System and Network Administration - Lab 2 - OS main components

Jaffar Totanji - j.totanji@innopolis.university

Exercise 1 - GPT partition:

```
kuro@kuro-VirtualBoxZorinOS:~/Desktop$ sudo fdisk -l
Disk /dev/sda: 25 GiB, 26843545600 bytes, 52428800 sectors
Disk model: VBOX HARDDISK
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: gpt
Disk identifier: 0B5BE532-4874-4961-BD78-43E910A12FFA
```

```
000001B0   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  ................
000001C0   02 00 EE FF  FF FF 01 00  00 00 FF FF  1F 03 00 00  ................
000001D0   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  ................
000001E0   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  ................
000001F0   00 00 00 00  00 00 00 00  00 00 00 00  00 00 55 AA  ..............U.
00000200
---   lba.0       --0x200/0x200-------------------------------------------------
```

Questions to answer:

1. The `fdisk` utility is used to view, create, and manipulate partition tables. It understands GPT, MBR, Sun, SGI and BSD partition tables.

2. I can use the `fdisk -lu` command to view all the bootable devices on my machine. The bootable partition is the one with the type EFI System, in this case it is `/dev/sda1`.

```
kuro@kuro-VirtualBoxZorinOS:~/Desktop$ sudo fdisk -lu
Disk /dev/sda: 25 GiB, 26843545600 bytes, 52428800 sectors
Disk model: VBOX HARDDISK
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: gpt
Disk identifier: 0B5BE532-4874-4961-BD78-43E910A12FFA

Device       Start      End  Sectors  Size Type
/dev/sda1     2048  1050623  1048576  512M EFI System
/dev/sda2  1050624 52426751 51376128 24,5G Linux filesystem
```

3. Logical Block Addressing (LBA) is a common scheme used for specifying the location of blocks of data stored on computer storage devices.

4. The flags have the following functionalities:

    - `count`: Specifies the number of input blocks we want to dump.
    - `bs`: Specifies the amount of bytes we want to read/write at a time.
    - `skip=N`: Skips `N` input blocks at the start of input.

We used `bs=512 count=1 skip=0` for the MBR dump because MBR starts at LBA 0 of the GPT layout. The flags mean that we don't want to read `512` bytes at a time (Size of LBA), and only dump one LBA, and skip nothing in the beginning.

We used `bs=512 count=1 skip=1` for the GPT dump because we wanted to skip the first LBA as it contains the MBR info we previoiusly looked it and we want to view the GPT header info located right after.

5. GPT formatted disks tend to include a “protective MBR.” This protective MBR ensures the old tools won’t mistake the GPT drive for an unpartitioned drive and overwrite its GPT data with a new MBR because this type of MBR says that the GPT drive has a single partition that extends across the entire drive. This way, the GPT formatted drive is protected from being overwritten.

6. A logical partition is a volume that is created inside an extended partition on a basic MBR-based disk. 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. This was used to bypass the hard limit of 4 partitions that could be created on a legacy MBR-based disk.

Exercise 2 - UEFI Booting:

```
kuro@kuro-VirtualBoxZorinOS:~/Desktop$ efibootmgr -v
BootCurrent: 0004
Timeout: 0 seconds
BootOrder: 0004,0000,0001,0002,0003
Boot0000* UiApp FvVol(7cb8bdc9-f8eb-4f34-aaea-3ee4af6516a1)/FvFile(462caa21-7614
-4503-836e-8ab6f4662331)
Boot0001* UEFI VBOX CD-ROM VB2-01700376         PciRoot(0x0)/Pci(0x1,0x1)/Ata(1,
0,0)N.....YM....R,Y.
Boot0002* UEFI VBOX HARDDISK VBb3248520-ccb2b45f        PciRoot(0x0)/Pci(0xd,0x0
)/Sata(0,65535,0)N.....YM....R,Y.
Boot0003* EFI Internal Shell    FvVol(7cb8bdc9-f8eb-4f34-aaea-3ee4af6516a1)/FvFi
le(7c04a583-9e3e-4f1c-ad65-e05268d0b4d1)
Boot0004* ubuntu        HD(1,GPT,5598ae77-deae-4bf4-8d44-b6beff389de9,0x800,0x10
0000)/File(\EFI\ubuntu\shimx64.efi)
```

```
kuro@kuro-VirtualBoxZorinOS:~/Desktop$ sudo ls -lah /boot/efi/EFI/ubuntu/
total 3,5M
drwx------ 2 root root 4,0K сен  9 19:08 .
drwx------ 4 root root 4,0K сен  9 19:08 ..
-rwx------ 1 root root  108 сен  9 19:08 BOOTX64.CSV
-rwx------ 1 root root  126 сен  9 19:08 grub.cfg
-rwx------ 1 root root 1,7M сен  9 19:08 grubx64.efi
-rwx------ 1 root root 837K сен  9 19:08 mmx64.efi
-rwx------ 1 root root 934K сен  9 19:08 shimx64.efi
```

```
kuro@kuro-VirtualBoxZorinOS:~/Desktop$ sudo ls -lah /boot/efi
total 12K
drwx------ 3 root root 4,0K янв  1  1970 .
drwxr-xr-x 4 root root 4,0K сен  9 19:23 ..
drwx------ 4 root root 4,0K сен  9 19:08 EFI
```

