verbose) option displays the installation message and the name of each special file as it is created.

NOTE: Using `insf -e` recreates **all** device files. As long as the naming convention is not violated, using the command will not do any harm.

Examples:

Install special files for all new devices belonging to the `tty` device class:

```
# insf -C tty
```

Install special files to the new device added at hardware path 2/4.0.0:

```
# insf -H 2/4.0.0
```

Remove Special File (rmsf)

If no options are specified, `rmsf` removes only the special files specified on the command line. Refer to the man page for details.

Example:

Removing special file at HW path `0/0/2` and all files below:

```
# rmsf -H 0/0/2
```

NOTE: `rmsf` does not remove the information about instance numbers (which are stored in the `/etc/ioconfig` file), i.e. when re-creating the previously removed special file with `insf` you do not get a different instance number.

Make Special File (mksf)

The `mksf` command is useful for creating single device files. Usually the `insf` command should be sufficient.

The `insf` command creates special files for DDS drives with the `BEST` option. The `BEST` option is used to achieve best possible density and compression that the tape drive supports. The default special file for the tape `/dev/rmt/0m` is identical to the BEST special file. To write without compression to a DDS drive another special file is required. It can be created with the `-b` option of `mksf`:

Example:

```
# mksf -H 40.4.0 -b DDS1 /dev/rmt/0m_DDS1      (for the 2000DC)
# mksf -H 40.4.0 -b DDS2 /dev/rmt/0m_DDS2      (for the 4000DC)
```

NOTE: The `-r` option specifies that `mksf` should make a character (raw) device file instead of the default block device file for drivers that support both, e.g. disk driver or MO-Autochanger.

Make Node (mknod)

The `mknod` command should only be used to create non-standard device files (e.g. for the SCSI pass through driver (`spt`) or the volume group control file):

```
# mknod name b|c major minor
```

`name`	name of the special file
`b|c`	block or character special
`major`	the major number
`minor`	the minor number

Example:

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

I/O Configuration (`ioinit`)

The `ioinit` command maintains the consistency between the kernel I/O data structures and file `/etc/ioconfig`, where the systems IO configuration is saved. `ioinit` is invoked by the init process when the system is booted, based on the ioin entry in `/etc/inittab`:

```
ioin::sysinit:/sbin/ioinitrc > /dev/console 2>&1
```

where `ioinitrc` is a script to invoke `ioinit` with the `-i` and `-r` options.

Given the `-i` option, `ioinit` checks consistency between the kernel I/O data structures (initialized with `/stand/ioconfig`, which is accessible for NFS-diskless support when the system boots up) and information read from `/etc/ioconfig`. If these are consistent, `ioinit` invokes `insf` to install special files for all new devices. If the kernel is inconsistent with `/etc/ioconfig`, `ioinit` updates `/stand/ioconfig` from `/etc/ioconfig`. It reboots the system, if the `-r` option is given.

If `/etc/ioconfig` is corrupted or missing when the system reboots, `ioinitrc` boots the system into single-user mode. The user should then restore `/etc/ioconfig` from backup or invoke the `ioinit` with the `-c` option to recreate `/etc/ioconfig` from the kernel (resulting in instance numbers that may be reordered, thus resulting in different device files).

The next section contains a procedure that describes how to reassign the instance numbers.

How to Change an Instance Number

Not only in cluster environments is it often useful to have a consistent device file naming. The following procedures may be used to change the instance numbers of such devices. These numbers determine the naming of the corresponding device files. `ioconfig` provides the mapping between instance numbers used by the kernel and the information the I/O system uses to communicate with peripheral devices (hardware paths). Two copies are maintained (`/stand/ioconfig` and `/etc/ioconfig`) or (`/stand/ioconfig_ext` and `/etc/ioconfig_ext`).

At boot time the `ioconfig` information is stored in the `io_tree` kernel data structure (see `ioinit(1M)`). The only purpose of `ioconfig` is to maintain configuration information when the system is not running. Even if hardware is removed from the system all mappings keep in place. This guarantees that no new device file names will appear after such changes. If removed hardware is added back to the system the original mapping can be reused, since it is still present in the `ioconfig` files.

Usually we want to change mappings for disk and LAN devices. Change the corresponding LAN instance numbers for LAN devices directly. For disk devices take care of the `ext_bus` instance numbers. The numbers of such "External Busses" (aka card instances) are responsible for the "c" numbers being part of disk device names. Look at the following extract of an `ioscan -fn` output:

```
ext_bus     3  2/0/1      c720      CLAIMED    INTERFACE   Built-in SCSI
target      0  2/0/1.3    tgt       CLAIMED    DEVICE
target      1  2/0/1.5    tgt       CLAIMED    DEVICE
disk        4  2/0/1.5.0  sdisk     CLAIMED    DEVICE      SEAGATE ST15150N
                          /dev/dsk/c3t5d0   /dev/rdsk/c3t5d0
```

The ext_bus instance number (3) is responsible for the "`c3`" in "`c3t5d0`". The target (`t5`) and the LUN (`d0`) are unchangeable by software. They are mapped directly to the underlying hardware configuration.

For HPUX 11.11, 11.23, there are two procedures to change an instance number available. The procedure I is easy and quick while the procedure II is more complicated. Usually procedure I is sufficient. If it fails, then use procedure II. Procedure II usually works in all cases but requires two reboots.

Procedure I

Default procedure, requires one reboot and works without additional tools.

1) Extract a configuration template from the current ioscan output

```
# ioscan -f | grep -e INTERFACE -e DEVICE | \
grep -v target | \
awk '{print $3, $1, $2}' > /infile
```

2) Edit `/infile` and change the `ext_bus` and `lan` instances as desired. No class is allowed to get more than one line for the same instance!

3) Gracefully change the system to runlevel 1

```
# init 1
```

4) Apply the ioconfig change

```
# /sbin/ioinit -f /infile -r
```

The system will reboot immediately if the change is successful. Warnings like "`Input is identical to kernel`" can be ignored.

If unsuccessful, the most likely error to happen is:

```
"ioinit: Instance number X already exists for class XXX"
```

The desired instance assignment conflicts with an existing instance number. If that instance is bound to hardware that is no longer visible in `ioscan`, then you are in trouble and need to perform procedure II or III.

5) Verify the changes. Once the system is up, verify that all instance numbers were changed as expected. It may be necessary to re-import volume groups to ensure that `/etc/lvmtab` contains the correct entries. The LAN configuration may need to be changed also.

Procedure II

Reliable, requires two reboots and works without additional tools.

1) Extract a configuration template from the current ioscan output

```
# ioscan -f | grep -e INTERFACE -e DEVICE | \
grep -v target | \
awk '{print $3, $1, $2}' > /infile
```

2) Edit /infile and change the ext_bus and lan instances as desired. No class is allowed to get more than one line for the same instance!

3) Move away the current ioconfig files and shutdown/reboot

```
# mv /stand/ioconfig /stand/ioconfig.sav
# mv /etc/ioconfig /etc/ioconfig.sav
# shutdown -ry 0
```

4) Recreate ioconfig files. Due to the missing ioconfig files the system will come to an ioinitrc prompt. Now recreate new ioconfig files from scratch. This prevents you from running into possible assignment conflicts.

```
(in ioinitrc)# /sbin/ioinit -c
```

5) Apply the ioconfig change with your prepared infile

```
(in ioinitrc)# /sbin/ioinit -f /infile -r
```

The system will reboot again now if the change was successful. Warnings like “`Input is identical to kernel`” can be ignored.

6) Verify the changes. Once the system is up, verify that all instance numbers were changed as expected. It may be necessary to re-import volume groups to ensure that /etc/lvmtab contains the correct entries. The LAN configuration may need to be changed also.

NOTE: If anything goes wrong, you can always put the `ioconfig` copy `/etc/ioconfig.sav` and `/stand/ioconfig.sav` back in place and reboot.

For 11.31, the procedures to change an instance number can be done online without the reboot. Here is the example for both legacy and persistent devices.

To change instance numbers for persistent and legacy device files, create an infile by using the following 2 commands:

```
# ioscan -kNf|grep -e INTERFACE -e DEVICE|awk '{printf "%s %s %s\n",$3,$1,$2}' |grep
disk > /tmp/infile (for persistent disk devices)

# ioscan -kf|grep -e INTERFACE -e DEVICE|awk '{printf "%s %s %s\n",$3,$1,$2}' |grep
disk >> /tmp/infile (for legacy disk devices)
```

Example of infile:

```
# cat /tmp/infile

64000/0xfa00/0x0 disk 3
64000/0xfa00/0x1 disk 4
64000/0xfa00/0x2 disk 5
64000/0xfa00/0x3 disk 7 <-instance to change
0/0/2/0.0.0.0 disk 2
0/1/1/0.0.0 disk 1
0/1/1/0.1.0 disk 0
0/1/1/1.2.0 disk 6
```

In this example, to change persistent device instance 7 to 6, the infile would need to be modified as follows:

```
# vi /tmp/infile
64000/0xfa00/0x0 disk 3
64000/0xfa00/0x1 disk 4
64000/0xfa00/0x2 disk 5
64000/0xfa00/0x3 disk 6 <-changed to 6 from 7
0/0/2/0.0.0.0 disk 2
0/1/1/0.0.0 disk 1
0/1/1/0.1.0 disk 0
0/1/1/1.2.0 disk 7 <-also needed to be changed
```

Since a legacy disk device 0/1/1/1.2.0 exists with instance number 6; it also required to changing with unique instance number.

NOTE: Persistent and legacy disk devices use the same class, called disk. So, all the disk instance numbers must be unique across legacy and persistent disk device instance numbers.

The other extra entries may be removed from the /tmp/infile which are not required any instance number changes. However, the other lines will ignored by the ioinit as the input is identical to kernel.

Example:

```
# cat /tmp/infile
64000/0xfa00/0x3 disk 6 <-changed to 6 from 7
0/1/1/1.2.0 disk 7 <-also needed to be changed
```

Now, run ioinit to change the instance numbers.

```
# ioinit -f /tmp/infile
```

This will change the instance numbers instantly without any system reboot. Refer to the manpage of ioinit(1M) for more details.

NOTE: To change more than 14 devices at one time without reboot, the following 11iv3 patches or its superseded patches must be present / installed in the system:

PHCO_36315	1.0	ioinit(1M) patch
PHKL_36333	1.0	Support over 14 instance number changes from ioinit

Adressing Storage Libraries

A **storage library** (as used in this document) is a SCSI or Fibre Channel mass storage device, consisting of a mechanical changer device (also known as picker device or robotic), and one or more storage devices (e.g. tape drives or optical disk drives).

The **picker device** moves media between slots and drives within the autochanger.

Two levels of functionality are provided withing HP-UX. The picker device can either be accessed **directly** to move media within the library. Alternatively, media surfaces of optical libraries can be accessed as unique devices, causing a medium changer device driver to **automatically** move the media into a drive to perform an I/O request. This feature is *not* available for tape libraries, which always need to be controlled directly via the picker device.

Depending on system architecture (WSIO or SIO), one of two medium changer device drivers (**schgr** or **autox0**) provides access to the medium changer device. Refer to the *I/O Architecture section* of this chapter for an explanation of the two system architectures.

A **surface driver** (`ssrfc`) provides access to the surfaces of the magneto-optical disks much like a disk device. An `open()` of a surface device file causes the corresponding medium to be moved into an idle (empty) drive, then the requested disk I/O operation (e.g `newfs(1M)`) is performed. Upon `close()` of the device, the medium is returned to its original storage location within the library.

The `ssrfc` module is provided specifically to support Hewlett-Packard magneto-optical disk autochanger products. It is not possible to concurrently mount more MO disks than drives are present. If there are e.g two drives available in an autochanger and two MO disks are already mounted, then a third MO disk can only be accessed if a MO disk is released before (umount). "Virtual mounting" was available up to HP-UX 9.X only.

Direct (manual) access to the picker device (mandatory for tape libraries, but also possible for optical libraries) is accomplished by a SCSI pass-through interface. Depending on system architecture (WSIO or SIO), one of two pass-through kernel drivers (`sctl` or `spt`) provides access to the medium changer device. The `schgr` and `autox0` drivers also offer a pass-through device (in addition to their surface device files). All of these pass-through drivers follow the SCSI specification for medium changer devices to provide a generic medium changer interface, making it feasible to construct an application level driver for any mechanical changer, jukebox, library, or autochanger device (MO, tape, CD-ROM).

Library Kernel Drivers

The system's IO architecture (SIO or WSIO) decides which driver to use for autochanger, medium changer device, disk and tape. Refer to the *I/O Architecture section* for details on SIO, WSIO, and neccessary device drivers.

The **medium changer** is called **`schgr`** (s_changer) for SIO- and **`autox0`** (autochanger0) for WSIO architecture. The **surface driver** is called **`ssrfc`** (s_surface) is identical for both SIO and WSIO.

The **picker device** needs the SCSI pass-through driver **`spt`** (SIO) or **`sctl`** (WSIO) to be configured in the kernel. For `sctl` the picker will not be claimed. `ioscan` will show it as `UNCLAIMED UNKNOWN`. This is not a problem!

Refer to the *Kernel chapter* to find out how to add drivers to the kernel.

Configuration using a Media Changer Device Driver

Device File Names

The device naming convention for the drivers enables accessing the changer device (via a pass-through interface), as well as individual media surfaces for optical libraries.

Block devices reside in **/dev/ac**, character devices reside in **/dev/rac**. Within these directories, names are derived from the usual c#t#d# device naming convention (explained in intro(7)), with a surface descriptor appended at the end:

cI**t**T**d**L**_surface**

I	Card instance number of the changer device
T	Target address (SCSI-ID) of the changer device
L	LUN of the changer device
surface	for the changer device: 0 or non specified
	for the surface: slot number and side (a or b)

Example: The media changer is connected to the SCSI interface with **instance number 1** and has the **SCSI ID 3** with **SCSI LUN 0**. This results in the pass-through device file:

/dev/rac/c**1**t**3**d0 character special

An optical medium in **slot 2** would have the following device files for **side B**:

/dev/rac/c1t3d0_**2b** character special
/dev/ac/c1t3d0_**2b** block special

NOTE : there is no block special file for the pass-through device.

Major and Minor Numbers

The **Major numbers** of autochanger slots are:

for schgr (WSIO): **230** (character special), **30** (block special)
for autox0 (SIO): **231** (character special), **29** (block special)

The **Minor numbers** are as follows:

0xIITSSS

II	Instance number of the SCSI interface (ext_bus), e.g. 01
T	SCSI ID of the changer controllers, e.g. 3
SSS	Surface name
	= 000 for the changer itself
	= 2 * slot -1 for side A and
	= 2 * slot for side B (in hex), e.g.

Slot	Side	SSS
changer device		000
1	A	001
1	B	002
2	A	003
2	B	004
...	…	…
143	A	11D

143	B	`11E`
n	A	`2n-1` (in hex)
n	B	`2n` (in hex)

E.g.: Slot = **32**, Side = **A**
==> `sss` = 2 * 32 - 1 = 63 (dec) = **03F** (hex)

Device files are created using the `insf(1M)` command. If your library has more than 32 slots then you need to specify an extended range using the –p option, e.g. for 105 slots:

```
# insf -C autoch -e -p 1:105
insf: Installing special files for autox0 instance 0 address 10/4/8.3.0
...............................
```

Following are long listings showing the major and minor numbers associated with the device special file names of the first surface and the changer:

for `schgr`:

```
brw-rw-rw- 1 root sys  29 0x043001 Apr 22 10:22 /dev/ac/c4t3d0_1a
crw-rw-rw- 1 root sys 231 0x043001 Apr 22 10:22 /dev/rac/c4t3d0_1a
crw-rw-rw- 1 root sys 231 0x043000 Apr 22 10:22 /dev/rac/c4t3d0
```

for `autox0`:

```
brw-rw-rw- 1 root sys  30 0x043001 Apr 24 11:35 /dev/ac/c4t3d0_1a
crw-rw-rw- 1 root sys 230 0x043001 Apr 24 11:35 /dev/rac/c4t3d0_1a
crw-rw-rw- 1 root sys 230 0x043000 Apr 24 11:35 /dev/rac/c4t3d0
```

Refer to the `autochanger(7)` manual page for more details.

Configuration of the Picker Device

You can take control over the picker without using a media changer device driver. This is done by a SCSI pass-through driver, `sctl` (for WSIO) or `spt` (for SIO). Applications like Omniback are usually configured to handle libraries with this option. The command line utility `mc(1M)` allows you to pick a medium and load/unload it to/from certain slots/drives (see section Testing the Picker below).

Device File Name

You may use any name for the picker device file. Frequently used names are `/dev/picker` or `/dev/robotic`. In this example we follow the naming convention `/dev/rscsi/c`**`I`**`t`**`T`**`d`**`L`** for Instance **I**, SCSI target **T**, SCSI LUN **L**.

Major and Minor Number

The **Major number** of the picker device is:

for SIO:	**75** usually	(use `lsdev -d spt`)
for WSIO:	**203**	(use `lsdev -d sctl`)

The **Minor number** is as follows (values are in Hex!):

```
0xIITL00
```

`II`	Instance number of the SCSI interface (ext_bus), e.g. `0a` (dec. 10)
`T`	SCSI ID of the changer controller (target) , e.g. `2`
`L`	SCSI LUN of the changer controller, e.g. `0`
`00`	this is fixed

To create the device file:

```
# mknod /dev/rscsi/cItTdL c <Major No. of spt/sctl> <Minor No.>
```

e.g.:

```
# mknod /dev/rscsi/c10t2d0 c 203 0x0a2000
```

An `ioscan` output for a WSIO configuration with a C1160F jukebox could look like:

```
# ioscan -fn
...
ext_bus    10  8/4      c720        CLAIMED   INTERFACE GSC add-on Fast/Wide SCSI
target      0  8/4.2    tgt         CLAIMED   DEVICE
ctl         3  8/4.2.0  sctl        UNCLAIMED UNKNOWN   HP      C1160F
target      1  8/4.3    tgt         CLAIMED   DEVICE
disk        0  8/4.3.0  sdisk       CLAIMED   DEVICE    HP      C1113F
                        /dev/dsk/c10t3d0   /dev/rdsk/c10t3d0
target      2  8/4.4    tgt         CLAIMED   DEVICE
disk        1  8/4.4.0  sdisk       CLAIMED   DEVICE    HP      C1113F
                        /dev/dsk/c10t4d0   /dev/rdsk/c10t4d0
...
```

NOTE: The **sctl driver does not claim** the device, so the `UNCLAIMED` state above is completely normal. On the other hand the **spt driver needs to claim the device**, but needs to be forced to do this by adding a driver statement to the system file (`/stand/system`). Add a line like the following to /stand/system, rebuild the kernel and reboot:

```
driver   <H/W Path>   spt
```

e.g.:

```
driver   8/4.2.0   spt
```

Refer to `scsi_ctl(1M)` or `scsi_pt(1M)` manual pages for details regarding the SCSI pass-through drivers.

Testing the Picker

The `mc(1M)` command (media changer manipulation utility) can be used to check the state of the library or to move media inside the library.

Inquiry of the Library

```
# mc -p <robotic device> -q
=> Vendor-ID; Product-ID; Firmware
```

Information about Slots and Drives

```
# mc -p <robotic device> -e DS
=> NUM_DT_ELEMENTS ; NUM_ST_ELEMENTS
```

Statistic of all Slots and Drives

```
# mc -p <robotic device> -r DS
```

Example:

```
# mc -p /dev/rscsi/c10t2d0 -r DS
DT_slot_1 FULL
ST_slot_1 EMPTY
ST_slot_2 FULL
ST_slot_3 FULL
ST_slot_4 FULL
ST_slot_5 FULL
ST_slot_6 FULL
```

In this example we have 6 storage slots and 1 drive slot. The storage slot 1 is empty whereas the other storage slots and the drive slot are full.

