Chapter 10:
Managing LVM Logical Volumes

Learning objectives

Upon completing this chapter, the learner should be able to:

- Understand LVM and its features that are not available in normal partitioning
- Understand the concept of snapshots
- Create partitions that can be used in an LVM configuration
- Create physical volumes, volume groups, and logical volumes to create an LVM infrastructure
- Learn how to resize volume groups and logical volumes

Key terms

Logical Volume Management (LVM)

LVM snapshot

Physical volume (PV)

Volume group (VG)

Logical volume (LV)

Physical extents (PEs)

Logical extents (LEs)

Device mapper

Table of content

1. Understanding LVM

In the early days of Linux servers, storage was handled by creating partitions on disks. Even if this approach does work, there are some disadvantages, the most important of which is that disks are inflexible. That is why the Logical Volume Manager (LVM) was introduced. Where it is not possible to dynamically grow a partition that is running out of disk space, this is possible when working with LVM. LVM offers many other advantages, which will be discussed later in this chapter.

1.1. LVM Architecture

In the LVM architecture, several layers can be distinguished. On the lowest layer, the storage devices are used. These can be any storage devices, such as complete disks, partitions, logical units (LUNs) on a storage-area network (SAN) and whatever else is made possible in modern storage topologies.

The storage devices need to be flagged as physical volumes (PVs), which makes them usable in an LVM environment, and other utilities which trying to access to logical volume. A storage device that is a physical volume can be added to the volume groups (VGs), which is the abstraction of all available storage. That means: the volume group is not something that is fixed, but that it can be resized when needed, which makes it possible to add more space on the volume group level when volumes are running out of disk space.

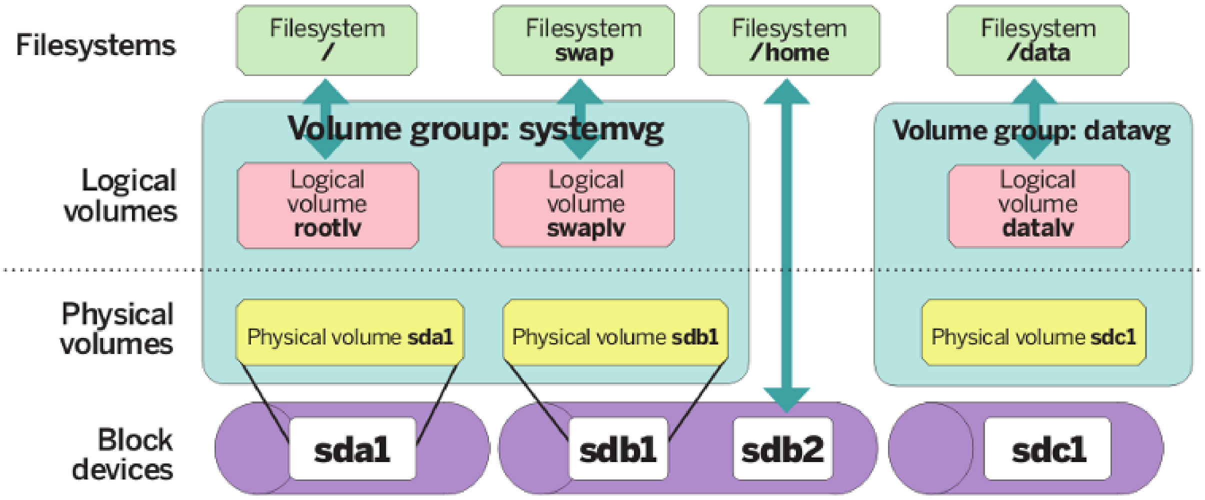
The idea is simple: If you are running out of disk space on a logical volume, you take available disk space from the volume group. And if there is no available disk space in the volume group, you just add it by adding a physical volume.

On top of the volume group are the logical volumes (LVs), they do not act on disks directly but get their disk space from available disk space in the volume group. That means: a logical volume may consist of available storage from multiple physical volumes, which does not really matter (as long as some level of redundancy is used on the disk layer).

