LVM Supported Limits

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

Parameter	Operation to change the value	Min Value	Default Value	Max Value
Max VGs per system Note, in 11.31, tunable was removed	Kernel tunable 'maxvgs'	0	16	256
Max LVs Per VG[1]	vgcreate(1M) -l 'max_lv'	1	255	255
Max PVs per VG[1]	vgcreate(1M) –p 'max_pv'	1	16	255
Max Extents per PV[1] The default value may get adjusted to (Disk space/ Extent size) of the first PV used for creating the VG, if that value is greater than 1016.	vgcreate(1M) –e 'max_pe'	1	1016	65535
Extent Size	vgcreate(1M) –s 'pe_size'	1 Mbytes	4 Mbytes	256 Mbytes
Effective size of a PV[1] Following patches are required for supporting disks larger than 256Gb: 11.00 – Patch superseding PHKL_30553 11.11 – PHKL_30622 11.23 -- PHKL_31500	pvcreate(1M) –s 'disk_size'	Extent Size	Disk Capacity	2 Tbytes
Size of a LV 11.11 & 11.23	lvcreate(1M)/lvextend(1M) -l 'le_number' | -L 'lv_size'	0	0	2 TBytes **[I/Os beyond 2TB will be rejected]**
Size of a LV 11.31				16 Tbytes
Mirror copies per LV	lvcreate(1M)/lvextend(1M) -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'	4k	8k	32768 Kbytes

Note 1: With 11.31, the new vgmodify(1M) command enables changing these values on an existing VG. Please refer to the man page and release notes for further details.

Calculating an optimal extent size for a volume group.

Sometimes when creating a volume group (VG), the *vgcreate(1M)* command may abort with a message that extent size is too small (too big error or with newer patches a more informative error explaining that the VGRA is too big). In this situation the user is expected to increase the extent size and re-issue the *vgcreate(1M)* command.

Increasing the extent size increases the data area marked stale when a write to a mirrored logical volume fails and that can increase the time required for re-synchronizing the stale data. Also, more space than intended may be allocated to the logical volume since 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 volume group is calculated using the maximum number of, logical volumes MAXLVs and physical volumes (MAXPVs) in the volume group and the maximum number of physical extents (MAXPXs) per each physical volume.

For a VG with bootable PVs, the metadata must fit within 768 Kbytes. Therefore, a *vgcreate(1M)* command with a set of values for MAXLVs, MAXPVs and MAXPXs that succeed on a VG without bootable PVs, may fail on a VG with bootable PVs. In this situation, if the user needs to add a bootable PV to a VG, they must recreate the VG by giving lesser values for these arguments. By far the biggest factor in the size of the metadata is the values for MAXPVs and MAXPXs. Alternatively, they can convert the bootable PV to a normal PV by rerunning *pvcreate(1M)* on that PV without '-B' option and then add it to the VG.

The following shell script will create and compile a small program that gives the minimum extent size for a given volume group:

```
#!/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 :
		 *
		 *	$ esize 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 the intention is not to have a boot disk in this \n"
		"VG then do not use '-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 this 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
```