LVM New Features in HP-UX 11i v3

Abstract

In HP-UX 11i v3 (11.31), LVM delivers significant performance, scalability, usability and availability enhancements. This whitepaper lists all the new LVM features in HP-UX 11i v3.

Some features have their own whitepaper and this document will only summarize them. See the referenced documents in the For More Information section for further details. Other features are presented in more detail, including their usage and benefits.

The document is intended for system administrators, operators, and customers who wish to utilize and know about new LVM features in HP-UX 11i v3.

New functionality

Support for Agile View of Mass Storage

HP-UX 11i v3 introduces a new representation of mass storage devices called the **agile view**. In this representation, the device special file (DSF) name for each disk no longer contains path (or link) information. The multi-pathed disk has a single **persistent DSF** regardless of the number of physical paths to it. The **legacy view**, represented by the **legacy DSF**, continues to exist. Both the DSF types can be used to access a given mass storage device independently and both can coexist on a system. See the For More Information section for a whitepaper - The Next Generation Mass Storage Stack.

Wherever applicable LVM configuration commands support both DSF naming models.

LVM allows volume groups to be configured with all persistent DSFs, all legacy DSFs or a mixture of persistent DSFs and legacy DSFs.

HP recommends the use of persistent DSFs for LVM configurations and encourages configuring new volume groups using persistent DSFs. To fully utilize all the capabilities of the new mass storage stack HP recommends migrating volume groups from legacy DSFs to persistent DSFs. HP provides */usr/contrib/bin/vgdsf* to facilitate this migration. The script works for both root and non-root volume groups. See the For More Information section for a whitepaper - LVM Migration from legacy to persistent naming model.

Multi-Pathing and Alternate Links (PVlinks)

Management of multi-pathed devices is available outside of LVM using the next generation mass storage stack. Agile addressing creates a single persistent DSF for each mass storage device regardless of the number of hardware paths to the disk. The mass storage stack in HP-UX 11i v3 uses that agility to provide transparent multi-pathing. See the For More Information section for a whitepaper - The Next Generation Mass Storage Stack.

LVM's Alternate Link functionality is now redundant, but this functionality is still supported with legacy DSFs. Alternate links will behave as they did in prior releases when the mass storage stack native multi-pathing feature is disabled via *scsimgr* command.

HP recommends converting volume groups with alternate links to use native multi-pathing by the use of persistent DSFs. The */usr/contrib/bin/vgdsf* script, *vgscan -N* command, or *vgimport -s -N* commands perform this conversion. See the For More Information section for a whitepaper - LVM Migration from legacy to persistent naming model.

Dynamic LUN Expansion (DLE)

Some disk arrays allow the dynamic resizing of their LUNs. With HP-UX 11i v3 LVM detects and handles physical volume size changes on invocation of a new command called *vgmodify*.

vgmodify(1M)

The *vgmodify* command provides a method to alter the attributes of a physical volume and volume group post *pvcreate* and *vgcreate* respectively. The *vgmodify* command must be run to update the LVM configuration to reflect any change to the physical volume size.

Refer to the vgmodify(1M) manual page for details and also see the [For More Information] section for a whitepaper - LVM Volume Group Dynamic LUN expansion (DLE)/vgmodify.

pvmove(1M)

The *pvmove* command has been enhanced in HP-UX 11i v3 to allow relocation of just the first extent of the physical volume. The *vgmodify* command can use this additional space to expand the LVM's on-disk configuration information.

The *pvmove* command honors the existing allocation policy of a logical volume containing the extent that is considered for relocation.

Refer to the pvmove(1M) manual page for more details.

Examples

- To relocate the first data extent to any free extent within the same physical volume:
  # pvmove /dev/disk/disk10:0 /dev/disk/disk10

- To relocate the first data extent to any free extent in the volume group:
  # pvmove /dev/dsk/c1t0d0:0

- To find a physical volume that has free space, pvdisplay(1M) can be used .The user can then relocate the first user extent to that physical volume using *pvmove*:

  # pvdisplay /dev/disk/disk22

```
--- Physical volumes ---
PV Name                     /dev/disk/disk22
VG Name                     /dev/vgname
PV Status                   available
Allocatable                 yes
VGDA                        2
Cur LV                      5
PE Size (Mbytes)            4
Total PE                    1279
Free PE                     779
Allocated PE                500
Stale PE                    0
IO Timeout (Seconds)        default
Autoswitch                  On
```

  Relocate first data extent from the source physical volume to the desired destination physical volume which was identified in the last step:

  # pvmove /dev/disk/disk10:0 /dev/disk/disk22

  Note: Relocation of the first data extent fails in case it violates the strict mirror allocation policy. So identify a suitable physical volume which does not hold any mirror copies of the same extent which is considered for relocation.

- To relocate the first data extent of a physical volume when *vgmodify* reports insufficient space for expanding the LVM configuration data on the disk. This happens when user tried to modify the physical volume setting using *vgmodify*.

  Consider a volume group with one physical volume and one logical volume as follows:

```
# vgdisplay -v /dev/vgdatabase
--- Volume groups ---
VG Name                     /dev/vgdatabase
VG Write Access             read/write
VG Status                   available
Max LV                      255
Cur LV                      1
Open LV                     1
Max PV                      16
Cur PV                      1
Act PV                      1
Max PE per PV               2559
VGDA                        2
PE Size (Mbytes)            4
Total PE                    2559
Alloc PE                    10
Free PE                     2549
Total PVG                   0
Total Spare PVs             0
Total Spare PVs in use      0

   --- Logical volumes ---
   LV Name                     /dev/vgtest/lvol1
   LV Status                   available/syncd
   LV Size (Mbytes)            40
   Current LE                  10
   Allocated PE                10
   Used PV                     1


   --- Physical volumes ---
   PV Name                     /dev/dsk/c11t0d5
   PV Status                   available
   Total PE                    2559
   Free PE                     2549
   Autoswitch                  On
```

The *vgmodify* command can report optimized volume group settings to adjust the number of extents and physical volume upwards, where possible, to make full use of the space reserved on each physical volume for the LVM configuration data:

```
# vgmodify -o -r /dev/vgtest
Current Volume Group settings:
                                          Max LV      255
                                          Max PV       16
                                   Max PE per PV     2559
                                PE Size (Mbytes)        4
                              VGRA Size (Kbytes)      400
New configuration requires "max_pes" are increased from 2559 to 6652
The current and new Volume Group parameters differ.
An update to the Volume Group IS required


New Volume Group settings:
                                          Max LV      255
                                          Max PV       16
                                   Max PE per PV     6652
                                PE Size (Mbytes)        4
                              VGRA Size (Kbytes)      896
Review complete. Volume group not modified
```

