Google IT Support Certificate: Course 3: Operating Systems and You

Week 4: Filesystems

Quiz: Partitioning and Formatting a Disk Drive in Linux

Introduction

In this lab, you'll learn how to partition and format a disk drive in Linux. Knowing how to do this is a critical skill to have as an IT Support Specialist. Partitions are important because a file system can't function without one. When you acquire a new disk drive, at least one partition is required in order to be able to write files to the file system. Different partitions can then have different file formats, depending on their purpose. For example, a disk partition that acts as a swap for your main memory may have a different file format than the default user-facing file systems. Partitions, like those used for system recovery, may also have different file formats. This shows you just how important this skill is to every IT Support Specialist out there.

What you'll do

You'll learn how to partition a disk drive into one or more partitions. You'll also learn how to format each of those partitions to a different file format. Your main learning objective for this lab is to practice the partitioning and formatting commands you'll find in this lab in the Linux VM.

Blocks and partitions

Before diving into the details of creating partitions and formatting them, let's kick things off with a review of blocks and partitions.

Blocks

Blocks are a layer of storage devices that allow individual access to each independently. They allow programs to access storage without worrying about whether the underlying hardware device is a hard drive, solid state drive, flash drive, etc.

In Linux, you can view block devices and file systems attached to your system using the **lsblk** command. This command gathers information about all devices attached to the system, and prints them out using a tree-like structure. To view the devices attached to your VM, use the **lsblk** command.

```
lsblk
```

Copied!

content_copy

```
NAME      MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda         8:0    0   10G  0 disk
|-sda1      8:1    0  4.9G  0 part
|-sda2      8:2    0   16M  0 part
|-sda3      8:3    0    2G  0 part
|-sda4      8:4    0   16M  0 part
|-sda5      8:5    0    2G  0 part
|-sda6      8:6    0  512B  0 part
|-sda7      8:7    0  512B  0 part
|-sda8      8:8    0   16M  0 part
|-sda9      8:9    0  512B  0 part
|-sda10     8:10   0  512B  0 part
|-sda11     8:11   0    8M  0 part
`-sda12     8:12   0   32M  0 part
sdb         8:16   0   10G  0 disk
|-sdb1      8:17   0  5.9G  0 part /etc/hosts
```

```
|-sdb2       8:18   0   16M  0 part
|-sdb3       8:19   0    2G  0 part
| `-vroot 253:0    0    2G  1 dm
|-sdb4       8:20   0   16M  0 part
|-sdb5       8:21   0    2G  0 part
|-sdb6       8:22   0  512B  0 part
|-sdb7       8:23   0  512B  0 part
|-sdb8       8:24   0   16M  0 part
|-sdb9       8:25   0  512B  0 part
|-sdb10      8:26   0  512B  0 part
|-sdb11      8:27   0    8M  0 part
`-sdb12      8:28   0   32M  0 part
```

You'll see that your instance has two block devices (sda or sdb) attached to it (disks). Each of them is 10GB in size. The column MOUNTPOINT shows where a block device is mounted. It’s a location of files on the disk, where files can be accessed by the Linux file system. In this case, the MOUNTPOINT **(/etc/hosts)** is displayed against **sdb**, which means the second disk (sdb) is mounted at the root of the Linux file system tree. Thus, the files you're seeing on your system right now are from this disk.

A first disk, **sda**, is also available, but it's not mounted (UNMOUNT). In this lab, you'll divide this disk into two partitions. You'll mount one of these partitions onto the file system, so you can start accessing files from it.

