

Name: Mosab Fathy Ramadan Mohamed

Group: B20-SD-01

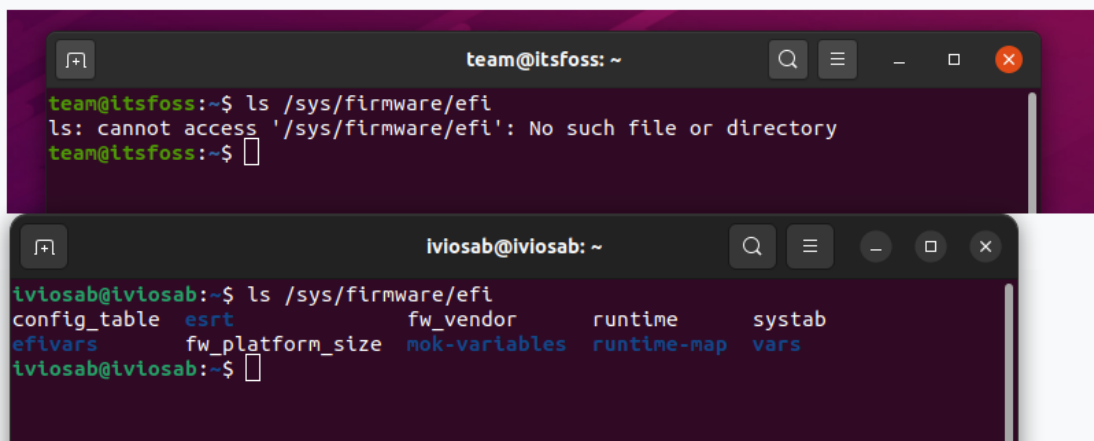
Lab 2: OS main components

Environment Preparation:

- Ensure that you enable the EFI standard of booting. How did you check this
 - It is enabled
 - I checked it by finding /sys/firmware/efi

Check if you are using UEFI or BIOS on Linux

The easiest way to find out if you are running UEFI or BIOS is to look for a folder /sys/firmware/efi. The folder will be missing if your system is using BIOS.



```
team@itsfoss: ~  
team@itsfoss:~$ ls /sys/firmware/efi  
ls: cannot access '/sys/firmware/efi': No such file or directory  
team@itsfoss:~$  
  
ivosab@ivosab: ~  
ivosab@ivosab:~$ ls /sys/firmware/efi  
config_table  esrt          fw_vendor      runtime        systab  
efivars       fw_platform_size  mok-variables  runtime-map    vars  
ivosab@ivosab:~$
```

Exercise 1: GPT partition

Questions to answer:

Q1: What is fdisk utility used for?

Answer:

Fdisk is used to manipulate disk partition tables

It is a dialog-driven program for creation and manipulation of partition tables



```
ivosab@ivosab: ~/Desktop/GitHub/F22-SNA/Week-2
FDISK(8)                                System Administration                                FDISK(8)

NAME
    fdisk - manipulate disk partition table

SYNOPSIS
    fdisk [options] device

    fdisk -l [device...]

DESCRIPTION
    fdisk is a dialog-driven program for creation and manipulation of partition
    tables. It understands GPT, MBR, Sun, SGI and BSD partition tables.

    Block devices can be divided into one or more logical disks called
    partitions. This division is recorded in the partition table, usually found
    in sector 0 of the disk. (In the BSD world one talks about 'disk slices'
    and a 'disklabel'.)

    All partitioning is driven by device I/O limits (the topology) by default.
    fdisk is able to optimize the disk layout for a 4K-sector size and use an
    alignment offset on modern devices for MBR and GPT. It is always a good
    idea to follow fdisk's defaults as the default values (e.g., first and last
    partition sectors) and partition sizes specified by the +/-<size>{M,G,...}

Manual page fdisk(8) line 1 (press h for help or a to quit)
```

Q2: Show the bootable device(s) on your machine, and identify which partition(s) are bootable.