The above output shows that "Max PV" is 16 and cannot scale beyond that with the current available space for LVM configuration data. An attempt to increase the maximum physical volume setting in the volume group to a value greater than 16 results in a failure as follows:

```
# vgchange -a n /dev/vgtest
Volume group "vgtest" has been successfully changed.

# vgmodify -n -p 64 /dev/vgtest
Current Volume Group settings:
                                          Max LV      255
                                          Max PV       16
                                   Max PE per PV     2559
                                PE Size (Mbytes)        4
                              VGRA Size (Kbytes)      400
vgmodify: This operation can only be completed if PE number zero on
"/dev/rdsk/c11t0d5" is freed
```

Note that the last output message indicates the user to free the first data extent. Use *pvmove* command to relocate the first data extent as follows:

```
# vgchange -a y /dev/vgtest
Activated volume group
Volume group "/dev/vgtest" has been successfully changed.

# pvmove /dev/dsk/c11t0d5:0
Transferring logical extents of logical volume "/dev/vgtest/lvol1"...
Physical volume "/dev/dsk/c11t0d5" has been successfully moved.
Volume Group configuration for /dev/vgtest has been saved in
/etc/lvmconf/vgtest.conf
```

Use *vgmodify* command to increase the maximum physical volume setting, this time it succeeds:

```
# vgchange -a n /dev/vgtest
Volume group "vgtest" has been successfully changed.

# vgmodify -n -p 64 /dev/vgtest
Current Volume Group settings:
                                          Max LV      255
                                          Max PV       16
                                   Max PE per PV     2559
                                PE Size (Mbytes)        4
                              VGRA Size (Kbytes)      400
The current and new Volume Group parameters differ.
An update to the Volume Group IS required


New Volume Group settings:
                                          Max LV      255
                                          Max PV       64
                                   Max PE per PV     2559
                                PE Size (Mbytes)        4
                              VGRA Size (Kbytes)     1488
New Volume Group configuration for "vgtest" has been saved in
"/etc/lvmconf/vgtest.conf"
Old Volume Group configuration for "vgtest" has been saved in
"/etc/lvmconf/vgtest.conf.old"
Starting the modification by writing to all Physical Volumes
Applying the configuration to all Physical Volumes from "/etc/lvmconf/vgtest.conf"
Completed the modification process.
```

```
New Volume Group configuration for "vgtest" has been saved in
"/etc/lvmconf/vgtest.conf.old"
Volume group "vgtest" has been successfully changed.
```

Now a *vgdisplay* on the volume group shows the modified values (in this case the maximum physical volume) for the volume group:

```
# vgchange -a y vgtest
Activated volume group
Volume group "vgtest" has been successfully changed.

# vgdisplay /dev/vgtest
VG Name                     /dev/vgtest
VG Write Access             read/write
VG Status                   available
Max LV                      255
Cur LV                      1
Open LV                     1
Max PV                      64
Cur PV                      1
Act PV                      1
Max PE per PV               2559
VGDA                        2
PE Size (Mbytes)            4
Total PE                    2558
Alloc PE                    10
Free PE                     2548
Total PVG                   0
Total Spare PVs             0
Total Spare PVs in use      0
```

Modification of Volume Group Settings

When an LVM volume group is created, several configuration parameters are set (such as max_pe, max_pv, max_lv). The new *vgmodify* command allows the user to change these configuration parameters on an existing volume group, which avoids having to migrate user data.

The *vgmodify* command can alter the following three configuration parameter set via *vgcreate*:

- The maximum number of physical extents that can be allocated per physical volume (max_pe setting set by *vgcreate -e*).
- The maximum number of physical volumes that the volume group can contain (max_pv setting set by *vgcreate -p*).
- The maximum number of logical volumes that the volume group can contain (max_lv setting set by *vgcreate -l*).

The *vgmodify* command displays the possible volume group max_pe and max_pv settings for this volume group to help optimize an existing volume group configuration.

Refer to the vgmodify(1M) manual page for details and see the For More Information section for a whitepaper - LVM Volume Group Dynamic LUN expansion (DLE)/vgmodify.

Modification of Physical Volume Type (Boot/Non-Boot)

When initializing a physical volume for LVM, *pvcreate* assigns a type to it, either boot or non-boot.
The *vgmodify* command allows the user to change a physical volume type from boot to non-boot or vice versa.

Refer to the vgmodify(1M) and pvcreate(1M) manual pages for the *-B* option. Also see the For More Information section for a whitepaper - LVM Volume Group Dynamic LUN expansion (DLE)/vgmodify.

Note that making a physical volume non-bootable increases the space available on that device for LVM configuration data. However, to take advantage of the additional space, every disk in a volume group must be marked non-bootable.

SLVM Single Node Online Reconfiguration (SNOR)

The SLVM SNOR feature allows changing the configuration of an active shared volume group in a cluster. Using new options in LVM commands, SLVM SNOR allows the system administrator to change the configuration of a shared volume group, and of logical and physical volumes in that volume group, while keeping it active in a single node. Using this procedure, applications on at least one node remain available during the volume group reconfiguration.

See the For More Information section for a whitepaper- SLVM Online Volume Re-configuration.
Also refer to vgchange(1M) manual page for more details.

LVM Device Online Replacement (OLR)

The LVM Online Disk Replacement (OLR) feature provides new methods for replacing or isolating path components or LVM disks within an active volume group:

- Using *-n* and *-N* options with *pvchange* command, a specific path or all paths to a physical volume can be detached respectively. LVM OLR enables the system administrator to follow a simpler procedure for replacing disks in an active volume group. The procedure does not require deactivating the volume group, modifying the volume group configuration or moving any user data.
- LVM OLR can also be easily employed to isolate troublesome paths or disks to allow running diagnostics against them.

In HP-UX 11i v3, the option of detaching an entire physical volume using *pvchange -a N* command, in order to perform an online disk replacement is still supported. The behavior is the same for both legacy and persistent DSFs and is compatible with previous releases.

Unless native multi-pathing is disabled and only legacy DSFs are configured for the volume group, the '*pvchange -a n'* command will not stop I/O operations for that path as they did in earlier releases. Instead, use the *scsimgr* command with the disable option to disable physical volume paths. Refer scsimgr(1M) manual pages for more details.

