scsimgr

SCSI Management and Diagnostics utility on HP-UX 11i v3

Abstract

This paper presents the scsimgr command introduced with HP-UX 11i v3 to provide SCSI management and diagnostics. scsimgr significantly enhances management and troubleshooting capabilities of the mass storage subsystem. The paper provides an overview of its features, its general syntax and examples showing how to use the command to accomplish some specific tasks.

Terms and definitions

Agile Addressing	The ability to address a LUN with the same device special file regardless of the location of the LUN, i.e. the device file for a LUN remains the same even if the LUN is moved from one HBA to another, from one switch/hub port to another or presented via a different target port to the host.
Agile naming model	DSF naming model introduced in HP-UX 11i v3 to support agile addressing.
Legacy DSF	A path dependent device special file following the legacy naming model conventions, wherein the DSF embeds the bus/target/lun/option for a specific path to a mass storage device (Ex. /dev/dsk/c#t#d#).
Legacy Hardware Path	Hardware path in the legacy format. Used for all components, mass storage or not.
Legacy naming model / legacy format	Device special file format convention used prior to HP-UX 11i v3. This model is maintained in HP-UX 11i v3 for backward compatibility purpose.
LUN	Logical unit that refers to an end storage device such as disk, tape, floppy, cdrom, changer, etc. This is the logical unit itself and does not represent the path to the logical unit.
LUN id	The SCSI address of the LUN within the target as defined in the SCSI standard.
lunpath	I/O path to a LUN in agile format.
Persistent DSF	DSF following the persistent naming model conventions. Does not contain any encoding of bus, target identifier or device specific option information.
SCSI class driver	A HP-UX device driver which manages one or more specific classes of mass storage devices. Ex: Disk Driver, Tape Driver, Changer Driver, Pass-through Driver.
Target Id	The target port identifier as defined in SCSI transport protocols.
WWID	SCSI Logical Unit World Wide identifier obtained from EVPD INQUIRY Page 0x83 of id_type 1, 2, 3, 7, 8 and association of 0.

Introduction

This paper presents the scsimgr command introduced with HP-UX 11i v3 (B.11.31) to provide SCSI management and diagnostics. It is intended for system administrators and other users of HP-UX 11i v3 (B.11.31) mass storage subsystem. The reader is assumed to have some basic knowledge of HP-UX mass storage subsystem configuration and troubleshooting.
Readers not familiar with mass storage features of HP-UX 11i v3 are encouraged to read the following documents first:

- The Next Generation Mass Storage Stack HP-UX 11i v3
- HP-UX 11i v3 Mass Storage Device Naming

The scsimgr command significantly extends and simplifies the management, diagnosis and troubleshooting capabilities of mass storage subsystem on HP-UX.

Mass storage tools on previous HP-UX releases can be classified as:

- SCSI transport specific: these tools manage HBAs of specific SCSI transport. Examples include fcmsutil for Fibre Channel (FC), and sasmgr for Serial Attached SCSI (SAS).
- Device class specific: these tools only manage a specific class of devices. An example is diskinfo for block devices.
- Non specific to a SCSI transport or a device class, but with relatively limited diagnostic capabilities: An example is scsictl.

The scsimgr command complements these tools by providing generic management capabilities for all SCSI objects and subsystems, and by enabling device class or SCSI transport type specific management capabilities through plug-in modules. Its extensible architecture allows future consolidation of management and diagnostic tasks for the mass storage subsystem into a single tool.

This paper is divided into three parts:

- The first part "Main features and benefits", provides an overview of the capabilities of the scsimgr command and their benefits.
- The second part "General syntax and summary of options", gives the general syntax of the scsimgr command and a brief description of its options.
- The third part "Examples of using scsimgr", provides examples of scsimgr commands and output. Readers interested in how to use scsimgr to accomplish some specific tasks can go directly to this section.

Main features and benefits

The main features and benefits of the scsimgr command include:

- Basic management and diagnostic capabilities for all SCSI objects (LUNs, lunpaths, target paths and SCSI HBA controllers) independently of the drivers managing them.
- Plug-ins to handle operations dependent on device class (block, tape, changers, etc.) or SCSI transport type (Fibre Channel (FC), Serial attached SCSI (SAS), parallel SCSI (pSCSI), etc.). When first released HP-UX 11i v3 provided three plug-ins respectively for: block devices, tapes and auto-changers. These plug-ins allow scsimgr to display and set status information, statistics and attributes specific to these classes of SCSI devices.
- Extended sets of statistics, status information, and attributes to aid in monitoring the operation of the mass storage subsystem, and to simplify its management and troubleshooting.

- A flexible tunable architecture. Tunables can be set at different levels: global, a set of devices meeting some criteria, and a SCSI object instance. This hierarchical tunable architecture is very powerful. It allows the system administrator to fine tune the behavior of the stack to optimally interoperate with devices with different operational requirements, to adjust the level of resources used by the SCSI stack, and to work around some interoperability problems discovered in the field without having to wait until a patch is rolled out.
- Support for online troubleshooting and maintenance operations to increase the system availability. This includes:
  - The ability to get a minimum set of information on devices in situations where normal applications cannot open the device due to error conditions. The scsimgr command uses a dedicated management device file to retrieve cached information without having to open the device. In some cases it uses the pass-through driver to get information from devices.
  - Explicit validation of device replacement to prevent data corruption. The SCSI stack implements mechanisms to authenticate devices based on the LUN Worldwide Identifier (WWID). When a LUN with a different WWID is discovered through a previously discovered lunpath, the system prevents I/O transfer to the device through that lunpath until the system administrator validates the replacement.
  - The capability to disable I/O transfers on a lunpath to a block device or on all lunpaths to a block device, while performing online troubleshooting or maintenance operations. For instance a maintenance operation can be carried out on the SAN, without having to unmount file systems using affected disks, by simply disabling affected lunpaths.
- Contextual help providing information based on the command options specified.
- Capability to address SCSI objects using different types of identifiers including: class and instance numbers, device files, and hardware paths.

General syntax and summary of options

The syntax of the scsimgr command is as follow:

scsimgr [-fpv] command [-d driver] [identifier] [keyword ...] [argument ...]
scsimgr [-h] [-d driver] [command]

The following tables summarize the meaning of different components

Option	Description
-v	Verbose or extended output of *command* options
-p	Parsable or scriptable output. Note: This option is ignored for *command* options, which do not provide a scriptable output.
-f	Force execution of a destructive command without requesting user confirmation. Note: This option is ignored for command options, which do not require user confirmation before proceeding.
-h	Help information. If a command option is specified, contextual help information is provided for the command option.
-d <driver>	Specifies the name of a driver. Used to explicitly direct an operation to a driver plug-in.

command: specifies the operation to perform. Only one command can be specified at a time. It should be the first keyword in the command line.

Command keyword	Description
get_stat	Get statistics on SCSI objects
clear_stat	Clear statistics on SCSI objects
clear_kmstat	Clear kernel metrics (kmetrics) data structures for SCSI objects
get_info	Get status information on SCSI objects
lun_map	Show mapping between LUNs and lunpaths
get_attr	Get attributes of SCSI objects or a SCSI subsystem
set_attr	Set current values of attributes for a SCSI object or a SCSI subsystem
save_attr	Set and save in a persistent manner, current values of attributes for a SCSI object or subsystem
disable	Disable a SCSI object for I/O transfer
enable	Enable a SCSI object for I/O transfer
replace_leg_dsf	Validate change of the binding of a legacy device file to a LUN
replace_wwid	Validate LUN replacement
lun_reset	Perform the LUN RESET SCSI management function
warm_bdr	Perform the WARM TARGET RESET SCSI management function
cold_bdr	Perform the COLD TARGET RESET SCSI management function
set_devid	Set user-friendly device identifier for a device supporting SET/REPORT DEVICE IDENTIFIER SCSI commands
get_devid	Read a user-friendly device identifier for a device supporting SET/REPORT DEVICE IDENTIFIER SCSI commands
sync_cache	Synchronize a block device write cache
erase	Erase optical memory device surface
inquiry	Perform the INQUIRY command to retrieve standard inquiry data or vital product data
ddr_add	Add a settable attribute scope
ddr_del	Delete a settable attribute scope
ddr_list	List registered settable attribute scopes

identifier: specifies the object on which the operation applies. It is of the following form:
-D (<dsf>| <leg_dsf>) | -H <hw_path> | -C <class> -I <instance> | -N <attr_scope>

Identifier Options	Description Comment
dsf	Full path of a persistent character device special file (DSF). Example: /dev/rdisk/disk0.
leg_dsf	Full path of a legacy character device file (DSF). Example: /dev/rdsk/c0t0d0
class	Class of the object. Must be used in combination with instance number to fully identify an object. Example: disk, lunpath, fc, ext_bus, tgtpath Note: the class of a SCSI object can be obtained from the ioscan command.
instance	Instance number of the object. Must be used in combination with the class to fully identify an object. Example: 0, 1, 2, 3 Note: the instance number of a SCSI object can be obtained from the ioscan command.
hw_path	Hardware path of the SCSI object. e.g.: 64000/0xfa00/0x5, 0/3/1/0.0x21000020371972eb.0x0 Note: The hardware path of a SCSI object can be obtained from the ioscan command.
attr_scope	Settable attribute scope. Examples: • */escsi/esdisk*: global to the esdisk driver. • */escsi/esdisk/0x07*: all optical memory (OM) devices managed by the esdisk driver. 0x7 is the peripheral device type corresponding to OM • */escsi/estape*: global to estape driver • */escsi/eschgr*: global to the eschgr driver.

Notes:

- *The scsimgr command only accepts object identifiers in the agile view format with the exception of the following commands, which also accept legacy device special files (DSFs): get_stat, clear_stat, replace_leg_dsf.*
- *Refer to the scsimgr(1M) manpage for information on building a settable attribute scope from driver name, device type, vendor identifier, product identifier and firmware revision.*
- *The different components of the settable attribute scope should be specified exactly as they would be returned by the device in the inquiry data. For instance to specify a scope corresponding to all disk devices from HP bound to the esdisk driver, the attribute scope should be specified as: "/escsi/esdisk/0x0/HP ". Note how space characters are added to make sure that the vendor identifier corresponds to what is returned in the standard inquiry data.*

keyword: provides additional information on the scope of the command. Not all keywords are valid for all commands. See scsimgr(1M) for more information.

Keyword	Description
all_lun	Apply the operation to all LUNs.
all_ctlr	Apply the operation to all SCSI HBA controllers.
all_lpt	Apply the operation to all I/O paths of a specified LUN or under a specified target path.

all_tp	Apply the operation to all target paths under a specified SCSI HBA controller.
current	Attribute/parameter current values.
saved	Attribute/parameter value saved in a persistent store.
default	Attribute/parameter default values.
all_ddr	Apply the operation to all settable attribute scopes.

argument: provides additional parameters depending on the command. For attribute related commands, argument is of the form: *-a (<attr> | <attr>=<value> | <attr>=default)*. Where:

- *attr* : name of an attribute (example: *esd_secs, load_bal_policy*)
- *value*: value to assign to the attribute

Examples of using scsimgr

The following scsimgr examples show how to use some of the command important features.

Gathering status information and statistics

The scsimgr command provides an extensive set of status information, statistics and attributes for SCSI subsystems (class drivers, interface drivers) and objects (LUNs, lunpaths, target paths and SCSI HBA controllers).
This simplifies the monitoring of the mass storage activity and enables to quickly root cause problems. You can analyze statistics at different levels (interface driver, SCSI services, class driver) to root cause a problem.

The following output of the scsimgr command shows:

- Generic information displayed on SCSI objects independent of the driver managing them.
- Class specific information displayed with the help of class driver plug-ins. If a plug-in is present, scsimgr displays automatically the information.
- How to send a command to a driver plug-in with the '-d' option.
- How scsimgr is capable of accepting different types of identifiers for SCSI objects: device file name, class and instance number, hardware path.