Answer:

We can find the bootable devices and partitions using the command : (lsblk)

```
ivlosab@ivlosab:~/Desktop/GitHub/F22-SNA/Week-2$ lsblk
NAME            MAJ:MIN RM   SIZE RO TYPE MOUNTPOINTS
loop0             7:0    0     4K  1 loop /snap/bare/5
loop1             7:1    0  908,1M  1 loop /snap/clion/203
loop2             7:2    0  909,4M  1 loop /snap/clion/204
loop3             7:3    0   55,6M  1 loop /snap/core18/2538
loop4             7:4    0   55,6M  1 loop /snap/core18/2560
loop5             7:5    0  164,8M  1 loop /snap/gnome-3-28-1804/161
loop6             7:6    0    62M  1 loop /snap/core20/1611
loop7             7:7    0   70,4M  1 loop /snap/core22/188
loop8             7:8    0  400,8M  1 loop /snap/gnome-3-38-2004/112
loop9             7:9    0  346,3M  1 loop /snap/gnome-3-38-2004/115
loop10            7:10   0   81,3M  1 loop /snap/gtk-common-themes/1534
loop11            7:11   0   91,7M  1 loop /snap/gtk-common-themes/1535
loop12            7:12   0   45,9M  1 loop /snap/snap-store/575
loop13            7:13   0   45,9M  1 loop /snap/snap-store/582
loop14            7:14   0    47M  1 loop /snap/snapd/16010
loop15            7:15   0    47M  1 loop /snap/snapd/16292
loop16            7:16   0   284K  1 loop /snap/snapd-desktop-integration/10
loop17            7:17   0   284K  1 loop /snap/snapd-desktop-integration/14
loop18            7:18   0   345M  1 loop /snap/telegram-desktop/4095
loop19            7:19   0   345M  1 loop /snap/telegram-desktop/4116
loop20            7:20   0  230,2M  1 loop /snap/racket/2
loop21            7:21   0  414,3M  1 loop /snap/gnome-42-2204/29
loop22            7:22   0   63,2M  1 loop /snap/core20/1623
loop23            7:23   0  169,4M  1 loop /snap/spotify/60
sda               8:0    0  931,5G  0 disk
├─sda1             8:1    0   999M  0 part
├─sda2             8:2    0  930,5G  0 part
nvme0n1          259:0    0  238,5G  0 disk
├─nvme0n1p1       259:1    0   450M  0 part
├─nvme0n1p2       259:2    0   100M  0 part /boot/efi
├─nvme0n1p3       259:3    0    16M  0 part
├─nvme0n1p4       259:4    0  130,9G  0 part
├─nvme0n1p5       259:5    0   777M  0 part
└─nvme0n1p6       259:6    0  106,3G  0 part /
```

We can see that the only bootable device is nvme0n1

And bootable partition in that device is the second one (nvme0n1p2)

Q3: What is logical block address?

Answer:

Logical block addressing (LBA) is a common scheme used for specifying the location of blocks of data stored on computer storage devices, generally secondary storage systems such as hard disk drives. LBA is a particularly simple linear addressing scheme; blocks are located by an integer index, with the first block being LBA 0, the second LBA 1, and so on.

Q4: Why did we specify the count, the bs, and the skip options when using dd?