It is always a good idea to avoid logical volumes from spanning multiple physical volumes, if one of the physical volumes breaks, all files on the LVM file system will become inaccessible.

The actual file systems are created on the logical volumes. As the logical volumes are flexible with regard to size, that makes the file systems flexible as well. If a file system is running out of disk space, it is relatively easy to extend the file system, or to reduce it if the file system allows that.

To brief it, LVM is a form of storage virtualization that offers system administrators a more flexible approach to managing disk storage space than traditional partitioning. This type of virtualization tool is located within the device-driver stack on the operating system. It works by chunking the physical volumes into physical extents (PEs). The PEs are mapped onto logical extents (LEs) which are then pooled into volume groups. These groups are linked together into logical volumes that act as virtual disk partitions and that can be managed by using LVM.



1.2. LVM Features

There are several reasons why LVM is great. The most important reason is that LVM offers a flexible solution for managing storage. Volumes are no longer bound to the restrictions of physical hard drives. If additional storage space is needed, the volume group can easily be extended so that disk space can be added to the logical volumes. It is also possible to reduce the size of a logical volume, but only if the file system that was created on that volume supports resizing. This is the case for the Ext4 file system but not for the XFS file system, which is used as the default file system on RHEL7.

Another important reason why administrators like using LVM is the support for snapshots. A snapshot keeps the current state of a logical volume and can be used to revert to a previous situation or to make a backup of the file system on the logical volume if the volume is open.

LVM snapshots are created by copying the logical volume administrative data (the metadata) that describe the current state of files to a snapshot volume. As long as nothing changes, from the LVM snapshot metadata the original blocks in the original volume are addressed. When blocks are modified, the blocks containing the previous state of the file are copied over to the snapshot volume, which for that reason will grow. Using this method ensures that by accessing an LVM snapshot volume the exact state of the files as they were when the snapshot was created can be accessed.

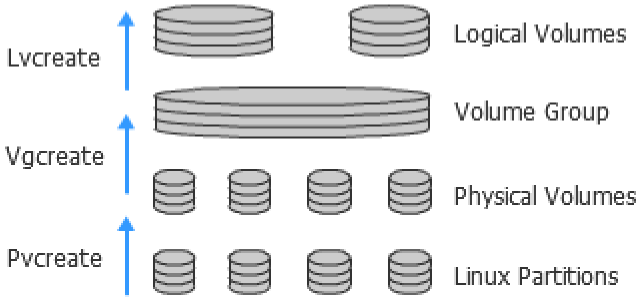
A third important advantage of using LVM logical volumes is the option to replace failing hardware easily. If a hard disk is failing, data can be moved within the volume group (through the pvmove command), the failing disk can then be removed from the volume group, and a new hard disk can be added dynamically, without requiring any downtime for the logical volume itself.

2. Creating LVM structure

Creating LVM logical volumes involves creating the three layers in the LVM architecture:

1. You first have to take care of the physical volume.
2. then you need to create the volume group and assign physical volumes to it.
3. As the last step, the logical volume itself has to be created.

In this section, you learn what is involved in creating these three layers, different utilities exist for creating LVM. This chapter focuses on using the command-line utilities. They are relatively easy to use, and they are available in all environments (whether you are running a graphical interface or not).



You absolutely do not need to learn the commands discussed in this chapter by heart. All you really need to remember is pv, vg and lv. Open a command line, type pv and press the Tab key twice. This will show all commands that start with pv, which are all commands that are used for managing physical volumes. After you have found the command you need, run this command with the --help option to show a usage summary that lists everything that needs to be done to create the element you need.