Note: These may be swapped for you, and your VM may be mounted on sda instead of sdb. This will change the commands used in the lab, so when you see \[MOUNT DRIVE\] replace it with your mount drive (sda or sdb) i.e the one which is displaying against /etc/hosts and when you see \[UNMOUNTED DRIVE\] replace it with the other one. If your VM is mounted on sda, the screenshots will also be flipped from what you will see based on the disk on which your VM is mounted on. Optionally, you can view disks mounted on the system using the **df** command. This command is normally used to display the amount of space available on the file system. It lists all block devices with the available space on them. Use the **-h** option to display file sizes in human readable format.

```
df -h
```

Copied!

content_copy

```
Filesystem      Size  Used Avail Use% Mounted on
overlay         5.7G  809M  4.9G  14% /
tmpfs            64M     0   64M   0% /dev
tmpfs           290M     0  290M   0% /sys/fs/cgroup
shm              64M     0   64M   0% /dev/shm
/dev/sdb1       5.7G  809M  4.9G  14% /etc/hosts
```

Partitions

Instead of using a storage block as a whole, it's common practice to divide a storage block into different partitions. Partitions can be different sizes, and formatted to different filesystems. This allows you to use a single storage device for different purposes.

You can display partition information using the **fdisk** command. You can also use the -l option to list partitions in the block. You can pass a device name to the fdisk command to list the partitions contained in that device.

To list all partitions, use **fdisk -l**

```
sudo fdisk -l
```

Copied!

content_copy

```
GPT PMBR size mismatch (18874524 != 20971519) will be corrected by write.
The backup GPT table is not on the end of the device. This problem will be
corrected by write.
Disk /dev/sda: 10 GiB, 10737418240 bytes, 20971520 sectors
Disk model: PersistentDisk
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 4096 bytes
I/O size (minimum/optimal): 4096 bytes / 4096 bytes
Disklabel type: gpt
Disk identifier: FDF53EEC-3010-3049-B247-42C11C16F682
Device       Start      End  Sectors  Size Type
/dev/sda1  8704000 18874476 10170477  4.9G Linux filesystem
/dev/sda2    20480    53247    32768   16M ChromeOS kernel
/dev/sda3  4509696  8703999  4194304    2G ChromeOS root fs
/dev/sda4    53248    86015    32768   16M ChromeOS kernel
/dev/sda5   315392  4509695  4194304    2G ChromeOS root fs
/dev/sda6    16448    16448        1  512B ChromeOS kernel
/dev/sda7    16449    16449        1  512B ChromeOS root fs
/dev/sda8    86016   118783    32768   16M Linux filesystem
/dev/sda9    16450    16450        1  512B ChromeOS reserved
/dev/sda10   16451    16451        1  512B ChromeOS reserved
/dev/sda11      64    16447    16384    8M BIOS boot
/dev/sda12  249856   315391    65536   32M EFI System
```

```
Partition 7 does not start on physical sector boundary.
Partition 9 does not start on physical sector boundary.
Partition 10 does not start on physical sector boundary.
Partition table entries are not in disk order.
Disk /dev/sdb: 10 GiB, 10737418240 bytes, 20971520 sectors
Disk model: PersistentDisk
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 4096 bytes
I/O size (minimum/optimal): 4096 bytes / 4096 bytes
```

To list partitions contained in **/dev/sdb**, pass **/dev/sdb** to the **fdisk** command.

```
sudo fdisk -l /dev/[MOUNT DRIVE]
```

Copied!

content_copy

```
Disk /dev/sdb: 10 GiB, 10737418240 bytes, 20971520 sectors
Disk model: PersistentDisk
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 4096 bytes
I/O size (minimum/optimal): 4096 bytes / 4096 bytes
Disklabel type: gpt
Disk identifier: FDF53EEC-3010-3049-B247-42C11C16F682
Device        Start      End  Sectors  Size Type
/dev/sdb1   8704000 20971486 12267487  5.9G Linux filesystem
/dev/sdb2     20480    53247    32768   16M ChromeOS kernel
/dev/sdb3   4509696  8703999  4194304    2G ChromeOS root fs
/dev/sdb4     53248    86015    32768   16M ChromeOS kernel
/dev/sdb5    315392  4509695  4194304    2G ChromeOS root fs
/dev/sdb6     16448    16448        1  512B ChromeOS kernel
/dev/sdb7     16449    16449        1  512B ChromeOS root fs
/dev/sdb8     86016   118783    32768   16M Linux filesystem
/dev/sdb9     16450    16450        1  512B ChromeOS reserved
/dev/sdb10    16451    16451        1  512B ChromeOS reserved
/dev/sdb11       64    16447    16384    8M BIOS boot
/dev/sdb12   249856   315391    65536   32M EFI System
```

```
Partition 7 does not start on physical sector boundary.
Partition 9 does not start on physical sector boundary.
Partition 10 does not start on physical sector boundary.
Partition table entries are not in disk order.
```

fdisk displays information contained in the partition table, where information about partitions is stored.