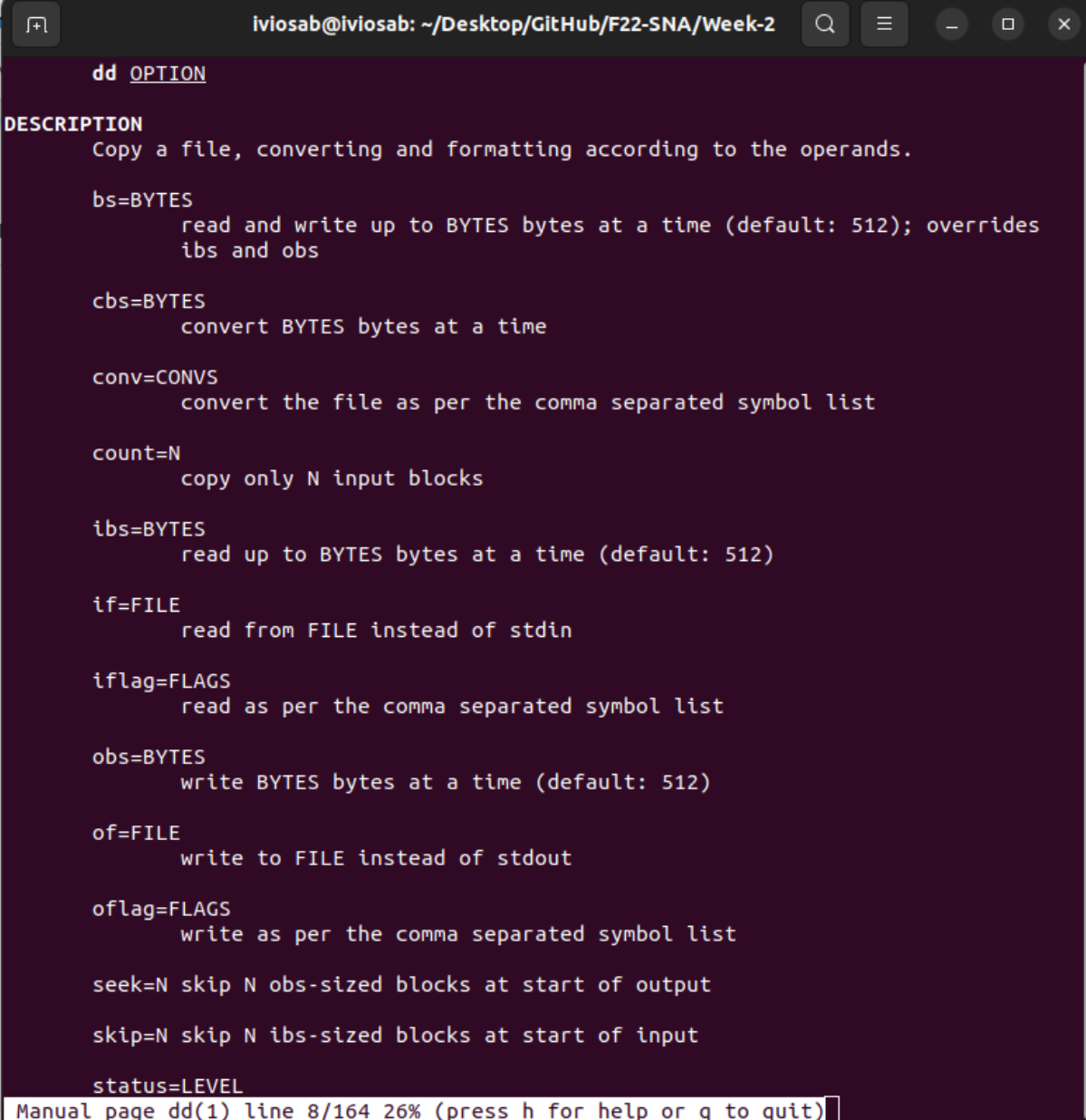
Answer:

“bs” means the number of bytes to read or write at a time

“count” means the number of input blocks you want to copy

“skip” means the number of blocks we want to skip before copying

So basically we indicate how much we read or write each time, then indicate the number of times we want to read, then indicate where we want to start copying



```
iviosab@iviosab: ~/Desktop/GitHub/F22-SNA/Week-2
dd OPTION
DESCRIPTION
Copy a file, converting and formatting according to the operands.

bs=BYTES
    read and write up to BYTES bytes at a time (default: 512); overrides
    ibs and obs

cbs=BYTES
    convert BYTES bytes at a time

conv=CONVS
    convert the file as per the comma separated symbol list

count=N
    copy only N input blocks

ibs=BYTES
    read up to BYTES bytes at a time (default: 512)

if=FILE
    read from FILE instead of stdin

iflag=FLAGS
    read as per the comma separated symbol list

obs=BYTES
    write BYTES bytes at a time (default: 512)

of=FILE
    write to FILE instead of stdout

oflag=FLAGS
    write as per the comma separated symbol list

seek=N skip N obs-sized blocks at start of output

skip=N skip N ibs-sized blocks at start of input

status=LEVEL
Manual page dd(1) line 8/164 26% (press h for help or q to quit)
```

Q5: Why does a GPT formatted disk have the MBR?