Listing below shows an example of the pvcreate --help command.

```
[root@server1 ~]# pvcreate --help
pvcreate: Initialize physical volume(s) for use by LVM
pvcreate
[--norestorefile]
[--restorefile file]
[-d|--debug]
[-f[f]|--force [--force]]
[-h|-?|--help]
[--labelsector sector]
[-M|--metadatatype 1|2]
[--pvmetadatacopies #copies]
[--bootloaderareasize
BootLoaderAreaSize[bBsSkKmMgGtTpPeE]]
[--metadatasize MetadataSize[bBsSkKmMgGtTpPeE]]
[--dataalignment Alignment[bBsSkKmMgGtTpPeE]]
[--dataalignmentoffset AlignmentOffset[bBsSkKmMgGtTpPeE]]
[--setphysicalvolumesize
PhysicalVolumeSize[bBsSkKmMgGtTpPeE]
[-t|--test]
[-u|--uuid uuid]
[-v|--verbose]
[-y|--yes]
[-Z|--zero {y|n}]
[--version]
PhysicalVolume [PhysicalVolume...]
```

2.1. Creating the physical volumes

Before the LVM tools can be used to create physical volumes, you need to create a partition marked as the LVM partition type. This is basically the same procedure of creating partitions, with the only difference that before writing changes to disk in fdisk or gdisk, you need to press t to change the partition type. If you are using an MBR disk, the partition type is 8e, and if you are using a GUID disk, use the partition type 8300.

After creating the partition and flagging it as an LVM partition type, you need to use pvcreate to mark it as a physical volume. This writes some metadata to the partition, which allows it to be used in a volume group.

Exercise: Creating the physical volume

In this exercise, you create a physical volume. To do this exercise, you need a hard disk that has free (unpartitioned) disk space available. The recommended method to make disk space available is by adding a new hard disk in your virtual machine environment.

In this exercise, I use a clean /dev/sdb device to create the partition. You may have to change the device name to match your configuration.

1. Open a root shell and type: fdisk –l to get the name of the new added hard disk (/dev/sdb), then fdisk /dev/sdb.
2. Type n to create a new partition. Select p to make it a primary partition, and use the partition number that is suggested as a default. If you are using a clean device, this will be partition number 1.
3. Press Enter when asked for the first sector and type +100M to accept the last sector.
4. Once you are back on the fdisk prompt, type t to change the partition type. Because there is one partition only, fdisk does not ask which partition to use this partition type on. You may have to select a partition if you are using a different configuration.
5. The partitioner asks for the partition type you want to use. Type 8e. Then, press w to write changes to disk and quit fdisk.

6. Now that the partition has been created, you need to flag it as an LVM physical volume. To do this, type pvcreate /dev/sdb1. You should now get this prompt: Physical volume "/dev/sbd1" successfully created.
7. Now type pvs to verify that the physical volume has been created successfully.

As an alternative to the pvs command, which shows a summary of the physical volumes and their attributes, you can also use the pvdisplay command to show some more details. Listing shows an example of the output of this command.

```
[root@server1 ~]# pvdisplay
--- Physical volume ---
PV Name /dev/sda2
VG Name centos
PV Size 3.51 GiB / not usable 3.00 MiB
Allocatable yes (but full)
PE Size 4.00 MiB
Total PE 898
Free PE 0
Allocated PE 898
PV UUID CILii7-DzOd-w4L0-yOxi-9NXg-D3nP-ZugJIj
```

If you want a very synthetic overview of the current storage configuration, you might also like the lsblk command. As can be seen in Listing below this command gives a hierarchical overview of which disks and partitions are used in what LVM volume groups and logical volumes.

```
[root@localhost ~]# lsblk
NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
fd0 2:0 1 4K 0 disk
sda 8:0 0 8G 0 disk
├─sda1 8:1 0 200M 0 part /boot
├─sda2 8:2 0 6.9G 0 part
│ ├─centos-swap 253:0 0 256M 0 lvm [SWAP]
│ └─centos-root 253:1 0 5.9G 0 lvm /
├─sda3 8:3 0 100M 0 part
└─sda4 8:4 0 887M 0 part
└─vgik-lvgroups 253:2 0 440M 0 lvm /groups
sr0 11:0 1 1024M 0 rom
```

2.2. Creating the volume groups

Now that the physical volume has been created, you can assign it to a volume group. It is possible to add a physical volume to an existing volume group, but you will now learn how to create a new volume group and add the physical volume to it. This is a simple one–command procedure.

Just type vgcreate followed by the name of the volume group you want to create and the name of the physical device you want to add to it. So, if the physical volume name is /dev/sdb1, the complete command is vgcreate vgdata /dev/sdb1.