Status information for a SCSI disk device identified by its device file name

```
# scsimgr get_info -D /dev/rdisk/disk20

        STATUS INFORMATION FOR LUN : /dev/rdisk/disk20

Generic Status Information

SCSI services internal state                 = UNOPEN
Device type                                  = Direct_Access
EVPD page 0x83 description code              = 1
EVPD page 0x83 description association       = 0
EVPD page 0x83 description type              = 3
World Wide Identifier (WWID)                 = 0x20000020371972eb
Serial number                                = LS24603100001008K74M
Vendor id                                    = SEAGATE
Product id                                   = ST39103FC
Product revision                             = HP06
Other properties                             =
SPC protocol revision                        = 2
Open count (includes chr/blk/pass-thru/class) = 0
Raw open count (includes class/pass-thru)    = 0
Pass-thru opens                              = 0
```

```
LUN path count                                  = 2
Active LUN paths                                = 2
Standby LUN paths                               = 0
Failed LUN paths                                = 0
Maximum I/O size allowed                        = 2097152
Preferred I/O size                              = 2097152
Outstanding I/Os                                = 0
I/O load balance policy                         = preferred_path
Path fail threshold time period                 = 0
Transient time period                           = 180
Tracing buffer size                             = 1024
LUN Path used when policy is path_lockdown      = NA

Driver esdisk Status Information :

Capacity in number of blocks                      = 17773524
Block size in bytes                               = 512
Number of active IOs                              = 0
Special properties                                =
Maximum number of IO retries                      = 45
IO transfer timeout in secs                       = 30
FORMAT command timeout in secs                    = 86400
START UNIT command timeout in secs                = 60
Timeout in secs before starting failing IO        = 240
IO infinite retries                               = true
```

Status information for a SCSI tape device identified by its device file name

```
# scsimgr get_info -D /dev/rtape/tape2_BEST

        STATUS INFORMATION FOR LUN : /dev/rtape/tape2_BEST

Generic Status Information

SCSI services internal state                    = UNOPEN
Device type                                     = Sequential_Access
EVPD page 0x83 description code                 = 1
EVPD page 0x83 description association          = 0
EVPD page 0x83 description type                 = 3
World Wide Identifier (WWID)                    = 0x50060b000059f20a
Serial number                                   = HU10617K90
Vendor id                                       = HP
Product id                                      = Ultrium 3-SCSI
Product revision                                = L54W
Other properties                                = Removable_Medium
SPC protocol revision                           = 3
Open count (includes chr/blk/pass-thru/class)   = 0
Raw open count (includes class/pass-thru)       = 0
Pass-thru opens                                 = 0
LUN path count                                  = 1
Active LUN paths                                = 1
Standby LUN paths                               = 0
Failed LUN paths                                = 0
Maximum I/O size allowed                        = 2097152
Preferred I/O size                              = 2097152
Outstanding I/Os                                = 0
I/O load balance policy                         = path_lockdown
Path fail threshold time period                 = 0
Transient time period                           = 0
Tracing buffer size                             = 1024
LUN Path used when policy is path_lockdown      = NA

Driver estape Status Information :

Drive type                                        = 0xf
File number                                       = 0
Block number                                      = 0
Head position                                     = Beginning of Tape (BOT)
Online                                            = 0
Write protected                                   = 0
Default command timeout in secs                   = 30
SPACE command timeout in secs                     = 1200
```

```
REWIND command timeout in secs                    = 600
UNLOAD command timeout in secs                    = 600
READ command timeout in secs                      = 600
WRITE command timeout in secs                     = 600
ERASE command timeout in secs                     = 18000
```

Status information for a lunpath identified by its class and instance

```
# scsimgr get_info -C lunpath -I 5

        STATUS INFORMATION FOR LUN PATH : lunpath5

Generic Status Information

SCSI services internal state                  = UNOPEN
Open close state                              = ACTIVE
Protocol                                      = fibre_channel
EVPD page 0x83 description code               = 1
EVPD page 0x83 description association        = 0
EVPD page 0x83 description type               = 3
World Wide Identifier (WWID)                  = 0x20000020371972eb
Outstanding I/Os                              = 0
Maximum I/O timeout in seconds                = 30
Maximum I/O size allowed                      = 2097152
Maximum number of active I/Os allowed         = 16
Current active I/Os                           = 0
Maximum queue depth                           = 16
Queue full delay count                        = 0
```

Status information for a target path identified by its hardware path

```
# scsimgr get_info -H 0/4/1/0.0x21000020371972eb

        STATUS INFORMATION FOR TARGET PATH : 0/4/1/0.0x21000020371972eb

Generic Status Information

SCSI services internal state                  = IDLE
Port id                                       = 0xe1
Protocol                                      = fibre_channel
Protocol revision                             = 4.3
Port name                                     = 0x21000020371972eb
Node name                                     = 0x20000020371972eb
LUN paths registered (active/inactive)        = 1
```

Status information for an HBA controller identified by its device file name

```
# scsimgr get_info -D /dev/td0

        STATUS INFORMATION FOR SCSI CONTROLLER : /dev/td0

Generic Status Information

SCSI services internal state                = IDLE
Target paths probed                         = 7
Target paths registered (active/inactive)   = 7
LUN paths registered                        = 7
Trace buffer size                           = 0
Port name                                   = 0x50060b000022cd16
Port id                                     = 0x1
Protocol                                    = fibre_channel
I/F driver version                          = B.11.31.ast_35
Firmware version                            = N/A
Operating negotiated/configured speed       = 1Gb
Maximum supported speed                     = 2Gb
Capability                                  = Boot Dump
Type                                        = Physical
Number of I/O objects                       = 1
I/O objects :
```

```
        Object index = 0, cpu = 0
```

Statistics for SCSI devices

LUN statistics can be divided into 3 categories:

- Generic statistics: They are commonly defined statistics independent of the class of the device. They track activities such as number of times the device has been opened, open/close failures, and so forth.
- I/O transfer statistics: They are commonly defined statistics independent of the class of the device. They track activity related to I/O transfer: number of bytes read or written, total number of I/O processed, etc. They are computed by consolidating I/O transfer statistics of lunpaths of the LUN.
- Class driver specific statistics: They depend on the class of the device. They are maintained by the class driver and require a plug-in module to be displayed.

Notes:

- *Not all the statistics actually displayed by scsimgr are shown.*
- *When the '-v' option is specified, an extended list of statistics is displayed for the object.*

Sample of a disk device statistics

```
# scsimgr get_stat -D /dev/rdisk/disk100

        STATISTICS FOR LUN :/dev/rdisk/disk100

Generic Statistics:

Overall attempted opens                          = 7097
Overall successfull opens                        = 7096
Attempted Pass-thru opens                        = 0
Successfull Pass-thru opens                      = 0
Overall closes                                   = 7096
Pass-thru closes                                 = 0
Offlines                                         = 3
Onlines                                          = 3
LUN path initializations                         = 4
Last time  cleared                               = N/A

I/O transfer Statistics:

Bytes read                                       = 217177561662
Bytes written                                    = 271582319104
Total I/Os processed                             = 33054140
I/O failures                                     = 0
Retried I/Os                                     = 119812
Retried I/O failures                             = 0
I/O failures due to invalid IO size              = 0
I/Os flushed                                     = 0
Check condition status                           = 5

Driver esdisk Statistics :

PR requests                                       = 0
Activation requests received                      = 0
Abort requests received                           = 0
Disable requests received                         = 1
Enable requests received                          = 1
LUN path addition requests                        = 4
LUN path deletion requests                        = 0
Persistent Registration failures                  = 0
LUN path offlines                                 = 0
All LUN paths offlines                            = 1
```

```
LUN path back online                                = 0
Capacity increases                                  = 0
Capacity reductions                                 = 0
Block size changes                                  = 0
IO failures due to misalignment or boundary       = 0
Unexpected media changes                            = 0
Last time  cleared                                  = N/A
```

Sample of estape driver specific statistics for a tape device

Note: The command is directed to the estape driver plug-in by specifying the '-d' option.

```
# scsimgr get_stat -d estape -D /dev/rtape/tape2_BEST

        Driver estape STATISTICS FOR LUN :/dev/rtape/tape2_BEST

Write medium errors                                 = 0
Read medium errors                                  = 0
Cleaning required errors                            = 0
Blocked rewind-on-close open attempts               = 0
Last time  cleared                                  = N/A
```

Statistics for lunpaths

Similar to LUNs, lunpaths statistics can be divided into 3 categories: generic statistics, I/O transfer statistics and class driver specific statistics. Class driver specific statistics depend on the class of the LUN and require a plug-in module to be displayed.

The following is a sample of lunpath statistics:

```
# scsimgr get_stat -C lunpath -I 27

        STATISTICS FOR LUN path : lunpath27

Generic Statistics:

Attempted class driver opens                      = 114
Successfull class driver opens                    = 114
Class driver closes                               = 114
Offline events                                    = 0
Online events                                     = 0
SPOC state: Active                                = 114
SPOC state Standby                                = 0
SPOC state: Failed                                = 0
SPOC state: Authentication Failure                = 0
SPOC state : Disabled                             = 0
Open failures due to suspended LUN path           = 0
LUN resets issued                                 = 0
Last time cleared                                 = N/A

I/O transfer Statistics:

Bytes read                                        = 53248433404
Bytes written                                     = 68912558080
Total I/Os processed                              = 8260955
I/O failures                                      = 0
Retried I/Os                                      = 32512
Retried I/O failures                              = 0
I/O failures due to invalid IO size               = 0
I/Os flushed                                      = 0
Check condition status                            = 1
Busy status                                       = 0
Queue full status                                 = 32511

Driver esdisk Statistics :

PR registrations                                   = 0
```

```
Activations                                    = 0
Disable requests                               = 2
Enable requests                                = 1
Last time  cleared                             = N/A
```

Sample of a target path statistics

Target path statistics can be divided into 3 categories:

- Generic statistics: They are commonly defined for all target paths independent of the SCSI transport. They are maintained by the SCSI services module.
- Interface driver (I/F) common statistics: They are commonly defined for all target paths independent of the SCSI transport. They are maintained by each interface driver.
- Interface driver (I/F) specific statistics: They are maintained by interface drivers and depend on the type of the SCSI transport. They require a plug-in module to be displayed. Currently no interface driver provides a plug-in module for scsimgr. Therefore scsimgr does not display interface driver specific statistics.

```
# scsimgr get_stat -H 0/4/1/0/4/0.0x50001fe150006e69

        SCSI STATISTICS FOR TARGET PATH : 0/4/1/0/4/0.0x50001fe150006e69

Generic Statistics:

CB_SCAN_ALL events received                    =
Target Probe events received                   = 647
Probe failures due to LUN 0 probe failures     = 0
Probe failures due to REPORT LUNS failures     = 0
LUN path probe failures                        = 0
Target path offline events from I/F driver     = 0
Target path online events from I/F driver      = 0
Port id change events from I/F driver          = 0
Target Warm Reset events                       = 0
Target Cold Reset events                       = 0
Target Warm reset failures                     = 0
Target Cold Reset failures                     = 0
Invalid port id changes                        = 0
Total I/Os processed                           = 183778806
Last time cleared                              = N/A

I/F Common Statistics:

Offline events                                  = 0
WARM/COLD target resets                         = 0
Time of last WARM/COLD target reset             = N/A
Bytes read                                      = 1139296631502
Bytes written                                   = 1691314886656
Outstanding I/Os                                = 0
```

Sample of a HBA controller statistics

Similar to target paths, SCSI HBA controller statistics can be divided in 3 categories: generic statistics, I/F common statistics and I/F driver specific statistics.

```
# scsimgr get_stat -H 0/4/1/0/4/0

        SCSI STATISTICS FOR CONTROLLER : 0/4/1/0/4/0

Generic Statistics:

Illegal events                                 = 0
Ctlr Probe events received                     = 645
BUS Reset events received                      = 0
BUS Reset event failures                       = 0
Target path probe failures                     = 0
Total I/Os processed                           = 735174372
Last time cleared                              = N/A
```

```
I/F Common Statistics:

Outstanding I/Os                                = 0
Bytes read                                      = 4670035451002
Bytes written                                   = 6719212338688
Last bus reset time                             = N/A
Target ports connected                          = 21
Bus resets attempted                            = 0
Offline state                                   = 0
Online state                                    = 1
```

Displaying information about lunpaths of a LUN

The user can display information about lunpaths of LUN by running the scsimgr command as shown below:

```
# scsimgr lun_map -D /dev/rdisk/disk20

        LUN PATH INFORMATION FOR LUN : /dev/rdisk/disk20

Total number of LUN paths     = 2
World Wide Identifier(WWID)    = 0x20000020371972eb

LUN path : lunpath5
Class                         = lunpath
Instance                      = 5
Hardware path                 = 0/3/1/0.0x21000020371972eb.0x0
SCSI transport protocol       = fibre_channel
State                         = UNOPEN
Last Open or Close state      = ACTIVE

LUN path : lunpath12
Class                         = lunpath
Instance                      = 12
Hardware path                 = 0/4/1/0.0x21000020371972eb.0x0
SCSI transport protocol       = fibre_channel
State                         = UNOPEN
Last Open or Close state      = ACTIVE
```

Displaying attributes

Most of the information on SCSI objects is managed via attributes, which can be queried individually and can be displayed in a scriptable format. This greatly simplifies the task of scripts, and other applications, which query information about SCSI objects. The following examples illustrate the flexibility provided by attributes for applications retrieving information on SCSI objects.