Disk partitioning with fdisk

When the *fdisk* command is used without options, it provides a menu-driven environment for creating and deleting partitions.

Caution!: Modifying partitions is destructive, and can lead to loss of data. Not good! Remember to always backup your data before modifying partitions on a live system.

Mount and umount

Mounting and unmounting mean making devices available or unavailable on a Linux file system. This is accomplished by the commands *mount* and *umount*. Before modifying a disk, you should first **unmount** it from the system, using the umount command. When modifications on the disk are done, you should **mount** it back onto the system. For this exercise, since the device we're partitioning isn't initially mounted, you can proceed with partitioning.

Go ahead and start *fdisk* in interactive mode by passing the name of the disk you want to partition. In this lab, we'll partition **/dev/sda** as it's unmounted drive.

Note: In this lab, we will partition the disk that's not currently mounted. You should select dev/[UNMOUNTED DRIVE] for creating a partition. You can still partition the disk even when the operating system is running from it, but a reboot will be required in order for the partition changes you make to take place.
Start fdisk by passing the disk you want to partition as the parameter.

```
sudo fdisk /dev/[UNMOUNTED DRIVE]
```

Copied!

content_copy

```
Welcome to fdisk (util-linux 2.33.1).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.
GPT PMBR size mismatch (18874524 != 20971519) will be corrected by write.
The backup GPT table is not on the end of the device. This problem will be
corrected by write.
Command (m for help):
```

fdisk will start in interactive mode. You can use **m** to use help provided by the command.

```
Command (m for help): m
Help:
  GPT
   M   enter protective/hybrid MBR
  Generic
   d   delete a partition
   F   list free unpartitioned space
   l   list known partition types
   n   add a new partition
   p   print the partition table
   t   change a partition type
   v   verify the partition table
   i   print information about a partition
  Misc
   m   print this menu
   x   extra functionality (experts only)
  Script
   I   load disk layout from sfdisk script file
   O   dump disk layout to sfdisk script file
  Save & Exit
   w   write table to disk and exit
   q   quit without saving changes
  Create a new label
   g   create a new empty GPT partition table
   G   create a new empty SGI (IRIX) partition table
   o   create a new empty DOS partition table
   s   create a new empty Sun partition table
Command (m for help):
```

You can use **p** to show details about partitions on the disk.

```
Command (m for help): p
Disk /dev/sda: 10 GiB, 10737418240 bytes, 20971520 sectors
Disk model: PersistentDisk
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 4096 bytes
I/O size (minimum/optimal): 4096 bytes / 4096 bytes
Disklabel type: gpt
Disk identifier: FDF53EEC-3010-3049-B247-42C11C16F682
Device       Start       End  Sectors  Size Type
```

```
/dev/sda1  8704000 18874476 10170477  4.9G Linux filesystem
/dev/sda2    20480    53247    32768   16M ChromeOS kernel
/dev/sda3  4509696  8703999  4194304    2G ChromeOS root fs
/dev/sda4    53248    86015    32768   16M ChromeOS kernel
/dev/sda5   315392  4509695  4194304    2G ChromeOS root fs
/dev/sda6    16448    16448        1  512B ChromeOS kernel
/dev/sda7    16449    16449        1  512B ChromeOS root fs
/dev/sda8    86016   118783    32768   16M Linux filesystem
/dev/sda9    16450    16450        1  512B ChromeOS reserved
/dev/sda10   16451    16451        1  512B ChromeOS reserved
/dev/sda11      64    16447    16384    8M BIOS boot
/dev/sda12  249856   315391    65536   32M EFI System
Partition 7 does not start on physical sector boundary.
Partition 9 does not start on physical sector boundary.
Partition 10 does not start on physical sector boundary.
Partition table entries are not in disk order.
Command (m for help):
```

Enter **q** to exit interactive mode when you are finished exploring.