You are completely free in your choice of name for the volume group. I like to start all volume group names with vg, which makes it easy to find the volume groups if there are many, but you are free to choose anything you like.

In this procedure, you learned how to create a volume group in a two-step procedure where first the physical volume is created with the pvcreate command, after which the volume group is added using the vgcreate command. You can do this in a one-step procedure as well (where using a separate pvcreate command will not be necessary). If you are adding a partition to the volume group, however, it must be marked as partition type 8e already.

The one-step procedure is particularly useful for adding a complete disk device (which does not need to be marked as anything). If you want to add the disk /dev/sdc, for instance, just type vgcreate vgdata /dev/sdc to create a volume group vgdata that contains the /dev/sdc device. When you are doing this to add a device that has not been marked as a physical volume yet, the vgcreate utility will automatically flag it as a physical volume.

When creating volume groups, a physical extent (PE) size is used. The physical extent size defines the size of the building blocks used to create logical volumes. A logical volume always has a size that is a multiple of the physical extent size. If you need to create huge logical volumes, it is more efficient to use a big physical extent size. If you do not specify anything, a default extent size of 4.00 MiB is used. The physical extent size is always specified as a multiple of 2 MiB, with a maximum size of 128 MiB. Use the vgcreate -s option to specify the physical extent size you want to use.

When working with an ext4 file system, logical extents are used. The extent size on LVM are in no way related to the extent sizes that are used on the file systems.

After creating the volume group, you can request details about the volume group using the vgs command for a short summary, or the vgdisplay command to get more information. Listing below shows an example of the output of the vgdisplay command.

```
[root@server1 ~]# vgdisplay
...
--- Volume group ---
VG Name vgdata
System ID
Format lvm2
Metadata Areas 1
Metadata Sequence No 2
VG Access read/write
VG Status resizable
MAX LV 0
Cur LV 1
Open LV 0
Max PV 0
Cur PV 1
Act PV 1
VG Size 248.00 MiB
PE Size 4.00 MiB
Total PE 62
Alloc PE / Size 31 / 124.00 MiB
Free PE / Size 31 / 124.00 MiB
VG UUID TutZ0F-Fe0q-VGR4-y0dO-O0RM-Mv7e-09zyar
```

2.3. Creating the logical volumes and file systems

Now that the volume group has been created, you can start creating logical volumes from it. The tool is lvcreate and this procedure is slightly more complicated than the creation of physical volumes or volume groups because there are more choices to be made. While creating the logical volume, you must specify a volume name and a size.

The volume size can be specified as an absolute value using the -L option. Use, for instance, -L 5G to create an LVM volume with a 5GB size. Alternatively, you can use relative sizes using the -l option. For instance, use -l 50%FREE to use half of all available disk space. You'll further need to specify the name of the volume group that the logical volume is assigned to, and optionally (but highly recommended), you can use -n to specify the name of the logical volume. For instance, use lvcreate -n lvvol1 -L 100M vgdata to create a logical volume with the name lvvol1 and add that to the vgdata volume group.

2.4. Understanding LVM device naming

Now that the logical volume has been created, you can start using it. To do this, you need to know the device name. LVM volume device names can be addressed in multiple ways. The simple method is to address the device as /dev/vgname/lvname. So, if you have created a volume with the name lvdata, which gets its available disk space from the vgdata volume group, the device name would be /dev/vgdata/lvdata.

For naming LVM volumes, another system plays a role: device mapper, abbreviated as dm, is a generic interface that the Linux kernel uses to address storage devices. Device mapper is used by multiple device types, such as LVM volumes, but also by software RAID and advanced network devices such as multipath devices. These devices are created in two locations:

- as devices that are sequentially numbered in the /dev directory, such as /dev/dm-0, /dev/dm-1, and further
- Because these device names do not provide any information about the device and therefore are confusing, symbolic links are created in the /dev/mapper directory