Notes:

- *Refer to the scsimgr (1M) manpage for information on how to obtain all attributes of a SCSI object.*
- *The same information can be accessed via web-based GUI such as fsweb or HP SMH (System Management Homepage).*

Displaying the LUN id of a lunpath

In normal output the 64 bit LUN id is decoded in the form of LUN number and the addressing method used.

```
# scsimgr  get_attr -C lunpath -I 27 -a lunid
```

```
        SCSI ATTRIBUTES FOR LUN PATH : lunpath27

name = lunid
current =0x400f000000000000 (LUN # 15, Flat Space Addressing)
default =
saved =
```

scriptable output

```
# scsimgr -p get_attr -C lunpath -I 27 -a lunid
0x400f000000000000
```

Displaying the port id of a target path

```
# scsimgr -v get_attr -H 0/4/1/0/4/0.0x50001fe150006e69 -a port_id

        SCSI ATTRIBUTES FOR TARGET PATH : 0/4/1/0/4/0.0x50001fe150006e69

name = port_id
current = 66048
default =
saved =
desc = target port identifier
```

Displaying hardware path, device file, WWID and serial number for all devices, in scriptable output

```
# scsimgr -p get_attr all_lun -a hw_path -a device_file -a wwid -a serial_number

64000/0xfa00/0x0:/dev/rdisk/disk15:0x0004cffffebbf737:3FD1M2SW
64000/0xfa00/0x1:/dev/rdisk/disk16:0x0004cffffebbe43e:3FD1LZ0S
64000/0xfa00/0x2:/dev/rdisk/disk17::
64000/0xfa00/0x3:/dev/rdisk/disk18:0x20000020371972ee:LS255190000010080N0V
64000/0xfa00/0x4:/dev/rdisk/disk19:0x20000020371972e3:LS271211000010080N1H
64000/0xfa00/0x5:/dev/rdisk/disk20:0x20000020371972eb:LS24603100001008K74M
64000/0xfa00/0x6:/dev/rdisk/disk21:0x200000203726a3af:LJS87900000029491GWF
64000/0xfa00/0x7:/dev/rdisk/disk22:0x20000020370fe8c8:LJ67387800002916H9A9
64000/0xfa00/0x8:/dev/rdisk/disk23:0x200000203726d3b9:LJT4848400002950H0R4
64000/0xfa00/0x9:/dev/pt/pt2::USSO08016323
64000/0xfa00/0xa:/dev/pt/pt3::USSO08016323
```

Displaying current values of a settable attribute for all devices on the system, in a scriptable format

A script, which needs current value of the *max_q_depth* tunable for all devices on the system, can issue the following scsimgr command. Note that the *device_file* attribute is also required to be able to associate devices and *max_q_depth* values.

```
# scsimgr -p get_attr all_lun -a device_file -a max_q_depth

/dev/rdisk/disk15:8
/dev/rdisk/disk16:8
/dev/rdisk/disk17:1
/dev/rdisk/disk18:8
/dev/rdisk/disk19:8
/dev/rdisk/disk20:16
/dev/rdisk/disk21:8
/dev/rdisk/disk22:8
/dev/rdisk/disk23:8
/dev/pt/pt2:1
/dev/pt/pt3:1
```

Notes: The following rules apply for the scriptable output of attributes:

- *Attribute values are displayed exactly in the order specified in the command line.*
- *By default current values are returned. If the script wants default or saved values, the script should specify the keyword 'default' and 'saved' respectively. Examples:*

    - *scsimgr -p get_attr -D /dev/rdisk/disk0 -a max_q_depth -a path_fail_secs default*
    - *scsimgr -p get_attr -D /dev/rdisk/disk0 -a max_q_depth -a path_fail_secs saved*
- *The field corresponding to an attribute will be empty if the value of the attribute is not available. Examples: (1) Invalid attribute; (2) Invalid value requested (for instance saved or default for a read-only attribute); (3) Value of the attribute not yet set (for instance querying the value of the alias attribute prior to setting it).*
- *If the values of some of the attributes are not available, scsimgr exits with an error code > 0 to indicate that an error has occurred. The corresponding fields will be empty in the output.*

Setting tunables

The HP-UX 11i v3 mass storage stack provides a rich set of attributes (tunables) that you can manage using the scsimgr command. Each attribute has a name and the following values types:

- current value: the value currently used for operation.
- saved value: the value saved in a persistent manner. If it is set this value is used to initialize the attribute at system boot.
- default value: the value used to initialize the attribute at system boot when there is no saved value, or upon user request to reset the attribute. The default value is usually hard-coded by the driver or is inherited from the current value at a higher level where the attribute is explicitly set.

Note: For read-only attributes (also called non settable attributes), only the current value is defined.

Attributes can be set at the following levels:

- Global: A value of an attribute set at this scope, affects the default behavior for the SCSI stack, a class driver or an interface driver. For instance, the default I/O timeout can be set to 60 seconds for all SCSI devices.
- Settable attribute scope: A settable attribute scope definition can include the following criteria for devices bound to a certain driver: device type, vendor identifier, product identifier, and a specific product revision. A value of an attribute set at this level, affects the default behavior for the set of devices bound to the specified driver, and meeting the criteria specified by the scope and only these devices. For instance, the default I/O timeout can be set to 60 seconds for disk devices from HP managed by the esdisk driver, and to 120 seconds for all tape devices managed by the estape driver.
- Specific instance of a SCSI object: A value of an attribute set at this scope overwrites the default behavior for a specific instance of a SCSI object. For instance I/O timeout can be set to 50 seconds for the disk device with instance number 0.

Not all attributes can be set at all levels. Some attributes can be set at all levels, others can only be set at a global level, a settable attribute scope level, or a SCSI object instance level.

The current value of an attribute for a SCSI object instance is determined according to the following algorithm in highest order of precedence:

1. If the attribute is explicitly set for the specific SCSI object instance, this value is used. The attribute is explicitly set when either one or all of the following is true:
    a. The user sets the saved and/or current values using scsimgr.

b. The driver has hard-coded a default value for the attribute for the object instance.

2. If the SCSI object matches a settable attribute scope and the attribute is set at this level, this value is used.
3. If the attribute is set at the global level, this value is used.

The default value of an attribute is determined according to the following algorithm in the highest order of precedence:

1. If the driver hard-codes a default value for this object instance, this value is used.
2. If the object matches a settable attribute scope and the attribute is set at this level, the current value at this level is used.
3. If the attribute is set at a global level, the current value at this level is used.

Note: the saved value of an attribute is never inherited from higher level. Either it is defined or is not.

Rules governing the update of attribute current values

The following rules govern the update of current values of attributes:

1. If the current value of an attribute is changed at a higher level for instance at a global level or a settable attribute scope, the new value will be taken into account for a specific SCSI object instance only at certain initialization points. For LUNs, the attributes are reloaded when the device is opened or re-opened.
2. Setting an attribute for a specific SCSI object instance affects immediately the current value of the attribute.

The tunable architecture provides a lot of flexibility when fine tuning the mass storage stack. The following examples are meant to help understand how the tunable architecture works, how to interpret attribute values at different levels, and how to use it to optimally solve interoperability problems.

Inheritance of attribute values

This example illustrates how attribute values are inherited from higher level and when change of an attribute value at a higher level take effect for a particular SCSI object.

leg_mpath_enable is an attribute, which controls whether multi-pathing is enabled or disabled for legacy DSFs of a LUN. This tunable can be set at a global level or for a particular device. By default the SCSI stack explicitly set this attribute to *true* at the global level to enable by default multi-pathing for legacy DSFs of all LUNs.

state is a read-only attribute, which provides the state of the device maintained by the SCSI stack. It is displayed here to show when the device is closed (state=UNOPEN), or when the device is opened (state=ONLINE).

In this example disk100 and disk101 are two disk devices on the same system. At system boot disk100 and disk101 inherit the value of *leg_mpath_enable* from the global level as shown in the following scsimgr output.

```
# scsimgr get_attr -a leg_mpath_enable

        SCSI GLOBAL ATTRIBUTES:

name = leg_mpath_enable
current = true
```

```
default = true
saved =

# scsimgr get_attr -D /dev/rdisk/disk100 -a state -a leg_mpath_enable

        SCSI ATTRIBUTES FOR LUN : /dev/rdisk/disk100

name = state
current = UNOPEN
default =
saved =

name = leg_mpath_enable
current = true
default = true
saved =

# scsimgr get_attr -D /dev/rdisk/disk101 -a state -a leg_mpath_enable

        SCSI ATTRIBUTES FOR LUN : /dev/rdisk/disk101

name = state
current = UNOPEN
default =
saved =

name = leg_mpath_enable
current = true
default = true
saved =
```

You can set *leg_mpath_enable* to be persistently false for disk100. This action overrides the default behavior (the setting at the global level). The LUN disk100 no longer inherits the current value of *leg_mpath_enable* from the global level, but its default value is still inherited from the current value of *leg_mpath_enable* at the global level.
Setting the attribute persistently updates the current and saved values. Hence, the saved value of *leg_mpath_enable* is now defined for disk100. If the system is rebooted, the current value of *leg_mpath_enable* for disk100 will be set to its saved value.

```
# scsimgr save_attr -D /dev/rdisk/disk100 -a leg_mpath_enable=false
Value of attribute leg_mpath_enable saved successfully

# scsimgr get_attr -D /dev/rdisk/disk100 -a leg_mpath_enable

        SCSI ATTRIBUTES FOR LUN : /dev/rdisk/disk100

name = leg_mpath_enable
current = false
default = true
saved = false
```

Note: it is possible to set the attribute in a non persistent manner with the 'scsimgr set_attr ' command. This updates the current value, but if the system is rebooted, this value will be lost.

Data transfer is initiated on disk100 and disk101. This causes both devices to be opened. The attributes values are loaded for disk100 and disk101.

Now if you change *leg_mpath_enable* at the global level after the data transfer is initiated, this does not affect current values of *leg_mpath_enable* for disk100 and disk101. It does not change *leg_mpath_enable* for disk100 because the attribute is explicitly set for disk100. It does not change *leg_mpath_enable* for disk101 because the attribute values are loaded only when the device is opened (so the change won't take effect until next time that the device is opened).

