# HP-UX LVM Supported Limits

Table of Contents

# Abstract

Starting with HP-UX 11i v2 (11.23), the supported limits of LVM volume groups has gradually increased. In HP-UX 11i v3 (11 .31), LVM delivers significant performance, scalability, usability, and availability enhancements. After the initial release of HP-UX 11iv3 in February 2007, three updates occurred in September 2007, March 2008, and September 2008. This white paper summarizes the LVM supported limits for different volume group versions supported on HP-UX releases.

This document is intended for system administrators, operators, and customers who are creating or managing volume groups. You must also have a basic knowledge of LVM.

# Introduction

Beginning with the September 2008 update to HP-UX 11i v3 (11.31), LVM supports three different versions of volume groups:

- Version 2.1 volume group (supported on 11iv3 September 2008 update and later)
- Version 2.0 volume group (supported on 11iv3 March 2008 update and later)
- Version 1.0 volume group (legacy volume group version, supported on all HP-UX releases and updates

This paper is structured into two parts: the original Version1.0 volume group and the Version2.0 and 2.1 volume groups. For information about the differences and how to use Version 2.*x* volume groups, see the "LVM Version 2.0 volume groups in HP-UX 11i v3" white paper.

# Supported Limits for Version 1.0 Volume Groups

The following table summarizes the supported limits in LVM for HP-UX 11.00 and later releases. Unless noted otherwise, the values are the same across releases.

| **Parameter** | **Operation to change the value** | **Min Value** | **Default Value** | **Max Value** |
|---|---|---|---|---|
| Max number of Version 1.0 VGs per system<br>Note: In 11iv3, tunable was removed. | Kernel tunable:<br>*kctune* maxvgs='value' | 0 | 10 | 256 |
| Max LVs Per VG[1] | *vgcreate* –l 'max_lv' | 1 | 255 | 255 |
| Max PVs per VG[1] | *vgcreate* –p 'max_pv' | 1 | 16 | 255 |
| Max Extents per PV[1]<br>If you specify a value greater than 1016, the default value might be adjusted to the first PV used for creating the VG (Disk space/ Extent size). | *vgcreate* –e 'max_pe' | 1 | 1016 | 65535 |
| Extent Size | *vgcreate* –s 'pe_size' | 1 MB | 4 MB | 256 MB |
| Effective size of a PV[1]<br>Following or superseding patches are required for supporting disks larger than 256GB:<br>11.00 –PHKL_30553<br>11.11 – PHKL_30622<br>11.23 – PHKL_31500 | *pvcreate* –s 'disk_size' | Extent Size | LUN Capacity | 2 TB |
| Size of a LV 11.11 & 11.23 | *lvcreate/lvextend* –l 'le_number' \| –L 'lv_size' | 0 | 0 | 2 TB<br>**[I/Os beyond 2TB will be rereturned with error]** |
| Size of a LV 11.23 with the following or superseding patches:- PHCO_36744, PHKL_36745, PHCO_37939 | | | | 16TB |
| Size of a LV 11.31 | | | | 16 TB |
| Mirror copies per LV | *lvcreate/lvextend* –m 'mirror_copies' | 0 | 0 | 2 |
| Stripes of LV | *lvcreate* –i 'stripes' | 2 | None – Must specify | Max PVs per VG |
| Stripe size of LV | *lvcreate* –I 'stripe_size' | 4KB | 8KB | 32768KB |

[1] With 11.31, the *vgmodify* command enables you to change these values on an existing VG. For more information, see the *vgmodify*(1M) man page and release notes.

## Calculating an Optimal Extent Size for a Version 1.0 Volume Group

Sometimes when creating a Version 1.0 volume group (VG), the *vgcreate* command might abort with a message indicating that the extent size is too small (too big error or a more informative error (with newer patches) explaining that the VGRA is too big). In this situation, you must manually increase the extent size and re-issue the *vgcreate* command.

Increasing the extent size increases the data area marked stale when a write to a mirrored logical volume (LV) fails, and can increase the time required for resynchronizing the stale data. Also, more space than intended might be allocated to the LV because the space is allocated in units of extent size. Therefore, the optimal extent size is the smallest value that can be used to successfully create the volume group with the given configuration parameters.

The minimum extent size for a VG is calculated using the maximum number of logical volumes (MAXLVs) and physical volumes (MAXPVs) in the VG and the maximum number of physical extents (MAXPXs) per each physical volume (PV).