These symbolic links use a name that uses the vgname-lvname pattern. So, the device /dev/vgdata/lvdata would also be known as /dev/mapper/vgdata-lvdata. When working with LVM logical volumes, you can use any of these device names. Listing shows an overview of the different LVM device names as provided by the device mapper.

```
[root@localhost ~]# \ls -l /dev/vgdata/lvdata
lrwxrwxrwx. 1 root root 7 Feb 11 05:00 /dev/vgdata/lvdata
-> ../dm-2
[root@localhost ~]# \ls -l /dev/mapper/
total 0
lrwxrwxrwx. 1 root root 7 Feb 11 03:50 centos-root ->
../dm-1
lrwxrwxrwx. 1 root root 7 Feb 11 03:50 centos-swap ->
../dm-0
crw-------. 1 root root 10, 236 Feb 11 03:50 control
lrwxrwxrwx. 1 root root 7 Feb 11 05:00 vgdata-lvdata ->
../dm-2
[root@localhost ~]# \ls -l /dev/dm-2
brw-rw----. 1 root disk 253, 2 Feb 11 05:00 /dev/dm-2
```

Table below summarizes the relevant commands for creating logical volumes.

Command	Explanation
pvcreate	Creates physical volumes
pvs	Shows a summary of available physical volumes
pvdisplay	Shows a list of physical volumes and their properties
vgcreate	Creates volume groups
vgs	Shows a summary of available volume groups
vgdisplay	Shows a detailed list of volume groups and their properties
lvcreate	Creates logical volumes
lvs	Shows a summary of all available logical volumes
lvdisplay	Shows a detailed list of available logical volumes and their properties

Exercise: Creating the volume group and logical volumes

In the previous Exercise, you created a physical volume. In this exercise, you continue working on that physical volume and assign it to a volume group. Then you add a logical volume from that volume group.

1. Open a root shell and type pvs to verify the availability of physical volumes on your machine. You should see the /dev/sdb1 physical volume that was created previously.
2. Type vgcreate vgdata /dev/sdb1. This will create the volume group with the physical volume assigned to it.
3. Type vgs to verify that the volume group was created successfully. Also type pvs. Notice that this command now shows the name of the physical volumes, with the names of the volume groups they are assigned to.
4. Type lvcreate –n lvdata –l 50%FREE vgdata. This creates an LVM logical volume with the name lvdata, which will use 50% of available disk space in the vgdata volume group.
5. Type lvs to verify that the volume was added successfully.
6. At this point, you are ready to create a file system on top of the logical volume. Type mkfs.xfs /dev/vgdata/lvdata to create the file system.
7. Type mkdir /files to create a folder on which the volume can be mounted.
8. Add the following line to /etc/fstab:
9. /dev/vgdata/lvdata /files xfs defaults 1 2
10. Type mount –a to verify that the mount works and mount the file system.

3. Resizing LVM structure

One of the major benefits of using LVM is that LVM volumes are easy to resize, which is very useful if your file system is running out of available disk space. If the XFS file system is used, a volume can be increased, but not decreased, in size. Other file systems such as Ext4 and Btrfs support decreasing of the file system size also.

Decreasing an Ext4 file system can be done offline only, which means that you need to unmount it before you can resize it. In this section, you learn how to increase the size of an LVM logical volume.

3.1. Resizing volume groups

The main part of LVM flexibility sits in the fact that it is so easy to resize the volume groups and the logical volumes that are using disk space from the volume group. The vgextend command is used to add storage to a volume group, and the vgreduce command is used to take physical volumes out of a volume group (which can lead to some additional complications). This procedure is relatively easy:

- Make sure that a physical volume or device is available to be added to the volume group
- Use vgextend to extend the volume group. The new disk space will show immediately in the volume group

After extending a volume group, you can use the vgs command to verify that a physical volume has been added to the volume group. In the following Listing, you can see that the centos VG contains two physical volumes, as indicated in the #PV column.

```
[root@server2 ~]# vgs
VG #PV #LV #SN Attr VSize VFree
centos 2 3 0 wz--n- 7.00g 560.00m
vgsan 1 2 0 wz--n- 496.00m 96.00m
```

3.2. Resizing logical volumes and file systems

Like volume groups can be extended with the vgextend command, logical volumes can be extended with the lvextend command. This command has a very useful option to take care of extending the file systems on the logical volume at the same time, it is recommended to use this option and not the alternative approach where logical volumes and the file systems on top of the logical volumes are extended separately.