Moving Media

```
# mc -p <robotic device> -s <source> -d <destination>
```

Examples:

This unloads tape from drive D1 into slot S1

```
# mc -p /dev/rscsi/c10t2d0 -s D1 -d S1
```

This loads tape from slot S4 into drive D1

```
# mc -p /dev/rscsi/c10t2d0 -s S4 -d D1
```

Querying the SCSI ID of a drive

e.g. for drive 1:

```
# mc -p /dev/rscsi/c10t2d0 -a 1
DT_slot_1   2
```

In this case drive 2 is configured with SCSI ID 2.

NOTE: If the `LIBRARY` environment variable is set then the `-p` option can be omitted:

```
# export LIBRARY=/dev/rscsi/c10t2d0
```

Fibre Channel Basics

Fibre Channel (FC) is a communication protocol that has been developed to satisfy the needs of a growing demand for data throughput. The advantages are as follows:

- different channels and protocols over one physical interface
- high bandwidth (200MB/s or higher)
- flexible setup (topology)
- connections across great distances (a couple of kilometers)
- supports different speeds, media and connections

Fibre Channel combines the advantages of channel and networking technology. A channel is a limited, direct, structured and predictable mechanism for data transmission between a few participants. A channel is typically used in situations where a high transfer rate is needed (e.g. connection of peripheral devices like disks, tape drives, printer or workstations). Well known channel protocols are e.g. SCSI or HIPPI.

Networks are non-structured and unpredictable. They are able to adapt automatically to changing environments and allow a higher number of participants. Hence a higher administration effort (mostly in software) is neccessary in order to establish a connection between two points in the network. Thus networks are slower than channels. Well known network protocols are Ethernet, Token Ring or FDDI.

This chapter covers the usage of Fibre Channel in order to address mass storage devices. This is called FCMS (Fibre Channel Mass Storage). FC is used as the transmission media for SCSI connections. The SCSI protocol is encapsulated.

The essential advantages of FC compared to SCSI:

	SCSI	FC
Distance	12m	10km
Speed	320MB/s	400MB/s
Number of devices	15 (4bit addressing)	126 ports in private Loop mode. 24bit addressing with fabric
Reliability		extremely low error rate
Robustness		no HW damages by pulling cables

FC Topologies

A physical connection at a port consists of two separate fibers because FC is a serial protocol. The fibers to receive (RX) and transmit (TX) are bundled in one cable (often colored orange). Basically you are able to distinguish between the different topologies by two criteria:

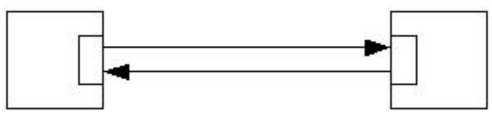
- Do we have a loop?
- Is a swich connected?

Loop	fabric	topology
Yes	no	private (arbitrated) loop
Yes	yes	public loop
No	no	direct point-to-point
no	yes	switched point-to-point (*)

(*): also known as direct fabric attach (DFA)

Point-to-Point (P2P)

In the p2p case (no loop) two FC devices are directly interconnected with each other. The transmitting diode of one device is connected to the receiving diode of the other and vice versa. The full bandwidth can be used for data transmission. The initialization of the link (login) is quite easy. HP did only implement the switched P2P topology. P2P without fabric can be regarded as a two port loop, i.e. the overhead of the arbitrated loop protocol is used even if there are only 2 ports.

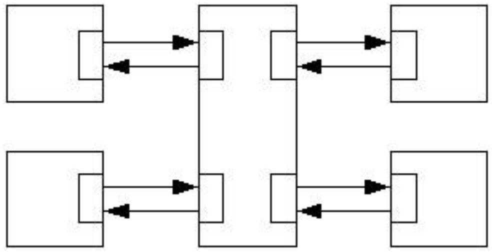


advantages	disadvantages
dix, full bandwidth for the link	high expense in terms of hardware
	no scalability

Fabric

Using the fabric topology you can have up to 2^{24} (approx. 16,7 million) nodes in a meshed configuration.

The advantage of this topology is the fact that multiple devices can communicate with each other simultaneously, each granted the full bandwidth. The use of FC switches is mandatory.



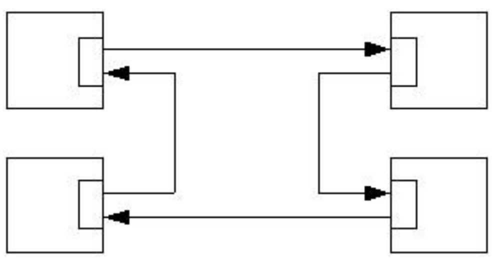
If an *N_Port* registers at a switch it is given a native address-ID (S_ID) by the switch (details in the FC addressing part). Further features of a fabric are multicast server, broadcast server, alias server, quality of service facilitator and directory server. Some switches have *FL_Ports* in order to run arbitrated loops.

advantages	disadvantages
high scalability, i.e. many devices (up to 16 mio)	higher initial cost compared to AL
multiple devices communicate simultaneously	slightly lower throughput compared to AL
loss of one component does not interrupt the link	
full bandwidth for each switch port	
performance only minor dependant to length	

NOTE: Fabric mode is supported as of UX 11.00 Tachyon TL driver version >= B.11.00.03. Be sure to install the current Fibre Channel Mass Storage Driver Patch.

Arbitrated Loop

The Arbitrated Loop (AL) topology is able to connect up to 126 ports that share the total bandwitdh. All nodes act as loop devices.



All devices in a loop need a unique ID (Loop ID). In a default configuration, Mass-Storage Devices should have a fixed ID given by the administrator. The host will get a Loop ID by sending a LIP (Loop Initialization Primitiv). A LIP occurs, if a host is rebooted or a device is added to the loop (via a hub). Many LIPs without those causes are an indicator for an unstable loop (bad cable, transmitter, receiver) and could decrease the performance of the Loop significantly!

AL is not a “token passing” protocol. If a port within an AL is about to transmits data it needs to arbitrate the loop in order to obtain the control over the loop. It sends the ARBx signal (arbitrate primitive; x=AL_PA of the ports). When this ARBx signal returns, the port has the control over the loop. After it has sent the OPN signal (open primitive) to the target port, a point-to-point connection (sortof) between the two ports has been established. The other ports act as repeater.

If more than one port arbitrates the loop at the same time, the port with the lower AL_PA (i.e. higher loop id) wins. Only if this port abandons the control over the loop, the other ports will be able to arbitrate the loop again. In contrast to a token passing protocol there is no limitation in how long a port can have control over the loop. Optionally, an access fairness algorithm may be used in order to give all other nodes the possibility to arbitrate the loop befor the same node gets access again..

Like with most of the ring topologies the setup of an AL is simplified by using **hubs**, because standardized cables can be used. A hub is able to detect connected or disconnected devices. A faulty device or a defect cable does no longer tear down the whole network.

Advantages	Disadvantages
cost effective	all nodes share the total bandwith
scalability	up to 126 nodes per loop
	if one component fails a new initialization is needed
	performance is strong dependant of length and number of nodes

There are private und public loops.

A **private loop** corresponds to the known implementation of the FC-AL. The adressing is done with the der AL_PA, an 8-bit identifier. All nodes in the loop can talk to each other, but not to nodes outside of the loop.

In a **public loop** the nodes register with their *World Wide Name* at the FL-port of the switch and get a 3-byte identifier assigned, i.e. they are capable of the **fabric login**. The lower byte corresponds to the AL_PA for the communication within the loop whereas the upper 2 bytes characterize the connected switchport. The name server table of the switch stores the mapping between world wide names and 3 byte identifiers. This enables public loop devices to communicate with nodes outside of the loop.

LoopID versus AL_PA

The *Arbitrated Loop Physical Address* (AL_PA) is used for addressing within an arbitrated loop. This hexadecimal value (10 bit) is determined dynamically during initialization of the loop. The equivalent to the AL_PA is the decimal LoopID (8 bit). The table below shows the relation between LoopID and AL_PA:

Loop ID	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00-0F	EF	E8	E4	E2	E1	E0	DC	DA	D9	D6	D5	D4	D3	D2	D1	CE
10-1F	CD	CC	CB	CA	C9	C7	C6	C5	C3	BC	BA	B9	B6	B5	B4	B3
20-2F	B2	B1	AE	AD	AC	AB	AA	A9	A7	A6	A5	A3	9F	9E	9D	9B
30-3F	98	97	90	8F	88	84	82	81	80	7C	7A	79	76	75	74	73
40-4F	72	71	6E	6D	6C	6B	6A	69	67	66	65	63	5C	5A	59	56
50-5F	55	54	53	52	51	4E	4D	4C	4B	4A	49	47	46	45	43	3C
60-6F	3A	39	36	35	34	33	32	31	2E	2D	2C	2B	2A	29	27	26

70-7F	25	23	1F	1E	1D	1B	18	17	10	0F	08	04	02	01	00	--

Example: To determine the AL_PA for **LoopID 22** (decimal):

0d22 = 0x16
LoopID can be found in 2nd row (0x10–0x1F)
7th column holds the appropriate **AL_PA: 0xC6**

The priority grows with increasing LoopID and decreasing AL_PA respectively. The highest priority has Loop ID 126 (or 0x7E which corresponds to AL_PA 0x00). This ID is reserved for a switchport (FL_Port) that may be connected to the loop. The Loop IDs below 126 are dynamically assigned. These so-called soft addresses are used to address e.g. HBAs. Storage devices like disk arrays or tape libraries have to get fixed/hard coded Loop IDs because the Loop ID determines the hardware path and therefor the name of the device file.

FC HBAs, Software and Patches

A very good Overview about the SW-drivers and patches:

- Starting with AR1201 release of the td driver, A6684A, A6685A, A5158A, A6795A share a common driver for both HP-UX 11.0 and 11.11.
- Starting with the web release of the fcd driver in Oct 28, 2003, A6826A, A9782A, A9784A share a common driver for HP-UX 11.11
- Starting with the web release of the fcd driver in March 24, 2004, A6826A, A9782A, A9784A share a common driver for HP-UX 11.23
- Starting with the AR0305 release of the fcd driver in March 2005, A6826A, A9782A, A9784A, AB465A share a common driver for HP-UX 11.11 and 11.23
- Starting with AR0512, release of fcd driver in December 2005, A6826A, A9782A, A9784A, AB465A, AB378A, AB379A share a common driver for HP-UX 11.11 and 11.23. They also share same 11.31 driver starting with HP-UX 11.31 release.

FCMS Dependency Patches:

- Dependency Patches for the driver. They must be installed prior to the installation of the driver.
- Driver Patches:
- The Driver Patch contains the latest fixes since the last release. It should be installed after the installation of the driver.

- To get the most current version of driver, make sure to install the driver from the latest AR or OE release plus the current driver patch (if any).

- For more details on the latest fixes of a particular release, check patchinfo file and/or AR/OE release notes at http://docs.hp.com, click on “I/O cards and Networking Software”, then “Fibre Channel”.

Fibre Channel Kernel Driver & Tunables

The following **kernel drivers** are FC related. Some are for legacy Tachyon adapters only, some are for Tachyon TL/TS/XL2 adapters only and some are suitable for all.

Driver	Tachyon	Tachyon TL/TS/XL2	Explanation
`fcms`	✓	✓	
`fcpmux`	✓	✓	if a FC-SCSI mux is used
`fcparray`	✓	✓	FCP Array Interface
`fcp_cdio`	✓	✓	
`fcpdev`	✓	✓	FCP Device Interface
`fcT1`	✓		Tachyon driver
`fcT1_fcp`	✓		
`fcT1_cntl`	✓		Fibre Channel Mass Storage Cntl
`fcp`	✓		FCP Protocol Adapter
`td`		✓	Tachyon TL/TS/XL2 driver
`fcd`			FC-Driver for all Combocards and 4Gbit HBA
`fclp`			Fibre Channel driver for Emulex-based FC HBAs
`fcoe`			Fibre Channel component of the HP NC551m Dual Port FlexFabric 10GbE Network Adapter.

There are the following FC-related **kernel tunables**. Do not modify them unless neccessary.

Tunable	Tachyon	Tachyon TL/TS/XL2	Explanation
fcp_large_ config	✓	✓	determines whether additional memory should be allocated for FC ports. A value of 0 (default) will handle up to 64 FC ports. A value of 1 indicates that more the 64 ports may be handled concurrently. Only for Arbitrated Loop!
num_tachyon_ adapters	✓		specifies how many Tachyon FCP adapters are installed in the system so that an appropriate amount of memory can be allocated for them at system start-up if the system does not provide I/O virtual addressing. Default of 0 lets the system decide.
max_fcp_reqs	✓		specifies the maximum number of concurrent FCP requests allowed on any Tachyon FCP installed in the machine. Default is 512.

See the following URL for details (Release Note & Installation Guide):

http://h20000.www2.hp.com/bizsupport/TechSupport/DocumentIndex.jsp?lang=en&cc=us&taskId=101&prodClassId=10007&contentType=SupportManual&docIndexId=64255&prodTypeId=12883&prodSeriesId=5039733

To determine which version of the drivers is installed:

```
# fcmsutil <FC-Devicefile> | grep "Driver Version"
Driver Version = @(#) libtd.a HP Fibre Channel Tachyon XL2 Driver
B.11.23.0512 $Date: 2005/09/20 12:22:47 $Revision: r11.23/1
```

Fibre Channel Addressing in HP-UX

The HP-UX addressing scheme assigns fixed and hierarchical hardware paths to the devices, based on the SCSI-2 specification. 14 bits are divided as follows:

Bus	(7 bit = 128 busses)
Target	(4 bit = 16 targets)
Lun	(3 bit = 8 luns)

The HW path to a SCSI disk would look like this:

```
[SCSI-HBA].[Bus].[Target].[Lun]
```

But Fibre Channel requires much more devices to be mapped:

- Every Loop can have up to 126 FC devices connected
- SAN address space is 224 FC ports (N_Port ID is 24 bit)
- SCSI-3 standard allows up to 2^{64} Luns for each target (i.e. per FC port)

Therefore the FC address space has to be adapted to the HP-UX structures. This is done by expanding the HW path by 3 more fields of 8 bit each, adding 24 bit address space.

The HW path of a FC device comprises the following three parts:

```
                additional FC fields      standard SCSI fields
[FC-HBA].[Domain].[Area].[Port].[Bus].[Target].[Lun]
```

- **HBA (host bus adapter) part:** It depends on the slot at the host where the FC adapter card is seated.
- **Fibre Channel part (N_Port ID):** This part comprises of the new 8 bit fields **Domain**, **Area** and **Port**. In general these fields are used to hold the **N_Port ID**. This 3 byte value can be regarded as the equivalence to the fixed 6 byte Media-Access Control identifier (MAC address) in the LAN world. In case of a fabric topology (a switch is present) the switch assigns the N_port ID dynamically during login and stores the information it's private name server table.
  - **Domain:** The Domain ID usually identifies the instance of the switch in a fabric environment. For private loops the value is 8. Therefore, SAN Switches should not have the Domain ID 8 !!
  - **Area:** In a fabric environment, the Area ID is generally associated with a physical port on a switch. For private loops, the value is 0.
  - **Port:** When interpreting the hardware path for FCMS devices, arrays are defined as having addressable controllers, and non-arrays are defined as not having addressable controllers. For hardware paths associated with non-array LUNs, the Port ID is set to the value 255 (255 for direct connect controller). For hardware paths associated with array LUNs, the Port ID is set to the Loop ID, for a private loop topology, AL_PA for public loop topology or 0 for direct fabric attach.
  - Domain and Area field hold the first two bytes of the N_Port ID. The third byte either comes from the Port field or from Bus and Target field of the SCSI part, depending on the addressing mode. The examples some pages ahead will help to understand this.

Topology	Fibre Channel Part		
	Domain_ID (8 bit)	Area_ID (8 bit)	Port_ID (8 bit)

Private Loop	Protocol Adapter defined in FC layer 4 (FC_4) **8** for Mass Storage 5 for Networking	always **0** for private loop	either **LoopID** or **255** in PDA addressing mode (see below)
Public Loop	usually identifies the Domain ID (i.e. **instance**) of the switch (at least for Brocade). 5 and 8 are reserved for private loop.	the physical connector (**slot**) on the switch. For Brocade subtract 16 from the Area ID to get the slot number	either **AL_PA** or **255** in PDA addressing mode (see below)
Direct Fabric Attach	*same as public loop*	*same as public loop*	either **0** or **255** in PDA addressing mode (see below)

NOTE: For the Brocade switches you can obtain the Domain ID of the switch either in the config menu of the front panel or when logged on via telnet using the `switchshow` command (user: admin, password: password).

- **(virtual) SCSI part:** This part reflects the connected devices (DLTs, LUNs etc.) in a matter that HP-UX is able to access them like SCSI. The SCSI part consists of the fields Bus, Target and Lun.

The grey blue orange color scheme will be used throughout the whole chapter in order to identify HBA, FC- and SCSI part. HP-UX differentiates between three **addressing modes**:

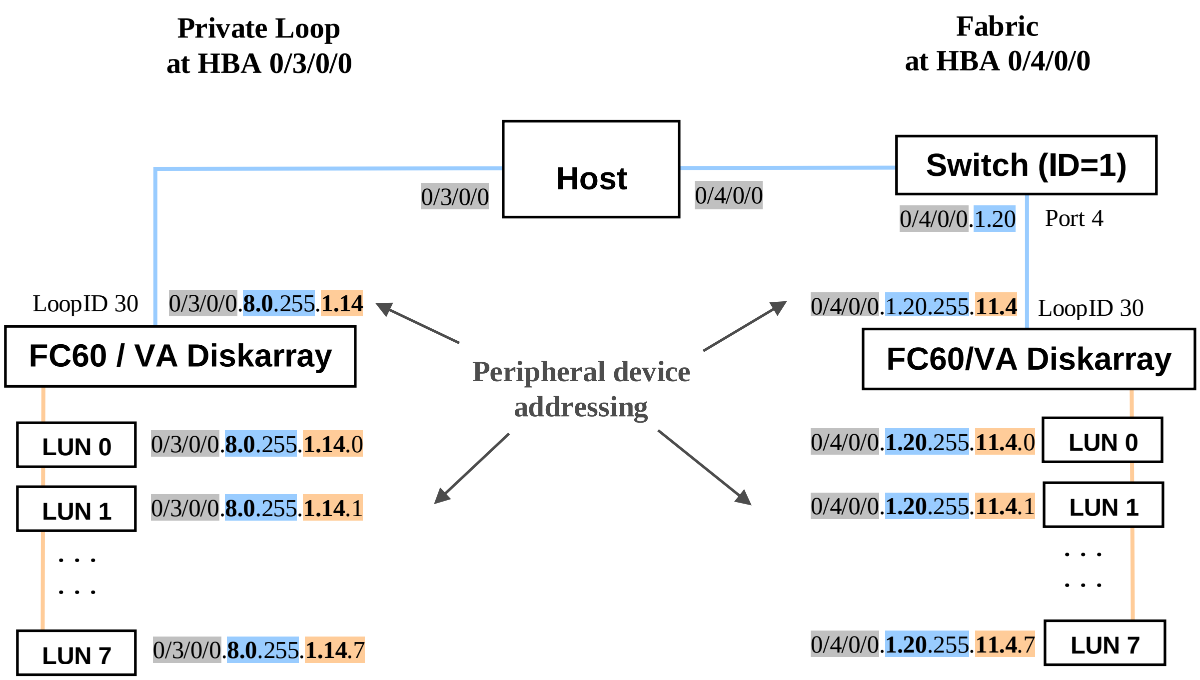
- Peripheral Device Addressing (PDA)
- Volume Set Adressing (VSA)
- Logical Unit Adressing (LUN)

The driver sends out an inquiry to get the type of the device and its capabilities. The addessing mode to use depends on the topology and the devices.