Answer:

The main purpose of inserting the MBR at the start of disk is strictly for protection. MBR-oriented disk utilities have the possibility of not recognizing or even writing over top of the GPT disks. To prevent this, the entire GPT disk is labeled as one partition. The System ID of this partition is established as 0xEE, indicating the implementation of the GPT.

Q6: Name two differences between primary and logical partitions in an MBR partitioning scheme

Answer:

Primary partition is a bootable partition and it contains the operating system/s of the computer, while logical partition is a partition that is not bootable. Logical partitions are similar to primary partitions. However, while only four primary partitions can exist on a single disk, the number of logical partitions that can exist on a disk is unlimited. A logical partition can be formatted and assigned a drive letter.

Exercise 2 - UEFI Booting

Questions to answer

Q1: Why is Shim used to load the GRUB bootloader?

Answer:

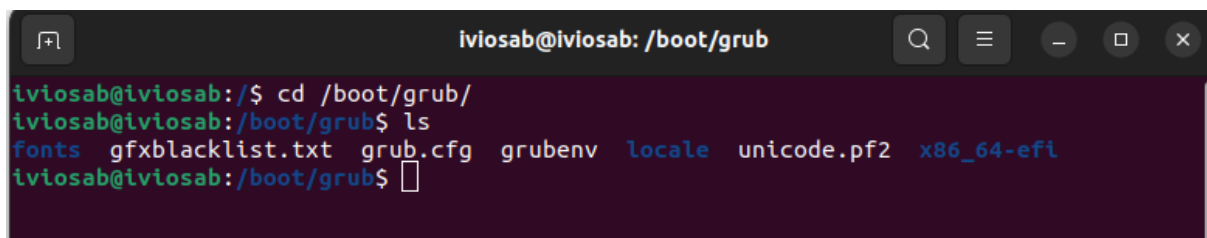
Shim acts as a pre-bootloader where it mounts all the efi drivers in the boot directory in ubuntu, so that the real bootloader GRUB loads utilizing the drivers the shim has written in the file system.

Q2: Can you locate your grub configuration file? Show the path.

Answer:

Yes we can, by using “cd /boot/grub”

So the path will be “/boot/grub/grub.cfg”

A terminal window with a dark background and light-colored text. The window title is 'ivosab@ivosab: /boot/grub'. The terminal shows the following commands and output:

```
ivosab@ivosab:/$ cd /boot/grub/  
ivosab@ivosab:/boot/grub$ ls  
fonts  gfxblacklist.txt  grub.cfg  grubenv  locale  unicode.pf2  x86_64-efi  
ivosab@ivosab:/boot/grub$
```

Q3: According to the boot order, what is the third boot device on your computer?
How did you check this?

Answer:

The third boot device is EFI USB Device RC

I checked this by opening the “efibootmgr -v” and checking the bootOrder, then locating the id of the third device which was 2001, then locating the information about that device.

```
ivlosab@ivlosab:/boot/grub$ efibootmgr -v
BootCurrent: 0000
Timeout: 0 seconds
BootOrder: 0000,0004,2001,2002,2003
Boot0000* ubuntu HD(2,GPT,7d202dd1-57c6-42ae-abff-ff88e849804a,0xe1800,0x32000)
)/File(\EFI\ubuntu\shimx64.efi)
Boot0002* EFI PXE 0 for IPv4 (00-2B-67-F1-63-5C) PciRoot(0x0)/Pci(0x1d,0x6)/Pc
i(0x0,0x0)/MAC(002b67f1635c,0)/IPv4(0.0.0.00.0.0.0,0,0)RC
Boot0003* EFI PXE 0 for IPv6 (00-2B-67-F1-63-5C) PciRoot(0x0)/Pci(0x1d,0x6)/Pc
i(0x0,0x0)/MAC(002b67f1635c,0)/IPv6([::]:<->[::]:,0,0)RC
Boot0004* Windows Boot Manager HD(2,GPT,7d202dd1-57c6-42ae-abff-ff88e849804a,0xe1800
,0x32000)/File(\EFI\Microsoft\Boot\bootmgfw.efi)WINDOWS.....x...B.C.D.O.B.J.E.C.T
.=.{.9.d.e.a.8.6.2.c.-.5.c.d.d.-.4.e.7.0.-.a.c.c.1.-.f.3.2.b.3.4.4.d.4.7.9.5.}...M...
.....
Boot2001* EFI USB Device RC
Boot2002* EFI DVD/CDROM RC
Boot2003* EFI Network RC
ivlosab@ivlosab:/boot/grub$
```