Creating Partitions

You'll now create new partitions using **fdisk**. You'll partition **the unmounted drive** into two partitions: one swap partition of size **1GB**, and another of size **9GB**. The file system type on the second partition will be ext4.

Open *fdisk* in interactive mode to do the partitioning:

```
sudo fdisk /dev/[UNMOUNTED DRIVE]
```

Copied!

content_copy

```
Welcome to fdisk (util-linux 2.33.1).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.
GPT PMBR size mismatch (18874524 != 20971519) will be corrected by write.
The backup GPT table is not on the end of the device. This problem will be
corrected by write.
Command (m for help):
```

To create a new partition, the command control **n** is used. However, since all the space on the disk is currently allocated, you'll need to first free up space by deleting the default partitions.

Use the **d** command control to delete the default partitions. When you issue the **d** command control, **fdisk** asks you to enter the number of partitions you want to delete. Since you have twelve partitions, **fdisk** automatically selects the last partition by default, and pressing **Enter** deletes the last partition. Repeat this process until you delete all the twelve partitions.

```
Command (m for help): d
Partition number (1-12, default 12):
Partition 12 has been deleted.
Command (m for help): d
Partition number (1-11, default 11):
Partition 11 has been deleted.
Command (m for help): d
Partition number (1-10, default 10):
```

```
Partition 10 has been deleted.
Command (m for help): d
Partition number (1-9, default 9):
Partition 9 has been deleted.
Command (m for help): d
Partition number (1-8, default 8):
Partition 8 has been deleted.
Command (m for help): d
Partition number (1-7, default 7):
Partition 7 has been deleted.
Command (m for help): d
Partition number (1-6, default 6):
Partition 6 has been deleted.
Command (m for help): d
Partition number (1-5, default 5):
Partition 5 has been deleted.
Command (m for help): d
Partition number (1-4, default 4):
Partition 4 has been deleted.
Command (m for help): d
Partition number (1-3, default 3):
Partition 3 has been deleted.
Command (m for help): d
Partition number (1,2, default 2):
Partition 2 has been deleted.
Command (m for help): d
Selected partition 1
Partition 1 has been deleted.
Command (m for help):
```

You're now able to create your new partitions. Enter the command control for creating a new partition, **n**.

You'll then need to provide the starting sector (memory location) of the new partition, from where you want to allocate. Here, press **Enter** to select the default value 2048.

```
Command (m for help): n
Partition number (1-128, default 1):
First sector (34-20971486, default 2048):
Last sector, +/-sectors or +/-size{K,M,G,T,P} (2048-20971486, default
20971486):
```

Provide the last sector of the new partition, up to where you want to allocate. The difference between the first and last sectors makes up the total size of the partition. Disk sector represents units used to measure the size on disks. Each sector stores a fixed amount of data. In lots of hard disks, for example, a sector stores 512 bytes. To create the first 1GB partition, enter **2097200** (divide the original partition by 10).

```
Command (m for help): n
Partition number (1-128, default 1):
First sector (34-20971486, default 2048):
Last sector, +/-sectors or +/-size{K,M,G,T,P} (2048-20971486, default
20971486): 2097200
Created a new partition 1 of type 'Linux filesystem' and of size 1023 MiB.
Command (m for help):
```

Two important things happen here: the partition size is set to **1GB**, and the partition type is set to **Linux filesystem**. (You'll see how to change partition types in the next section.) Voila! One partition is now created. You'll now move on to the second one.