Peripheral Device Addressing (PDA)

This addressing mode can be identified by Loop ID (for private loop) or AL_PA (for public loop) as coded in the two fields' `bus` and `target`, whereas the SCSI devices are coded in the `LUN` field. The upper 4 bits of the 8 bit Loop ID/AL_PA go to the bus field, the lower 4 bit go to the target field. Given the HW path the Loop ID/AL_PA can be calculated: 16*Bus + Target. Only the 3 bit of the Lun field are left for addressing Luns, resulting in $2^3 = 8$ possible Luns.

There is no translation at the devices behind the FC controller. This is why the `port` field is always set to 255 (i.e. direct attach). The 255 in the port field is an indicator for PDA addressing. PDA is the standard addressing scheme. It is used whenever there is no need to exceed 8 Luns.



Example private loop:

Loop ID = **30** = 0x1E $\Rightarrow$ bus=0x1 = **1**, target=0xE = **14**

This results in the following HW path:

```
0/3/0/0.8.0.255.1.14.7
   |     | |  |  |  | |_Lun
   |     | |  |  |  |___Target  (lower 4 bits of Loop ID)
   |     | |  |  |______Bus  (upper 4 bits of Loop ID)
   |     | |  |_________Port  (255 for PDA)
   |     | |____________Area  (0 for private loop)
   |     |______________Domain/Protocol  (8 für Mass Storage)
   |____________________HBA
```

Example public loop:

Loop ID = 30 = 0x1E ==> AL_PA = **0xB4** $\Rightarrow$ bus=0xB = **11**, target=0x4 = **4**

This results in the following HW path:

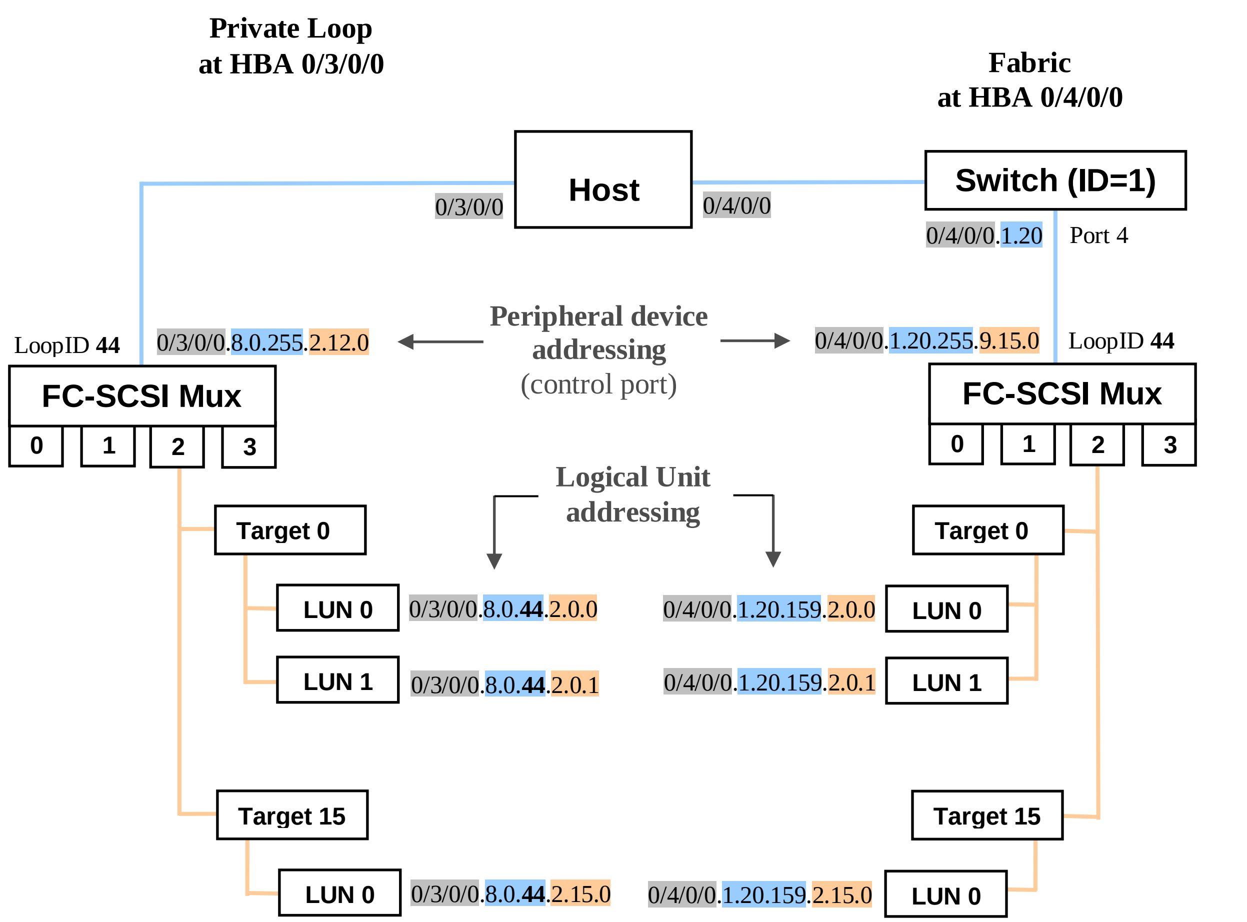
```
0/4/0/0.1.20.255.11.4.7
   |     |  |  |   | | |_Lun
   |     |  |  |   | |___Target  (lower 4 bits of AL_PA)
   |     |  |  |   |_____Bus  (upper 4 bits of AL_PA)
   |     |  |  |_________Port  (255 for PDA)
   |     |  |____________Area  (slot at switch)
   |     |_______________Domain  (Instance no. of the switch)
   |_____________________HBA
```

The N_Port ID is coded in the Domain field, the Area field and the AL_PA:
Domain= 1 = 0x01, Area = 20 = 0x14, AL_PA = 0xB4 ==> **N_Port ID=0x0114B4**

PDA addressed devices are:

- FC controller at XP, EMC, VA, FC60 disk arrays and FC-SCSI Mux
- FC4/2 bridge, FC 2/1 bridge
- Galactica DLT libraries
- FC10 disk devices
- Luns in Galaxy diskarray and Hitachi Data System (HDS) diskarray. NOTE: Galaxy has no addressable controller like XP- or EMC-diskarray, although the HDS is identical to HP's XP it cannot be addressed using VSA. See http://wtec.cup.hp.com/~hpux/io/current_issues/articles/9962369007867.html (HP internal)

Logical Unit Adressing (LUN)



In LUN adressing mode the Loop ID/AL_PA can be found in the port field. Bus, target and LUN fields are used to address SCSI devices as usual.

This addressing mode is used, because the bridge does a conversion from FC to; the SCSI devices are more structured. The `port` field shows the Loop-ID/AL_PA of the previous component (FC-port of the FC-SCSI bridge). The `bus` field holds the instance number of the SCSI bus whereas the fields `target` (SCSI ID) and `LUN` (LUN) are adressed like SCSI.

This addressing mode is required whenever more than 16 targets exist. The driver decides on SCSI inquiry wether LUN addressing will be used.

Example private loop:

Loop ID of the FC controller of the bridge = **44**

This results in the following HW path:

```
0/3/0/0.8.0.44.2.15.0
   |    | | |  |  | |_Lun
   |    | | |  |  |___Target  (SCSI ID)
   |    | | |  |______Bus  (virtual SCSI bus)
   |    | | |_________Port  (LoopID of the FC controller)
   |    | |___________Area  (0 for private loop)
   |    |_____________Domain/Protocol  (8 für Mass Storage)
   |__________________HBA
```

Example public loop:

Loop ID of the FC controller of the bridge = 44 = 0x2C ==> AL_PA = 0x9F = **159**

This results in the following HW path:

```
0/4/0/0.1.20.159.2.15.0
   |    |  |  |   |  | |_Lun
   |    |  |  |   |  |___Target (SCSI ID)
   |    |  |  |   |______Bus (virtual SCSI bus)
   |    |  |  |__________Port (AL_PA of the FC controller)
   |    |  |_____________Area (slot at switch)
   |    |________________Domain (Instance no. of the switch)
   |_____________________HBA
```

The N_Port ID is coded in the Domain field, the Area field and the AL_PA:
Domain= 1 = 0x01, Area = 20 = 0x14, AL_PA = 0x9F ==> **N_Port ID=0x01149F**

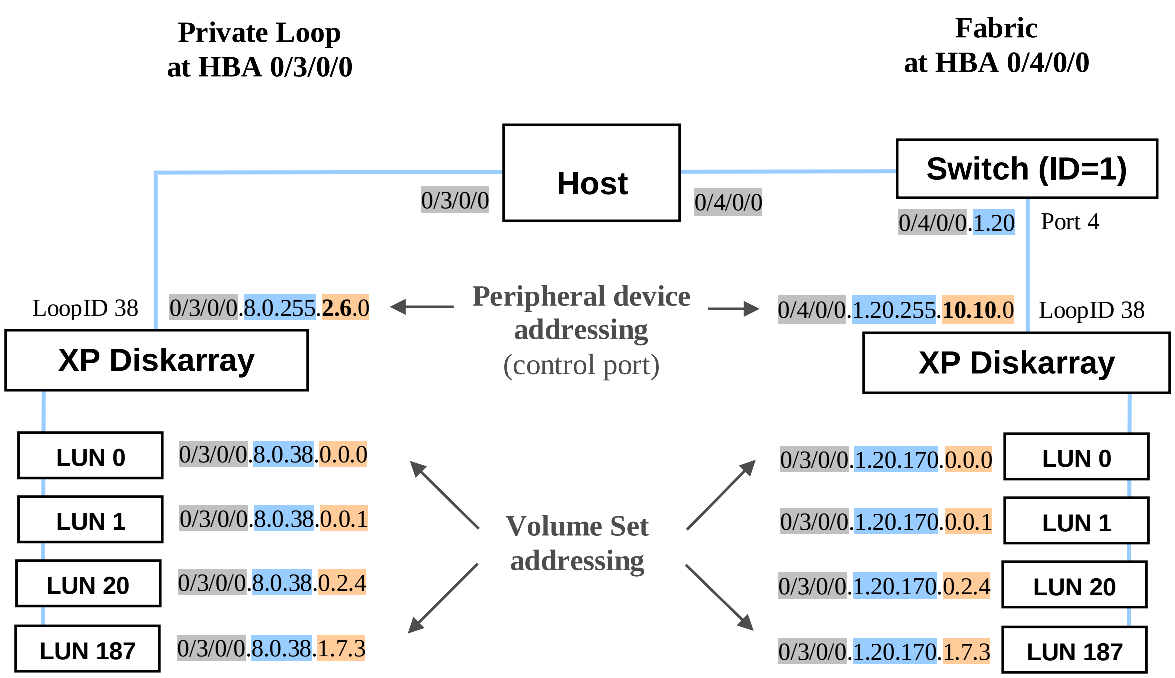
Examples for LUN addressed devices:

- SCSI interfaces of the FC-SCSI Mux and the devices behind this bus
- HP AutoRAID 12H
- XP diskarray in private loop

Volume Set Addressing (VSA)

This addressing mode was introduced to overcome the limitation of 8 Luns for disk arrays in PDA addressing, where each Volume is mapped to a SCSI LUN of a single SCSI target. The whole 14 bits of bus, target and LUN field are now used for the addressing of the LUNs, resulting in a maximum of 2^{14} = 16384 addressable Luns.

Similar to LUN addressing, the `port` field holds the Loop-ID/AL_PA of the previous component on the FC side (FC-Port of the disk array). The fields `bus`, `target` und `LUN` are used to map the configured LUNs on the disk array. The driver decides on SCSI inquiry wether VSA addressing will be used.



Example:

The Lun number (or Volume ID) generated on an XP disk array is **187**.

Bus Target Lun

Vol ID = 187 = $(0000001\ \ 0111\ \ 011)_2$

==> Bus = $(0000001)_2$ = **1**, Target = $(0111)_2$ = **7** und Lun = $(011)_2$ = **3**

typically you calculate like this:

Bus   = Vol ID DIV 2^7
Target = (Vol ID MOD 2^7) DIV 2^3
Lun   = (Vol ID MOD 2^7) MOD 2^3

and vice versa, starting with bus and target: and lun:

Vol ID = 128*Bus + 8*Target + Lun

Example private loop:

Loop ID of the FC controller of the XP = 38.

```
0/3/0/0.8.0.38.1.7.3
   |     | | |  | | |_Lun (= (Vol ID MOD 128) MOD 8)
   |     | | |  | |___Target  (= (Vol ID MOD 128) DIV 8)
   |     | | |  |_____Bus  (= Vol ID DIV 128)
   |     | | |________Port  (LoopID of the FC controller)
   |     | |__________Area  (0 for private loop)
   |     |____________Domain/Protocol  (8 für Mass Storage)
   |__________________HBA
```

Example public loop:

Loop ID of the FC controller of the XP = 38 = 0x26 ==> AL_PA = 0xAA = **170**.

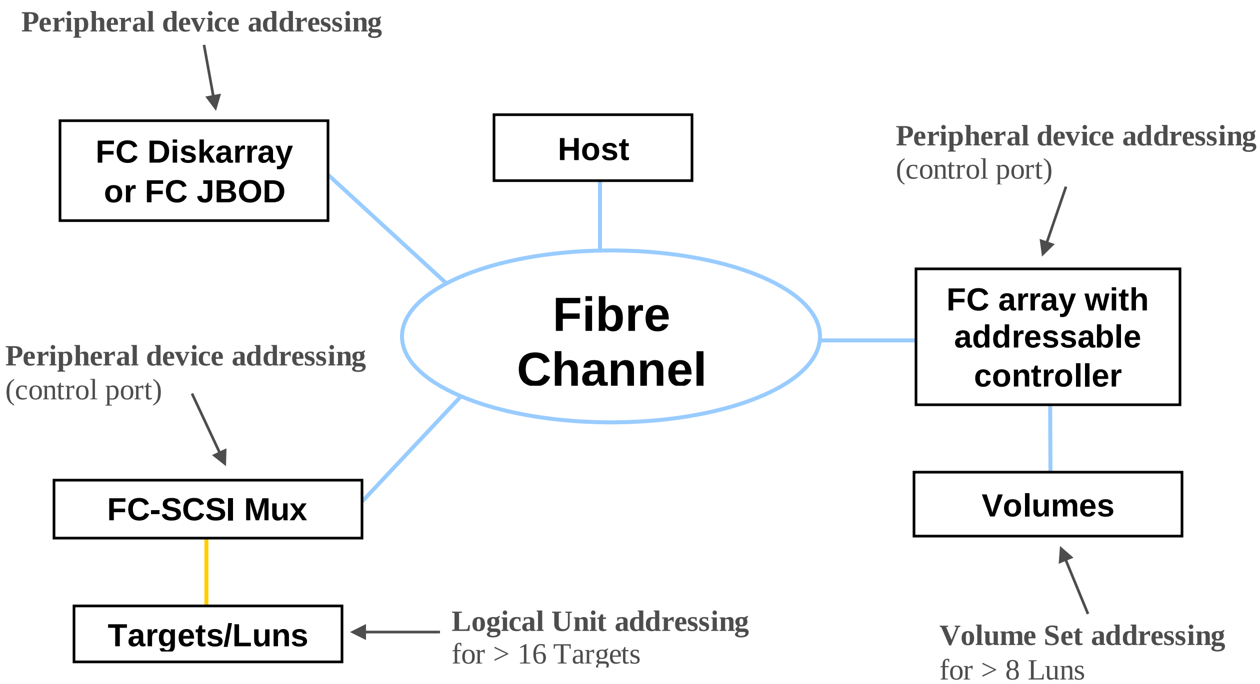
```
0/4/0/0.1.20.170.1.7.3
   |     |  |   |  | | |_Lun (= (Vol ID MOD 128) MOD 8)
   |     |  |   |  | |___Target  (= (Vol ID MOD 128) DIV 8)
   |     |  |   |  |_____Bus  (= Vol ID DIV 128)
   |     |  |   |________Port (AL_PA of the FC controller)
   |     |  |____________Area (slot at switch)
   |     |_______________Domain (Instance no. of the switch)
   |_____________________HBA
```

The N_Port ID is coded in the Domain field, the Area field and the AL_PA:
Domain= 1 = 0x01, Area = 20 = 0x14, AL_PA = 0xAA ==> **N_Port ID=0x0114AA**

According to PDA addressing the AL_PA is coded in the bus and target field of the HW path of the XP's control port:
AL_PA = 0xAA ==> target = 0xA = 10, bus = 0xA = 10 ==> 0/4/0/0.1.20.255.**10.10**.0.

VSA addressed devices are XP-, VA-, FC60- and EMC Diskarray Volumes.

The following picture summarizes the adressing modes explained above.



some basic rules to remember resulting from the above:

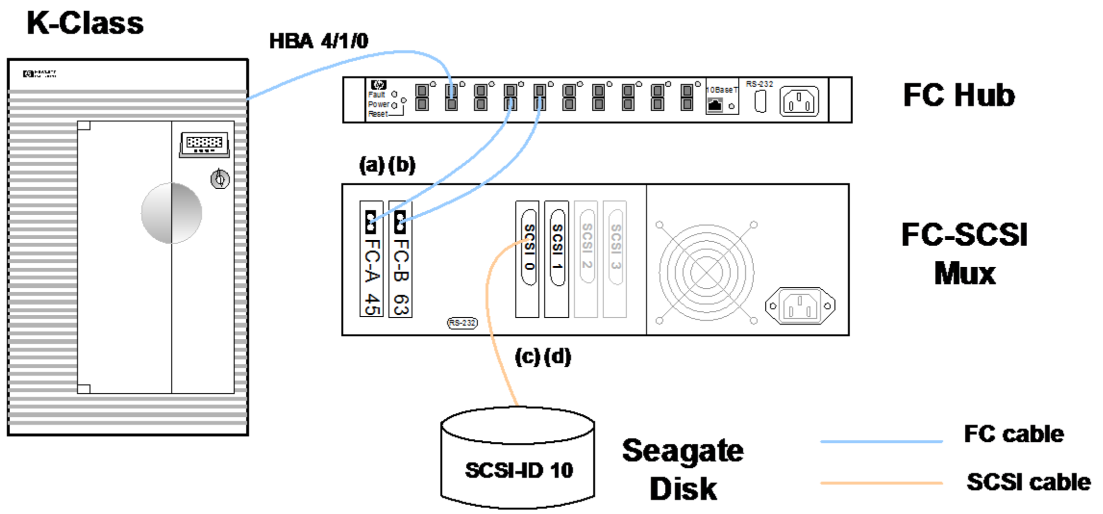
- If the **Domain ID is 8** the Area ID will be 0 and we are in a private loop, i.e without a switch.
- If the **Domain ID is not 0** it holds the ID of the attached switch and the Area ID will hold the switchport (subtract 16 for Brocade) where the device is connected. The device is operating in a public loop or in direct fabric attach mode (Port ID = 0).
- If the **Port ID is 255** the device is directly attached to the host over a hub or a switch and operates in PDA mode
- If the **Port ID is 0** the device operates in **direct fabric attach mode** (if the device supports it), otherwise the Port ID holds the Loop ID for private loop or AL_PA for public loop.
- switch means fabric, hub means (private) loop

The utility "SAN Toolbox" is able to do all the conversions explained above. It is a Windows executable: http://hprtnt06.grc.hp.com/central_san_cc/san_toolbox/index.htm (HP internal)

Example: Private Loop with FC-SCSI Mux

Tachyon HBA --- HUB --- FC-SCSI Mux --- SCSI Disk

The following example explains PDA and LUN addressing in a private loop. A FC-SCSI Mux with 2 FC adapter and 2 SCSI cards has one disk connected.