See the For More Information section for whitepapers – LVM Online Disk Replacement (LVM OLR) and When Good Disks Go Bad: Dealing with Disk Failures under LVM.

Volume Group Quiesce and Resume

The LVM volume group Quiesce/Resume feature allows quiescing I/O operations to the disks in a volume group to facilitate creating a consistent snapshot of an otherwise active volume group for backup purposes. This feature is designed to work with backup management and disk array management software to allow them to create a consistent snapshot of the disks that make up an LVM volume group.

The Quiesce/Resume feature prevents the disk images from changing and allows a snapshot of the disks to be taken without having to unmount or close the open logical volumes and deactivate the volume group.

The *vgchange* command provides new options *-Q* and *-R* to allow quiescing the volume group prior to creating a snapshot and to resume the volume group afterward. Optionally, both reads and writes or just writes to the volume group can be quiesced.

See the For More Information section for a whitepaper - LVM Volume Group Quiesce/Resume.

Boot Resiliency

Root volume group scanning is a new LVM feature in HP-UX 11i v3. The feature can prevent boot failures that can occur on prior HP-UX releases.

During boot, the root volume group activation can fail if the LVM boot configuration information is incorrect or out-of-date with the systems current I/O configuration. Two of the possible causes are:

- The root volume group is configured using legacy DSFs representing the devices in a Storage Area Network(SAN) and the SAN is reconfigured such that DSFs of the devices changed.
- The root disk is relocated to a different slot such that the DSF name changes.

With the new root volume group scanning, LVM automatically handles such situations; LVM now scans all the disks to identify the ones belonging to the root volume group and retries activation. If the activation succeeds, it's likely that the LVM's in-memory boot configuration information for the root volume group is out of sync with the DSF in the /etc/lvmtab for the root volume group. To assist recovery in this case the LVM driver prints a warning message to the console and logs into /var/adm/syslog/syslog.log to the effect -

```
"LVM: WARNING: Root VG activation required a scan. The PV information in the on-
disk BDRA may be out-of-date from the system's current IO configuration. To update
the on-disk BDRA, first update /etc/lvmtab using vgscan(1M), then update the on-
disk BDRA using lvlnboot(1M).
For example, if the root VG name is /dev/vg00:
      vgscan -k -f /dev/vg00
      lvlnboot -R /dev/vg00"
```

In case some physical volumes in the root volume group are not available but the quorum is met, no root volume group scan is performed. Also, during a single user mode boot with '*-is*' or maintenance mode boot with '*-lm*' root volume group scanning is skipped.

Striped and Mirrored Logical Volumes

HP-UX LVM now introduces support for striped and mirrored logical volumes at a smaller granularity than the extent size (4KB being the smallest possible stripe size).

RAID 1+0 and RAID 0+1

RAID 0, commonly referred to as *stripping*, refers to the segmentation of logical sequences of data across disks. RAID 1, commonly referred to as *mirroring*, refers to creating exact copies of logical sequences of data. When implemented in a device hardware, RAID 10 (or RAID 1+0) and RAID 01 (or RAID 0+1) is nested RAID levels. The difference between RAID 0+1 and RAID 1+0 is the location of each RAID system — RAID 0+1 is a mirror of stripes whereas RAID 1+0 is a stripe of mirrors. Figure 1 depicts the RAID 10 and RAID 01 configurations (A1, A2…Ax are stripe chunks of a logical volume).

With a hardware-based RAID 10 configuration, I/O operations are striped first then each strip is mirrored. With hardware-based RAID 01, I/Os are mirrored first then striped. RAID 10 and RAID 01 can have the same physical disk layout.

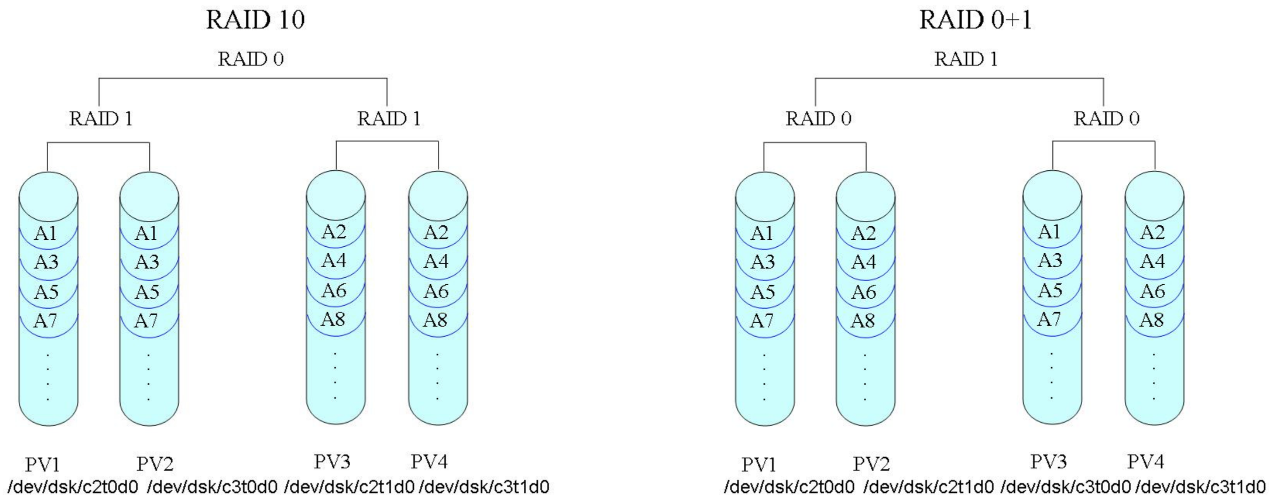

Figure 1: RAID 1+0 and RAID 0+1

The advantages of hardware-based RAID 10 over RAID 01:

- When one disk fails and is replaced, only the amount of data on this disk needs to be copied/re-synchronized.
- RAID 10 is more tolerant to multiple disk failures before data becomes unavailable.

The advantages of hardware-based RAID 01 over RAID 10:

- Simpler to configure striped volumes and then extend mirroring.
- Able to split the mirror copy and have two usable volume sets.

LVM's Implementation of RAID levels in HP-UX

LVM's implementation of RAID management is different from the hardware based solutions, because it does not nest the RAID levels, but processes them simultaneously. Typically with hardware solutions, you create a LUN with a RAID level and the RAID functions are stacked. LVM provides more flexibility on how logical volumes are created amongst a set of disks as compared to hardware solutions.