Note the following:

- The value of the *state* attribute of the devices is now ONLINE indicating that the devices are opened.
- The SCSI stack does not explicitly define a default value for the *leg_mapth_enable* attribute, for specific device instances. The default value is inherited from the current value of this attribute at the global level.

```
# dd if=/dev/rdisk/disk100 of=/dev/null count=10000000 &
[1]     12618
# dd if=/dev/rdisk/disk101 of=/dev/null count=10000000 &
[2]     12678

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

# scsimgr get_attr -a leg_mpath_enable

        SCSI GLOBAL ATTRIBUTES:

name = leg_mpath_enable
current = false
default = true
saved =

# scsimgr get_attr -D /dev/rdisk/disk100 -a state -a leg_mpath_enable

        SCSI ATTRIBUTES FOR LUN : /dev/rdisk/disk100

name = state
current = ONLINE
default =
saved =

name = leg_mpath_enable
current = false
default = false
saved = false

# scsimgr get_attr -D /dev/rdisk/disk101 -a state -a leg_mpath_enable

        SCSI ATTRIBUTES FOR LUN : /dev/rdisk/disk101

name = state
current = ONLINE
default =
saved =

name = leg_mpath_enable
current = true
default = false
saved =
```

Terminating the dd commands on disk100 and disk101 causes these devices to be closed. The values of the *leg_mpath_enable* attribute for disk101 are not affected because attribute values are not re-loaded upon device close.

```
# jobs
[2] +  Running                 dd if=/dev/rdisk/disk101 of=/dev/null count=10000000 &
[1] -  Running                 dd if=/dev/rdisk/disk100 of=/dev/null count=10000000 &
# kill %1 %2
# jobs
[2] + Terminated               dd if=/dev/rdisk/disk101 of=/dev/null count=10000000 &
[1] + Terminated               dd if=/dev/rdisk/disk100 of=/dev/null count=10000000 &

# scsimgr get_attr -a leg_mpath_enable

        SCSI GLOBAL ATTRIBUTES:
```

```
name = leg_mpath_enable
current = false
default = true
saved =

# scsimgr get_attr -D /dev/rdisk/disk100 -a state -a leg_mpath_enable
        SCSI ATTRIBUTES FOR LUN : /dev/rdisk/disk100

name = state
current = UNOPEN
default =
saved =

name = leg_mpath_enable
current = false
default = false
saved = false

# scsimgr get_attr -D /dev/rdisk/disk101 -a state -a leg_mpath_enable

        SCSI ATTRIBUTES FOR LUN : /dev/rdisk/disk101

name = state
current = UNOPEN
default =
saved =

name = leg_mpath_enable
current = true
default = false
saved =
```

Starting new data transfers on disk100 and disk101 causes the attributes to be re-loaded. In this case, disk101 inherits the default behavior set at the global level while disk100 behavior is unaffected as the attribute is explicitly set for disk100.

```
# dd if=/dev/rdisk/disk100 of=/dev/null count=10000000 &
[1]     12932
# dd if=/dev/rdisk/disk101 of=/dev/null count=10000000 &
[2]     12934
# scsimgr get_attr -D /dev/rdisk/disk100 -a state -a leg_mpath_enable

        SCSI ATTRIBUTES FOR LUN : /dev/rdisk/disk100

name = state
current = ONLINE
default =
saved =

name = leg_mpath_enable
current = false
default = false
saved = false

# scsimgr get_attr -D /dev/rdisk/disk101 -a state -a leg_mpath_enable

        SCSI ATTRIBUTES FOR LUN : /dev/rdisk/disk101

name = state
current = ONLINE
default =
saved =

name = leg_mpath_enable
current = false
default = false
saved =
```

The leg_mpath_enable attribute is set to *true* for disk100 while data transfers are still going on (in other words disk100 is still opened). Note that the new attribute value is taken into account immediately.

```
# scsimgr set_attr -D /dev/rdisk/disk100 -a leg_mpath_enable=true
Value of attribute leg_mpath_enable set successfully

# scsimgr get_attr -D /dev/rdisk/disk100 -a state -a leg_mpath_enable

        SCSI ATTRIBUTES FOR LUN : /dev/rdisk/disk100

name = state
current = ONLINE
default =
saved =

name = leg_mpath_enable
current = true
default = false
saved = false
```

Fine tuning the SCSI stack to interoperate with devices with different operational requirements

This use case shows how the tunable architecture can be used to fine tune interoperability with devices.

The SCSI stack defines the following tunables for disk devices bound to the esdisk class driver:

- *path_fail_secs*: Timeout in seconds before declaring a lunpath offline when the device no longer responds to I/O requests sent through this I/O path. The default value is 120 seconds.
- *max_q_depth*: Maximum number of concurrent I/O requests that can be sent to the device on a given I/O path. The default value is 8.
- *load_bal_policy*: Policy used to distribute I/O requests between lunpaths of a device. The default policy is *round_robin* (I/Os are distributed between I/O paths to the device in a round robin manner).

These tunables need to be fine tuned to meet the following requirements for an optimal operation:

- All HP disk devices with the product identifier (pid) ST3360LCB, require *path_fail_secs* to be set at least to 160 seconds. In addition these disk arrays exhibit some degradation in performance if I/O requests are distributed in a round robin manner. Optimal performances are obtained with the *least_cmd_load* policy, which distributes I/O to the I/O path with the least load.
- All HP disk devices with pid ST39103FC2, can handle only up to 4 concurrent I/O requests.
- disk0 is a JBOD connected to the system through a very slow and unreliable link. It requires *path_fail_secs* to be set at least to 240 seconds.

Note: This example of different operational requirements is theoretical. It is just intended to illustrate the capabilities of the stack.

The values of these attributes at the esdisk driver global level, determine the default behavior for all disk devices bound to the esdisk driver.

```
# scsimgr get_attr -N /escsi/esdisk -a path_fail_secs -a load_bal_policy -a max_q_depth

        SCSI ATTRIBUTES FOR DDR ENTRY : /escsi/esdisk
```

```
name = path_fail_secs
current = 120
default = 120
saved =

name = load_bal_policy
current = round_robin
default = round_robin
saved =

name = max_q_depth
current = 8
default = 8
saved =
```

You can perform the following to meet the above requirements:

1 Add 2 settable attribute scopes covering respectively HP disk devices with pid: ST3360LCB and ST39103FC2.

```
# scsimgr -f ddr_add -N "/escsi/esdisk/0x0/HP      /ST3360LCB       "
scsimgr: settable attribute scope '/escsi/esdisk/0x0/HP      /ST3360LCB       '
added successfully

# scsimgr ddr_add -N "/escsi/esdisk/0x0/HP      /ST39103FC2      "
scsimgr: settable attribute scope '/escsi/esdisk/0x0/HP      /ST39103FC2      '
added successfully
```

2 Set *path_fail_secs* to 160 seconds, *load_bal_policy* to *least_cmd_load* for HP disk devices with pid: ST3360LCB.

```
# scsimgr save_attr -N "/escsi/esdisk/0x0/HP      /ST3360LCB       " -a
path_fail_secss=160 -a load_bal_policy=least_cmd_load
Value of attribute path_fail_secs saved successfully
Value of attribute load_bal_policy saved successfully

# scsimgr get_attr -N "/escsi/esdisk/0x0/HP      /ST3360LCB       " -a
path_fail_secss -a load_bal_policy

        SCSI ATTRIBUTES FOR SETTABLE ATTRIBUTE SCOPE : /escsi/esdisk/0x0/HP
/ST39103FC2

name = path_fail_secs
current = 160
default = 120
saved = 160

name = load_bal_policy
current = least_cmd_load
default = round_robin
saved = least_cmd_load
```

3 Set *max_q_depth* to 4 for HP disk devices with pid: ST39103FC2.

```
scsimgr save_attr -N "/escsi/esdisk/0x0/HP      /ST39103FC2      " -a
max_q_depth=4
Value of attribute max_q_depth saved successfully

scsimgr get_attr -N "/escsi/esdisk/0x0/HP      /ST39103FC2      " -a max_q_depth

        SCSI ATTRIBUTES FOR SETTABLE ATTRIBUTE SCOPE : /escsi/esdisk/0x0/HP
/ST39103FC2

name = max_q_depth
```

```
current = 4
default = 8
saved = 4
```

4 Set *path_fail_secs* to 240 seconds for disk0.

```
# scsimgr save_attr -D /dev/rdisk/disk0 -a path_fail_secs=240
Value of attribute path_fail_secs saved successfully

# scsimgr get_attr -D /dev/rdisk/disk0 -a path_fail_secs
name = path_fail_secs
current = 240
default = 120
saved = 240
```

After fine tuning these tunables, examine their values for 3 disk devices: disk0, disk1 and disk2. The disks are all manufactured by HP and have the following product identifiers respectively: OPEN-V, ST3360LCB and ST39103FC2.

The scsimgr command output below show the values of the tunables before the disks are opened or re-opened - in other words before the attributes are re-loaded.

Note the following:

- For all disks, the current values of the attributes remain unchanged. This is due to the fact that the attributes are not yet refreshed and the current values still reflect the last used values.
- disk0:
  - The *path_fail_secs* attribute, is explicitly set for disk0. This is reflected in the current value of this attribute, which correspond to what has been set through scsimgr.
  - disk0 does not match any of the 2 settable attribute scopes added. *load_bal_policy* and *max_q_depth* attributes inherit their default values from the global level.
- disk1:
  - Default values of the *path_fail_secs* and *load_bal_policy* attributes are inherited from the settable attribute scope covering devices with pid: ST3360LCB.
  - The *max_q_depth* is not explicitly set for the settable attribute scope covering disk1. So it inherits its default value from the global level.
- disk2:
  - The default value of the *max_q_depth* attribute is inherited from the settable attribute scope covering devices with pid: ST39103FC2.
  - *path_fail_secs* and *load_bal_policy* are not explicitly set at the settable attribute scope covering disk2. They inherit their default values from the global level.

```
# scsimgr get_attr -D /dev/rdisk/disk0 -a path_fail_secs -a max_q_depth -a load_bal_policy

   SCSI ATTRIBUTES FOR LUN : /dev/rdisk/disk0

name = path_fail_secs
current = 240
default = 120
saved = 240

name = max_q_depth
current = 8
default = 8
saved =

name = load_bal_policy
current = round_robin
default = round_robin
```

```
saved =


# scsimgr get_attr -D /dev/rdisk/disk1 -a max_q_depth -a load_bal_policy

   SCSI ATTRIBUTES FOR LUN : /dev/rdisk/disk1

name = path_fail_secs
current = 120
default = 160
saved =

name = max_q_depth
current = 8
default = 8
saved =

name = load_bal_policy
current = round_robin
default = least_cmd_load
saved =


# scsimgr get_attr -D /dev/rdisk/disk2 -a max_q_depth -a load_bal_policy

   SCSI ATTRIBUTES FOR LUN : /dev/rdisk/disk2

name = path_fail_secs
current = 120
default = 120
saved =

name = max_q_depth
current = 8
default = 4
saved =

name = load_bal_policy
current = round_robin
default = round_robin
saved =
```

The scsimgr command output below shows the values of the tunables after the disks are opened or re-opened - In other words after the attributes are re-loaded.

Note the following:

- Now current values of the attributes are set based on the inheritance rules. Most importantly these values correspond to the operational requirements outline earlier.
- disk0:
  - The *path_fail_secs* attribute, is explicitly set for disk0. So the current value is unchanged. It is not inherited from a higher level.
  - disk0 does not match any of the 2 settable attribute scopes added. The default and current values of *load_bal_policy* and *max_q_depth* attributes are inherited from the global level.
- disk1:
  - Default and current values of the *path_fail_secs* and *load_bal_policy* attributes are inherited from the current value of corresponding attributes set at the settable attribute scope covering devices with pid: ST3360LCB.
  - The *max_q_depth* attribute is not explicitly set at the attribute scope covering disk1. It inherits its default and current values from the global level.
- disk2:
  - The default and current values of the *max_q_depth* attribute are inherited from the current value of this attribute at the settable attribute scope covering devices with pid: ST39103FC2.