- The FC-SCSI Mux has a unique **Loop ID** in the range 0 - 125 assigned for each of its FC adapter. In this example: FC Adapter A = 45, FC Adapter B = 63
- It needs to be clear which of the 4 **SCSI Slots** (0 - 3) of the FCMS is equipped with a SCSI controller card. In this example: SCSI Slot 0, SCSI Slot 1. The connected disk is attached to the SCSI controller in Slot 0
- The **SCSI-ID** of the device needs to be note. For disk arrays you would also need the Lun number. In this example: SCSI-ID of the SEAGATE disk is 10

How is the `ioscan` output for this setup?

```
# ioscan -fn
...
...
ba          1  4                      epic       CLAIMED    BUS_NEXUS   PCI Bus Bridge -
epic
fc          1  4/1/0                  fcT1       CLAIMED    INTERFACE   HP Fibre Channel
Mass Storage Adapter
lan         2  4/1/0.5                fcT1_cntl  CLAIMED    INTERFACE   HP Fibre Channel
Mass Storage Cntl
                                     /dev/fcms2
fcp         1  4/1/0.8                fcp        CLAIMED    INTERFACE   FCP Protocol Adapter

(c) ext_bus 6  4/1/0.8.0.45.0         fcpmux     CLAIMED    INTERFACE   HP A3308 FCP-SCSI
MUX Interface
target      4  4/1/0.8.0.45.0.7       tgt        CLAIMED    DEVICE
ctl         5  4/1/0.8.0.45.0.7.0     sctl       CLAIMED    DEVICE      Initiator
                                     /dev/rscsi/c6t7d0
target      5  4/1/0.8.0.45.0.10      tgt        CLAIMED    DEVICE
disk        7  4/1/0.8.0.45.0.10.0    sdisk      CLAIMED    DEVICE      SEAGATE ST32550W
                                     /dev/dsk/c6t10d0   /dev/rdsk/c6t10d0

(d) ext_bus 7  4/1/0.8.0.45.1         fcpmux     CLAIMED    INTERFACE   HP A3308 FCP-
SCSI MUX Interface
```

```
      target      6   4/1/0.8.0.45.1.7       tgt        CLAIMED     DEVICE
      ctl         6   4/1/0.8.0.45.1.7.0     sctl       CLAIMED     DEVICE      Initiator
                                           /dev/rscsi/c7t7d0

(c)*  ext_bus     9   4/1/0.8.0.63.0         fcpmux     CLAIMED     INTERFACE   HP A3308 FCP-SCSI
      MUX Interface
      target     11   4/1/0.8.0.63.0.7       tgt        CLAIMED     DEVICE
      ctl         8   4/1/0.8.0.63.0.7.0     sctl       CLAIMED     DEVICE      Initiator
                                           /dev/rscsi/c9t7d0
      target     12   4/1/0.8.0.63.0.10      tgt        CLAIMED     DEVICE
      disk        8   4/1/0.8.0.63.0.10.0    sdisk      CLAIMED     DEVICE      SEAGATE ST32550W
                       /dev/dsk/c9t10d0   /dev/rdsk/c9t10d0

(d)*  ext_bus    10   4/1/0.8.0.63.1         fcpmux     CLAIMED     INTERFACE   HP A3308 FCP-SCSI
      MUX Interface
      target     13   4/1/0.8.0.63.1.7       tgt        CLAIMED     DEVICE
      ctl         9   4/1/0.8.0.63.1.7.0     sctl       CLAIMED     DEVICE      Initiator
                                           /dev/rscsi/c10t7d0

(a)   ext_bus     8   4/1/0.8.0.255.2        fcpdev     CLAIMED     INTERFACE   FCP Device
      Interface
      target      7   4/1/0.8.0.255.2.13     tgt        CLAIMED     DEVICE
      ctl         7   4/1/0.8.0.255.2.13.0   sctl       CLAIMED     DEVICE      HP   HPA3308
                                           /dev/rscsi/c8t13d0

(b)   ext_bus    11   4/1/0.8.0.255.3        fcpdev     CLAIMED     INTERFACE   FCP Device
      Interface
      target     14   4/1/0.8.0.255.3.15     tgt        CLAIMED     DEVICE
      ctl        10   4/1/0.8.0.255.3.15.0   sctl       CLAIMED     DEVICE      HP   HPA3308
```

(*alternative link to the same device)

Interpretation:

Host interface (HBA):

```
fc   1   4/1/0   fcT1   CLAIMED INTERFACE HP Fibre Channel Mass Storage Adapter
```

FC protocol adapter is 8 for mass storage:

```
fcp 1   4/1/0.8    fcp      CLAIMED      INTERFACE     FCP Protocol Adapter
```

The **FC adapter** (a) and (b) of the FC-SCSI Mux are addressed in PDA mode with their Loop-IDs. In ioscan they turn up behind the devices connected to the FC-SCSI Mux:

```
(a)   ext_bus     8   4/1/0.8.0.255.2        fcpdev     CLAIMED     INTERFACE   FCP Device
      Interface
      target      7   4/1/0.8.0.255.2.13     tgt        CLAIMED     DEVICE
      ctl         7   4/1/0.8.0.255.2.13.0   sctl       CLAIMED     DEVICE      HP   HPA3308
                                           /dev/rscsi/c8t13d0

(b)   ext_bus    11   4/1/0.8.0.255.3        fcpdev     CLAIMED     INTERFACE   FCP Device
      Interface
      target     14   4/1/0.8.0.255.3.15     tgt        CLAIMED     DEVICE
      ctl        10   4/1/0.8.0.255.3.15.0   sctl       CLAIMED     DEVICE      HP   HPA3308
```

Bus and target of the FC-adapter are derived from the Loop-ID according to PDA addressing:

for FC-A (Loop-ID 45):	for FC-B (Loop ID 63):

Bus = 45 DIV 16 = 2 Target = 45 MOD 16 = 13	Bus = 63 DIV 16 = 3 Target = 63 MOD 16 = 15

The **SCSI controller** of the FC-SCSI mux and the Seagate disks behind are accessed using LUN addressing mode:

```
# ioscan -fd fcpmux
Class     I  H/W Path        Driver S/W State   H/W Type     Description
========================================================================
ext_bus   6  4/1/0.8.0.45.0  fcpmux CLAIMED     INTERFACE    HP A3308 FCP-SCSI MUX
Interface
ext_bus   7  4/1/0.8.0.45.1  fcpmux CLAIMED     INTERFACE    HP A3308 FCP-SCSI MUX
Interface
ext_bus   9  4/1/0.8.0.63.0  fcpmux CLAIMED     INTERFACE    HP A3308 FCP-SCSI MUX
Interface
ext_bus  10  4/1/0.8.0.63.1  fcpmux CLAIMED     INTERFACE    HP A3308 FCP-SCSI MUX
Interface
```

For each FC adapter is an entry for each SCSI card (here: 0 and 1). This shows the number of SCSI cards in the FC-SCSI mux. The SCSI-ID of these cards is shown in the target field and is usually 7:

```
ext_bus      6  4/1/0.8.0.45.0        fcpmux      CLAIMED     INTERFACE     HP A3308
FCP-SCSI MUX Interface
target       4  4/1/0.8.0.45.0.7      tgt         CLAIMED     DEVICE
ctl          5  4/1/0.8.0.45.0.7.0    sctl        CLAIMED     DEVICE        Initiator
                                     /dev/rscsi/c7t7d0
```

The connected **Seagate disk** is shown straight behind this:

```
target       5  4/1/0.8.0.45.0.10     tgt         CLAIMED     DEVICE
disk         7  4/1/0.8.0.45.0.10.0   sdisk       CLAIMED     DEVICE        SEAGATE
ST32550W
              /dev/dsk/c6t10d0   /dev/rdsk/c6t10d0
```

The disk devicefile is composed of the Instancenumber of the FC-SCSI Mux SCSI cards (6), the SCSI-ID of the disk (10) and the Lun-ID (0) zusammen:

```
/dev/dsk/c6t10d0
```

In this example the disk can be accessed over a second FC path:

```
target      12  4/1/0.8.0.63.0.10     tgt         CLAIMED     DEVICE
disk         8  4/1/0.8.0.63.0.10.0   sdisk       CLAIMED     DEVICE     SEAGATE
ST32550W
                /dev/dsk/c9t10d0   /dev/rdsk/c9t10d0
```

Example: Private Loop versus Public Loop

This shows how to migrate an HP XP256 from a private loop configuration to a fabric topology. A hub has been replaced with a switch resulting in different `ioscan` outputs.

Tachyon TL/TS/XL2 HBA --- HUB --- XP256

The `ioscan` output for a **Private Loop** configuration could be as follows:

```
Class     I    H/W Path            Driver    S/W State    H/W Type    Description
--------------------------------------------------------------------------------
fc        0    0/1/2/0               td          CLAIMED    INTERFACE    HP Tachyon TL/TS
Fibre Channel Mass Storage Adapter
fcp       1    0/1/2/0.8             fcp         CLAIMED    INTERFACE    FCP Protocol
Adapter
ext_bus   4    0/1/2/0.8.0.11.0      fcparray    CLAIMED    INTERFACE    FCP Array
Interface
target    6    0/1/2/0.8.0.11.0.0    tgt         CLAIMED    DEVICE
disk      3    0/1/2/0.8.0.11.0.0.0  sdisk       CLAIMED    DEVICE       HP OPEN-8
/dev/dsk/c3t0d0 /dev/rdsk/c3t0d0
disk     10    0/1/2/0.8.0.11.0.0.7  sdisk       CLAIMED    DEVICE       HP OPEN-8
/dev/dsk/c3t0d7 /dev/rdsk/c3t0d7
target    7    0/1/2/0.8.0.11.0.1    tgt         CLAIMED    DEVICE
disk     11    0/1/2/0.8.0.11.0.1.0  sdisk       CLAIMED    DEVICE       HP OPEN-9
     /dev/dsk/c3t1d0      /dev/rdsk/c3t1d0
disk     18    0/1/2/0.8.0.11.0.1.7  sdisk       CLAIMED    DEVICE       HP OPEN-9
     /dev/dsk/c3t1d7      /dev/rdsk/c3t1d7
```

Tachyon TL/TS/XL2 HBA --- SWITCH --- XP256

The ioscan output for a **Public Fabric Loop** configuration could be as follows:

```
Class     I    H/W Path            Driver     S/W State    H/W Type    Description
---------------------------------------------------------------------------------
fc        0    0/1/2/0               td         CLAIMED     INTERFACE    HP Tachyon TL/TS
Fibre Channel Mass Storage Adapter
fcp       1    0/1/2/0.1             fcp        CLAIMED     INTERFACE    FCP Domain
ext_bus   4    0/1/2/0.1.19.0.0      fcparray   CLAIMED     INTERFACE    FCP Array
Interface
target    6    0/1/2/0.1.19.0.0.0    tgt        CLAIMED     DEVICE
disk      3    0/1/2/0.1.19.0.0.0.0  sdisk      CLAIMED     DEVICE       HP OPEN-8
        /dev/dsk/c4t0d0        /dev/rdsk/c4t0d0
disk     10    0/1/2/0.1.19.0.0.0.7  sdisk      CLAIMED     DEVICE       HP OPEN-8
        /dev/dsk/c4t0d7        /dev/rdsk/c4t0d7
target    7    0/1/2/0.1.19.0.0.1    tgt        CLAIMED     DEVICE
disk     11    0/1/2/0.1.19.0.0.1.0  sdisk      CLAIMED     DEVICE       HP OPEN-9
        /dev/dsk/c4t1d0        /dev/rdsk/c4t1d0
disk     18    0/1/2/0.1.19.0.0.1.7  sdisk      CLAIMED     DEVICE       HP OPEN-9
        /dev/dsk/c4t1d7        /dev/rdsk/c4t1d7
fcp       0    0/1/2/0.8             fcp        SCAN        INTERFACE FCP Protocol
Adapter
```

Looking at the `iotree` examples, you can see the following:

- There has been no change to the adapter path or the associated device file which is used for the `fcmsutil` diagnostic tool.

- The node 0/1/2/0.8, FCP Protocol Adapter, is shown in both `ioscan` outputs. In a private loop configuration, the interface and target devices will reside behind this node. In a Fabric environment, this node may be created as a dummy node, if the

HBA is scanned while it cannot see the Fabric (for example, no cable attached, switch down, etc.).

- In the original private loop implementation of the fibre channel driver, this node of the `iotree` was used to indicate the fibre channel FC4 "TYPE". A type of "8" denotes that the FCP protocol is being used to encapsulate the SCSI protocol. With the introduction of fabric, this node contains the "Domain" portion of the N_Port address. To maintain backward compatibility, the domain of 8 is reserved for use with private loop devices.
- **CAUTION:** Do not configure switches with a Domain of 8. This is an unsupported configuration and will not work. The Domain of 8 is reserved for Private Loop devices on HP systems.
- The Fabric configuration now contains an `iotree` node of 0/1/2/0.1 described as FCP domain. A node of this type will be built for each domain the fabric contains (Ddmains usually correspond one to one with a switch instance).
- The FCP Array Interface `iotree` node has changed from 0/1/2/0.**8.0.11**.0 to 0/1/2/0.**1.19.0**.0. The address is still at hardware Path 0/1/2/0, but the next three elements of the path, which represent the N_Port address, have changed. The old N_Port address of 8.0.11 uses the reserved Domain of 8 and area of 0. In this case, the HPA or Port byte of the N_Port address is 11. In the Fabric `iotree`, the new N_Port address is 1.19.0. This corresponds to a domain ID of 1, an area ID of 19 and a port ID of 0.
- Note that for most switches, the domain will map to a switch instance, an area ID will map to a physical connector on the switch, and the port ID will only be used (is non-zero), if there is an Arbitrated Loop configured behind the switch connector. The HPA of the device is then used as the port portion of the `iotree` address.
- All targets and disk devices retain their original `iotree` addresses with the exception that the new Fabric N_Port address has been substituted for the old Arbitrated Loop address.
- New device files have been generated for the new `iotree` nodes. The old device files will continue to exist until removed with the `rmsf(1M)` command.

Troubleshooting Utilities fcmsutil, tdutil, tdlist, tddiag, fcddiag, fclpdiag

fcmsutil

The `fcmsutil` utility can be found in the directory `/opt/fc/bin/` and/or `/opt/fcms/bin`. It helps troubleshooting the FC Loop of the SAN. It is invoked using the device file of the FC adapter (see `ioscan -fnk`).

Tachyon Example:

```
# fcmsutil /dev/fcms2

Local N_Port_ID is = 0x000001
N_Port Node World Wide Name = 0x10000060B03EF669
N_Port Port World Wide Name = 0x10000060B03EF669
Topology = IN_LOOP
Speed = 1062500000 (bps)
HPA of card = 0xFFB4C000
EIM of card = 0xFFFA2009
Driver state = READY
Number of EDB's in use = 0
Number of OIB's in use = 0
Number of Active Outbound Exchanges = 1
Number of Active Login Sessions = 3
```

Tachyon TL/TS/XL2 Example:

```
# fcmsutil /dev/td0

                           Vendor ID is = 0x00103c
                           Device ID is = 0x001029
                XL2 Chip Revision No is = 2.2
            PCI Sub-system Vendor ID is = 0x00103c
                   PCI Sub-system ID is = 0x00128c
                               Topology = PTTOPT_FABRIC
                             Link Speed = 1Gb
                     Local N_Port_id is = 0x0d1200
            N_Port Node World Wide Name = 0x50060b000010dcef
            N_Port Port World Wide Name = 0x50060b000010dcee
                           Driver state = ONLINE
                       Hardware Path is = 0/4/1/0
                 Number of Assisted IOs = 471
        Number of Active Login Sessions = 0
                   Dino Present on Card = NO
                     Maximum Frame Size = 1024
                         Driver Version = @(#) libtd.a HP Fibre Channel
Tachyon XL2 Driver B.11.23.0512 $Date: 2005/09/20 12:22:47 $Revision: r11.23/1
```

Another possibility for the Topology is: **`PRIVATE_LOOP`**

After pulling the FC cable the Driver state will change to:

```
                           Driver state = AWAITING_LINK_UP
```

In order to get a summary of the link statistics:

```
# fcmsutil /dev/td0 stat -s
Fri Apr 26 16:05:55 2002
Channel Statistics

Statistics From Link Status Registers ...
Loss of signal                   2      Bad Rx Char                   182
Loss of Sync                    40      Link Fail                       4
Received EOFa                    0      Discarded Frame                 0
Bad CRC                          0      Protocol Error                  0
```

Do not look for high values, as some counters will increase during boot or LIP. Only values that are increasing over time indicate a problem. In the following we see a typical error message:

`0/4/0/0: Unable to access previously accessed device at nport ID 0xae.`

Here's how to troubleshoot:

```
# ioscan -fnkH0/4/0/0
fc          0  0/4/0/0               td        CLAIMED     INTERFACE    HP Tachyon TL/TS Fibre
Channel Mass Storage Adapter
                                    /dev/td0
fcp         0  0/4/0/0.8             fcp       CLAIMED     INTERFACE    FCP Protocol Adapter
ext_bus     4  0/4/0/0.8.0.255.0     fcpdev    CLAIMED     INTERFACE    FCP Device Interface
target      8  0/4/0/0.8.0.255.0.12  tgt       CLAIMED     DEVICE
disk        3  0/4/0/0.8.0.255.0.12.0 sdisk    CLAIMED     DEVICE       SEAGATE ST39102FC
                                    /dev/dsk/c4t12d0   /dev/rdsk/c4t12d0
target      9  0/4/0/0.8.0.255.0.13  tgt       CLAIMED     DEVICE
disk        4  0/4/0/0.8.0.255.0.13.0 sdisk    CLAIMED     DEVICE       SEAGATE ST39102FC
                                    /dev/dsk/c4t13d0   /dev/rdsk/c4t13d0
target     10  0/4/0/0.8.0.255.0.14  tgt       CLAIMED     DEVICE
disk        5  0/4/0/0.8.0.255.0.14.0 sdisk    CLAIMED     DEVICE       SEAGATE ST39102FC
                                    /dev/dsk/c4t14d0   /dev/rdsk/c4t14d0
target     11  0/4/0/0.8.0.255.0.15  tgt       CLAIMED     DEVICE
disk        6  0/4/0/0.8.0.255.0.15.0 sdisk    CLAIMED     DEVICE       SEAGATE ST39102FC
                                    /dev/dsk/c4t15d0   /dev/rdsk/c4t15d0
ext_bus     5  0/4/0/0.8.0.255.1     fcpdev    CLAIMED     INTERFACE    FCP Device Interface
target     12  0/4/0/0.8.0.255.1.6   tgt       CLAIMED     DEVICE
ctl         4  0/4/0/0.8.0.255.1.6.0 sctl      CLAIMED     DEVICE       HP      A5236A
target     13  0/4/0/0.8.0.255.1.8   tgt       CLAIMED     DEVICE
disk        7  0/4/0/0.8.0.255.1.8.0 sdisk     CLAIMED     DEVICE       SEAGATE ST39102FC
                                    /dev/dsk/c5t8d0   /dev/rdsk/c5t8d0
target     14  0/4/0/0.8.0.255.1.9   tgt       CLAIMED     DEVICE
disk        8  0/4/0/0.8.0.255.1.9.0 sdisk     CLAIMED     DEVICE       SEAGATE ST39102FC
                                    /dev/dsk/c5t9d0   /dev/rdsk/c5t9d0
target     15  0/4/0/0.8.0.255.1.10  tgt       CLAIMED     DEVICE
disk        9  0/4/0/0.8.0.255.1.10.0 sdisk    CLAIMED     DEVICE       SEAGATE ST39102FC
                                    /dev/dsk/c5t10d0   /dev/rdsk/c5t10d0
target     16  0/4/0/0.8.0.255.1.11  tgt       CLAIMED     DEVICE
disk       10  0/4/0/0.8.0.255.1.11.0 sdisk    CLAIMED     DEVICE       SEAGATE ST39102FC
                                    /dev/dsk/c5t11d0   /dev/rdsk/c5t11d0
ext_bus     6  0/4/0/0.8.0.255.2     fcpdev    NO_HW       INTERFACE    FCP Device Interface
```

N_port ID (= AL_PA, because it is a private loop) = 0xae
Regarding the conversion table this corresponds to LoopID 34.

```
# fcmsutil /dev/td0 devstat all | grep -e Nport -e Failed
Device Statistics for Nport_id 0x0000ae(Loop_id 34)
Failed Open of previously opened device                  9
Device Statistics for Nport_id 0x0000b9(Loop_id 27)
Failed Open of previously opened device                  0
Device Statistics for Nport_id 0x0000ba(Loop_id 26)
Failed Open of previously opened device                  0
Device Statistics for Nport_id 0x0000bc(Loop_id 25)
Failed Open of previously opened device                  0
Device Statistics for Nport_id 0x0000c3(Loop_id 24)
Failed Open of previously opened device                  0
Device Statistics for Nport_id 0x0000c6(Loop_id 22)
Failed Open of previously opened device                  0
Device Statistics for Nport_id 0x0000ce(Loop_id 15)
Failed Open of previously opened device                  0
Device Statistics for Nport_id 0x0000d1(Loop_id 14)
Failed Open of previously opened device                  0
Device Statistics for Nport_id 0x0000d2(Loop_id 13)
```

```
Failed Open of previously opened device                     0
Device Statistics for Nport_id 0x0000d3(Loop_id 12)
Failed Open of previously opened device                     0
```

This is a private loop with PDA Addressing (8.0.255), i.e

LoopID = 16*Bus+Target ==> Bus = 34 DIV 16 = 2
==> Target = 34 MOD 16 = 2

and results in the following HW path:

HBA	Domain	Area	Port	Bus	Target	Lun
0/4/0/0 .	8 .	0 .	255 .	2 .	2 .	0

This path is not part of `ioscan`, because the interface above this path is shown as `NO_HW`, i.e someone disconnected it without rebooting:

```
ext_bus    6    0/4/0/0.8.0.255.2    fcpdev    NO_HW    INTERFACE    FCP Device Interface
```

A5236A is a FC10 JBOD.