For a VG with bootable PVs, the metadata must fit within 768 kilobytes. Therefore, a *vgcreate* command with a set of values for MAXLVs, MAXPVs, and MAXPXs that succeedson a VG without bootable PVs might fail on a VG with bootable PVs. In this situation, if you must add a bootable PV to a VG, recreate the VG by giving smaller values for these arguments. By far the biggest factor in the size of the metadata is the values for MAXPVs and MAXPXs. Alternatively, convert the bootable PV to a normal PV by rerunning *pvcreate* on that PV without the '–B' option and then add it to the VG. For a PV that is included in a VG, you can use *vgmodify* to change a PV from bootable to a normal PV.

## Sample shell script

The following shell script creates and compiles a small program that gives the minimum extent size for a given VG:

```
#!/usr/bin/sh
cat << EOF > vgrasize.c
#include <stdio.h>

#define BS 1024 /* Device block Size */
#define roundup(val, rnd) (((val + rnd - 1) / rnd) * rnd)

main(int argc, char *argv[])
{
	int i, length, lvs, pvs, pxs;

	if (argc != 4) {

		/* Usage example:
		 *	Maximum LVs in the VG = 255,
		 *	Maximum PVs in the VG = 16
		 *	Maximum extents per PV = 2500 :
		 *
		 *	$ vgrasize 255 16 2500
		 */
		printf("USAGE: %s <MAXLVs> <MAXPVs> <MAXPXs>\n", argv[0]);
		exit(1);
	}
	lvs = atoi(argv[1]); pvs = atoi(argv[2]); pxs = atoi(argv[3]);
```

```
        length = 16 + 2 * roundup(2 +
                                (roundup(36 + ((3 * roundup(pvs, 32)) / 8) +
                                        (roundup(pxs, 8) / 8) * pvs, BS)  +
                                roundup(16 * lvs, BS) +
                                roundup(16 + 4 * pxs, BS) * pvs) / BS, 8);

        if (length > 768) {
        printf("Warning: A bootable PV cannot be added to a VG \n"
                "created with the specified argument values. \n"
                "The metadata size %d Kbytes must be less \n"
                "than 768 Kbytes.\n"
                "If you do not want a boot disk in this \n"
                "VG, do not use the '-B' option during pvcreate(1M) \n"
                "for the PVs to be part of this VG. \n", length);
        }

        length = roundup(length, 1024) / 1024;

        if (length > 256 ) {
                printf("Cannot configure a VG with the maximum values"
                        " for LVs, PVs and PXs\n");
                exit(1);
        }

        for (i = 1; i < length ; i = i << 1) { }

        printf("\nMinimum extent size for this configuration = %d MB\n", i);
}
EOF
make vgrasize
```

The following example shows the usage of the executable created from the previous script.

**# vgrasize 255 16 2550**

Minimum extent size for this configuration = 1 MB

# Supported Limits for Version 2.0 and 2.1 Volume Groups

The following table summarizes the supported limits in LVM for Version 2.0 and 2.1 VGs. Unless noted otherwise, the values are the same across releases. Note: Creation of Version 2.*x* VGs and the inherent limitations are very different from Version1.0 VGs. For more information, see the "LVM Version 2.0 volume groups in HP-UX 11i v3" white paper.

| Parameter | Operation to change the value | Version 2.0 VG | | | Version 2.1 VG | | |
|---|---|---|---|---|---|---|---|
| | | Min Value | Default Value | Max Value | Min Value | Default Value | Max Value |
| Max number of VGs per system | None – Not needed | 0 | n/a | 512[2] | 0 | n/a | 2048[2] |
| Max LVs per VG | None – Not needed | 511 | 511 | 511 | 2047 | 2047 | 2047 |
| Max PVs per VGs | None – Not needed | 511 | 511 | 511 | 2047 | 2047 | 2047 |
| Max VG size | vgcreate –S 'max_vgsize' | 1MB[3] | None – Must specify | 2PB | 1MB[3] | None – Must specify | 2PB |
| Max size of a PV | | Extent size | LUN capacity | 16TB | Extent size | LUN capacity | 16TB |
| Max extents per PV | None – not needed | 1 | LUN capacity/ extent size | 16777216 | 1 | LUN capacity/ extent size | 16777216 |
| Extent size | vgcreate –s 'pe_size' | 1MB | None – Must specify | 256MB | 1MB | None – Must specify | 256MB |
| Size of a LV | lvcreate/lvexted – L 'lv_size' | 0 | 0 | 256TB | 0 | 0 | 256TB |
| Max extents per LV | lvcreate/lvextend –L 'le_number' | 0 | 0 | 33554432 | 0 | 0 | 33554432 |
| Mirror copies per LV | lvcreate/lvextend –m 'mirror_copies' | 0 | 0 | 5 | 0 | 0 | 5 |
| Stripes of LV | lvcreate –i 'stripes' | 2 | None – Must specify | Max PVs per VG (511) | 2 | None – Must specify | 511 |
| Stripe size of LV | lvcreate –I 'stripe_size' | 4KB | None – Must specify | 262144KB | 4KB | None – Must specify | 262144KB |

[2] The maximum combined limit of 2.0 and 2.1 volume group is 2048.
[3] If the disk is larger than the max VG size specified, *vgcreate* adjusts the minimum VG size to match the disk size.