- The *path_fail_secs* and *load_bal_policy* attributes are not explicitly set at a settable attribute scope covering disk2. They inherit their current and default values from the global level.

```
# scsimgr get_attr -D /dev/rdisk/disk0 -a path_fail_secs -a max_q_depth -a load_bal_policy

   SCSI ATTRIBUTES FOR LUN : /dev/rdisk/disk0

name = path_fail_secs
current = 240
default = 120
saved = 240

name = max_q_depth
current = 8
default = 8
saved =

name = load_bal_policy
current = round_robin
default = round_robin
saved =



# scsimgr get_attr -D /dev/rdisk/disk1 -a max_q_depth -a load_bal_policy

   SCSI ATTRIBUTES FOR LUN : /dev/rdisk/disk1

name = path_fail_secs
current = 160
default = 160
saved =

name = max_q_depth
current = 8
default = 8
saved =

name = load_bal_policy
current = least_cmd_load
default = least_cmd_load
saved =


# scsimgr get_attr -D /dev/rdisk/disk2 -a max_q_depth -a load_bal_policy

   SCSI ATTRIBUTES FOR LUN : /dev/rdisk/disk2

name = path_fail_secs
current = 120
default = 120
saved =

name = max_q_depth
current = 4
default = 4
saved =

name = load_bal_policy
current = round_robin
default = round_robin
saved =
```

Assigning user-friendly device identifiers and aliases for ease of inventory

Device identifiers and aliases can be assigned in a way to simplify device inventory for instance by integrating the usage and the asset inventory number in the identifier or alias. User-friendly device identifiers can only be set for devices supporting the *SET DEVICE IDENTIFIER and REPORT DEVICE IDENTIFIER* SCSI commands. In this case the identifier resides on non volatile memory on the device and can be queried by all systems accessing the device.
An alias is a settable attribute that can be assigned to any device. It is stored locally on the system registry. Therefore it must be set on each system accessing the device.

To assign the following user-friendly device identifier to disk device disk0: “Engineering - XPD0890-0”

```
# scsimgr -f set_devid -D /dev/rdisk/disk0 "Engineering - XPD0890-1"
scsimgr: Device Identifier successfully set
```

To display the device identifier assigned to disk device disk0:

```
# scsimgr get_devid -D /dev/rdisk/disk0
Device Identifier for /dev/rdisk/disk0 = Engineering - XPD0890-1
```

To display device identifiers for all devices in a scriptable output format:

```
# scsimgr -p get_devid all_lun
/dev/rdisk/disk0: Engineering - XPD0890-1
/dev/rdisk/disk1:
/dev/rdisk/disk2: PQS - XPD0893-2
/dev/rdisk/disk0: Engineering - XPD0890-2
```

Note: If the device does not support the SET DEVICE IDENTIFIER and REPORT DEVICE IDENTIFIER SCSI commands, and you attempt to set or get the device identifier, scsimgr will display an error message as shown in the output below

```
# scsimgr set_devid -D /dev/rdisk/disk20 "Engineering department disk20"
Do you really want to set device id? (y/n)? y
scsimgr: ERROR: LUN /dev/rdisk/disk20 does not support Device Identifier

# scsimgr get_devid -D /dev/rdisk/disk20
scsimgr: ERROR: LUN /dev/rdisk/disk20 does not support Device Identifier
```

To assign the alias “Engineering - XPD08934-1” to disk device disk1

```
# scsimgr save_attr -D /dev/rdisk/disk1 -a alias="Engineering - XPD08934-1"
Value of attribute alias saved successfully
```

To view the alias assigned to disk1

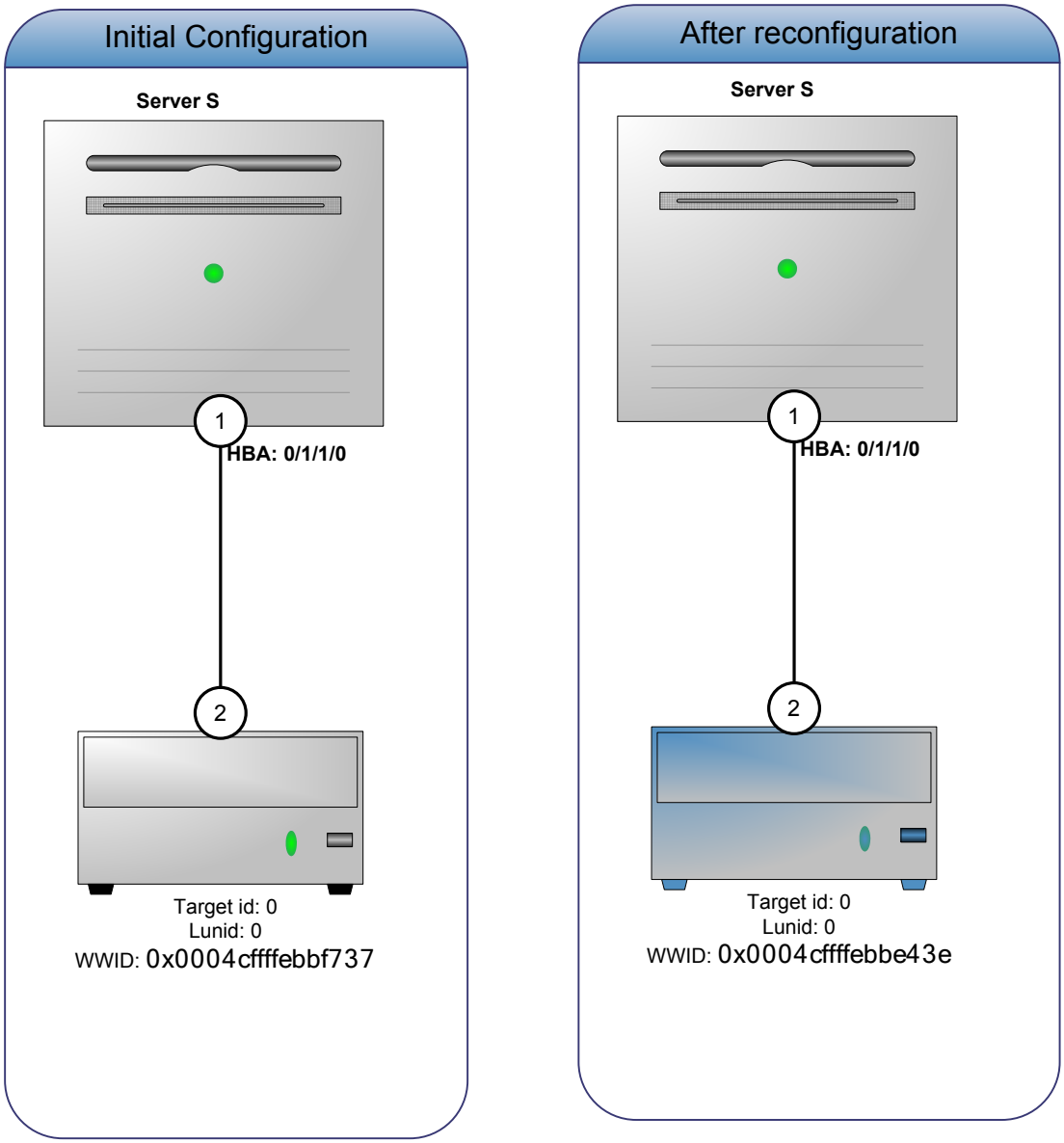
```
# scsimgr get_attr -D /dev/rdisk/disk20 -a alias

        SCSI ATTRIBUTES FOR LUN: /dev/rdisk/disk1

name = alias
current =  Engineering - XPD08934-1
default =
saved =  Engineering - XPD08934-1
```

Validating online replacement of devices and SAN reconfigurations

When a LUN with a different WWID is discovered through an existing lunpath, to prevent data corruption, the SCSI stack puts the lunpath Last Open or Close (SPOC) state in "authentication failure". In this state the lunpath cannot be used for I/O transfer. The lunpath remains in this state until the system administrator validates the change.

Note: Various SAN reconfigurations may result in lunpath "authentication failure". When that happens, the mass storage stack logs a message in /var/adm/syslog/syslog.log and on the console, to alert the user. The user should rely on these messages and should apply the action recommended. Usually the action consists of running the 'scsimgr replace_wwid' command.

The following example illustrates an online replacement of a device leading to a lunpath "authentication failure". It shows how to recognize the situation and validate the change by running the appropriate scsimgr commands.



The above diagram shows a disk connected to Server S via parallel SCSI and through the HBA with hardware path: 0/1/1/0. This disk is assigned the instance 15 by the system. Its target identifier is: 0x0, its LUN identifier is: 0x0, and its WWID is: 0x0004cffffebbf737. The disk has one lunpath: 0/1/1/0.0x0.0x0. The following output of the ioscan command shows information about disk15

```
# ioscan -kfnNCdisk
Class     I  H/W Path  Driver S/W State   H/W Type     Description
===================================================================
disk      13  64000/0xfa00/0x2  esdisk CLAIMED     DEVICE       TEAC    DV-28E-B
                      /dev/disk/disk17   /dev/rdisk/disk17
disk      14  64000/0xfa00/0x3  esdisk CLAIMED     DEVICE       SEAGATE ST39103FC
                      /dev/disk/disk18   /dev/rdisk/disk18
disk      15  64000/0xfa00/0x0  esdisk CLAIMED     DEVICE       HP 36.4GST336706LC
                      /dev/disk/disk15      /dev/rdisk/disk15
```

disk15 hardware path, lunpaths and DSFs:

```
# ioscan -m lun /dev/rdisk/disk15
Class     I  Lun H/W Path  Driver  S/W State   H/W Type     Health  Description
===================================================================
disk      15  64000/0xfa00/0x0   esdisk  CLAIMED     DEVICE       online  HP 36.4GST336706LC
              0/1/1/0.0x0.0x0
                      /dev/disk/disk15      /dev/rdisk/disk15
```

disk15 WWID:

```
# scsimgr -p get_attr -D /dev/rdisk/disk15 -a wwid
0x0004cffffebbf737
```

The system administrator replaces disk15 with a back-up disk containing the same data, but with a different WWID (0x0004cffffebbe43e). The target id and LUN id remain the same.
As the new disk is accessed through the same HBA and the target identifier and LUN identifier remain the same, its lunpath hardware path remains: 0/1/1/0.0x0.0x0.

The system administrator re-initializes device probing by running the ioscan command.

The SCSI stack discovers a different disk at the lunpath with the hardware path 0/1/1/0.0x0.0x0. It sets this lunpath's SPOC state to "authentication failure" to prevent data corruption in case the change was not intentional. The SCSI stack also logs the following message to alert the system administrator:

```
Evpd inquiry page 83h/80h failed or the current page 83h/80h data do not match the previous
known page 83h/80h data on LUN id 0x0 probed beneath the target path (class = tgtpath,
instance = 3). The lun path is (class = lunpath, instance = 0). Run 'scsimgr replace_wwid'
command to validate the change.
```

The system administrator can view the SPOC state of the lunpath in the log message by running the following command:

```
# scsimgr get_info -C lunpath -I 0

        STATUS INFORMATION FOR LUN PATH : lunpath0

Generic Status Information

SCSI services internal state                 = ACTIVE
Open close state                             = AUTH_FAILED
Protocol                                     = parallel_scsi
EVPD page 0x83 description code              = 1
EVPD page 0x83 description association       = 0
EVPD page 0x83 description type              = 2
World Wide Identifier (WWID)                 = 0x0004cffffebbe43e
Outstanding I/Os                             = 0
Maximum I/O timeout in seconds               = 30
```

```
Maximum I/O size allowed                    = 1380352
Maximum number of active I/Os allowed       = 8
Current active I/Os                         = 0
Maximum queue depth                         = 8
Queue full delay count                      = 0
```

To validate disk15 replacement, run the '*scsimgr replace_wwid'* command. This command has three options:

- Validate the replacement for a specific lunpath
- Validate the replacement for all paths of disk15
- Validate the replacement for all lunpaths beneath the target paths through, which disk15 was discovered

Validating disk replacement on a single lunpath

To validate the replacement for the lunpath run the following command:

```
# scsimgr -f replace_wwid -C lunpath -I 0
Binding of LUN path 0/1/1/0.0x1.0x0 with new LUN validated successfully
```

The advantage of this method is that it directly specifies the lunpath class and instance provided in the logging message. This is also sufficient when the replaced disk only has one lunpath, or if only a few lunpaths are affected by the SAN reconfiguration.