Overview of fcmsutil options

T: option available with Tachyon, TL: option available with Tachyon TL/TS/XL2, red font: destructive task, i.e the communication gets interrupted

fcmsutil Option	Explanation	T/TL
`fcmsutil <dev file>`	State of the link	T+TL
`fcmsutil <dev file> lgninfo_all`	Info about devices in the loop	T
`fcmsutil <dev file> devstat all`	Info about devices in the loop	TL
`fcmsutil <dev file> reset`	Reset the FC card	T+TL
`fcmsutil <dev file> lb tachyon`	Internal loopback test, tests the Tachyon chip	T
`fcmsutil <dev file> lb plm`	External loopback test (needs loopback cable)	T+TL
`fcmsutil <dev file> enable`	Enable the card (after a HW failure)	T+TL
`fcmsutil <dev file> disable`	Disable the card	T+TL
`fcmsutil <dev file> stat`	Obtain statistics maintained by the driver	T+TL
`fcmsutil <dev file> stat -s`	Obtain statistics summary maintained by the driver	TL
`fcmsutil <dev file> clear_stat`	Reset the statistics	TL
`fcmsutil <dev file> echo <nport_id>`	Send packet to other N-port ID (within priv. loop)	T+TL
`fcmsutil <dev file> rls <nport_id>`	Send packet to another N-port ID (across switch)	TL
`fcmsutil <dev file> replace_dsk <nport_id>`	Replace disk (disallow authentication)	TL

tdutil, tdlist, tddiag, fcddiag, fclpdiag

In the directory `/opt/fcms/bin` are additional utilities:

`tdutil` this is fcmsutil but for td driver only.

`tdlist` this is a shell script that uses ioscan and tdutil in order to list all devices that are are handled by the td driver, i.e. all devices attached to the Tachyon TL/TS/XL2 adapters on the system. The script contains a nice function that translates Loop ID to AL_PA.

`tddiag` this is a shell script that gathers the following fibre channel related information of the system: system name, system model, system uptime, memory information, mounted file systems, TachLite version in kernel, system file, patches installed, device special files for TachLite, ioscan output, tdlist output, running processes, infos about each `/dev/td#`: device info, device vpd info, device topology sensing mode, chip registers, device statistics, Name Server device statistics, CT Server device statistics, all remote statistics, name server port info (from kernel), name server port info (from switch), device statistics on all targets.

fcddiag this is similar to the tddiag. That means it is a shell script that gathers the following fibre channel related information of the system: system name, system model, system uptime, memory information, mounted file systems, info about `/dev/fcd#`

fclpdiag this is similar to the tddiag, fcddiag. That means it is a shell script that gathers the following fibre channel related information of the system: system name, system model, system uptime, memory information, mounted file systems, info about `/dev/fclp#`

How to Replace Disks at Tachyon TL/TS/XL2 HBAs

Before a server can talk to a target it has to authenticate at the Tachyon TL/TS/XL2 HBA with its WWN (World Wide Name). The authentication (PLOGI) ensures that the system is talking to the correct device, avoiding data corruption due to a user accidentally connecting another device at the same `nport_id`.

The **adapter** holds a table where the native address (S_ID or AL_PA and WWN) of every known device of the loop is stored. This table is created upon initialization of the link or when the first communication between host and device occurs.

NOTE: The authentication applies to the devices connected to the TL/TS/XL2 adapter only, Tachyon adapters do not go through the same level of authentication. (the `replace_dsk` option is supported by TL/TS/XL2 only).

The `replace_dsk` option of `fcmsutil` is required to change a device and keep the same `nport_id`. It should be used during disk replacement. When this option is used, no authentication on that device is performed the next time the system communicates with it avoiding the following error (`syslog`):

```
0/4/0/0: 'World-wide name' (unique identifier) for device at Loop ID
0x5 has changed. If the device has been replaced intentionally, please
use the fcmsutil replace_dsk command to allow the new device to be
used.
```

In this example the disk with `loop_id 5` at Tachyon TL/TS/XL2-Adapter `/dev/td0` is going to be replaced:

- Identify the `nport_id` or loop_id of the disk being moved or replaced (this step is optional if nport_id or loop_id of the device is known from syslog, dmesg, or other error logs).
- Disconnect the device (i.e remove it from the loop).
- List all devices with which the TL/TS/XL2 card has successfully communicated using the devstat all option:

```
# fcmsutil /dev/td0 devstat all | grep Loop
Device Statistics for Nport_id 0x0000E8 (Loop_id 1)
Device Statistics for Nport_id 0x0000DA (Loop_id 5)
```

Using the `echo` option, try to login to each device using the `loop_id`, a message "`unable to login`" is returned for the disconnected device.

```
# fcmsutil /dev/td0 echo -l 1
Data came back intact
...
# fcmsutil /dev/td0 echo -l 5
Unable to login
```

- Run `replace_dsk` using `nport_id`:

```
# fcmsutil /dev/td0 replace_dsk 0x0000DA
```

in a private loop you can also use the `loop_id`:

```
# fcmsutil /dev/td0 replace_dsk -l 5
```

After running the command, a message appears indicating the port will not be authenticated:

```
Disk at nportid 0x0000da (Loop_id 5) will not be authenticated
```

ATTENTION: This step has to be repeated for **any** Tachyon TL/TS/XL2 adapter on the host as well as for other hosts (e.g in a ServiceGuard cluster) that access the same device!

- Replace bad disk with new disk.

The new disk (`loop_id=5`) can login without getting an authentication failure. The next time the system issues any read/write operations (e.g. `ioscan`) to the new disk, the new World-Wide Name (WWN) will be recognized and recorded to associate with `nport_id 0x0000da`.

It is important to make sure the correct `nport_id` or `loop_id` is used with `replace_dsk`. But if you accidentally type in the wrong `nport_id` or `loop_id`, nothing harmful will happen. Just keep in mind that there will be no authentication done on that device during the next login. The device will be authenticated on subsequent logins.

Fibre Channel Storage Devices

This is a short overview about existing devices. For details go to HP's storage website http://www.hp.com/go/storage or follow one of the links at section Additional Information.

NIKE Model 30 FC Disk Array (legacy)

NOTE: This is a legacy product which should not be ordered anymore. FC30 are not supported in SAN-environments, private loop only!

FC10 Fibre Channel Disk System (Product-#: A5236A)

The HP SureStore E Disk System FC10 is a 10-slot Fiber Channel JBOD Disk Enclosure that features redundant Link Control Cards (LCC), redundant power supplies and fan modules. The disk modules are native Fibre Channel with 2 FC-ALports per disk. This allows the use of alternate paths for high availability configurations.

HP SureStore Disk System 2405 (DS2405, Product-# A6250A)

The HP Surestore Disk System 2405 (ds2405) supports the latest 2 GB Fibre Channel technology. With its modular design, the ds2405 is a highly scalable and flexible storage system. In a compact 3U form factor, each enclosure holds up to 15 disk drives or over 1 TB of capacity. The ds2405 can be upgraded from a standalone disk system to add-on storage for the HP Virtual Array.

A HP StorageWorks 2500 Disk System (DS2500, Product-#: AG572A)

The HP StorageWorks 2500 Disk System (DS2500) delivers an industry leading, high-capacity 3U storage solution. This rack-optimized enclosure features system compatibility with HP-UX operating systems. Allows up to 7 TB of storage in a single enclosure with 14 Disks (72GB/146 GB FC, 500 GB FATA drives).

FC60 Fibre Channel Disk Array (Product-# A5277A)

High availability disk array comprised of a controller enclosure and up to six disk enclosures. Each disk enclosure can contain up to ten disks. Disk capacities are 18.2GB, 36.4GB or 73.4GB. RAID Level 0*, 1, 5, and 0/1 are supported.

The HP Array Manager/60 cumulative patch provides the Manager Software for FC60 administration. It contains the commands with manual pages and the startup script (`/sbin/init.d/hparamgr`). The commands can be found in the directory `/opt/hparray/bin/`:

`amdsp`, `amcfg`, `amdload`, `amfmt`, `amlog`, `ammgr`, `amutil`.

The communication with the array is done by a daemon (`/usr/lbin/hparray/hparamgrd`) that is invoked by the startup script.

HP Virtual Array VA7100/VA7400

The HP Virtual Array VA7100 and VA7400 are utilizing AutoRAID technology. All modules at the front and rear are hot-pluggable.

The chassis accomodates:

- 3 EIA rack mountable chassis.
- 15 low profile 3.5 inch fibre channel hard disk drives loaded from the front.
- 1 or 2 fibre channel controllers mounted in the backside. Each controller has one 1 GB fibre channel interface to the host.
- 2 power supply and fan modules mounted in the rear. The power supply and fan components are combined into a single module.
- PSP: http://wwwpsp.atl.hp.com/lmx_mount/supplan/psp/12/psp12062.htm (HP internal)

Compared to the VA7100, this array has the following additional features:

- Two types of chassis enclosures: The main that holds the AutoRAID controllers and up to six add on enclosures that allow more drives to be added behind the AutoRAID controllers.
- Optional 2 GB configurable fibre channel interface to the host
- Add-on chassis has 2 fibre channel LCC (Link Controller Card) that allows a daisy chain type of connection to the main. Each LCC has two 1 GB fibre channel interface.

You can manage the VA either by the front panel or by installing a graphical and command line user interface – HP Command View SDM. The administration of the VA is similar to the administration of the FC60 or the AutoRAID. The commands start with `arm*` instead of `am*` or `array*` at the FC60 or AutoRAID respectively. The table at http://ren.nsr.hp.com/howto/array.html (HP internal) shows a comparison between the admin commands of the three disk arrays. You can download SDM from an external website. Just search for “sdm ux” at http://www.hp.com. It can also be found (together with latest Firmware) at the HP intranet: http://tce-web.boi.hp.com/prod_port/ (HP internal).

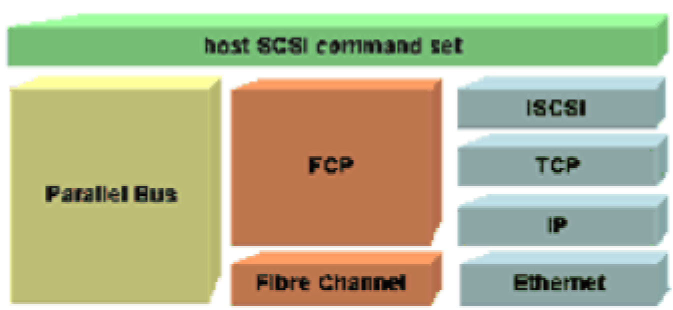
XP Disk Array Family

Information can be found in the XP Disk Arrays Chapter.

iSCSI devices on HPUX hosts

Overview

iSCSI is a TCP/IP-based transport protocol for SCSI, operating at the same level as Parallel SCSI and Fibre Channel. It is a low cost alternative to Fibre Channel (IP equipment can be used) but has lower performance due to additional layers.



In contrast to other network storage protocols like NFS, Samba or CIFS it operates on the block device level, not on a file level. Thus, iSCSI targets are seen as local disks on HPUX

Currently, no iSCSI server exists for HPUX. Refer to the HP-UX iSCSI Software Initiator Support Matrix for supported iSCSI targets.

`ioscan` example:

```
iscsi       0  255/0          iscsi     CLAIMED     VIRTBUS      iSCSI Virtual
Node
ext_bus     4  255/0/0.0      iscsial   CLAIMED     INTERFACE    iSCSI-SCSI
Protocol Interface
target      7  255/0/0.0.0    tgt       CLAIMED     DEVICE
disk        3  255/0/0.0.0.0  sdisk     CLAIMED     DEVICE       NETAPP   LUN
                             /dev/dsk/c4t0d0    /dev/rdsk/c4t0d0
```

The external representation of the hardware path for an iSCSI LUN is:

255/<iscsi_virtual_HBA>/<session_instance>.<virtual_bus>.<virtual_target>.<lun>

- **iscsi_virtual_HBA** represents a virtual HBA off of the virtual root node. The value of **iscsi_virtual_HBA** is typically zero (0).
- **session_instance** is the iSCSI session instance number. The iSCSI session defines a SCSI I-T nexus for transactions between the iSCSI initiator and the iSCSI target.
- **virtual_bus** is an instance number (`ext_bus`) of a SCSI-2 bus.
- **virtual_target** is a virtual representation of a SCSI-2 target.
- **lun** is the representation of a SCSI-2 logical unit number.

The mapping used is equivalent to the mapping used by the HP-UX Fibre Channel driver.

To view more detailed information regarding the targets use:

```
iscsiutil /dev/iscsi -p -D

Discovery Target Information
----------------------------

Target # 1
-----------
       IP Address                : 192.168.155.2
       iSCSI TCP Port            : 3260
       iSCSI Portal Group Tag    : 1

  User Configured:
  ----------------

       Authenticaton Method      :
       CHAP Method               : CHAP_UNI
       Initiator CHAP Name       :
       CHAP Secret               :
       Header Digest             : None,CRC32C (default)
       Data Digest               : None,CRC32C (default)
```

In case of more severe problems like network connectivity issues, the syslog loglevel can be increased . A tool `iscsidiag` exists that collects all information from a system that is needed for iSCSI troubleshooting. An example of the iscsidiag output can be seen here: http://herkules.deu.hp.com/documents/docs/EMEA-Newsletter/iscsidiag.txt.

Installation

You need the iscsi driver. It can be downloaded from http://software.hp.com (search for iSCSI Software Initiator)

```
swlist -l product | grep iSCSI
...
iSCSI-00                  B.11.11.03e   HP-UX iSCSI Software Initiator
iSCSI-00.ISCSI-SWD        B.11.11.03e   HP-UX iSCSI Software Initiator
```

To add an iSCSI target the customer first has to allow the access to the iSCSI device on the iSCSI server (f.e.a Proliant Server). Several authentication methods exist but at least the IQN (iSCSI qualified name, similar to a WWN) of the virtual iSCSI adapter has to be configured on the iSCSI server.

You can see the IQN(s) on the HPUX host with:

```
# iscsiutil -l
        iqn.1986-03.corn.hp:hpuxiscsi.615349383
```

to add one (or several) discovery target(s):

```
# iscsiutil [/dev/iscsi] -a -I <ip-address> [-P <tcp-port>] [-M <portal-grp-tag>]
```

where

- `-a` adds a discovery target address into iSCSI persistent information. Only discovery target addresses can be added using this option.
- `-I` requires the IP Address or the Hostname of the discovery target portal address as an argument.
- `<ip-address>` is the IP address or hostname component of the target network portal.
- `[-P <tcp-port>]` is the listening TCP port component of the discovery target network portal (optional). The default iSCSI TCP port number is 3260.
- `[-M <portal-grp-tag>]` is the target portal group tag (optional). The default target portal group tag for discovery targets is 1.

Secure Path for active/passive disk arrays

Because of the active/passive controller implementation of the EVA 3000/5000 (CVS-Firmware Version: 3.x and before) Secure Path for HP-UX is required for accessing the EVA over multiple paths from HP-UX. Secure Path is able to handle controller failovers and hides all “physical” paths (usually 4 paths when using 2 HBAs connected to 2 separate SANs) to the EVA from HP-UX. It creates one single virtual path/devicefile per vdisk instead. SecurePath can be used for active/active controller implementation to use all the other features.

Installation Secure Path

The HP-UX system must be patched before Secure Path can be installed (see release notes of Secure Path). The main binaries of Secure Path (`spmgr`,`spagent`) is installed in `/sbin`. Keep

Note: Volume `/dev/vg00/lvol3` cannot be increased in size easily.

To install Secure Path, today we get a package with installation script:

```
# ./install.sh

Please select the software that you want to install

 1) Secure Path Ver 3.0F for HP-UX for Active-Passive Disk Arrays
 2) Secure path Ver 3.0F for HP-UX for Active-Active Disk Arrays
 3) Exit Secure path Installation
```

There is an automatic reboot at the end of the installation!

Install Secure Path on a SAN boot device

- Install HP-UX 11i v1.0 or 11i v2.0 on an EVA LUN.
- See the HP StorageWorks Secure Path 3.0F Service Pack 2 for HP-UX 11i v1.0 and 11i v2.0 release notes (part number T3549-96605) to verify that all required patches are installed (reboot required).
- Install Secure Path 3.0F SP2 (reboot again).
- Determine the hardware path of the stand-by path for the EVA boot LUN by using the `spmgr` display and `ioscan` commands. Set the alternate boot path to this hardware path by using the command `setboot -a hardware_path`.

Removing Secure Path software

```
# swremove -x autoreboot=true CPQswsp,r=*
```

NOTE: After removing Secure Path, reconfigure the array for transparent failover mode and restore the server to a single-path RAID storage environment. In a single-path configuration, the HSG80 controller is set to transparent failover mode. See the *HSG80 ACS Solution Software for HP-UX installation and configuration guide* for details.

Limitations for SAN boot

The following are limitations to Secure Path for HP-UX on a SAN boot device:

- Root/Swap/Dump is not supported with HSG LUNs.
- Root/Swap/Dump must be kept under LVM while installing HP-UX.
- Root/Swap/Dump must not be under VxVm configuration while installing HP-UX.
- Do not configure multiple Swap/Dump devices as a whole disk before installing Secure Path 3.0F SP2.

- After the system reboots on a SAN LUN, do not run any applications before installing Secure Path.
- Removing Secure Path for HP-UX is not supported on a SAN boot device.