LVM allocates the physical extents for striped and mirrored logical volumes in sets of stripe width multiplied by the number of copies of the data. For instance, if the logical volume is 1-way mirrored and striped across two disks, extents are allocated to the logical volume 4 at a time. LVM enforces that the physical extents of a single set are from different physical volumes. Within this set, the logical extents are stripped and mirrored, to obtain the data layout displayed in Figure 1.

Striping and mirroring in LVM combines the advantages of the hardware implementation of RAID 1+0 and RAID 0+1, and provides the following benefits:

- Better write performance. Write operations takes place in parallel and each physical write operation is directed to a different physical volume.
- Excellent performance for read. Even in the case where several disks are out of service the read of a stripe can be done in parallel on different physical volumes with one I/O operation per physical volume.
- High availability of data. With multiple copies of the user data residing on different physical volumes, LVM avoids single point of failure ensuring high availability.

Configuration

A user can create a new Striped and Mirrored logical volume from scratch, or extend (add mirror to) an existing striped logical volume. With the use of physical volume groups (PVGs), mirroring and striping can be directed to specific physical volumes. Without the constraint created by the nesting of levels in hardware based RAID solutions, LVM is combining the striping and mirroring processing which allows LVM to provide the best of both RAID 10 and RAID 01.

To create a pure RAID 10 ensure that the physical volumes have enough extents in terms of size and stripe width to accommodate the logical volume. Otherwise, LVM optimizes the extent allocation to allow the user to create a striped and mirrored logical volume as long as there are enough free extents in the volume group. The logical volume layout then uses a slight variation of RAID 10 and all the benefits of RAID 10 are retained. For example, consider a volume group with 7 physical volumes and a logical volume striped across 3 disks with 1 mirror copy as shown in the figure below:

	First mirror			Second mirror		
Stripes from PV →	PV0	PV2	PV3	PV4	PV5	PV6
	PV0	PV2	PV3	PV4	PV5	PV6
	PV1	PV2	PV3	PV4	PV5	PV6
	PV1	PV2	PV3	PV4	PV5	PV6

Figure 2: Example of LVM's implementation of RAID 10 variation.

If the first physical volume used in the creation had enough extents (size of LV/3) to create RAID 10, only 6 physical volumes would be used resulting in a strict RAID 10 configuration. But as one of the physical volumes, PV0 didn't have enough free extents; LVM used another physical volume, PV1, to complete the operation. PV0 was short by 2 extents so 2 extents were allocated from PV1. LVM created the logical volume using 7 physical volumes resulting in a slight variation of RAID 10.

Customers can extend their existing striped logical volumes to a striped and mirrored configuration easily, using the *lvextend* command to add mirrors. Existing mirrored logical volumes cannot be converted to a striped and mirrored configuration. They will be able to create any new logical volumes with striped and mirrored configuration as explained in the 'Examples' section.

LVM allows the splitting and merging of striped and mirrored logical volumes. These logical volumes are also supported in a clustered environment with shared and in exclusive mode activation.

Examples

Use any of the following procedures to create a striped and mirrored logical volume:

- Creating a logical volume of size 90MB striped across two physical volumes with one mirror copy and stripe size of 64 KB.

  ```
  # lvcreate -L 90 -i 2 -I 64 -m 1 /dev/vgtest
  ```

  Note: Striping with mirroring always uses strict allocation policies where copies of data don't exist on the same physical disk. This result in a configuration similar to the RAID 01 as illustrated in the above Figure1.

- Creating a logical volume of size 90MB striped across two physical volumes with one mirror copy and stripe size of 64 KB and creating the mirror copies on specific disks (configuration equivalent to RAID 01 as illustrated in the Figure1 above)
  ```
  # lvcreate -L 90 -i 2 -I 64 –m 1 /dev/vgtest
  ```

Contents of /etc/lvmpvg

```
VG /dev/vgtest
PVG PVG0
/dev/dsk/c2t0d0
/dev/dsk/c2t1d0
PVG PVG1
/dev/dsk/c3t0d0
/dev/dsk/c3t1d0
```

- Adding a mirror to existing striped logical volumes.

  The existing logical volume has the striped property as shown by the lvdisplay output:

  # lvdisplay /dev/vgtest/lvol1

```
--- Logical volumes ---
LV Name                     /dev/vgtest/lvol1
VG Name                     /dev/vgtest
LV Permission               read/write
LV Status                   available/syncd
Mirror copies               0
Consistency Recovery        MWC
Schedule                    striped
LV Size (Mbytes)            1024
Current LE                  256
Allocated PE                256
Stripes                     2
Stripe Size (Kbytes)        64
Bad block                   on
Allocation                  strict
IO Timeout (Seconds)        default
```

  To get a striped and mirrored configuration extend the logical volume as follows:

  # lvextend -m 1 /dev/vgtest/lvol1

  Note that the volume group must have enough physical volumes and extents to accommodate the mirror copy.

  Now the *lvdisplay* output shows the logical volume is striped and mirrored:

  # lvdisplay /dev/vgtest/lvol1

```
--- Logical volumes ---
LV Name                     /dev/vgtest/lvol1
VG Name                     /dev/vgtest
LV Permission               read/write
LV Status                   available/syncd
Mirror copies               1
Consistency Recovery        MWC
Schedule                    striped
LV Size (Mbytes)            1024
Current LE                  256
Allocated PE                256
Stripes                     2
Stripe Size (Kbytes)        64
Bad block                   on
Allocation                  strict
IO Timeout (Seconds)        default
```

Compatibility Note

Releases prior to HP-UX 11i v3 only support striped or mirrored logical volume and do not support combination of striped and mirrored logical volume. If a logical volume using simultaneous mirroring and striping is created on HP-UX 11i v3, attempts to import or activate its associated volume group will fail on a previous HP-UX release.

To import the volume group with striped and mirrored logical volume to releases prior to HP-UX 11i v3, you must remove the incompatible logical volumes or reduce them to no mirrors.

Better Co-existence with Other Disk Users

LVM co-exists better with other disk users: the LVM commands *pvcreate* and *vgcfgrestore* now check if their target device is in use by other subsystems.

LVM has a provision to disown or unmark a LVM disk. This is facilitated by the *pvremove* command. Refer to the pvremove(1M) man page for complete details.

Note: *pvcreate* with the force option *-f* overrides this checks and takes ownership of the disk, overwriting any other subsystem's data. HP recommends NOT using the *-f* option by default in any of their scripts or during initializing a physical volume using *pvcreate* command.