Use the command control **n** again for a new partition.

```
Command (m for help): n
Partition number (2-128, default 2):
```

Select partition number **2** to issue partition numbers in sequence.

```
Command (m for help): n
Partition number (2-128, default 2):
First sector (2097201-20971486, default 2099200):
```

Select the default partition starting sector, which is the next sector from the last partition you allocated.

```
Command (m for help): n
Partition number (2-128, default 2):
First sector (2097201-20971486, default 2099200):
Last sector, +/-sectors or +/-size{K,M,G,T,P} (2099200-20971486, default
20971486):
```

Also select the default last sector, which will be the last sector of the remaining disk space.

```
Command (m for help): n
Partition number (2-128, default 2):
First sector (2097201-20971486, default 2099200):
Last sector, +/-sectors or +/-size{K,M,G,T,P} (2099200-20971486, default
20971486):
Created a new partition 2 of type 'Linux filesystem' and of size 9 GiB.
Command (m for help):
```

The second partition is now created. Sweet!

Before committing your changes, you'll change the second partition to a different partition type. You'll change the first partition type to a Linux swap type. Enter command control **t** to change the partition type, and select the first partition.

```
Command (m for help): t
Partition number (1,2, default 2): 1
Partition type (type L to list all types):
```

You can use the command control **L** to view a list of all partition types.

```
Command (m for help): t
Partition number (1,2, default 2): 1
Partition type (type L to list all types): L
  1 EFI System                     C12A7328-F81F-11D2-BA4B-00A0C93EC93B
  2 MBR partition scheme           024DEE41-33E7-11D3-9D69-0008C781F39F
  3 Intel Fast Flash               D3BFE2DE-3DAF-11DF-BA40-E3A556D89593
  4 BIOS boot                      21686148-6449-6E6F-744E-656564454649
  5 Sony boot partition            F4019732-066E-4E12-8273-346C5641494F
  6 Lenovo boot partition          BFBFAFE7-A34F-448A-9A5B-6213EB736C22
  7 PowerPC PReP boot              9E1A2D38-C612-4316-AA26-8B49521E5A8B
  8 ONIE boot                      7412F7D5-A156-4B13-81DC-867174929325
  9 ONIE config                    D4E6E2CD-4469-46F3-B5CB-1BFF57AFC149
 10 Microsoft reserved             E3C9E316-0B5C-4DB8-817D-F92DF00215AE
 11 Microsoft basic data           EBD0A0A2-B9E5-4433-87C0-68B6B72699C7
 12 Microsoft LDM metadata         5808C8AA-7E8F-42E0-85D2-E1E90434CFB3
 13 Microsoft LDM data             AF9B60A0-1431-4F62-BC68-3311714A69AD
 14 Windows recovery environment   DE94BBA4-06D1-4D40-A16A-BFD50179D6AC
 15 IBM General Parallel Fs        37AFFC90-EF7D-4E96-91C3-2D7AE055B174
 16 Microsoft Storage Spaces       E75CAF8F-F680-4CEE-AFA3-B001E56EFC2D
 17 HP-UX data                     75894C1E-3AEB-11D3-B7C1-7B03A0000000
 18 HP-UX service                  E2A1E728-32E3-11D6-A682-7B03A0000000
 19 Linux swap                     0657FD6D-A4AB-43C4-84E5-0933C84B4F4F
 20 Linux filesystem               0FC63DAF-8483-4772-8E79-3D69D8477DE4
 21 Linux server data              3B8F8425-20E0-4F3B-907F-1A25A76F98E8
 22 Linux root (x86)               44479540-F297-41B2-9AF7-D131D5F0458A
```

Enter **19** to change the partition type to ‘Linux swap', and press **Enter**.

Head's up: Some of the characters in the partition type name **Linux swap** are truncated.

```
Partition type (type L to list all types): 19
Changed type of partition 'Linux filesystem' to 'Linux swap'.
Command (m for help):
```

The partition type will be changed to match the selection.

Up to this point, you've just been editing the partition table in memory. You can use the **q** command here to quit **fdisk** without committing changes to the disk. You can also update your partitions by using the **d** and **n** commands to remove and add new partitions.

You can also use the **v** command here to verify your changes before proceeding.

```
Command (m for help): v
No errors detected.
Header version: 1.0
Using 2 out of 128 partitions.
A total of 4013 free sectors is available in 2 segments (the largest is
1007 KiB).
```

If you're satisfied with the changes you've made so far, you can commit them to the disk by using the **w** command.

```
Command (m for help): w
The partition table has been altered.
Calling ioctl() to re-read partition table.
Syncing disks.
```

Congrats! You've successfully partitioned the second disk using **fdisk**.