PV timeout for Secure Path devices

It is recommended to increase the PV timeout values from default (30s) to 60s:

```
# pvchange -t 60 /dev/dsk/c100t0d0
```

`ioscan` with Secure Path

It may be necessary to reboot the machine once to make the EVA vdisks visible to HP-UX. Here is how `ioscan` looks before Secure Path is installed:

```
# ioscan -fkH1/4
Class      I  H/W Path        Driver S/W State   H/W Type     Description
=========================================================================
Ba        11 1/4             lba    CLAIMED     BUS_NEXUS    Local PCI Bus Adapter
Fc         3 1/4/0/0         td     CLAIMED     INTERFACE    HP Tachyon XL2
Fcp        1 1/4/0/0.82      fcp    CLAIMED     INTERFACE    FCP Domain
ext_bus 112 1/4/0/0.82.9.0.0 fcparray     CLAIMED INTERFACE    FCP Array Interface
target   14 1/4/0/0.82.9.0.0.0    tgt     CLAIMED DEVICE
ctl      82 1/4/0/0.82.9.0.0.0.0  sctl    CLAIMED DEVICE COMPAQ HSV110 (C)COMPAQ
hsx      12 1/4/0/0.82.9.0.0.0.1  hsx     CLAIMED DEVICE HSV110 (C)COMPAQ PATH
0x600508B4000108960002100000AD0000
ext_bus  17 1/4/0/0.82.9.255.0    fcpdev  CLAIMED INTERFACE FCP Device Interface
target   15 1/4/0/0.82.9.255.0.0  tgt     CLAIMED DEVICE
ctl     114 1/4/0/0.82.9.255.0.0.0 sctl   CLAIMED DEVICE COMPAQ HSV110 (C)COMPAQ
ext_bus 122 1/4/0/0.82.10.0.0 fcparray    CLAIMED INTERFACE FCP Array Interface
target   16 1/4/0/0.82.10.0.0.0    tgt    CLAIMED DEVICE
ctl     116 1/4/0/0.82.10.0.0.0.0  sctl   CLAIMED DEVICE COMPAQ HSV110 (C)COMPAQ
hsx      16 1/4/0/0.82.10.0.0.0.1  hsx    CLAIMED DEVICE HSV110 (C)COMPAQ PATH
0x600508B4000108960002100000AD0000
ext_bus  20 1/4/0/0.82.10.255.0    fcpdev CLAIMED INTERFACE FCP Device Interface
target   17 1/4/0/0.82.10.255.0.0  tgt    CLAIMED DEVICE
ctl     142 1/4/0/0.82.10.255.0.0.0 sctl  CLAIMED DEVICE COMPAQ HSV110 (C)COMPAQ
```

After the Secure Path installation all physical paths to the EVA disappear, instead one path per vdisk appears under HW path 255/255:

```
# ioscan -fkCdisk
Class      I  H/W Path        Driver S/W State   H/W Type     Description
=========================================================================
disk       0 0/0/1/1.2.0      sdisk  CLAIMED     DEVICE       HP 36.4GST336753LC
disk       1 0/0/2/0.2.0      sdisk  CLAIMED     DEVICE       HP 36.4GST336753LC
disk       2 0/0/2/1.2.0      sdisk  CLAIMED     DEVICE       HP DVD-ROM 305
disk       3 255/255/0.0.0    sdisk  CLAIMED     DEVICE       HSV110 (C)COMPAQ LUN
0x600508B4000108960002100000AD0000
```

All vdisks (LUNs) presented to the host can be displayed using the following Secure Path command:

```
# spmgr display
 Server:  hostname    Report Created: Thu, May 15 17:53:53 2003
 Command: spmgr display
 = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
 Storage:  5000-1FE1-5000-E0E0
 Load Balance: On  Auto-restore: Off
 Path Verify: On   Verify Interval: 30
 HBAs: td0 td1
 Controller: P5849D4AAOJ05U, Operational
             P5849D4AAOJ05G, Operational
 Devices:   c100t0d0  c100t0d1  c100t0d2

 TGT/LUN    Device           WWLUN_ID              H/W_Path           #_Paths
   0/  0    c100t0d0         6005-08B4-0001-0896-0002-1000-00AD-0000    4
                                                   255/255/0.0.0
         Controller  Path_Instance     HBA              Preferred?  Path_Status
         P5849D4AAOJ05U                                     no
                     c111t0d1          td0              no           Active
                     c112t0d1          td1              no           Available

         Controller  Path_Instance     HBA              Preferred?  Path_Status
         P5849D4AAOJ05G                                     no
                     c122t0d1          td0              no            Standby
                     c121t0d1          td1              no            Standby
```

Note that there is no relation between the SCSI address of the virtual Secure Path device (device file) and the LUN number of the vdisk. The Secure Path device file addresses are simply increased beginning with `t0d0`. In order to save the relation between the vdisk and the device file (and to make it persistent across reboots) the following procedure must be executed every time a LUN change is done for the HP-UX host. Please note that the procedure is documented here for Secure Path version 3.0b. The procedure for Secure Path 3.0a after "`spmgr update`" is slightly different, but "`spmgr update`" always displays the next steps automatically.

spmgr – commands

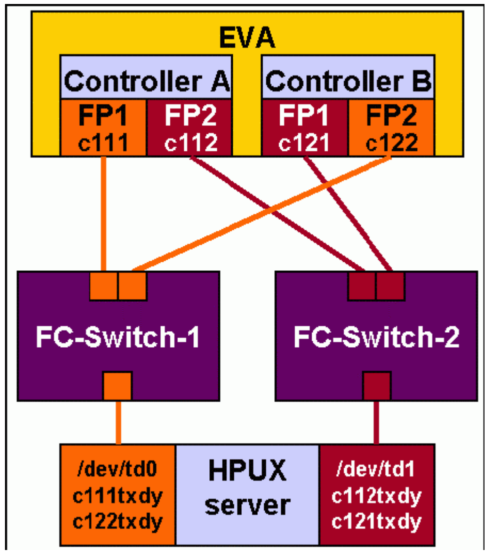
Command	Options or Arguments	Description
spmgr add	WWLUNID target LUN -r WWNN all	Adds a new device to the Secure Path for Active-Passive configuration Note: specifying a LUN address has been removed from SecurePath SW.
spmgr alias	alias_name old_name no argument	Assigns an alias to an object
spmgr clean	-d [WWLUNID] -r [WWNN] all	Cleans device data from the Secure Path's stale device list
spmgr delete	WWLUNID device -r WWNN all	Removes a device from the Secure Path for Active-Passive configuration
spmgr display	-a[v] adapter -c[v] controller_ser_num -d[v] device -r[v] WWNN -p path-Instance -s	Displays information about configured Secure Path for Active-Passive devices

	-u no argument	
spmgr log	-c 0, 1...3 -l 0, 1...3 -n 0, 3 no argument	Sets logging to the console, system log file, and e-mail notification
spmgr notify	add severity_level e-mail_address delete e-mail_address no argument	Manages e-mail address and event logging severity to each e-mail recipient
spmgr passwd	no argument	Provides security on the server side to restrict client access
spmgr prefer	path_instance	Assigns a preferred attribute to a path
spmgr quiesce	-a adapter -c controller_ser_num -p path_instance	Moves I/O to an alternative object and temporarily removes the selected object from use
spmgr refreshdisplay	no argument	Refreshes the array controller serial numbers NOTE: Only the controller serial number is updated and not the controller status. This is because the controller status is determined by the status of the paths going through that controller.
spmgr restart	-a adapter -c controller_ser_num -p path_instance all	Returns a previously quiesced object to an active, available, or standby state
spmgr restore	-d device -r WWNN all	Restores one or more devices to their preferred I/O path
spmgr select	-c controller_ser_num [-d device] -p path_instance	Selects a path or controller for I/O. Also, initiates a LUN movement if necessary
spmgr set	-a on | off WWNN -b on | off | rr | lb | li | ls | WWNN -f interval -p on | off WWNN	Enables or disables Secure Path features like load balancing, path verification
spmgr unalias	alias_name old_name	Removes an alias
spmgr unprefer	path_instance	Removes a preferred path attribute

Quiescing an object means to move all active I/O from an object to an alternate path, and to mark all paths to the object as quiesced to temporarily remove the object from use. Quiescing individual path is supported to allow other fabric infrastructures, such as switches, to remove and replace.

Selection of preferred paths

Using the `spmgr select` command, Secure Path can switch the active paths to the EVA either from one HBA to a second HBA or from one EVA controller to another EVA controller. **Example**: HP-UX device file instances have been renumbered (using `ioinit`), second digit behind "c" refers to 1 for controller A and 2 for controller B of the EVA; the third digit refers to 1 for FP1 and 2 for FP2.



Secure Path for Active-Active disk arrays (autopath)

HP StorageWorks Secure Path 3.0F SP2 for active/active disk arrays provides automatic I/O path failover, failback, recovery, and load balancing for host systems configured with multiple host adapters and connections to disk arrays.

IMPORTANT: You must disable the VERITAS DMP multipath feature on a Secure Path owned device, or remove the device from Secure Path's control by running the `autopath delete` command to avoid interoperatebility issues with VxVM 4.1 and Secure Path 3.0F SP2 on HP-UX 11i v2.0.

ULM services

Secure Path uses the SCSI Upper Layer Module (ULM) interface to provide flexible configuration and layering of modules above the SCSI disk drivers. Register Secure Path with the ULM services for each LUN path Secure Path should control, if Secure Path is configured into HP-UX 11i v1.0 64-bit operating systems and should coexist with other pseudo drivers.

Command line interface

You can use the Secure Path command line interface (CLI) only on those hosts that are connected to devices with the HPswsp driver installed.

This section contains information about the commands supported by this interface along with their syntax, including `autopath` options: `delete, add, set, display, help, discover, retrieve, set_lbpolicy, set_prefpath`.

Functionality

Autopath does not create virtual device files; it uses the device file given by HP-UX instead. The command `autopath discover` checks for alternate links to each device, which can be verified with `autopath display`. It is sufficient to use one path only from the desired LUN (eg: in `lvmtab`); autopath will handle the failover and loadbalancing, if needed. For further information see the installation and reference guide at [HP StorageWorks Secure Path 3.0F Service Pack 2 for HP-UX 11i v1.0 and 11i v2.0](#).

Example (LUN on XP512, alternate links, loadbalancing: round-robin, with LVM)

```
# ioscan -fknCdisk
Class     I  H/W Path       Driver   S/W State   H/W Type     Description
[…]
disk    189  1/10/0/0.3.16.0.0.13.5  sdisk    CLAIMED     DEVICE     HP      OPEN-E
                           /dev/dsk/c5t13d5   /dev/rdsk/c5t13d5
[…]
disk    268  1/10/0/0.3.30.0.0.13.5  sdisk    CLAIMED     DEVICE     HP      OPEN-E
                           /dev/dsk/c7t13d5   /dev/rdsk/c7t13d5

# autopath display:
==================================================================
 Lun WWN                 : 0400-78FB-00C7
 Load Balancing Policy : Round Robin
==================================================================
 Device Path                       Status
==================================================================
 /dev/dsk/c5t13d5                  Active
 /dev/dsk/c7t13d5                  Active
```

Putting one physical path to a Volume Group (example: vg77)

```
# strings /etc/lvmtab | grep dev
/dev/vg00
/dev/dsk/c2t6d0
/dev/vg77
/dev/dsk/c7t13d5

# lvdisplay -v /dev/vg77/lvol1
--- Logical volumes ---
LV Name                     /dev/vg77/lvol1
VG Name                     /dev/vg77
LV Permission               read/write
LV Status                   available/syncd
Mirror copies               0
Consistency Recovery        MWC
Schedule                    parallel
LV Size (Mbytes)            252
Current LE                  63
Allocated PE                63
Stripes                     0
```

```
Stripe Size (Kbytes)         0
Bad block                    on
Allocation                   strict
IO Timeout (Seconds)         default

   --- Distribution of logical volume ---
   PV Name            LE on PV   PE on PV
   /dev/dsk/c7t13d5   63         63

   --- Logical extents ---
   LE    PV1                 PE1   Status 1
   00000 /dev/dsk/c7t13d5   00000 current
   00001 /dev/dsk/c7t13d5   00001 current
   00002 /dev/dsk/c7t13d5   00002 current
   00003 /dev/dsk/c7t13d5   00003 current
[…]
```

During access of the logical volume, both HW paths are used because of autopath's loadbalancing policy round-robin:

```
# sar -d 3 3

HP-UX smart B.11.11 U 9000/800    08/08/07

15:14:44   device   %busy   avque   r+w/s  blks/s  avwait  avserv
15:14:47   c2t6d0    3.67    0.50       3      76    3.06   14.21
          c5t13d5   33.67    0.50     854    6835    4.95    0.40
          c7t13d5   43.33    0.50     854    6835    4.96    0.48
```

In case of a link failure of one FC link, autopath recovers it and hides the failure from the LVM layer. If the path `c7t13d5` (which is the only one configured in the `lvmtab`) has failed, the above scenario looks as follows:

```
# ioscan -fknCdisk
Class     I  H/W Path       Driver   S/W State   H/W Type     Description
[…]
disk    189  1/10/0/0.3.16.0.0.13.5  sdisk    CLAIMED     DEVICE     HP      OPEN-E
                           /dev/dsk/c5t13d5   /dev/rdsk/c5t13d5
[…]
disk    268  1/10/0/0.3.30.0.0.13.5  sdisk    NO_HW       DEVICE     HP      OPEN-E
                           /dev/dsk/c7t13d5   /dev/rdsk/c7t13d5

# autopath display:
==================================================================
 Lun WWN                : 0400-78FB-00C7
 Load Balancing Policy : Round Robin
==================================================================
 Device Path                    Status
==================================================================
 /dev/dsk/c5t13d5               Active
 /dev/dsk/c7t13d5               Failed
```

autopath shows the link `c7t13d5` correctly as failed, but it presents the device file to LVM, because autopath "knows" the alternate path:

```
# lvdisplay -v /dev/vg77/lvol1
--- Logical volumes ---
LV Name                     /dev/vg77/lvol1
VG Name                     /dev/vg77
LV Permission               read/write
LV Status                   available/syncd
Mirror copies               0
Consistency Recovery        MWC
Schedule                    parallel
LV Size (Mbytes)            252
Current LE                  63
Allocated PE                63
Stripes                     0
Stripe Size (Kbytes)        0
Bad block                   on
Allocation                  strict
IO Timeout (Seconds)        default

   --- Distribution of logical volume ---
   PV Name            LE on PV  PE on PV
   /dev/dsk/c7t13d5   63        63

   --- Logical extents ---
   LE    PV1                PE1   Status 1
   00000 /dev/dsk/c7t13d5   00000 current
   00001 /dev/dsk/c7t13d5   00001 current
   00002 /dev/dsk/c7t13d5   00002 current
   00003 /dev/dsk/c7t13d5   00003 current
[…]

# sar -d 3 3

HP-UX smart B.11.11 U 9000/800    08/08/07

14:56:27   device    %busy   avque   r+w/s  blks/s  avwait  avserv
14:56:30   c2t6d0     4.00    0.50       4      40    7.25   13.85
          c5t13d5    86.67    0.50    2449   19589    5.03    0.34
```

autopath adds entries into the syslog.log-file:

```
# grep AUTOPATH /var/adm/syslog/syslog.log
Aug  8 14:39:59 smart vmunix: AUTOPATH : Path 0xbc07d500 failed! Rerouting to
alternate path
Aug  8 14:58:08 smart vmunix: AUTOPATH : Path 0xbc07d500 recovered
```

Update to SP2

The basic version of Secure Path can be updated via downloads. The current version of SecurePath acive/active is `v3.0F-SP2 Patch 2`. Use the script for installation. After the installation, you may notice, that virtual device files are created:

```
/hpap/dsk/hpapX
/hpap/rdsk/hpapX
```

`autopath display` shows the LUNs with corresponding device file:

```
# autopath display | more

==================================================================
 HPswsp Version        : A.3.0F.02F.02F
==================================================================
 Auto Discover         : OFF
==================================================================
 Array Type            : XP
 Array WWN             : 3097-1
 Path Verification Period : 00:10
==================================================================

[...]

==================================================================
 Lun WWN               : 0400-78FB-00C9
 Virtual Device File   : /hpap/dsk/hpap110
 Load Balancing Policy : Round Robin
 Lun Timeout           : Infinite Retry (-1)
==================================================================
 Device Path                    Status
==================================================================
 /dev/dsk/c5t13d7               Active
 /dev/dsk/c7t13d7               Active

==================================================================
```

The virtual device file should be used, if the LUN is mounted directly to a mount point. This special file is the same as the first disk’s device file.

```
# ll /hpap/dsk/hpap110 /dev/dsk/c5t13d7
brw-r-----   1 bin        sys          31 0x05d700 Mar 26 13:21 /dev/dsk/c5t13d7
brw-r-----   1 root       sys          31 0x05d700 Aug 23 13:31 /hpap/dsk/hpap110

# lssf /hpap/dsk/hpap110 /dev/dsk/c5t13d7
sdisk card instance 5 SCSI target 13 SCSI LUN 7 section 0 at address
1/10/0/0.3.16.0.0.13.7 /hpap/dsk/hpap110
sdisk card instance 5 SCSI target 13 SCSI LUN 7 section 0 at address
1/10/0/0.3.16.0.0.13.7 /dev/dsk/c5t13d7
```

The virtual device files cannot be used within an LVM configuration:

```
# vgextend vg77 /hpap/dsk/hpap110
vgextend: /hpap/dsk/hpap110 has no correspoding valid raw device file under
/dev/rdsk.
Verification of unique LVM disk id on each disk in the volume group
vg77 failed.
```

New Type of Device Special Files

Format of a persistent DSF

So-called legacy device special files (DSF) were used with HP-UX 11i v2 and previous releases. DSFs were mainly path-oriented and associated with a specific legacy hardware path. Each multi-path device has more than one legacy DSF. Further information about the legacy DSFs is available in the beginning of this chapter.

HP-UX 11.31 introduced a new type of DSFs, the persistent (or agile) DSFs. They are connected directly to the LUN and not to the path to a LUN. A persistent DSF is a path-independent and persistent representation of a logical unit (LUN). It is bound to a LUN using the World Wide Identifier (WWID).

Legacy DSFs are known in the format

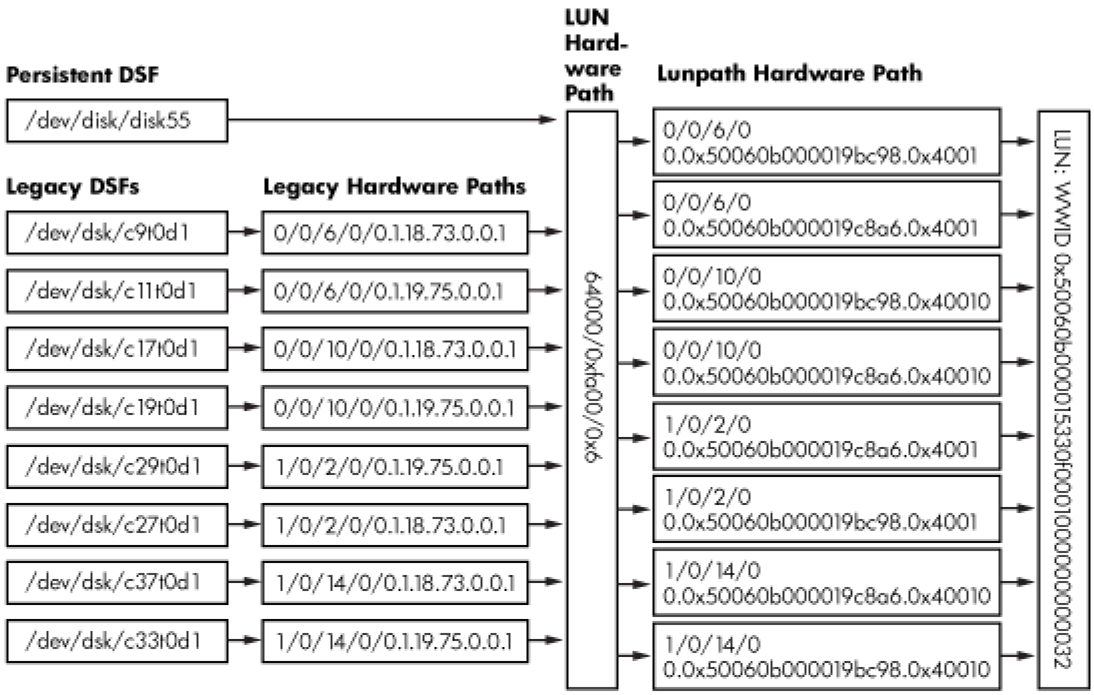
```
/dev/(r)dsk/cXtYdZ
```

`with X`: instance number, `Y`: target address (example: SCSI ID), `Z`: Device unit number at the target address.