Example

Co-existence with Oracle ASM disk: Consider the disk /dev/dsk/c3t1d1 is marked for use by Oracle ASM. An initialization operation on this disk identifies that the disk belongs to some other subsystem as below:

```
# pvcreate /dev/rdsk/c3t1d1
pvcreate: Could not perform LVM operation on Oracle ASM disk "/dev/rdsk/c3t1d1"
```

To disown or unmark an LVM disk which is not part of any volume group use *pvremove* command:

```
# pvremove /dev/rdisk/disk26
The physical volume associated with "/dev/rdisk/disk26" has been removed.
```

Better Utilization of Disk Space – No Bad Block Reserve Area

Software bad block relocation refers to capability of LVM to handle I/O errors on disk blocks which are bad. In previous releases of HP-UX, upon a media failure (detection of a bad block of data on disk), LVM would mark the failed block in the Bad Block Directory, and attempted to relocate the block to a new location in the Bad Block Reserve Area(BBRA) on the disk.

The BBRA was created when the disk was initialized using the *pvcreate* command.

All modern disks now offer hardware based bad sector relocation and don't trigger the LVM software relocation. With HP-UX 11i v3, LVM does not reserve a BBRA at the end of disk to allocate more space for user data.

Compatibility Note

In HP-UX 11i v3 LVM is compatible with disks which already have bad blocks relocated to the BBRA. For example, consider a disk device configured for LVM on HP-UX 11i v2. This disk has bad blocks relocated to the BBRA. On importing such a disk onto a HP-UX 11i v3 release, LVM will continue to support this disk and will be capable of reading the relocated blocks from the BBRA.

Elimination of 'maxvgs' Tunable

This static tunable specified the maximum number of LVM volume groups in the range 1 to 256, which could be created or activated on the system. With HP-UX 11i v3 release, the maxvgs tunable has been removed. Users can now create up to 256 volume groups in a system without requiring setting this kernel tunable, thus **avoiding a reboot**.

Note: Scripts which use or modify maxvgs tunable will require changes for HP-UX 11i v3.

Example

On a HP-UX 11i v2 system:

```
# kctune maxvgs
Tunable  Value  Expression
maxvgs      10  Default
```

Modifying the maxvgs tunable on a HP-UX 11i v2 release results in a reboot:

```
# kctune maxvgs=64

NOTE:    The configuration being loaded contains the following change(s)
         that cannot be applied immediately and which will be held for
         the next boot:
      -- The tunable maxvgs cannot be changed in a dynamic fashion.
WARNING: The automatic 'backup' configuration currently contains the
         configuration that was in use before the last reboot of this
         system.
     ==> Do you wish to update it to contain the current configuration
         before making the requested change? y
       * The automatic 'backup' configuration has been updated.
       * The requested changes have been saved, and will take effect at
         next boot.
Tunable               Value  Expression
maxvgs   (now)           10  Default
         (next boot)     64  64
```

On a HP-UX 11i v3 system, no reboot is required as the maxvgs tunable is removed:

```
# kctune maxvgs
ERROR:   There is no tunable named 'maxvgs'.
```

See the maxvgs(5) manual page for more details.

Performance Improvements

Mirror Write Cache (MWC) Enhancements

The MWC allows a fast resynchronization of data following a system crash or failure.

The MWC keeps track of where I/O writes are occurring on the volume group and periodically records this activity in an on-disk data structure. An extra disk write is required for every mirrored write not already recorded on the physical volume. Upon system reboot after crash, the operating system uses the MWC to resynchronize inconsistent data blocks quickly.

In HP-UX 11i v3, the MWC is larger than in previous releases. This leads to a better logical volume I/O performance by allowing more concurrent writes. MWC has also been enhanced to support large I/O sizes.

To enable or disable MWC, use the '*-M y*' or '*-M n*' option with *lvcreate* or *lvchange* command respectively. Refer to the lvcreate(1M) and lvchange(1M) manual page for more information on MWC.

Compatibility Note:

When a volume group containing a logical volume (created on pre-11i v3 releases) using the MWC is activated on HP-UX 11i v3, its MWC format is converted. Any subsequent activation of this volume group on previous releases will not recognize the new MWC format and LVM performs a full resynchronization of the entire logical volume. But if the logical volumes are marked "NOMWC" and the logical volume is synced, then activation of the volume group on previous releases will avoid full resynchronization. Note that this would happen during a Serviceguard rolling update configuration. Previous releases may be enhanced to recognize the new MWC format; contact your Hewlett-Packard support representative for up-to-the-moment information, or consult the Hewlett-Packard IT Resource Center site: http://itrc.hp.com (Americas and Asia Pacific) or http://europe.itrc.hp.com (Europe) for LVM patches.

LVM Support for Large I/O Sizes

LVM now supports larger I/O sizes (up to the extent size). As I/O requests don't need to be split into smaller chunks, the I/O throughput is improved by issuing reads and writes of larger sizes through LVM.

Note that the actual I/O size may be limited by the interface card.

See the For More Information section for a System Admin guide – HP-UX System Administrator's Guide Logical Volume Management.

Also see the For More Information for a whitepaper – The Next Generation Mass Storage Stack.
HP recommends users to refer the whitepaper for information on benefits and capabilities introduced in the New Mass Storage Stack in HP-UX 11i v3.

Increased Limits

Maximum Logical Volume Size Increased up to 16TB

LVM in HP-UX 11i v3 release supports logical volume of sizes up to 16 terabyte (TB) (versus 2TB in earlier releases).

In LVM, logical volume sizes must be a multiple of the extent size. Extent size is expressed in units of megabytes (MB) in the range of 1 to 256 and number of extents per logical volume is a decimal value in the range of 1 to 65535. Thus a logical volume with 65535 extents and 256MB extent size will have the maximum supported size (16TB – 256MB).

The following are the advantages of16TB logical volumes:

- These logical volumes offer better scalability.
- These logical volumes can be mirrored, striped or striped and mirrored.
- These logical volumes are supported in standalone, Shared LVM and exclusive mode.

Existing logical volumes using 256MB extents which are of 2TB or lesser size can be extended up to 16TB.

Note: For the logical volumes to extend up to 16TB, physical volumes in volume group should have sufficient LVM configuration space on the disk. To increase the LVM configuration space see the For More Information section for a whitepaper on: LVM Volume Group Dynamic LUN expansion (DLE)/vgmodify.