## Calculating an Optimal Size for a Version 2.0 and 2.1 Volume Group

Like Version 1.0 VGs, there is a relationship between extent size and maximum VG size. There is also a limitation on the number of extents an individual VG can contain. To determine the proper size, the *vgcreate* command has a new –E option. This option displays the maximum VG size based on the given physical extent size or minimum physical extent size based on the max VG size.

**Example**

After you know the VG size you want to provision for, use *vgcreate* with the –E option to determine the minimum extent size required to achieve it.

What is the minimum extent size to provision a 2.0 VG for 1 PB?

```
# vgcreate -V 2.0 -E -S 1p
Max_VG_size=1p:extent_size=32m
```

What is the maximum 2.0 volume group size with an extent size of 16MB?

```
# vgcreate -V 2.0 -E -s 16
Max_VG_size=512t:extent_size=16m
```

What is the minimum extent size to provision a 2.1 VG for 2 PB?

```
# vgcreate -V 2.1 -E -S 2p
Max_VG_size=2p:extent_size=64m
```

What is the maximum 2.1 volume group size with an extent size of 32MB?

```
# vgcreate -V 2.1 -E -s 32
Max_VG_size=1p:extent_size=32m
```

# Determining LVM's Maximum Limits on a System

The March 2008 update to HP-UX 11i v3 (11.31) introduced a new command that enables the system administrator to determine the maximum LVM limits supported on the target system for a given volume group version. The *lvmadm* command displays the implemented limits for Version 1.0 and Version 2.*x* VGs. It is impossible to create a VG that exceeds these limits.

**Example**

```
# lvmadm -t

--- LVM Limits ---
VG Version                  1.0
Max VG Size (Tbytes)        510
Max LV Size (Tbytes)        16
Max PV Size (Tbytes)        2
Max VGs                     256
Max LVs                     255
Max PVs                     255
Max Mirrors                 2
Max Stripes                 255
Max Stripe Size (Kbytes)    32768
Max LXs per LV              65535
Max PXs per PV              65535
Max Extent Size (Mbytes)    256

VG Version                  2.0
Max VG Size (Tbytes)        2048
Max LV Size (Tbytes)        256
Max PV Size (Tbytes)        16
Max VGs                     512
Max LVs                     511
Max PVs                     511
Max Mirrors                 5
Max Stripes                 511
Max Stripe Size (Kbytes)    262144
Max LXs per LV              33554432
Max PXs per PV              16777216
Max Extent Size (Mbytes)    256

VG Version                  2.1
Max VG Size (Tbytes)        2048
Max LV Size (Tbytes)        256
Max PV Size (Tbytes)        16
Max VGs                     2048
Max LVs                     2047
Max PVs                     2048
Max Mirrors                 5
Max Stripes                 511
Max Stripe Size (Kbytes)    262144
Max LXs per LV              33554432
Max PXs per PV              16777216
Max Extent Size (Mbytes)    256
```

# Glossary

**KB**

A kilobyte unit of information equal to $2^{10}$ or 1024 bytes

**MB**

A megabyte unit of information equal to $2^{20}$ or 1,048,576 bytes

**GB**

A gigabyte unit of information equal to $2^{30}$ or 1,073,741,824 bytes

**TB**

A terabyte unit of information equal to $2^{40}$ or 1,099,511,627,776 bytes

**PB**

A petabyte unit of information equal to $2^{50}$ or 1,125,899,906,842,624 bytes

# For More Information

To learn more about some of the LVM features, see the following documents on the 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 Version 2.0 Volume Groups in HP-UX 11i v3
- SLVM Single-Node Online Reconfiguration (SLVM SNOR)
- LVM Online Disk Replacement (LVM OLR)
- When Good Disks Go Bad: Dealing with Disk Failures under LVM
- LVM Volume Group Dynamic LUN expansion (DLE)/vgmodify
- LVM Volume Group Quiesce/Resume
- LVM New Features in HP-UX 11i v3

To learn more about configuring LVM and migration of LVM VG configuration from legacy to agile naming model, see the following documents 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 Agile Naming Model
- HP-UX System Administrator's Guide: Logical Volume Management
- HP-UX LVM Performance Assessment ( The whitepaper will be available soon)

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

# 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.