The second disk device is now made up of two partitions of **1GB** and **9GB**, respectively.

Click *Check my progress* to verify the objective.

Partitioning

Check my progress

Formatting partitions using mkfs

Next, you'll create different file systems in the partitions you just created. You'll do this by using the command **mkfs** in Linux. Multiple filesystem types exist, and it's important to know all of them, along with the functions they're best suited for. In this lab, you'll format the second partition into ext4, the most widely used Linux filesystem type.

To do this, use **lsblk** again to find the disk you want to create the file system type in.

```
lsblk
```

Copied!

content_copy

```
NAME      MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda         8:0    0   10G  0 disk
|-sda1      8:1    0 1023M  0 part
`-sda2      8:2    0    9G  0 part
sdb         8:16   0   10G  0 disk
|-sdb1      8:17   0  5.9G  0 part /etc/hosts
|-sdb2      8:18   0   16M  0 part
|-sdb3      8:19   0    2G  0 part
| `-vroot 253:0    0    2G  1 dm
|-sdb4      8:20   0   16M  0 part
|-sdb5      8:21   0    2G  0 part
|-sdb6      8:22   0  512B  0 part
|-sdb7      8:23   0  512B  0 part
|-sdb8      8:24   0   16M  0 part
|-sdb9      8:25   0  512B  0 part
|-sdb10     8:26   0  512B  0 part
|-sdb11     8:27   0    8M  0 part
`-sdb12     8:28   0   32M  0 part
```

Format the second partition **in your unmounted drive** (sdb2 or sda2) to ext4 using this command:

```
sudo mkfs -t ext4 /dev/[UNMOUNTED DRIVE]2
```

Copied!

content_copy

```
mke2fs 1.44.5 (15-Dec-2018)
Discarding device blocks: done
Creating filesystem with 2359035 4k blocks and 589824 inodes
Filesystem UUID: 3e68d65f-3029-4232-8f45-b924de3862bd
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632
Allocating group tables: done
Writing inode tables: done
Creating journal (16384 blocks): done
```

```
Writing superblocks and filesystem accounting information: done
```

Click *Check my progress* to verify the objective.

EXT4

Check my progress

Now, You can mount **/dev/[UNMOUNTED DRIVE]2** to a location on the file system to start accessing files on it. Mount it on the directory **/home/my_drive**.

```
sudo mount /dev/[UNMOUNTED DRIVE]2 /home/my_drive
```

Copied!

content_copy

You can verify the file systems and block devices attached to your system using **lsblk** command.

```
lsblk
```

Copied!

content_copy

```
NAME      MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda         8:0    0   10G  0 disk
|-sda1      8:1    0 1023M  0 part
`-sda2      8:2    0    9G  0 part /home/my_drive
sdb         8:16   0   10G  0 disk
|-sdb1      8:17   0  5.9G  0 part /etc/hosts
|-sdb2      8:18   0   16M  0 part
|-sdb3      8:19   0    2G  0 part
| `-vroot 253:0    0    2G  1 dm
|-sdb4      8:20   0   16M  0 part
```

```
|-sdb5       8:21   0    2G  0 part
|-sdb6       8:22   0  512B  0 part
|-sdb7       8:23   0  512B  0 part
|-sdb8       8:24   0   16M  0 part
|-sdb9       8:25   0  512B  0 part
|-sdb10      8:26   0  512B  0 part
|-sdb11      8:27   0    8M  0 part
`-sdb12      8:28   0   32M  0 part
```

From now on, accessing "/home/my_drive" will be accessing files on the disk.

That's it! You've successfully partitioned and formatted a disk in Linux.

Click *Check my progress* to verify the objective.

Mount

Check my progress

Conclusion

In this lab, we've gone through the process of creating partitions, formatting them to specific filesystems, and mounting them onto accessible locations in Linux. You should continue to practice these commands so that you become comfortable using them.