Compatibility Note

Releases prior to HP-UX 11i v3 can only access data within the first 2TB of a logical volume. If a logical volume larger than 2TB is created on HP-UX 11i v3, its activation and use is not recommended on any previous HP-UX release. Though the logical volume can be activated and used on earlier releases, the data beyond 2TB will be inaccessible.

Usability Enhancements

Compact, Parsable Command Output

LVM display commands have been enhanced to produce a compact and easily parsable output. A new option *-F* is introduced in *pvdisplay, vgdisplay and lvdisplay* commands. The commands generate colon separated fields as key=value[,value…]:key=value[,value].

The *-F* option is designed to be used by scripts. The resulting command output may be split across multiple lines. The output may include new keys and/or values in the future. The positioning of a given key and value pair is not guaranteed. If a key is deprecated, its associated value is set to NAM (key=NAM). HP recommends all user scripts depending on LVM display commands' output to be modified to use *-F* option.

Examples

- Given a logical volume name and field name as specified in the *lvdisplay -F* output, the script below extracts the value for the given key:

  ```
  # cat get_lvinfo

  #!/bin/sh

  LV_NAME=$1;
  LVDISPLAY_F_OUTPUT_KEY=$2;

  lvdisplay -F $LV_NAME | /usr/bin/awk -F':' '{for(i=1;i<=NF;i++) print $i;}' | \
          awk /$LVDISPLAY_F_OUTPUT_KEY/ | cut -f2 -d "="

  # get_lvinfo /dev/vgtest/lvol1 lv_size
  40
  ```

- Given a volume group name, the script below lists all the physical volume belonging to the volume group

  ```
  # cat get_pv_in_vg

  #!/bin/sh

  VG_NAME=$1;

  vgdisplay -Fv $VG_NAME | sed -n -e "s/^.*pv_name=\([^:.]*\):.*$/\1/p"

  # get_pv_in_vg /dev/vgtest
   /dev/dsk/c11t0d5
  /dev/dsk/c7t0d1
  ```

- Perl Sample example: Given a physical volume name and field name as specified in the *pvdisplay -F* output , the script below extracts the value for the given key:

  ```
  # cat get_pvinfo

  #!/usr/bin/perl

  my $PV_NAME = $ARGV[0];
  my $PVDISPLAY_KEY = $ARGV[1];
  my %pvdisplay_hash = ();
  ```

```
my $pvdisplay_out = `pvdisplay -F $PV_NAME 2> /dev/null`;

if($? != 0) {
 print("The PV $PV_NAME does not belong to any VG\n");
 print("Provide a PV which belong to a VG\n");
 exit(1);
}

my @pvdisplay_pairs = split(':',$pvdisplay_out);

foreach my $key_value (@pvdisplay_pairs) {
        my @pair = ();
        @pair = split('=',$key_value);
        $pvdisplay_hash{$pair[0]} = $pair[1];
}

print "$pvdisplay_hash{$PVDISPLAY_KEY}\n";
```

 # get_pvinfo /dev/dsk/c11t0d5 vg_name
/dev/vgtest

pvdisplay – Displays user data offset and check disk under HP-UX LVM's control

The *pvdisplay* command displays information about physical volumes in LVM. It is enhanced to list additional details for a physical volume and to report if the disk is owned by LVM.

The *pvdisplay* command has the following new options:

-d
: For each physical volume, display the offset to the start of the user data in 1024 byte blocks from the beginning of the physical volume, specify if pv_path is a bootable physical volume, and display the number of bad blocks that were relocated. These details are displayed in addition to other information.

-l
: Check whether physical volume refers to a disk device under HP Logical Volume Manager (LVM) control. This option can be used on any disk in the system. For example, on a system with many disks, this option can be used to determine which disks are marked for LVM usage and which are not.

Refer to the pvdisplay(1M) manual page for more information.

Examples

 # pvdisplay -d /dev/dsk/c22t6d2

```
--- Physical volumes ---
PV Name                     /dev/dsk/c22t6d2
VG Name                     /dev/vgname
PV Status                   available
Allocatable                 yes
VGDA                        2
Cur LV                      5
PE Size (Mbytes)            4
Total PE                    1279
Free PE                     779
Allocated PE                500
Stale PE                    0
IO Timeout (Seconds)        default
Autoswitch                  On
```

```
Data Start                  1024
Boot Disk                   no
Relocated Blocks            61
```

To check whether the physical volume is owned by LVM do the following:

```
# pvdisplay -l /dev/dsk/c4t3d0
/dev/dsk/c4t3d0:LVM_Disk=yes

# pvdisplay -l /dev/disk/disk96
/dev/disk/disk96:LVM_Disk=no
```

vgscan - Scans Faster, Per Volume Group Scanning and Supports Persistent DSF

The *vgscan* command recovers the LVM configuration file, /etc/lvmtab. It scans all physical volumes for LVM volume groups and adds entries for volume groups that are missing from /etc/lvmtab. The vgscan command recovers volume group information by using LVM data structures in kernel memory, and by probing all devices, searching for LVM disks.

In HP-UX 11i v3 *vgscan* has the following new options:

-f <vgs>

With this option *vgscan* works on a **per volume group** basis rather than scanning for all the volume groups. This option **replaces** any existing entries related to these volume groups in /etc/lvmtab with updated entries. If the specified volume groups are missing, *vgscan* adds them to /etc/lvmtab. Note: *vgscan -f* does not search for additional volume groups other than the one specified with the command and does not report unconfigured volume groups.

-k

This option provides a mechanism to make *vgscan* **faster**. It skips the disk probe portion of the vgscan, and retrieves volume group information only from the LVM data structures in kernel memory. Note: with this option, *vgscan* only adds activated volume groups to /etc/lvmtab. For deactivated volume groups no information is added to /etc/lvmtab.

-N

Recover /etc/lvmtab file **using persistent DSFs**, with the exception of active volume groups configured with legacy DSFs. In this case *vgscan* populates the /etc/lvmtab file using legacy DSFs.

-B

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

With the above options, *vgscan* provides the following improvements:

- With the -f option, an incorrect volume group entry in /etc/lvmtab can be corrected without modifying the rest of /etc/lvmtab. Previously, this could only be done in a cumbersome way using *vgexport* and *vgimport* to recover lvmtab entries for the root volume group.
- With the -f option, a user can specify the ordering of volume groups in /etc/lvmtab, such as placing the boot volume group first.
- With the *-k* option *vgscan* can recover activated volume groups in /etc/lvmtab quickly (in seconds), even if the system has a large number of LVM disks configured.