Exercise 3: Filesystem

Questions to answer

Q1: How many inodes are in use on your system?

Answer:

I have 559180 inodes in use

```
bash: /: is a directory
ivosab@ivosab:/$ df -i
Filesystem      Inodes    IUsed   IFree  IUse% Mounted on
tmpfs            2036168    1426  2034742    1% /run
/dev/nvme0n1p6  6971392  557152  6414240    8% /
tmpfs            2036168     409  2035759    1% /dev/shm
tmpfs            2036168      3  2036165    1% /run/lock
/dev/nvme0n1p2      0         0         0     - /boot/efi
tmpfs            407233    190   407043    1% /run/user/1000
ivosab@ivosab:/$
```

Q2: What is the filesystem type of the EFI partition?

Answer:

FAT32

Using “lsblk -r” command

```
nvme0n1
├─nvme0n1p1
│   ntfs          Recovery
│                   BA1C8E0B1C8DC33D
├─nvme0n1p2
│   vfat  FAT32    248F-14E7          65,8M    31% /boot/efi
├─nvme0n1p3
│
├─nvme0n1p4
│   ntfs          01D88E5CAFF9D1B0
├─nvme0n1p5
│   ntfs          AA8AB64C8AB614B3
└─nvme0n1p6
    ext4  1.0      ad47611f-ca53-4a26-9508-28ee50c32a40  58,5G    39% /
ivosab@ivosab:/$
```


Q3: What device is mounted at your root / directory? Show proof.

Answer:

nvme0n1 is the device mounted at my root directory
Specifically nvme0n1p6 partition

```
nvme0n1
├─nvme0n1p1
│   ntfs          Recovery
│                   BA1C8E0B1C8DC33D
├─nvme0n1p2
│   vfat  FAT32    248F-14E7          65,8M   31% /boot/efi
├─nvme0n1p3
├─nvme0n1p4
│   ntfs          01D88E5CAFF9D1B0
├─nvme0n1p5
│   ntfs          AA8AB64C8AB614B3
└─nvme0n1p6
    ext4  1.0      ad47611f-ca53-4a26-9508-28ee50c32a40  58,5G   39% /
iviosab@iviosab:/$ mount /
mount: /: must be superuser to use mount.
iviosab@iviosab:/$ sudo mount /
mount: /: /dev/nvme0n1p6 already mounted on /.
iviosab@iviosab:/$
```

Q4: What is your partition UUID?

Answer:

UUID="ad47611f-ca53-4a26-9508-28ee50c32a40"

```
iviosab@iviosab:/$ sudo blkid | grep UUID=
/dev/nvme0n1p6: UUID="ad47611f-ca53-4a26-9508-28ee50c32a40" BLOCK_SIZE="4096" TYPE="ext4" PARTUUID="ed142395-9c13-44a4-83a7-059fd48656af"
/dev/nvme0n1p5: BLOCK_SIZE="512" UUID="AA8AB64C8AB614B3" TYPE="ntfs" PARTUUID="d1995c63-6fb8-430b-8918-1aaba8a5e6f0"
/dev/nvme0n1p1: LABEL="Recovery" BLOCK_SIZE="512" UUID="BA1C8E0B1C8DC33D" TYPE="ntfs" PARTLABEL="Basic data partition" PARTUUID="35488e0e-8f39-4313-9531-ac2c7162f322"
/dev/nvme0n1p4: BLOCK_SIZE="512" UUID="01D88E5CAFF9D1B0" TYPE="ntfs" PARTLABEL="Basic data partition" PARTUUID="35e0822d-e769-46ba-af27-a2f660c683f0"
/dev/nvme0n1p2: UUID="248F-14E7" BLOCK_SIZE="512" TYPE="vfat" PARTLABEL="EFI system partition" PARTUUID="7d202dd1-57c6-42ae-abff-ff88e849804a"
/dev/sda2: LABEL="New Volume" BLOCK_SIZE="512" UUID="B69448D794489C2D" TYPE="ntfs"
/dev/sda1: UUID="EE5C-6E50" BLOCK_SIZE="512" TYPE="vfat"
/dev/nvme0n1p3: PARTLABEL="Microsoft reserved partition" PARTUUID="7af755cb-c080-4331-a191-b3bcb3a07e6c"
iviosab@iviosab:/$
```

Q5: Show at least two methods of viewing the UUID of a block device.

Answer:

1) "blkid"

```
lviosab@lviosab:/$ sudo blkid | grep UUID=
/dev/nvme0n1p6: UUID="ad47611f-ca53-4a26-9508-28ee50c32a40" BLOCK_SIZE="4096" TYPE="ext4" PARTUUID="ed142395-9c13-44a4-83a7-059fd48656af"
/dev/nvme0n1p5: BLOCK_SIZE="512" UUID="AA8AB64C8AB614B3" TYPE="ntfs" PARTUUID="d1995c63-6fb8-430b-8918-1aaba8a5e6f0"
/dev/nvme0n1p1: LABEL="Recovery" BLOCK_SIZE="512" UUID="BA1C8E0B1C8DC33D" TYPE="ntfs" PARTLABEL="Basic data partition" PARTUUID="35488e0e-8f39-4313-9531-ac2c7162f322"
/dev/nvme0n1p4: BLOCK_SIZE="512" UUID="01D88E5CAFF9D1B0" TYPE="ntfs" PARTLABEL="Basic data partition" PARTUUID="35e0822d-e769-46ba-af27-a2f660c683f0"
/dev/nvme0n1p2: UUID="248F-14E7" BLOCK_SIZE="512" TYPE="vfat" PARTLABEL="EFI system partition" PARTUUID="7d202dd1-57c6-42ae-abff-ff88e849804a"
/dev/sda2: LABEL="New Volume" BLOCK_SIZE="512" UUID="B69448D794489C2D" TYPE="ntfs"
/dev/sda1: UUID="EE5C-6E50" BLOCK_SIZE="512" TYPE="vfat"
/dev/nvme0n1p3: PARTLABEL="Microsoft reserved partition" PARTUUID="7af755cb-c080-4331-a191-b3bcb3a07e6c"
lviosab@lviosab:/$
```

2) "lsblk -f"

```
lviosab@lviosab:/$ lsblk -f | grep -v loop
NAME            FSTYPE  FSVER LABEL          UUID                                FSAVAIL FS
USE% MOUNTPOINTS
sda
├─sda1          vfat    FAT32              EE5C-6E50
└─sda2          ntfs                    New Volume B69448D794489C2D
nvme0n1
├─nvme0n1p1    ntfs                    Recovery  BA1C8E0B1C8DC33D
├─nvme0n1p2    vfat    FAT32              248F-14E7                    65,8M
31% /boot/efi
├─nvme0n1p3
├─nvme0n1p4    ntfs                    01D88E5CAFF9D1B0
├─nvme0n1p5    ntfs                    AA8AB64C8AB614B3
└─nvme0n1p6    ext4    1.0                ad47611f-ca53-4a26-9508-28ee50c32a40 58,5G
39% /
lviosab@lviosab:/$
```

Q6: What is the function of /dev/zero?

Answer:

/dev/zero is a special file in Unix-like operating systems that provides as many null characters (ASCII NUL, 0x00) as are read from it. One of the typical uses is to provide a character stream for initializing data storage.