The new persistent DSFs have the following format:

```
/dev/(r)disk/diskX
```

with `X`: instance number, assigned automatically by the operating system.



New DSF Naming Conventions

Persistent DSF name	Legacy DSF name	Description
/dev/disk/disk#	/dev/dsk/c#t#d#	The entire disk(Block Access)
/dev/rdisk/disk#	/dev/rdsk/c#t#d#	The entire disk (Raw Access)
/dev/disk/disk#_p#	/dev/dsk/c#t#d#s#	Partition on the disk(Block Access)
/dev/rdisk/disk#_p#	/dev/rdsk/c#t#d#s#	Partition on the disk(Raw Access)
/dev/rtape/tape#options	/dev/rmt/c#t#d#options	Tape device (Raw access)
/dev/rchrgr/autoch#	/dev/rac/c#t#d#_options	Autochanger device (raw access)
/dev/pt/ptinstance	/dev/rscsi/c#t#d#	Pass-through device

Creating persistent DSFs manually

During a cold installation of HP-UX 11i v3 both types of DSFs are created. Legacy DSFs are used mainly for backward compatibility. If an HP-UX update is performed, only persistent DSFs are created. Persistent DSFs can be created manually to

- restore device files deleted accidentally
- create a pass-through device file for a device

- create a custom device special file corresponding to certain device options

Following commands are used to create DSFs manually: `insf, mksf, mknod`

- `insf` re-creates device special files created by the system for devices of a particular class; devices bound to a specific driver; or for all devices on the system. If new devices are present on the system, it allocates new minor numbers for these devices. It stores minor numbers, device options and device in a registry.

```
# insf -e
insf: Installing special files for sdisk instance 2 address 0/0/2/0.0.0.0
insf: Installing special files for sdisk instance 0 address 0/1/1/0.0.0
insf: Installing special files for sdisk instance 1 address 0/1/1/0.1.0
insf: Installing special files for sdisk instance 30 address 0/3/1/0.3.2.0.0.0.1
insf: Installing special files for sdisk instance 31 address 0/3/1/0.3.2.0.0.0.2
insf: Installing special files for sctl instance 9 address 0/3/1/0.3.2.255.0.0.0
insf: Installing special files for asio0 instance 0 address 0/7/1/0
```

Note: If no message appears while executing the command, there are either no new devices on the system, or the "–e" option was omitted.

- `mksf` creates all device special files created by the system by default for a particular device; a custom device special file corresponding to certain device options. The device special file name can be specified. If a minor number was not already allocated for the device options specified, the system allocates a new minor number, and stores the binding between it and the device options in a registry.

Example: Create the default block device file for a disk device with instance number 4 and hardware path `64000/0xfa00/0x1` using the instance number to identify the device:

```
# mksf -C disk -I 4
# ls -l /dev/disk/disk4
brw-r-----   1 bin     sys        1 0x000001 Jul 21 09:52 /dev/disk/disk4
```

Example: Create the default raw device file for a disk device with instance number 4 and hardware path `64000/0xfa00/0x1` using the hardware path to identify the device:

```
# mksf -r -H 64000/0xfa00/0x1
# ls -l /dev/rdisk/disk4
crw-r-----   1 bin     sys       11 0x000001 Jul 21 10:34 /dev/rdisk/disk4
```

You can test creating a device file for a partition of a disk the following way:

```
# mkdir /mydev
# mksf -C disk -I 4 -s 1 /mydev/disk4_partition1
# ls /mydev
disk4_partition1
```

Note that the directory `/mydev` should be created before creating the DSF. It is not recommended to create DSFs outside `/dev` other than test issues.

- `mknod` creates a device special file by excplicitly specifying the major number, the minor number, and the device special file name. Unlike the legacy minor numbers, the new persistent minor numbers are opaque values assigned by the system. They cannot be constructed by the user from the hardware addressing components. Using `mknod` to create a persistent DSF requires the use of a minor number already assigned to the device by the system.

`ioscan` output from the new DSFs

There are new options added to the `ioscan` command to show the new device special files, and to map the persistent to the new DSFs.

The following outputs show the different formats of legacy and persistent DSFs for hard disks:

```
# ioscan -fnC disk
Class     I  H/W Path       Driver S/W State   H/W Type     Description
===================================================================
disk      3  0/0/2/0.0.0.0  sdisk   CLAIMED     DEVICE       TEAC    DV-28E-N
                           /dev/dsk/c0t0d0   /dev/rdsk/c0t0d0
disk      0  0/1/1/0.0.0    sdisk   CLAIMED     DEVICE       HP 36.4GMAX3036NC
                           /dev/dsk/c2t0d0   /dev/rdsk/c2t0d0
disk      1  0/1/1/0.1.0    sdisk   CLAIMED     DEVICE       HP 36.4GMAX3036NC
                           /dev/dsk/c2t1d0   /dev/rdsk/c2t1d0
disk      2  0/1/1/1.2.0    sdisk   CLAIMED     DEVICE       HP 36.4GMAX3036NC
                           /dev/dsk/c3t2d0     /dev/rdsk/c3t2d0
                           /dev/dsk/c3t2d0s1   /dev/rdsk/c3t2d0s1
                           /dev/dsk/c3t2d0s2   /dev/rdsk/c3t2d0s2
                           /dev/dsk/c3t2d0s3   /dev/rdsk/c3t2d0s3
```

```
# ioscan -fnC disk -N
Class     I  H/W Path  Driver S/W State   H/W Type     Description
===================================================================
disk      4  64000/0xfa00/0x0  esdisk   CLAIMED     DEVICE       HP 36.4GMAX3036NC
                      /dev/disk/disk4   /dev/rdisk/disk4
disk      5  64000/0xfa00/0x1  esdisk   CLAIMED     DEVICE       HP 36.4GMAX3036NC
                      /dev/disk/disk5   /dev/rdisk/disk5
disk      6  64000/0xfa00/0x2  esdisk   CLAIMED     DEVICE       HP 36.4GMAX3036NC
                      /dev/disk/disk6      /dev/rdisk/disk6
                      /dev/disk/disk6_p1   /dev/rdisk/disk6_p1
                      /dev/disk/disk6_p2   /dev/rdisk/disk6_p2
                      /dev/disk/disk6_p3   /dev/rdisk/disk6_p3
disk      7  64000/0xfa00/0x3  esdisk   CLAIMED     DEVICE       TEAC    DV-28E-N
                      /dev/disk/disk7   /dev/rdisk/disk7
```

Another new option added to `ioscan` is "`-m`", which maps legacy to persistent DSFs. It has three options:

```
# ioscan -m lun [-F] [-d driver | -C class] [-I instance] [-H lun hw_path] [devfile]
       -m: show mappings

# ioscan [-F] -m dsf [devfile]
```

```
# ioscan -m hwpath [-F] [-H hw_path]
```

It returns following outputs:

```
root@riga:/ # ioscan -m lun
Class     I  Lun H/W Path  Driver  S/W State   H/W Type     Health  Description
======================================================================
disk      5  64000/0xfa00/0x0   esdisk  CLAIMED     DEVICE       online  HP 36.4GMAX3036NC
             0/1/1/0.0x5000c50003e9c469.0x0
                      /dev/disk/disk4      /dev/rdisk/disk4
disk      6  64000/0xfa00/0x1   esdisk  CLAIMED     DEVICE       online  HP 36.4GMAX3036NC
             0/1/1/0.0x5000c50003ee3109.0x0
                      /dev/disk/disk5   /dev/rdisk/disk5
disk      7  64000/0xfa00/0x2   esdisk  CLAIMED     DEVICE       online  HP 36.4GMAX3036NC
             0/1/1/0.0x5000c50003eff69d.0x0
                      /dev/disk/disk6   /dev/rdisk/disk6
                      /dev/disk/disk6_p1   /dev/rdisk/disk6_p1
                      /dev/disk/disk6_p2   /dev/rdisk/disk6_p2
                      /dev/disk/disk6_p3   /dev/rdisk/disk6_p3
```

```
# ioscan -m dsf

Persistent DSF            Legacy DSF(s)
========================================
/dev/rdisk/disk4          /dev/rdsk/c2t0d0
/dev/rdisk/disk5          /dev/rdsk/c2t1d0
/dev/rdisk/disk6          /dev/rdsk/c3t2d0
/dev/rdisk/disk6_p1       /dev/rdsk/c3t2d0s1
/dev/rdisk/disk6_p2       /dev/rdsk/c3t2d0s2
/dev/rdisk/disk6_p3       /dev/rdsk/c3t2d0s3
/dev/rdisk/disk7          /dev/rdsk/c0t0d0

# ioscan -m hwpath

Lun H/W Path      Lunpath H/W Path                Legacy H/W Path
====================================================================
64000/0xfa00/0x0
                  0/1/1/0.0x0.0x0                 0/1/1/0.0.0
64000/0xfa00/0x1
                  0/1/1/0.0x1.0x0                 0/1/1/0.1.0
64000/0xfa00/0x2
                  0/1/1/1.0x2.0x0                 0/1/1/1.2.0
64000/0xfa00/0x3
                  0/0/2/0.0.0x0.0x0               0/0/2/0.0.0.0
```

For a particular device the hardware path can be used:

```
# ioscan -m hwpath -H 0/1/1/1.2.0
Lun H/W Path      Lunpath H/W Path                Legacy H/W Path
====================================================================
64000/0xfa00/0x2
                  0/1/1/1.0x2.0x0                 0/1/1/1.2.0
```

There are three types of hardware paths as seen above: LUN HW path, LUN path HW path and Legacy HW Path. The **LUN hardware path** is a **virtualised** hardware path for a LUN. **LUN path hardware path** is the **physical hardware path** to the LUN and similar to the legacy hardware path. **Legacy hardware path** is the hardware path known from previous releases.

io_redirect_dsf

`io_redirect_dsf` is a new command introduced with HP-UX 11.31. It redirects the persistent DSF from one device to a different device. This is equivalent of the command `sasmgr replace_tgt`.

`sasmgr` with the `replace_tgt` option was used to remap the DSF of the failed SAS disk to the replacement disk with HP-UX 11.23. With HP-UX 11.31 `sasmgr replace_tgt` can still be used, but it supports legacy DSFs only. `io_redirect_dsf` uses and remaps the persistent DSFs. It redirects the persistent DSF from one device to another device. The command can be used only, if both devices are from the same class. `io_redirect_dsf` redirects the DSF only to an existing device. If it does not exist it has to be created with `insf` or `mksf`.

Follow this procedure to replace a failed disk (for non-root disks only):

- Save the hardware paths of the disk:

  ```
  # ioscan -m lun /dev/disk/disk14
  Class   I  Lun H/W Path  Driver  S/W State   H/W Type     Health  Description
  ======================================================================
  disk   14  64000/0xfa00/0x0   esdisk  CLAIMED   DEVICE   online  HP
  36.4GMAX3036NC
                0/1/1/1.0x3.0x0
                         /dev/disk/disk14   /dev/rdisk/disk14
  ```

  The instance number is `14`, the LUN hardware path is `64000/0xfa00/0x0` and the LUN path hardware path is `0/1/1/1.0x3.0x0`

  Note: When the disk has been replaced, a new LUN instance and LUN hardware path will be created. To identify the disk (once it has been replaced) the LUN path hardware path should be used (here: `0/1/1/1.0x3.0x0`)

- Halt LVM access to the disk. If the disk is not hot-swappable, the system has to be shut down. Otherwise use:

  ```
  pvchange -a N /dev/disk/disk14
  ```

- Replace the disk

- Notify the mass storage sub-system that the disk has been replaced. If the system was not rebooted to replace the failed disk, then `scsimgr` has to be run before the new disk can be used:

  ```
  # scsimgr replace_wwid –D /dev/rdisk/disk14
  ```

This command allows the storage sub-system to replace the old disk's WWID with the new disk's LUN WWID. The storage sub-system will create a new LUN instance and new device special files for the replacement disk. `scsimgr replace_wwid` in combination with `scsimgr replace_leg_dsf` replaces `fcmsutil replace_dsk`.

- Determine the new instance number for the disk by running `ioscan –m`:

```
# ioscan -m lun
Class   I  Lun H/W Path  Driver  S/W State   H/W Type     Health  Description
========================================================================
disk   14  64000/0xfa00/0x0   esdisk  NO_HW      DEVICE   online  HP
36.4GMAX3036NC
                      /dev/disk/disk14   /dev/rdisk/disk14
…

disk   28  64000/0xfa00/0x1c   esdisk  CLAIMED     DEVICE online  HP
36.4GMAX3036NC
             0/1/1/1.0x3.0x0
                      /dev/disk/disk28   /dev/rdisk/disk28
```

In this example a new instance `28` has been created for the new disk with LUN Hardware Path `64000/0xfa00/0x1c`, devices `/dev/disk/disk28` and `/dev/rdisk/disk28` and LUN path hardware path `0/1/1/1.0x3.0x0`. The failed disk `disk14` has no LUN path associated with it.

- Run `io_redirect_dsf` to reassign the old LUN instance number to the new disk:

```
#io_redirect_dsf -d /dev/disk/disk14 -n /dev/disk/disk28
```

- Check with `ioscan -m lun`:

```
# ioscan -m lun
Class   I  Lun H/W Path  Driver  S/W State   H/W Type     Health  Description
========================================================================
disk   14  64000/0xfa00/0x1c   esdisk  CLAIMED     DEVICE nline  HP
36.4GMAX3036NC
             0/1/1/1.0x3.0x0
                      /dev/disk/disk14   /dev/rdisk/disk14
```

The LUN representation of the old disk with the LUN hardware path `64000/0xfa00/0x0` has been removed. The LUN representation of the new disk with LUN Hardware path `64000/0xfa00/0x1c` has been reassigned from LUN instance `28` to LUN instance `14`. Its device special files have been renamed to `/dev/disk/disk14` and `/dev/rdisk/disk14`.

- Restore the data (for LVM, see the LVM chapter of this handbook).

Mass Storage Stack Native Multipathing

Native Multipathing as a replacement to Secure Path

HP-UX 11iv2 and earlier versions use HP Storage Works Secure Path to provide high-availability access to HP active/active disk arrays. Secure Path for HP-UX 11.23 and previous versions are described earlier in this chapter.

HP-UX 11iv3 does not support Secure Path. Multipathing is fully integrated into the operating system. Multipathing is the ability to use multiple paths to a LUN to provide the following benefits:

- Availability: transparent recovery from path failures via failover to an alternate path.
- Performance: increased I/O performance via load-balancing of I/O requests across available paths.

After upgrades to HP-UX 11iv3, the virtual aliases created by Secure Path (for active/active DAs) are no longer available. Applications use the persistent DSFs provided by the operationg system, instead of Secure Path. Persistent DSFs are created for each LUN in the system, and they are no longer path depedant in nature.

Secure Path migration is very easy. Migration scripts are implemented in the HP-UX 11iv3 upgrade kit and are invoked automatically when the operating system is upgraded to HP-UX 11iv3. The following actions take place during the upgrade:

- Deinstallation of HP Storage Works Secure Path for active/active and active/passive disk arrays (if present on the system).
- Map the pseudo-DSF (provided to the LUNs by Secure Path for active/passive disk arrays) to their corresponding LUN WWN. This information is stored in `/opt/HPswsp_migration/SecurePathVDsf_To_NativeMultipathingDsf.txt`. Each line in the file is in the format: <CTD><LunWWN>, where <CTD> is the CTD assigned to a LUN by Secure Path for active/passive disk arrys. <LunWWN> is the WWN of the LUN.
- Maps the virtual aliases (provided to the LUNs by Secure Path for active/active disk arrays) to their corresponding HP-UX 11i v3 disk device special files. This information is stored in `/opt/HPswsp_migration/SecurePathVDsf_To_NativeMultipathing.txt` file. Each line in the file is of the format: <SecurePathAlias>:<NewStyleDSF>, where <SecurePathAlias> is the alias provided by Secure Path for active/active disk arrays. <NewStyleDSF> is the new DSF introduced in HP-UX 11i v3 to represent each LUN.
- Update `/etc/fstab` with HP-UX 11i v3 disk device special files by replacing the virtual aliases provided by Secure Path for active/active disk arrays.

Following issues are important to know:

- Load-balancing policies for Secure Path LUNs are not migrated to the corresponding native multipathing load-balancing policies.

HP-UX 11iv3 provides native multipathing built into the operating system. If there are multiple hardware paths from the host to the LUN, the operating system distributes the I/O requests across all available paths to the LUN using a choice of load-balancing policies. Policies include:

- round-robin: a path is selected in round-robin principle. This is the default policy for random access devices.

- least-command-load: the path with the smallest number of outstanding I/O requests is selected.

- cell-local round-robin: a path belonging to the cell (in cell-based systems), where the I/O request was initiated, is selected.

- path lock down: a single specified path is selected for all I/O requests to the LUN. This is primarily used for sequential access devices.

- preferred path: a path belonging to a user-specified “preferred” path is selected. This is similar to “path lockdown”, Additionally, it also provides automatic path failover, when the preferred path fails.

The load-balancing policies operate the list of available paths to a LUN. They can change as paths go offline or online due to error conditions or disabling via `scsimgr`.

The performance benefits of load-balancing include:

- Increased I/O throughput via balanced utilization of all available paths.

- Decreased CPU utilization (and increased aggregate throughput capability) on cell-based machines using cell-local load-balancing.

Further information:

http://bizsupport1.austin.hp.com/bc/docs/support/SupportManual/c01915741/c01915741.pdf

The HP-UX 11iv3 mass storage stack provides new or improved performance tracking capabilities, including information about:

- Per HBA performance

- Tape performance

- Read/Write performance
- Per LUN path performance
- Statistics at various level of mass storage stack

`sar` has new option to show the aforementioned information:

Per LUN path performance

```
# sar -L 1 5

HP-UX stormpea B.11.31 U 9000/800    07/22/07

20:33:32        lunpath    %busy   avque     r/s      w/s  blks/s  avwait  avserv
                                   %age      num      num     num     num    msec     msec
20:33:33        disk3_lunpath0     2.97     0.50        0       4      63    0.00    12.48
20:33:34
20:33:35        disk3_lunpath0     3.00     0.50        0       3      48    0.00    16.47
20:33:36
20:33:37        disk3_lunpath0     2.00     0.50        0       4      82    0.00    12.91

Average         disk3_lunpath0     1.60     0.50        0       2      39    0.00    13.72
```

Per HBA performance:

```
# sar -H 1 5

HP-UX stormpea B.11.31 U 9000/800    07/22/07

20:38:03     ctlr    util  t-put  IO/s   r/s   w/s   read  write avque avwait avserv
                     %age   MB/s   num   num   num   MB/s   MB/s   num   msec   msec
20:38:04
20:38:05    c8xx2       2   0.02     3     0     3   0.00   0.02     1      0     12
20:38:06    c8xx2       1   0.01     1     0     1   0.00   0.01     1      0      9
20:38:07
20:38:08    c8xx2       2   0.03     3     0     3   0.00   0.03     1      0     12

Average     c8xx2       1   0.01     1     0     1   0.00   0.01     1      0     12
```

In addition to the `-H` and `-L` options, `sar` provides the following new options in HP-UX 11i v3:

- `-R` used with `-d` to add read vs. write breakdown on per LUN reporting.
- `-t` report tape device performance data.

The `iostat` command provides a new option `-L` to display I/O statistics for each active lunpath:

```
# iostat -L

                        lunpath     bps     sps    msps

                 disk3_lunpath0      16     1.9     1.0
```

The command `scsimgr` provides options to display and clear a variety of I/O statistics globally, or for various components of the mass storage stack.

scsimgr

`scsimgr` performs management and diagnostic operations on SCSI objects (single LUN, LUN path, target path, HBA controller, or a group of these, specified as an *identifier* in the command line) and sub-systems (SCSI services module, SCSI class driver, or SCSI interface [I/F] driver).