Examples

- For the volume group /dev/vg01, overwrite the existing physical volume DSFs in /etc/lvmtab with the physical volume DSFs found belonging to /dev/vg01 during a hardware probe of all devices.

```
# vgscan -f /dev/vg01
Physical Volume "/dev/dsk/c2t0d0" contains no LVM information
Couldn't stat physical volume "/dev/dsk/c0t0d0":
Invalid argument
Physical Volume "/dev/dsk/c13t0d5" is not part of a Volume Group
Physical Volume "/dev/dsk/c11t0d5" is not part of a Volume Group
Physical Volume "/dev/dsk/c20t0d6" is not part of a Volume Group
Physical Volume "/dev/dsk/c17t0d6" is not part of a Volume Group
Physical Volume "/dev/dsk/c17t1d0" is not part of a Volume Group
Physical Volume "/dev/dsk/c20t1d0" is not part of a Volume Group
Physical Volume "/dev/dsk/c17t13d3" is not part of a Volume Group
Physical Volume "/dev/dsk/c20t13d3" is not part of a Volume Group
Physical Volume "/dev/dsk/c17t13d6" contains no LVM information
Physical Volume "/dev/dsk/c20t13d6" contains no LVM information

*** LVMTAB has been updated successfully.
*** Do the following to resync the information on the disk.
*** #1.  vgchange -a y
*** #2.  lvlnboot -R
```

- For the volume group /dev/vg01, overwrite the existing physical volume DSFs in /etc/lvmtab with the physical volume DSFs used in kernel memory. The volume group /dev/vg01 must be activated, or this command will fail.

```
# vgscan -k -f /dev/vg01
```

- For the volume group /dev/vg01, overwrite the existing physical volume DSFs in /etc/lvmtab with the physical volumes persistent DSFs found belonging to /dev/vg01 during a hardware probing of all devices.

```
# vgscan -N -f /dev/vg01
vgscan: Warning: couldn't query physical volume "/dev/dsk/c7t0d1":
The specified path does not correspond to physical volume attached to
this volume group
vgscan: Warning: couldn't query physical volume "/dev/dsk/c9t0d1":
The specified path does not correspond to physical volume attached to
this volume group
vgscan: Warning: couldn't query all of the physical volumes.
vgscan: The physical volume "/dev/disk/disk25_p2" is already recorded in the
"/etc/lvmtab" file.
vgscan: The physical volume "/dev/disk/disk30" is already recorded in the
"/etc/lvmtab" file.
vgscan: The physical volume "/dev/disk/disk31" is already recorded in the
"/etc/lvmtab" file.
Physical Volume "/dev/disk/disk26" contains no LVM information
Couldn't stat physical volume "/dev/disk/disk27":
Invalid argument
Physical Volume "/dev/disk/disk36" is not part of a Volume Group
Couldn't stat physical volume "/dev/disk/disk80":
Invalid argument
Couldn't stat physical volume "/dev/disk/disk81":
Invalid argument
Physical Volume "/dev/disk/disk82" is not part of a Volume Group
Physical Volume "/dev/disk/disk83" is not part of a Volume Group
Physical Volume "/dev/disk/disk93" is not part of a Volume Group
Physical Volume "/dev/disk/disk96" contains no LVM information
*** LVMTAB has been updated successfully.
*** Do the following to resync the information on the disk.
*** #1.  vgchange -a y
*** #2.  lvlnboot -R
```

- Recreate the /etc/lvmtab file for volume groups activated since the last boot. For activated and deactivated volume groups, use both persistent and legacy DSFs. Report all physical volume persistent and legacy DSFs belonging to unconfigured volume groups.

```
# mv /etc/lvmtab /etc/lvmtab.BCK
# vgscan -B
Creating "/etc/lvmtab".
Physical Volume "/dev/disk/disk26" contains no LVM information
Couldn't stat physical volume "/dev/disk/disk27":
Invalid argument
Physical Volume "/dev/disk/disk36" is not part of a Volume Group
Physical Volume "/dev/disk/disk82" is not part of a Volume Group
Physical Volume "/dev/disk/disk83" is not part of a Volume Group
Physical Volume "/dev/disk/disk93" is not part of a Volume Group
Physical Volume "/dev/disk/disk96" contains no LVM information
Physical Volume "/dev/dsk/c2t0d0" contains no LVM information

Following Physical Volumes belong to one Volume Group.
Unable to match these Physical Volumes to a Volume Group.
Use the vgimport command to complete the process.
/dev/disk/disk28
/dev/disk/disk29
/dev/dsk/c5t0d1
/dev/dsk/c7t0d0

The Volume Group /dev/vgtest was not matched with any Physical Volumes.
*** LVMTAB has been created successfully.
*** If PV links are configured in the system.
*** Do the following to resync information on disk.
*** #1.   vgchange -a y
*** #2.  lvlnboot -R
```

Refer to the vgscan(1M) manual page for more information.

vgcfgrestore – Enhanced Listing of Backup File Contents

The *vgcfgrestore* command restores the LVM configuration data from a default (-n option) or alternate (-f option) configuration backup file to a physical volume, or it displays the configuration backup file content (-l option).

In HP-UX 11i v3 the *vgcfgrestore -l* command provides the following additional information from the configuration backup file when invoked together with the *-v* option:

- For each physical volume, display the disk size in kilobytes
- For each physical volume, display the starting block number (KB) of the user data
- For each physical volume, display the PVkey (See the lvdisplay(1M) manual page for more information).
- For each volume group, display max_pv, max_pe, and max_lv.

Example

```
# vgcfgrestore -l -v -n vg00
Volume Group Configuration information in "/etc/lvmconf/vg00.conf"
VG Name /dev/vg00
 ---- Physical volumes : 1 ----
     PV         Type         Size (kb)          Start (kb)   PVkey
   disk95_p2   Bootable      34643968           2912         0

max_pv 16  max_pe 4238 max_lv 255
```

Error Management Technology (EMT)

Error Management Technology (EMT) provides an online searchable repository containing error descriptions, probable causes and recommended actions on systems running HP-UX. With HP-UX 11i v3 LVM supports EMT and LVM error messages are available in the EMT repository.

Long Host Name Support for Display Commands

Shared LVM (SLVM) allows a volume group to be shared across different nodes within a Serviceguard cluster. In this environment, the *vgdisplay* command displays the host names of the nodes participating in the Serviceguard cluster. One of the host names is represented as "Server" and all other nodes as "Client".