When resizing a logical volume with the file system it contains, nothing will happen to the file system, and its data will remain intact. Most file system resizing operations can even be done online, without any need to unmount the file system.

To grow the logical volume size, use lvresize, followed by the -r option to resize the file system used on it. Then, specify the size you want the resized volume to be. The easiest and most intuitive way to do that is by using -L followed by a + sign and the amount of disk space you want to add, as in lvresize -L +1G -r /dev/vgdata/lvdata.

An alternative way to resize the logical volume is by using the -l option, this option is followed by the number of extents that are added to the logical volume or by the absolute or relative percentage of extents in the volume group that will be used. You can, for example, use the following commands to resize the logical volume:

- lvresize -r -l 75%VG /dev/vgdata/lvdata, this resizes the logical volume so that it will take 75% of the total disk space in the volume group
- lvresize -r -l +75%VG /dev/vgdata/lvdata, this tries to add 75% of the total size of the volume group to the logical volume. (Notice the difference with the previous command)
- lvresize -r -l +75%FREE /dev/vgdata/lvdata, this adds 75% of all free disk space to the logical volume
- lvresize -r -l 75%FREE /dev/vgdata/lvdata, this resizes the logical volume to a total size that equals 75% of the amount of free disk space. (Notice the difference with the previous command)

While resizing a logical volume, you can also use the –l option, followed by the number of logical extents (LEs) that you want to add or remove. A logical extent is the logical building block used when creating logical volumes, and it maps to a physical extent, the size of which can be specified when creating a volume group. All resize operations need to match complete logical extents. You will sometimes notice that the resize size is rounded up or down to the logical extent size. You can also specify the number of logical extents that need to be added or removed directly by using the –l option with the lvresize command.

Exercise: Creating the volume group and logical volumes

In previous Exercises, you have created a physical volume, volume group, and logical volume. In this exercise, you extend the size of the logical volume and the file system used on top of it. Open a root shell. Type pvs to verify the availability of physical volumes on your machine. You should see the /dev/sdb1 physical volume that was created previously.

1. Type pvs and vgs to show the current physical volume and volume group configuration.
2. Use fdisk to add another partition with a size of 100M. Do not forget to flag this partition with the LVM partition type. I'll assume this new partition is /dev/sdb2 for the rest of this exercise. Replace this name with the name used on your configuration.
3. Type vgextend vgdata /dev/sdb2 to extend vgdata with the total size of the /dev/sdb2 device.
4. Type vgs to verify that the available volume group size has increased.
5. Type lvs to verify the current size of the logical volume lvdata.
6. Type df − h to verify the current size of the file system on lvdata.
7. Type lvextend −r −l +50%FREE /dev/vgdata/lvdata to extend lvdata with 50% of all available disk space in the volume group.
8. Type lvs and df −h again to verify that the added disk space has become available.
9. Type lvreduce −r −L −150M /dev/vgdata/lvdata. This shrinks the lvdata volume with 50MB. Notice that while doing this the volume is temporarily unmounted, which happens automatically.

Quiz

Chapter review questions

1. Which of the following is not a standard component in an LVM setup?
   a. Logical volume
   b. File system
   c. Volume group
   d. Physical volume
2. Which of the following is not an LVM feature?
   a. Volume resizing
   b. Hot replacement of failing disk
   c. Copy on write
   d. Snapshots
3. Which partition type do you need on a GPT partition to mark it with the LVM partition type?
   a. 83
   b. 8e
   c. 8300
   d. 8e00
4. Which of the following commands shows correctly how to create a logical volume that uses 50% of available disk space in the volume group?
   a. vgadd –n lvdata –l+50%FREE vgdata
   b. lvcreate lvdata –l 50%FREE vgdata
   c. lvcreate –n lvdata –l 50%FREE vgdata
   d. lvadd –n lvdata –l 50% FREE /dev/vgdata
5. Which command shows an overview of available physical volumes?
   a. pvshow
   b. pvdisplay
   c. pvs
   d. Pvlist

6. Which of the following statements about physical volumes is/are true?

a. Partitions should be flagged as LVM partition before assigning them as a physical volume.

b. Raw disks can be flagged as a physical volume.

c. Physical volumes can only be created on the Ext4 or XFS file system.

d. Physical volumes cannot be created on LUNs.