Validating disk replacement for all lunpaths of a replaced LUN

If disk15 had multiple lunpaths, it would be cumbersome to repeat the *'scsimgr replace_wwid'* command for each lunpath. It is more convenient to validate the replacement for all lunpaths of the disk at once by specifying the replaced LUN as shown in the command below.

```
# scsimgr -f replace_wwid -D /dev/rdisk/disk15
scsimgr: Successfully validated binding of LUN paths with new LUN.
```

Note: To be able to validate the replacement by specifying the replaced LUN, either write down its identifier (device file name, hardware path, or class and instance number of the replace LUN) before proceeding with the replacement, or determine the replace LUN hardware path by running the following commands after the authentication failure is logged:

1. *Determine the hardware path of the lunpath:*
   ```
   # scsimgr -p get_attr -C lunpath -I 0 -a hw_path
   0/1/1/0.0x0.0x0
   ```
2. *Determine the hardware path of the replaced LUN*
   ```
   # ioscan -m hwpath -H 0/1/1/0.0x0.0x0
   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
   ```
3. *Validate the replacement using the hardware path of the replaced LUN*
   ```
   # scsimgr -f replace_wwid -H 64000/0xfa00/0x0
   scsimgr: Successfully validated binding of LUN paths with new LUN.
   ```

Once the replacement is validated, data transfer to the new disk can be performed through the lunpaths. Since the new disk has a different WWID, the system assigns it a new instance and creates new DSFs. To avoid reconfiguring applications and upper layer modules (Volume Managers, file systems, etc.) using disk15, assign disk15 device special files and instance to the new disk by specifying the *'dsf'* keyword at the command line as shown below.

```
# scsimgr -f replace_wwid -D /dev/rdisk/disk15 dsf
scsimgr: Successfully validated binding of LUN paths with new LUN.
Device special file of replaced LUN successfully re-assigned to replacing LUN
```

Notes:

- *scsimgr calls the io_redirect_dsf script to re-assign the device files of the replaced disk to the new disk. For more information about this script, see the io_redirect_dsf (1M) man page.*
- *If the device is used by the Logical Volume Manager (LVM), you must use the 'dsf' keyword or use the io_redirect_dsf script during the LVM online disk replacement procedure.*

Validating disk replacement for all lunpaths under a target path

When a LUN is replaced or the SAN is reconfigured, several lunpaths under a given target can be put in "authentication failure" state. This is true when for instance a reconfiguration of a disk array result to different WWIDs being assigned to all LUNs of the disk array seen by the system. The disk array can have thousand of LUNs, but only a few target ports. All lunpaths of the affected LUNs will be put in "Authentication failure". In this case it is more convenient for the user to validate the change by specifying the target paths. All lunpaths beneath each target path specified will be validated. Since the number of target paths is very limited compared to the number of lunpaths or disks affected, the system administrator will have to run *'scsimgr replace_wwid'* only a few times. The system administrator can use target information provided in the logging as shown in the example below.

```
# scsimgr -f replace_wwid -C tgtpath -I 3
scsimgr: Successfully validated binding of LUN paths with new LUN.
```

Note: If verbose output is requested with the -v option, scsimgr displays each lunpath for which the replacement is validated. The following example shows a verbose output for target path with several lunpaths beneath it.

```
# scsimgr -v -f replace_wwid -C tgtpath -I 10
Binding of LUN path 0/4/1/0/4/0.0x50001fe150006e69.0x0 with new LUN validated successfully
Binding of LUN path 0/4/1/0/4/0.0x50001fe150006e69.0x4001000000000000 with new LUN validated
successfully
Binding of LUN path 0/4/1/0/4/0.0x50001fe150006e69.0x4002000000000000 with new LUN validated
successfully
Binding of LUN path 0/4/1/0/4/0.0x50001fe150006e69.0x4003000000000000 with new LUN validated
successfully
Binding of LUN path 0/4/1/0/4/0.0x50001fe150006e69.0x4004000000000000 with new LUN validated
successfully
```

Note: If the target path is specified to validate the change, scsimgr cannot re-assign the device special files of the replaced disks to the new disks. You should either reconfigure the applications, which were using the replaced disks, or invoke the io_redirect_dsf script for each replaced disk.

Validating change of the binding of legacy DSFs to LUNs

In HP-UX 11i v3 legacy DSFs and persistent DSFs coexist. A legacy DSF represents a path to the LUN. By default the native multi-pathing is enabled on legacy DSFs. It means that even if the lunpath corresponding to a legacy DSF becomes unavailable, applications using it can continue to transfer data to the device through other available lunpaths. This behavior affects how the SCSI stack deal with reconfigurations impacting legacy DSFs.

In releases prior to HP-UX 11i v3, if a new LUN is discovered at a specific lunpath, the corresponding legacy DSF is automatically associated with the new LUN. In HP-UX 11i v3 the SCSI stack maintains internally the binding between a legacy DSF and a LUN. If a new LUN is discovered at a specific

path, the SCSI stack keeps the existing binding between the corresponding legacy DSF and the LUN previously discovered at this path until the change is explicitly validated or the system is rebooted. This allows applications using the affected legacy DSF to continue to perform I/O transfers to the device previously seen at this path, through other available paths.

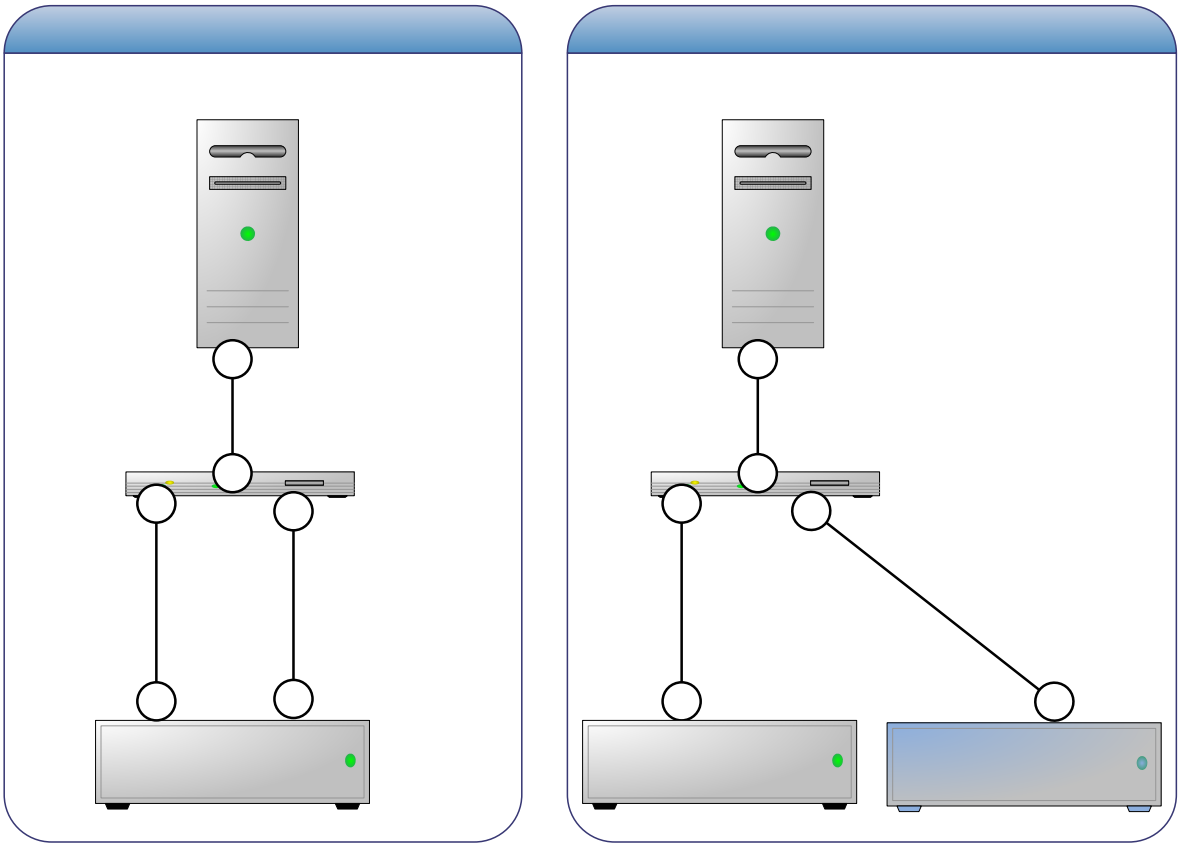
Notes:

- *Various SAN reconfigurations can impact legacy DSF. In cases where the system administrator has to explicitly confirm the change of the binding between a legacy DSF and a LUN, the SCSI stack logs a message and the recommended action in /var/adm/syslog/syslog.log and on the console. The system administrator should rely on these messages and perform the action indicated to validate the change.*

- *Even in case a message is logged, the system administrator may choose not to confirm the change for example to allow applications using the legacy DSF to continue I/O transfers through other available paths. However, if the system is rebooted some time later, that legacy DSF will be bound to the new LUN. This could impact applications or upper layers using this legacy DSF as they would now access a different LUN. For instance the user may have to reconfigure a volume group to use a different legacy DSF corresponding to another path to the original LUN.*

Example of a SAN reconfiguration affecting legacy DSFs

The following is an example of a SAN reconfiguration, whose impact on legacy DSFs requires explicit confirmation of the change of the legacy DSF binding.



In the diagram above, Server SV1 is initially connected to the disk array DA1 through: (1) the HBA with the hardware path: 0/4/1/0/4/0; (2) the switch SW1 with the domain address: 1; (3) two target ports of the disk array DA1 with respectively the following information:

- Target port '4':
  - Area address: 2
  - Port address: 0
  - WWN: 0x50001fe150006e69
- Target port '6' :
  - Area address: 4
  - Port address: 0
  - WWN: 0x50001fe150006e68

Based on this information, you can determine the following:

- The hardware paths of the 2 target ports of DSA1 are: 0/4/1/0/4/0.0x50001fe150006e69 and 0/4/1/0/4/0.0x50001fe150006e68.
- The corresponding legacy target hardware paths will start with: 0/4/1/0/4/0.1.2 and 0/4/1/0/4/0.1.4.