Example: `scsimgr` on HP-UX 11i v3

```
# scsimgr -h

    command : command to execute. Can be one of the following :
        get_stat             Get statistics
        clear_stat           Clear statistics
        clear_kmstat         Clear Kmetric data
        get_info             Get status information
        lun_map              List LUN paths of a LUN
        get_attr             Get values and description of attributes
        set_attr             Change current values of settable attributes
        save_attr            Save values of settable attributes
        disable              Disable a SCSI object
        enable               Enable a SCSI object
        replace_wwid         Validate binding between a new LUN and LUN paths
        replace_leg_dsf      Change binding of a legacy device file to a LUN
        lun_reset            reset a LUN
        warm_bdr             Perform a warm  reset on a target
        cold_bdr             Perform a cold reset on a target
        set_devid            Set device identifier for a LUN
        get_devid            Get device identifier for a LUN
        sync_cache           Request synchronization of a block device's cache
        erase                Erase blocks of an optical block device
        inquiry              Perform SCSI command INQUIRY
        ddr_add              Add a new settable attribute scope for a driver
        ddr_del              Delete a settable attribute scope for a driver
        ddr_list             List all registered settable attribute scopes
```

Examples: Some options of `scsimgr`

```
# scsimgr get_stat

        SCSI SERVICES GLOBAL STATISTICS:

bioerrors                                                    = 18
Device lookup failures                                       = 1407
Legacy device lookup failures                                = 0
ctlr registration failures                                   = 0
Ctlr unregistrations failures                                = 0
tgt reg failures due to invalid parms/cxt                    = 0
I/F Invalid Asynchronous Event Notifications                 = 0
tgt unreg failures due to invalid parms/cxt                  = 0
Legacy bus instance overflows                                = 0
Legacy bus registration failures                             = 0
Legacy target registration failures                          = 0
Legacy LUN registration failures                             = 0
I/F driver invalid calls for inquiry data                    = 0
device open failure due to invalid mode                      = 0
IOCTL failures due to lack of user privileges                = 0
Open failures due to legacy mode disabled                    = 0
last time statistics were cleared                            = N/A
```

Note: Option `-D` and the name of the LUN displays information about the lunpaths of a LUN. Without any options information for all LUNs are shown:

```
# scsimgr lun_map

        LUN PATH INFORMATION FOR LUN : /dev/rdisk/disk4

Total number of LUN paths     = 1
World Wide Identifier(WWID)    = 0x50014c30020baaf4

LUN path : lunpath0
Class                         = lunpath
Instance                      = 0
Hardware path                 = 0/1/1/0.0x0.0x0
SCSI transport protocol       = parallel_scsi
State                         = ACTIVE
Last Open or Close state      = ACTIVE
…..
```

More information about `scsimgr` and examples can be found in the document: "`scsimgr` SCSI Management and Diagnostics utility on HP-UX 11i v3" (www.docs.hp.com).

msv2v3check

Run `msv2v3check` on HP-UX 11i v2 (11.23) to validate whether the drivers, mass storage I/O cards, and mass storage devices on your system are supported by HP-UX 11i v3. This script will report any unsupported drivers, mass storage I/O cards, and mass storage devices on your system.The `msv2v3check` script is availabe here:

http://h20293.www2.hp.com/portal/swdepot/displayProductInfo.do?productNumber=MSV2V3Check

Example:

```
# msv2v3check

======= STARTING at Fri Jul 20 11:17:10 EETDST 2007

WARNING:  The utility could not determine the latest firmware version
          for the device at H/W path 0/0/2/1.0.16.0.0. Please
          manually verify that this device is running the minimum
          required firmware version for HP-UX 11i v3.
…………

NOTE:     For information on supported HP Storage devices and
          third-party storage devices which have been qualified as
          interoperable with HP-UX 11i v3, please refer to the following
          web-site:
          http://www.hp.com/products1/serverconnectivity/index.html ,
          respectively under Mass Storage Connectivity and HP-UX Third
          Party Connectivity. For support of all other storage devices
          on HP-UX 11i v3, HP recommends that you contact your
          third party storage vendor to determine compatibility of
```

```
          third-party (non-HP) storage with HP-UX 11i v3.

        * VALIDATION RESULT: 0 errors, and 3 warnings.

          Congratulations! This system will install and boot HP-UX 11i
          v3 successfully. However, please review all WARNING
          messages as your current mass-storage configuration may
          not fully operate on HP-UX 11i v3.

======= ENDED at Fri Jul 20 11:17:32 EETDST 2007
```

Native multipathing with LVM

HP-UX 11iv3 LVM supports both legacy and persistent DSFs. A volume group can be configured using both types of DSFs. Some volume groups can use legacy device special files, some can use persistent ones, and some can use a mixture of both types. A volume group currently using one DSF naming model can be extended (by `vgextend`) with the same or another physical volume using a different DSF naming model. Such operations result in a mixed mode volume group and facilitate a phased DSF migration from legacy to agile naming model.

Example: `/etc/lvmtab in` a mixed mode

```
strings /etc/lvmtab
/dev/vg00
/dev/disk/disk6_p2
/dev/disk/disk5
/dev/disk/disk4
```

Note: If mixed mode is activated (the legacy naming model is not de-activated), volume groups, physical volumes, and logical volumes can use both naming models. Operations on physical volume can utilize both types of DSFs. If the physical volume was created using persistent DSF, `ioscan -m dsf` shows the legacy equivalent:

```
# ioscan -m dsf
Persistent DSF           Legacy DSF(s)
========================================
/dev/rdisk/disk4         /dev/rdsk/c2t0d0
/dev/rdisk/disk5         /dev/rdsk/c2t1d0
/dev/rdisk/disk6         /dev/rdsk/c3t2d0
/dev/rdisk/disk6_p1      /dev/rdsk/c3t2d0s1
/dev/rdisk/disk6_p2      /dev/rdsk/c3t2d0s2
/dev/rdisk/disk6_p3      /dev/rdsk/c3t2d0s3
/dev/rdisk/disk7         /dev/rdsk/c0t0d0
```

HP recommends using persistent DSFs for LVM configurations, especially for new volume groups. Existing volume groups configured with legacy DSFs should be migrated to use corresponding persistent DSFs.

New options have been added to `vgimport` (option `-N`) and `vgscan` (options `-N -B`) to specify the persistent naming model.

- `vgimport -N`: Configure the volume group using persistent DSFs. Omit `-N` to use legacy DSFs. This option can be used with the scan option `-s` only.

- `vgscan -N`: Recover `/etc/lvmtab` file using persistent DSFs. Exception: Active volume groups configured with legacy DSFs; then, `vgscan` populates `/etc/lvmtab` using legacy DSFs.

- `vgscan -B`: Recover `/etc/lvmtab` file using both persistent and legacy DSFs. This option can be used to migrate a volume group configured with legacy DSFs to use corresponding persistent DSFs.

Example: After renaming `lvmtab` to `lvmtab.old`, and re-creating `lvmtab` with `vgscan -B`, the new `lvmtab` includes both types of DSFs (legacy and persistent).

```
#strings /etc/lvmtab
/dev/vg00
/dev/disk/disk6_p2
/dev/disk/disk5
/dev/disk/disk4
/dev/dsk/c2t0d0
/dev/dsk/c2t1d0
/dev/dsk/c3t2d0s2

# strings /etc/lvmtab.old
/dev/vg00
/dev/disk/disk6_p2
/dev/disk/disk5
/dev/disk/disk4
```

LVM Alternate Links

LVM supports alternate links to a device to secure continued access to the device in the event of a primary link failure. HP-UX releases prior to 11iv3 are able to use multiple paths alternately, but not simultaneously. The maximum link of a LUN is still 8 (1 primary and 7 alternate link).

Note: LVM still supports this functionality in HP-UX 11i v3, but with the introduction of HP-UX mass storage stack's native multipathing it is not recommended to configure LVM alternate links. If legacy LVM alternate link is preferred, the native multipathing feature can be disabled using `scsimgr`.

HP-UX 11i v3 LVM alternate links differ in behavior compared to previous releases:

- Regardless of whether LVM alternate links are configured or not, the mass storage stack processes the I/O operations to the device, using all available paths. LVM selects the LUN and the mass storage stack selects the lunpath.

- If LVM alternate links are configured, no link switching happens unless the mass storage stack's native multipathing feature is disabled (using `scsimgr`) and only legacy DSFs are used in the volume group configuration.

- Path switch options of `pvchange` (`-s` and `-S` options) do not switch links or stop I/O operations as they did in earlier releases. In order to disable I/O operations on a given path, use `scsimgr` (`scsimgr [-f] disable lunpath`).

- LVM alternate link functionality supports up to 8 paths to a physical volume. Mass storage stack's native multipathing feature supports up to 32 lunpaths.

Example: Enable legacy LVM alternate link functionality in a downward compatible manner. vg01 is configured with a single persistent DSF:

- List the DSFs configured in the volume group:

```
#vgdisplay -v -F vg01 | grep pv_name
```

- Find legacy DSFs corresponding to the persistent DSF:

```
# ioscan -m dsf /dev/disk/disk5
Persistent DSF                  Legacy DSF(s)
/dev/disk/disk5               /dev/dsk/c2t1d0
                              /dev/dsk/c3t1d0
```

- Add legacy DSFs as alternate links to the persistent DSF:

```
# vgextend vg01 /dev/dsk/c2t1d0 /dev/dsk/c3t1d0
```

- Remove the persistent DSF from the volume group:

```
# vgreduce vg01 /dev/disk/disk5
```

- Disable native multipathing through legacy DSFs for a specific LUN in a non-persistent way:

```
#scsimgr set_attr -D /dev/rdisk/disk5 -a leg_mpath_enable=false
Value of attribute leg_mpath_enable set successfully
```

  Disable native multipathing globally for all LUNs in the system in a non-persistent way:

```
#scsimgr set_attr -a leg_mpath_enable=false
Value of attribute leg_mpath_enable set successfully
```

- Verify the configuration. `scsimgr get_attr` shows the current setting of `leg_mpath_enable attribute value set to false` for all associated persistent DSFs in the volume group.:

```
#scsimgr get_attr -D /dev/rdisk/disk5 -a leg_mpath_enable
SCSI ATTRIBUTES FOR LUN : /dev/rdisk/disk5
name = leg_mpath_enable
current = false
default = false
```

```
saved =
```

LVM Migration

It is recommended to use persistent DSFs with HP-UX 11i v3. There are different ways to migrate LVM configurations from legacy to persistent DSFs. All of them update `/etc/lvmtab` or `/etc/lvmtab_p` automatically, and no explicit user action is required. However, some alternatives require manual editing of `/etc/lvmpvg`.

Note: Each migration updates one volume group at a time. The steps have to be repeated for all volume groups.

Procedure 1:

- Backup `/etc/lvmpvg`, if it exists. De-activate and export the volume group. Re-import the same volume group using persistent DSFs in place of legacy DSFs. Persistent DSFs and corresponding legacy DSFs can be found using

  ```
  ioscan -m dsf <legacy_dsf>
  ```

  Restore `/etc/lvmpvg` from backup. Replace legacy DSFs listed in the volume group directory with corresponding persistent DSFs. Mulitipath devices with a single persistent DSF can correspond to multiple legacy DSFs. The disadvantage of this method is the necessity of volume group de-activation.

- Run `vgscan-B`, followed by volume group activation. For each of the physical volumes configured in the volume group, `vgscan -B` populates `/etc/lvmtab` with both persistent and legacy DSFs. `/etc/lvmtab` supports a maximum of 8 paths per physical volume. If a physical volume has more than 8 paths configured, only 7 are retained in order to allow additional persistent DSF.

  Note: `vgscan` will not update existing volume group entries `/etc/lvmtab` unless option `-f` is used. It restricts the operation to the specified volume group (it overwrites existing volume group entries in `/etc/lvmtab`). If `/etc/lvmtab` is moved to a new location before running `vgscan -B`, `/etc/lvmtab` is recreated in its entirety, affecting all volume groups.

  After `vgscan -B`, use `vgchange` to reactivate the corresponding volume group. This reconfigures the volume group with the DSFs (both legacy and persistent DSFs) in `/etc/lvmtab`. Legacy DSFs can be removed later using `vgreduce`, leaving behind corresponding persistent DSFs. This facilitates a phased migration from legacy to agile naming model, while the volume group continues to remain active.

Procedure 2:

- Backup `/etc/lvmpvg`, if it exists. After the `vgreduce` operations, restore `/etc/lvmpvg` from back-up and replace the legacy DSFs with the corresponding persistent DSFs.

- The shell script `/usr/contrib/bin/vgdsf` performs the migration while the volume group remains active. Migrate the volume group to persistent DSFs

  - Identify the legacy DSFs configured in the volume group, using `vgdisplay -v`. Ignore alternate links.

  - For each legacy DSF found, look for its corresponding persistent DSF, using `ioscan -m`.

  - If a physical volume is configured with 8 paths, remove one using `vgreduce`. This will make room for additional persistent DSF.

  - Extend the volume group to add the persistent DSFs, keeping it in a physical volume group configuration, if any (`/etc/lvmpvg` gets updated automatically).

- Reduce the volume group of all legacy DSFs:

  - Identify all legacy DSFs configured in the volume group, using `vgdisplay -v`

  - For each of the legacy DSF with a corresponding persistent DSF, reduce the legacy DSF from the volume group. If the corresponding persistent DSF is not configured in the volume group a message is displayed on screen and the command continues with other DSFs.

  - Back up the configuration with `vgcfgbackup`.

  - Activate all configured volume groups in the system and identify the ones requiring a migration, using `vgchange -a`.

  - Take a note of the volume group configuration. Save the outputs from LVM commands in verbose mode: `vgdisplay`, `lvdisplay`, `pvdisplay`.

  - Ensure all physical volumes configured in the volume group are online using `ioscan -P health`.

  - Run `/usr/contrib.bin.vgdsf -c <vg_name>` to migrate each volume group. Ensure that no failures are reported.

  - Verify the resulting volume group configuration using LVM display commands. Use `ioscan -m dsf` to validate DSF mappings.

Below the vgdsf migration script:

```
#!/usr/bin/ksh
  DEV=$1
  LAST=""
  if [[ -s /etc/lvmpvg ]]; then
    if /usr/bin/grep -q "$DEV" /etc/lvmpvg; then
      TOKS=`/usr/bin/grep -e PVG -e "$DEV" /etc/lvmpvg|sed "s/PVG *//"| \
           /usr/bin/awk '{ print $1 }'`
      for TOK in $TOKS
      do
        if [[ $TOK = $DEV ]]; then
          if [[ -n $LAST ]]; then
            PVG="-g $LAST"
          fi
          return
        fi
        LAST=$TOK
      done
    fi
  fi
}
```

For further information of LVM, please refer to the following links:

HP-UX LVM Supported Limits

http://bizsupport1.austin.hp.com/bc/docs/support/SupportManual/c02019992/c02019992.pdf

LVM Migration from Legacy to Agile Naming Model HP-UX 11iv3

http://bizsupport1.austin.hp.com/bc/docs/support/SupportManual/c01916036/c01916036.pdf

Literature and Additional Information

The technical development on the fibre channel market is fast and rapidly growing. Answers to questions regarding supported configurations and newest technology can be found on the following websites:

- http://docs.hp.com/en/oshpux11iv3.html

HP-UX 11i v2 to 11i v3 Mass Storage Stack Update Guide

http://bizsupport1.austin.hp.com/bc/docs/support/SupportManual/c01915888/c01915888.pdf

HP-UX 11i v3 Mass Storage Device Naming

http://bizsupport1.austin.hp.com/bc/docs/support/SupportManual/c01906504/c01906504.pdf

HP-UX 11i v3 Persistent DSF Migration Guide

http://bizsupport1.austin.hp.com/bc/docs/support/SupportManual/c01914759/c01914759.pdf

Maximum LUN Configuration and Considerations for HP-UX

http://bizsupport1.austin.hp.com/bc/docs/support/SupportManual/c02029814/c02029814.pdf

Manuals and Release Notes

- http://docs.hp.com/hpux/netcom/#Fibre%20Channel

Manual pages

`intro(7)` (manual page about device special files), `ioscan(1M)`, `ioinit(1M)`, `ioconfig(4)`, `lsdev(1M)`, `lssf(1M)`, `insf(1M)`, `mksf(1M)`, `rmsf(1M)`, `utochanger(7)`, `mc(1M)`, `scsictl(1M)`, `mt(1)`, `scsi_pt(7)`, `scsi_ctl(7)`, `scsimgr(1M)`

FC Glossary

AL_PA ArbitratedLoop_PhysicalAddress; the address of a fibre channel node in an *arbitrated loop*. (1byte; 0x00..0x7E)

Arbitrated Loop An interconnection scheme supporting up to 126 ports on a loop in a shared medium topology. This is HP's standard implementation of a loop. Same as *FC-AL*.

Cascading An interconnection of individual switches used to create larger Fabric configuration.

DFA Direct Fabric Attach – the connection between an N-port and an F-port.

E-Port Switch port to cascades switches.

EPL Enhanced Private Loop; HP's FC-AL implementation with limitations in order to use legacy devices (FC-devices in *PLDA* environment; e.g. Galaxy Disk Array) in a SAN; access like *PLDA*.

F-Port Switch port that operates according to the *P2P* protocol.

Fabric	A fibre channel interconnection method allowing multiple hosts and/or storage devices connected through a multi-port hub; multiple, simultaneous and concurrent data transfers. FC switch required.
FC-AL	Abbreviation for Fibre Channel Arbitrated Loop; see *Arbitrated Loop*.
FC-AL-2	Whereas FC-AL allows private loop only, FC-AL-2 allows both private and public loops. To run a public loop the device must handle 8-bit addresses within the private loop (FC-AL) and 24-bit addresses when communicating over the switch.
FC-FLA	FC Fabric Loop Attach; ANSI standard describing the communication between *NL*- und *FL-Ports*.
Fibre	Thin filament of glass. An optical wave quide consisting of a core and a cladding; capable of carrying information in the form of light. Fibre is also a general term used to cover all physical media types supported by Fibre Channel, such as optical fiber, twisted pair, and coaxial cable.
Fibre Channel	A high-speed, serial, bi-directional, topology-independent, multi-protocol, highly scalable interconnection between computers, peripherals, and networks.
FL-Port	Switch port that operates in Loop protocol.
Frame	Smallest unit of the transmission protocol between 2 N-ports. It consists of start sequence, header, payload (SCSI data), CRC and end sequence.
ISL	Inter Switch Link. A cable to cascade FC switches.
Loop	All nodes are attached in a ring topology (similar to FDDI or Token Ring).
Loop ID	Counterpart to AL_PA. Address of the node in *FC-AL* (126 to 0).
N-Port	Port of a storage device or host that operates according to *P2P* protocol.
NL-Port	A storage device or host that operates according to *FC-AL* protocol.
Node	Device that is connected over fibre channel (host, disk array, bridge, …).
P2P	Point-To-Point; direct connection between two nodes.
PLDA	Private Loop Direct Attach (synonym: private loop).
Port	FC-interconnection of a node.

Private Loop A private loop is an *Arbitrated Loop* which has no attached switch port (NL-port to NL-port). In theory, private loop devices can only communicate with other devices on the same loop (see *Translative Mode*).

Public Loop Arbitrated Loop with a switch port (FL-Port) that allows the nodes to communicate with other nodes outside the loop.

QL Quick Loop - a technique by which a Brocade switch can combine several fibre channel links into a single virtual private arbitrated loop.

TranslativeMode Switch function to allow communication of private targets (private loop protocol only) with public initiators not on the loop. The switch creates "phantom" fabric addresses. Private targets appear as physically connected (via private loop ports) to the fabric. The switch reroutes the data to the correct address.