7. Which of the following shows correct syntax for creating a volume group?

a. vgcreate /dev/vgdata sda1

b. vgcreate sda1 /dev/vgdata

c. vgcreate vgdata /dev/sda1

d. vgcreate vgdata sda1

8. Which of the following commands would set the extent size that is used when creating a logical volume to a default of 4 MiB?

a. pvcreate –s 4M /dev/sda4

b. vgcreate –s 4M vgdata /dev/sda4

c. lvcreate –s 4M –L 200M –n lvdata vgdata

d. lvcreate –x 4M –L 200M –n lvdata vgdata

9. Which of the following is not a likely correct name for the logical volume lvdata that was created in the volume group vgdata?

a. /dev/vgdata–lvdata

b. /dev/vgdata/lvdata

c. /dev/dm–1

d. /dev/mapper/vgdata–lvdata

10. Which of the following commands resizes the logical volume /dev/vgdata/lvdata as well as the file system that is used on top of it, and adds 1GB to the existing volume size?

a. lvresize –l+1G –r /dev/vgdata/lvdata

b. lvresize –L+1G –r /dev/vgdata/lvdata

c. lvresize –L 1G –r /dev/vgdata/lvdata

d. lvresize –l 1G –f /dev/vgdata/lvdata

11. Which partition type is used on a GUID partition that needs to be used in LVM?

a. 8e

b. dm0

c. Ext4

d. 8300

12. Which command enables you to create a volume group with the name vggroup that contains the physical device /dev/sdb3 and uses a physical extent size of 4MiB?

a. vgcreate -s 4MiB vgdata /dev/sdb3

b. vgcreate -s 4MiB /dev/sdb3 vggroup

c. vgcreate -s 4MiB vggroup /dev/sdb3

d. vgcreate -s 4MiB vggroup /dev/sdb

13. Which command shows a short summary of the physical volumes on your system as well as the volume group to which these belong?

a. pvdisplay

b. lvdisplay

c. pvcreate

d. Lvcreate

14. What do you need to do to add an entire hard disk /dev/sdd to the volume group vgroup?

a. vgextend vggrp /dev/sdb2

b. vgextend vgdata /dev/sdd

c. vgextend vgroup /dev/sdd2

d. vgextend vgroup /dev/sdd

15. Which command enables you to create a logical volume lvvol1 with a size of 6MiB?

a. lvcreate -s lvvol1 -L 6MB lvvol1

b. lvcreate -n lvvol1 -L 6MiB vgdata

c. lvcreate -n lvvol1 -L 6MB lvvol1

d. lvcreate -s 4M -L 200M - n lvdata

16. How do you show which logical volumes are available?

a. lvdisplay

b. vgdisplay

c. vgcreate

d. lvcreate

17. Which command enables you to add 100MB to the logical volume lvvol1, assuming that the disk space is available in the volume group?

a. lvresize +100MB –r /dev/mapper/lvvol1

b. lvresize –L+100MB –r /dev/mapper/lvvol1

c. lvextend –r 100MB lvvol1

d. lvextend –m 100MB lvvol1

18. Which option do you use when using lvextend to make sure that the file system is also resized?

a. L

b. f

c. r

d. l

19. This command shows a summary of available volume groups:

a. vgs

b. vgcreate

c. lvdisplay

d. Lvextend

20. The physical extent size defines the size of the building blocks used to create logical volumes.

a. True

b. False

Answers to chapter Review Questions:

1. b
2. c
3. b
4. c
5. b
6. a
7. c
8. b
9. a
10. a
11. d
12. c
13. a
14. d
15. b
16. a
17. b
18. c
19. a
20. a