The following output of the ioscan command shows the target paths of disk array DA1:

```
# ioscan -kfNCtgtpath
Class     I  H/W Path       Driver S/W State   H/W Type     Description
===================================================================
tgtpath   5  0/4/1/0/4/0.0x50001fe150006e68  estp   CLAIMED     TGT_PATH     fibre_channel
target served by fcd driver
tgtpath   3  0/4/1/0/4/0.0x50001fe150006e69  estp   CLAIMED     TGT_PATH     fibre_channel
target served by fcd driver
```

The following output of the ioscan command shows the disks of DA1:

```
# ioscan -kfnNCdisk
Class     I  H/W Path  Driver S/W State   H/W Type     Description
===================================================================
disk     86  64000/0xfa00/0x6   esdisk   CLAIMED     DEVICE       COMPAQ  HSV111 (C)COMPAQ
                     /dev/disk/disk86   /dev/rdisk/disk86
disk     87  64000/0xfa00/0x7   esdisk   CLAIMED     DEVICE       COMPAQ  HSV111 (C)COMPAQ
                     /dev/disk/disk87   /dev/rdisk/disk87
disk     88  64000/0xfa00/0x8   esdisk   CLAIMED     DEVICE       COMPAQ  HSV111 (C)COMPAQ
                     /dev/disk/disk88   /dev/rdisk/disk88
disk     89  64000/0xfa00/0x9   esdisk   CLAIMED     DEVICE       COMPAQ  HSV111 (C)COMPAQ
                     /dev/disk/disk89   /dev/rdisk/disk89
disk     90  64000/0xfa00/0xa   esdisk   CLAIMED     DEVICE       COMPAQ  HSV111 (C)COMPAQ
                     /dev/disk/disk90   /dev/rdisk/disk90
```

Each disk of DA1 has two lunpaths. The following output of the ioscan command, shows the hardware path of each disk, its lunpaths' hardware paths, and the corresponding legacy hardware paths.

```
# ioscan -m hwpath
Lun H/W Path      Lunpath H/W Path                 Legacy H/W Path
====================================================================
64000/0xfa00/0x6
                  0/4/1/0/4/0.0x50001fe150006e69.0x4001000000000000
0/4/1/0/4/0.1.2.0.0.0.1
                  0/4/1/0/4/0.0x50001fe150006e68.0x4001000000000000
0/4/1/0/4/0.1.4.0.0.0.1
64000/0xfa00/0x7
```

```
                0/4/1/0/4/0.0x50001fe150006e69.0x4002000000000000
0/4/1/0/4/0.1.2.0.0.0.2
                0/4/1/0/4/0.0x50001fe150006e68.0x4002000000000000
0/4/1/0/4/0.1.4.0.0.0.2
64000/0xfa00/0x8
                0/4/1/0/4/0.0x50001fe150006e69.0x4003000000000000
0/4/1/0/4/0.1.2.0.0.0.3
                0/4/1/0/4/0.0x50001fe150006e68.0x4003000000000000
0/4/1/0/4/0.1.4.0.0.0.3
64000/0xfa00/0x9
                0/4/1/0/4/0.0x50001fe150006e69.0x4004000000000000
0/4/1/0/4/0.1.2.0.0.0.4
                0/4/1/0/4/0.0x50001fe150006e68.0x4004000000000000
0/4/1/0/4/0.1.4.0.0.0.4
64000/0xfa00/0xa
                0/4/1/0/4/0.0x50001fe150006e69.0x4005000000000000
0/4/1/0/4/0.1.2.0.0.0.5
                0/4/1/0/4/0.0x50001fe150006e68.0x4005000000000000
0/4/1/0/4/0.1.4.0.0.0.5
```

The instance numbers of FCP array interfaces (virtual buses) in the legacy view, corresponding to the two target ports of the disk array DA1 are shown in the output of the ioscan command below. These instance numbers can be correlated with the legacy DSFs.

```
# ioscan -kfnCext_bus -H 0/4/1/0/4/0
Class     I  H/W Path                Driver     S/W State H/W Type     Description
=====================================================================
ext_bus  15  0/4/1/0/4/0.1.2.0.0     fcd_vbus   CLAIMED   INTERFACE    FCP Array Interface
ext_bus  14  0/4/1/0/4/0.1.2.255.0   fcd_vbus   CLAIMED   INTERFACE    FCP Device Interface
ext_bus  17  0/4/1/0/4/0.1.4.0.0     fcd_vbus   CLAIMED   INTERFACE    FCP Array Interface
ext_bus  16  0/4/1/0/4/0.1.4.255.0   fcd_vbus   CLAIMED   INTERFACE    FCP Device Interface
```

The following output of ioscan shows the legacy DSFs corresponding to the lunpaths of each disk.

```
# ioscan -m dsf
Persistent DSF           Legacy DSF(s)
========================================
/dev/rdisk/disk86        /dev/rdsk/c15t0d1
                         /dev/rdsk/c17t0d1
/dev/rdisk/disk87        /dev/rdsk/c15t0d2
                         /dev/rdsk/c17t0d2
/dev/rdisk/disk88        /dev/rdsk/c15t0d3
                         /dev/rdsk/c17t0d3
/dev/rdisk/disk89        /dev/rdsk/c15t0d4
                         /dev/rdsk/c17t0d4
/dev/rdisk/disk90        /dev/rdsk/c15t0d5
                         /dev/rdsk/c17t0d5
```

An application 'A' opened the legacy DSF /dev/rdsk/c17t0d1 to transfer data to disk86. While the I/O transfers are going on, reconfigure the SAN by connecting the port '5' of the switch SW1 to the target port '7' of the disk array DA2. The target port '7' has the following information:

- Area address: 4
- Port address: 0
- WWN: 0x50002fe250006e6c

Based on this information, the hardware path of the target path of DS2 is: 0/4/1/0/4/0.0x50002fe250006e6c. The corresponding legacy target path hardware paths start with: 0/4/1/0/4/0.1.4.

After the SAN reconfiguration, the target path of the disk array DA1 with the hardware path: 0/4/1/0/4/0.0x50001fe150006e68 as well as lunpaths beneath it are longer available.

Run the ioscan command to re-initiate the device probing.

The following output of the ioscan shows the new disks (instance 91 to 95) from DSA2.

```
# ioscan -knNfCdisk
Class     I  H/W Path  Driver S/W State   H/W Type     Description
===================================================================
disk     86  64000/0xfa00/0x6   esdisk   CLAIMED     DEVICE       COMPAQ  HSV111 (C)COMPAQ
                     /dev/disk/disk86   /dev/rdisk/disk86
disk     87  64000/0xfa00/0x7   esdisk   CLAIMED     DEVICE       COMPAQ  HSV111 (C)COMPAQ
                     /dev/disk/disk87   /dev/rdisk/disk87
disk     88  64000/0xfa00/0x8   esdisk   CLAIMED     DEVICE       COMPAQ  HSV111 (C)COMPAQ
                     /dev/disk/disk88   /dev/rdisk/disk88
disk     89  64000/0xfa00/0x9   esdisk   CLAIMED     DEVICE       COMPAQ  HSV111 (C)COMPAQ
                     /dev/disk/disk89   /dev/rdisk/disk89
disk     90  64000/0xfa00/0xa   esdisk   CLAIMED     DEVICE       COMPAQ  HSV111 (C)COMPAQ
                     /dev/disk/disk90   /dev/rdisk/disk90
disk     91  64000/0xfa00/0xb   esdisk   CLAIMED     DEVICE       COMPAQ  HSV111 (C)COMPAQ
                     /dev/disk/disk91   /dev/rdisk/disk91
disk     92  64000/0xfa00/0xc   esdisk   CLAIMED     DEVICE       COMPAQ  HSV111 (C)COMPAQ
                     /dev/disk/disk92   /dev/rdisk/disk92
disk     93  64000/0xfa00/0xd   esdisk   CLAIMED     DEVICE       COMPAQ  HSV111 (C)COMPAQ
                     /dev/disk/disk93   /dev/rdisk/disk93
disk     94  64000/0xfa00/0xe   esdisk   CLAIMED     DEVICE       COMPAQ  HSV111 (C)COMPAQ
                     /dev/disk/disk94   /dev/rdisk/disk94
disk     95  64000/0xfa00/0xf   esdisk   CLAIMED     DEVICE       COMPAQ  HSV111 (C)COMPAQ
                     /dev/disk/disk95   /dev/rdisk/disk95
```

The following output of the ioscan command shows the new target path (0/4/1/0/4/0.0x50002fe250006e6c) corresponding to target port '7' of DA2. Notice that the target path corresponding to target port '6' of DA1 is now in NO_HW state. This target path and all lunpaths beneath it are no longer available.

```
# ioscan -kfNCtgtpath
Class     I  H/W Path                          Driver S/W State H/W Type     Description
=========================================================================
tgtpath   5  0/4/1/0/4/0.0x50001fe150006e68  estp   CLAIMED   TGT_PATH     fibre_channel
target served by fcd driver
tgtpath   3  0/4/1/0/4/0.0x50001fe150006e69  estp   NO_HW     TGT_PATH     fibre_channel
target served by fcd driver
tgtpath   6  0/4/1/0/4/0.0x50002fe250006e6c  estp   CLAIMED   TGT_PATH     fibre_channel
target served by fcd driver
```

The following ioscan output shows the mapping between the lunpath hardware path of the disks and corresponding legacy hardware paths. Note the following:

- Lunpaths beneath the target path corresponding to the target port '6' of the disk array DA1 have no corresponding legacy hardware path. This is because this target path is no longer available.
- Lunpaths beneath the target path corresponding to the target port '7' of the disk array DA2 have no corresponding legacy hardware paths. This is because the SCSI stack has not yet associated the legacy lunpaths with the new disks from the disk array DA2.

```
# ioscan -m hwpath
Lun H/W Path      Lunpath H/W Path                 Legacy H/W Path
=====================================================================
64000/0xfa00/0x6
                  0/4/1/0/4/0.0x50001fe150006e69.0x4001000000000000
0/4/1/0/4/0.1.2.0.0.0.1
                  0/4/1/0/4/0.0x50001fe150006e68.0x4001000000000000
64000/0xfa00/0x7
                  0/4/1/0/4/0.0x50001fe150006e69.0x4002000000000000
0/4/1/0/4/0.1.2.0.0.0.2
                  0/4/1/0/4/0.0x50001fe150006e68.0x4002000000000000
64000/0xfa00/0x8
                  0/4/1/0/4/0.0x50001fe150006e69.0x4003000000000000
0/4/1/0/4/0.1.2.0.0.0.3
                  0/4/1/0/4/0.0x50001fe150006e68.0x4003000000000000
64000/0xfa00/0x9
```

```
                0/4/1/0/4/0.0x50001fe150006e69.0x4004000000000000
0/4/1/0/4/0.1.2.0.0.0.4
                0/4/1/0/4/0.0x50001fe150006e68.0x4004000000000000
64000/0xfa00/0xa
                0/4/1/0/4/0.0x50001fe150006e69.0x4005000000000000
0/4/1/0/4/0.1.2.0.0.0.5
                0/4/1/0/4/0.0x50001fe150006e68.0x4005000000000000
64000/0xfa00/0xb
                0/4/1/0/4/0.0x50002fe250006e6c.0x4001000000000000
64000/0xfa00/0xc
                0/4/1/0/4/0.0x50002fe250006e6c.0x4002000000000000
64000/0xfa00/0xd
                0/4/1/0/4/0.0x50002fe250006e6c.0x4003000000000000
64000/0xfa00/0xe
                0/4/1/0/4/0.0x50002fe250006e6c.0x4004000000000000
64000/0xfa00/0xf
                0/4/1/0/4/0.0x50002fe250006e6c.0x4005000000000000
```