In HP-UX 11i v3 the *vgdisplay* command has been enhanced to support large host names of more that 32 character (as in prior releases) up to 255 characters.

Note that the though LVM supports host name length up to 255 characters, the host name displayed depends on the host name length supported by Serviceguard.

See the For More Information for a whitepaper on: Node and Host Name Sizes on HP-UX. The whitepaper provides information on supported host name lengths.

Example

Display the hostname information (machine-server and machine-client)

```
# vgdisplay -v /dev/vgtest1
--- Volume groups ---
VG Name                     /dev/vgtest1
VG Write Access             read/write
VG Status                   available, shared, server
Max LV                      255
Cur LV                      1
Open LV                     1
Max PV                      16
Cur PV                      1
Act PV                      1
Max PE per PV               8750
VGDA                        2
PE Size (Mbytes)            8
Total PE                    8749
Alloc PE                    50
Free PE                     8699
Total PVG                   0
Total Spare PVs             0
Total Spare PVs in use      0
   ---  ---

   machine1-server                     Server
   machine2-client                     Client

   --- Logical volumes ---
   LV Name                     /dev/vgtest1/lvol1
   LV Status                   available/syncd
   LV Size (Mbytes)            400
   Current LE                  50
   Allocated PE                50
```

```
Used PV                     1

--- Physical volumes ---
PV Name                     /dev/dsk/c4t13d0
PV Status                   available
Total PE                    8749
Free PE                     8693
Autoswitch                  On
```

Miscellaneous

Commands enhanced to prevent misconfigurations through alternate links

In HP-UX 11i v3 LVM commands have been enhanced to avoid overwriting LVM configuration data on physical volumes which already belong to a volume group and are recorded in the /etc/lvmtab.

In prior releases, *pvcreate* with *-f* (force) option on an alternate physical volume path which was not present in /etc/lvmtab succeeded even though the corresponding physical volume belonged to a volume group recorded in /etc/lvmtab. This behavior has been changed with this release so that *pvcreate* fails in such situations.

In addition to *pvcreate* the following commands have been enhanced similarly:
vgcreate, pvremove, vgimport and *vgchgid.*

Example

As an example, consider a multi-pathed physical volume whose DSFs are **`/dev/dsk/c9t1d0 and /dev/dsk/c6t1d0.`** Let the volume group be configured with only **`/dev/dsk/c9t1d0`**.

`'pvcreate -f /dev/rdsk/c6t1d0'` succeeds on prior releases to HP-UX 11i v3. But on HP-UX 11i v3 *pvcreate* fails and prints the following:

```
"pvcreate: The physical volume "/dev/dsk/c9t1d0" is already present in the
"/etc/lvmtab" file and represents an alternate link to "/dev/dsk/c6t1d0"."
```

Mirror Disk Installation No Longer Requires a Reboot

With HP-UX 11i v3 release, the Mirror Disk product installation is much easier. As in previous releases Mirror Disk product installation does not require a reboot.

Glossary

Agile Addressing

The ability to address a LUN with the same device special file regardless of the physical location of the LUN or the number of paths leading to it. In other words, the device special file for a LUN remains the same even if the LUN is moved from one HBA to another, moved from one switch/hub port to another, presented via a different target port to the host, or configured with multiple hardware paths. Also referred to as **persistent LUN binding**.

Agile View

The representation of LUNs using lunpath hardware paths, LUN hardware paths, and persistent DSFs, introduced in HP-UX 11i v3.

DSF

Device Special File. A file associated with an I/O device. DSFs are read and written the same as ordinary files, but requests to read or write are sent to the associated device.

Hardware Path

A series of numbers representing the physical or virtualized location of a device. The path is a sequence of I/O addresses that share a hierarchical relationship. The address elements may not correspond to physical hardware addresses, and may represent only a "handle" to a device rather than a physical path to it.

Legacy View

The representation of legacy hardware paths and legacy DSFs, as in releases prior to HP-UX 11i v3.

Legacy Hardware Path

The representation of a hardware path as it exists in releases prior to HP-UX 11i v3. It is composed of a series of bus-nexus addresses separated by '/' leading to the HBA; beneath the HBA, additional address elements (such as domain, area, port, target, and LUN) are separated by '.'. The string "0/2/1/0.1.4.0.0.2.7" is an example of a legacy hardware path.

Legacy DSF

A DSF with the hardware path information such as SCSI bus, target, and LUN embedded in the file's minor name and file name, such as `/dev/dsk/c2t3d4`.

LUN

A SCSI logical unit. This refers to an end storage device such as a disk, tape, floppy, or CD. This is the logical unit itself and does not represent the path to the logical unit.

Lunpath

The physical hardware path leading to a SCSI logical unit. A SCSI LUN can have more than one lunpath.

Persistent DSF

A DSF conforming to the naming model introduced in HP-UX 11i v3 to support agile addressing. The device file name contains an instance number, such as `/dev/disk/disk#`, and the minor number has no hardware path information.

For More Information

To learn more about some of the LVM features, see the following document on HP documentation website:
http://docs.hp.com (Use search with the given name of the whitepaper)
http://www.docs.hp.com/en/oshpux11iv3#LVM%20Volume%20Manager

- SLVM Online Volume Re-configuration
- LVM Online Disk Replacement
- When Good Disks Go Bad: Dealing with Disk Failures under LVM
- LVM Volume Group Dynamic LUN expansion (DLE)/vgmodify (Available May 2007)
- LVM Volume Group Quiesce/Resume

To learn more about configuring LVM and migration of LVM volume group configuration from legacy to agile naming model, see the following document on HP documentation website:
http://docs.hp.com (Use search with the given name of the whitepaper)
http://www.docs.hp.com/en/oshpux11iv3#LVM%20Volume%20Manager

- LVM Migration from legacy to persistent naming model
- HP-UX System Administrator's Guide Logical Volume Management

To learn more about the agile view and the new mass storage stack, see the following document on HP documentation website: http://docs.hp.com/en/netsys.html#Storage%20Area%20Management

- The Next Generation Mass Storage Stack

To learn more about supported node and host name sizes on HP-UX, see the following document on HP documentation website: http://www.docs.hp.com/en/oshpux11iv3#White%20Papers

- Node and Host Name Sizes on HP-UX

Call to Action

HP welcomes your input. Please give us comments about this whitepaper, or suggestions through our technical documentation feedback website: http://docs.hp.com/en/feedback.html.