The SCSI stack detects the change of the disk seen through the legacy lunpaths (legacy DSFs) with hardware path starting with: 0/4/1/0/4/0.1.4. It logs the following message for the legacy lunpaths impacted; One message for each legacy lunpath.

```
The legacy lun path (b 17 - t 0 - l 1) registration failed because it has been re-mapped
from its original LUN (default dev 0x0C000005) to a different LUN (default dev 0xC0000006).
The administrator has to close the original LUN and then validate this LUN re-mapping using
the scsimgr:
     scsimgr [-f] replace_leg_dsf -D /dev/rdsk/cxtydz


The legacy lun path (b 17 - t 0 - l 2) registration failed because it has been re-mapped
from its original LUN (default dev 0x0C000005) to a different LUN (default dev 0xC0000006).
The administrator has to close the original LUN and then validate this LUN re-mapping using
the scsimgr:
     scsimgr [-f] replace_leg_dsf -D /dev/rdsk/cxtydz


The legacy lun path (b 17 - t 0 - l 3) registration failed because it has been re-mapped
from its original LUN (default dev 0x0C000005) to a different LUN (default dev 0xC0000006).
The administrator has to close the original LUN and then validate this LUN re-mapping using
the scsimgr:
     scsimgr [-f] replace_leg_dsf -D /dev/rdsk/cxtydz


The legacy lun path (b 17 - t 0 - l 4) registration failed because it has been re-mapped
from its original LUN (default dev 0x0C000005) to a different LUN (default dev 0xC0000006).
The administrator has to close the original LUN and then validate this LUN re-mapping using
the scsimgr:
     scsimgr [-f] replace_leg_dsf -D /dev/rdsk/cxtydz


The legacy lun path (b 17 - t 0 - l 5) registration failed because it has been re-mapped
from its original LUN (default dev 0x0C000005) to a different LUN (default dev 0xC0000006).
The administrator has to close the original LUN and then validate this LUN re-mapping using
the scsimgr:
     scsimgr [-f] replace_leg_dsf -D /dev/rdsk/cxtydz
```

In these messages, the legacy lunpath is identified by the legacy bus instance (b), the legacy target identifier (t), and the legacy LUN identifier (l). The impacted legacy DSFs can be derived from these identifiers by following the legacy DSF naming convention (see intro(7)). The legacy DSFs impacted are:

- Raw legacy DSFs:
  - /dev/rdsk/c17t0d1
  - /dev/rdsk/c17t0d2
  - /dev/rdsk/c17t0d3
  - /dev/rdsk/c17t0d4
  - /dev/rdsk/c17t0d5
- Block legacy DSFs:

/dev/rdsk/c17t0d1
/dev/rdsk/c17t0d2
/dev/rdsk/c17t0d3
/dev/rdsk/c17t0d4
/dev/rdsk/c17t0d5

The disk86 of the disk array DA1 is still accessible through the lunpath: 0/4/1/0/4/0.0x50001fe150006e69.0x4001000000000000, and the multi-pathing is enabled by default on legacy DSFs. The application 'A' I/O transfers to disk86 through the legacy DSF /dev/rdsk/c17t0d1 are not interrupted. Another application can even open /dev/rdsk/c17t0d1 to access disk86 even if the corresponding lunpath is no longer available. The following example shows the dd command opening /dev/rdsk/c17t0d1 and transferring data successfully.

```
# dd if=/dev/rdsk/c17t0d1 of=/dev/null count=100
100+0 records in
100+0 records out
```

Validating the change of LUN binding for a single legacy DSF

You can choose to immediately validate the binding of some of the legacy DSFs with new LUNs and deferred it for others for example to let applications using these legacy DSFs complete. In this case, run the '*scsimgr replace_leg_dsf*' command for each legacy DSF for which you want to validate the change in binding. The example below validates the change in binding of the legacy DSF: c17t0d2.

```
# scsimgr -f replace_leg_dsf -D /dev/rdsk/c17t0d2
scsimgr: Legacy device file '/dev/rdsk/c17t0d2'  binding to LUN changed successfully
```

Validating the change of LUN binding for all legacy DSFs corresponding to lunpaths beneath a target path

You can choose to validate at once the change of the binding for all legacy DSFs impacted, by specifying the target path to the '*scsimgr replace_leg_dsf*' command. This is convenient if there are several lunpaths beneath each target path as in our example. You should make sure that all affected legacy DSFs are closed, then run *'scsimgr replace_leg_dsf'* for each target path beneath which legacy DSFs are impacted. For this example of reconfiguration, only one target path is affected: 0/4/1/0/4/0.0x50002fe250006e6c. You can validate the change in binding for all legacy DSFs by running the command below.

```
# scsimgr -fv replace_leg_dsf -H 0/4/1/0/4/0.0x50002fe250006e6c
scsimgr: Legacy device file '/dev/rdsk/c17t0d1'  binding to LUN changed successfully
scsimgr: Legacy device file '/dev/rdsk/c17t0d2'  binding to LUN changed successfully
scsimgr: Legacy device file '/dev/rdsk/c17t0d3'  binding to LUN changed successfully
scsimgr: Legacy device file '/dev/rdsk/c17t0d4'  binding to LUN changed successfully
scsimgr: Legacy device file '/dev/rdsk/c17t0d5'  binding to LUN changed successfully
```

Note: if you invoke 'scsimgr replace_leg_dsf' while the legacy DSF is still opened, the SCSI stack fails the validation of change in binding for the legacy DSF, and scsimgr displays an error message as shown below.

```
# scsimgr -f replace_leg_dsf -D /dev/rdsk/c17t0d1
scsimgr: ERROR: The Legacy device file is opened. replace_leg_dsf cannot proceed

# scsimgr -fv replace_leg_dsf -H 0/4/1/0/4/0.0x50002fe250006e6c
```

```
scsimgr: ERROR: The Legacy device file '/dev/rdsk/c17t0d1' is opened. replace_leg_dsf
operation cannot proceed on it.
scsimgr: Legacy device file '/dev/rdsk/c17t0d2'  binding to LUN changed successfully
scsimgr: Legacy device file '/dev/rdsk/c17t0d3'  binding to LUN changed successfully
scsimgr: Legacy device file '/dev/rdsk/c17t0d4'  binding to LUN changed successfully
scsimgr: Legacy device file '/dev/rdsk/c17t0d5'  binding to LUN changed successfully
```

Note: In case the SAN reconfiguration affects both legacy DSFs binding and lunpath authentication, if you run 'scsimgr replace_wwid' while the legacy DSF is closed, the SCSI stack automatically binds the legacy DSF to the new disk. However, if the legacy DSF is opened when 'scsimgr replace_wwid' is run, it remains bound to the original disk. The SCSI stack then logs a message similar to the one below:

```
Legacy lun path (default minor = 0x110100) mapping to LUN (default minor = 0x6) could not be
cleared in the context of rmsf on new LUN path
(0/4/1/0/4/0.0x50002fe250006e6c.0x4001000000000000).
Close the legacy LUN path before running ioscan to re-map the legacy lun path to a new LUN.
```

After the change is validated, the impacted legacy DSFs are bound to the disks of the disk array DA2 as shown in the following ioscan output.

```
# ioscan -m hwpath
Lun H/W Path      Lunpath H/W Path                 Legacy H/W Path
====================================================================
64000/0xfa00/0x6
                  0/4/1/0/4/0.0x50001fe150006e69.0x4001000000000000
0/4/1/0/4/0.1.2.0.0.0.1
                  0/4/1/0/4/0.0x50001fe150006e68.0x4001000000000000
64000/0xfa00/0x7
                  0/4/1/0/4/0.0x50001fe150006e69.0x4002000000000000
0/4/1/0/4/0.1.2.0.0.0.2
                  0/4/1/0/4/0.0x50001fe150006e68.0x4002000000000000
64000/0xfa00/0x8
                  0/4/1/0/4/0.0x50001fe150006e69.0x4003000000000000
0/4/1/0/4/0.1.2.0.0.0.3
                  0/4/1/0/4/0.0x50001fe150006e68.0x4003000000000000
64000/0xfa00/0x9
                  0/4/1/0/4/0.0x50001fe150006e69.0x4004000000000000
0/4/1/0/4/0.1.2.0.0.0.4
                  0/4/1/0/4/0.0x50001fe150006e68.0x4004000000000000
64000/0xfa00/0xa
                  0/4/1/0/4/0.0x50001fe150006e69.0x4005000000000000
0/4/1/0/4/0.1.2.0.0.0.5
                  0/4/1/0/4/0.0x50001fe150006e68.0x4005000000000000
64000/0xfa00/0xb
                  0/4/1/0/4/0.0x50002fe250006e6c.0x4001000000000000
0/4/1/0/4/0.1.2.0.0.0.1
64000/0xfa00/0xc
                  0/4/1/0/4/0.0x50002fe250006e6c.0x4002000000000000
0/4/1/0/4/0.1.2.0.0.0.2
64000/0xfa00/0xd
                  0/4/1/0/4/0.0x50002fe250006e6c.0x4003000000000000
0/4/1/0/4/0.1.2.0.0.0.3
64000/0xfa00/0xe
                  0/4/1/0/4/0.0x50002fe250006e6c.0x4004000000000000
0/4/1/0/4/0.1.2.0.0.0.4
64000/0xfa00/0xf
                  0/4/1/0/4/0.0x50002fe250006e6c.0x4005000000000000
0/4/1/0/4/0.1.2.0.0.0.5
```

Disabling LUNs and lunpaths for SAN maintenance

For maintenance and troubleshooting, you can disable I/O transfer on a lunpath of a block device or all lunpaths of a block device. When a lunpath or a LUN is disabled, its health state is set to *'disabled'*.

To disable the lunpath 0/4/1/0/4/0.0x50001fe150006e69.0x400f000000000000 you can run the scsimgr command below. Note the *health* state displayed with the ioscan command.

```
# scsimgr -f disable -H 0/4/1/0/4/0.0x50001fe150006e69.0x400f000000000000
LUN path 0/4/1/0/4/0.0x50001fe150006e69.0x400f000000000000 disabled successfully

# ioscan -P health -H 0/4/1/0/4/0.0x50001fe150006e69.0x400f000000000000
Class     I  H/W Path  health
================================
lunpath  27  0/4/1/0/4/0.0x50001fe150006e69.0x400f000000000000  disabled
```

To disable disk100 for I/O transfer.

```
# scsimgr -f disable -D /dev/rdisk/disk100
scsimgr: LUN /dev/rdisk/disk100 disabled successfully

# ioscan -P health /dev/rdisk/disk100
Class     I  H/W Path  health
==============================
disk    100  64000/0xfa00/0x15  disabled
```

Notes:

- *When all lunpaths to a LUN are disabled, applications can no longer open the LUN, and attempts to open the LUN will fail with the following error: 'no such device or address'.*
- *The system performs critical resource analysis (CRA) when the user requests to disable a lunpath or a LUN. If the lunpath or the LUN is critical for the operation of the system (for instance root/boot device), the system notifies and fails the operation.*

To enable a lunpath, which was previously disabled

```
# scsimgr enable -H 0/4/1/0/4/0.0x50001fe150006e69.0x400f000000000000
LUN path  0/4/1/0/4/0.0x50001fe150006e69.0x400f000000000000 enabled successfully
```

To enable a disk device, which was previously disabled

```
# scsimgr enable -D /dev/rdisk/disk100
scsimgr: LUN /dev/rdisk/disk100 enabled successfully
```

Enabling I/O transfer to a LUN after fixing an unrecoverable deferred error

When the SCSI stack detects that a device has experienced an unrecoverable deferred error, or a change of I/O block size, it disables the device for I/O transfer and logs a message to alert the system administrator. For example if an unrecoverable error occurred on disk100, the SCSI stack will log a message similar to the following:

```
An unrecoverable deferred error occurred on the device dev=0x0b000015, lba=0x0020f0.
The LUN has been disabled to minimize any further loss of data integrity.
The LUN may be enabled by running : scsimgr enable -D <raw_lun_dsf> after
the content of the LUN has been restored from the last known good copy.
```

After restoring the content of disk100, you can run the following command to re-enable I/O transfers.

```
# scsimgr enable -D /dev/rdisk/disk100
scsimgr: LUN /dev/rdisk/disk100 enabled successfully
```

Additional Resources

White papers
White papers can be found at: http://docs.hp.com/en/netsys.html#Storage%20Area%20Management

- The Next Generation Mass Storage Stack
- HP-UX 11i v3 Persistent DSF Migration Guide
- HP-UX 11i v3 Mass Storage Device Naming

Man pages

- intro(7), insf(1M), ioscan(1M), lssf(1M), mksf(1M), mknod(1M), rmsf(1M)
- scsimgr(1M), scsimgr_esdisk(7), scsimgr_estape(7), scsimgr_eschgr(7)