Questions to answer:

1. It is related to the way Secure Boot works which is by using a set of keys embedded in the computer's firmware. These keys (or more precisely, their private counterparts) are used to sign boot loaders, drivers, option ROMs, and other software that the firmware runs. Most commodity PCs sold today include keys that Microsoft controls. Thus, to be able to install/run any Linux distribution, you must disable Secure Boot, find a Linux boot loader that's signed with Microsoft's keys, or replace your computer's standard keys with ones that you control. Using a pre-signed boot loader, such as the popular Shim program, is one of the more popular ways to do that, because using a pre-signed boot loader with the default key set means that your computer will accept as valid Microsoft's boot loaders and any others that Microsoft decides to sign.

2. The `GRUB` configuration file (`grub.cfg`) is located in `/boot/grub`.

```
kuro@kuro-VirtualBoxZorinOS:/boot/grub$ readlink -f grub.cfg
/boot/grub/grub.cfg
```

3. According to the boot order, the third boot device on my virtual machine is the virtual CD-ROM. The info can be checked using `efibootmgr -v`.

```
kuro@kuro-VirtualBoxZorinOS:~/Desktop$ efibootmgr -v
BootCurrent: 0004
Timeout: 0 seconds
BootOrder: 0004,0000,0001,0002,0003
Boot0000* UiApp FvVol(7cb8bdc9-f8eb-4f34-aaea-3ee4af6516a1)/FvFile(462caa21-7614
-4503-836e-8ab6f4662331)
Boot0001* UEFI VBOX CD-ROM VB2-01700376         PciRoot(0x0)/Pci(0x1,0x1)/Ata(1,
0,0)N.....YM....R,Y.
Boot0002* UEFI VBOX HARDDISK VBb3248520-ccb2b45f        PciRoot(0x0)/Pci(0xd,0x0
)/Sata(0,65535,0)N.....YM....R,Y.
Boot0003* EFI Internal Shell    FvVol(7cb8bdc9-f8eb-4f34-aaea-3ee4af6516a1)/FvFi
le(7c04a583-9e3e-4f1c-ad65-e05268d0b4d1)
Boot0004* ubuntu        HD(1,GPT,5598ae77-deae-4bf4-8d44-b6beff389de9,0x800,0x10
0000)/File(\EFI\ubuntu\shimx64.efi)
```

Exercise 3 - Filesystem:

```
kuro@kuro-VirtualBoxZorinOS:/boot/grub$ lsblk -f
NAME FSTYPE LABEL UUID                                 FSAVAIL FSUSE% MOUNTPOINT
sda
├─sda1
│    vfat         3BB0-DDBB                            505,8M     1% /boot/efi
└─sda2
     ext4         866d1769-2b9f-417e-82f0-d7a56166781d   14,3G    35% /
sr0
```

```
kuro@kuro-VirtualBoxZorinOS:/boot/grub$ ls -lah /dev
total 4,0K
drwxr-xr-x  19 root root        4,0K сен  9 20:06 .
drwxr-xr-x  20 root root        4,0K сен  9 19:06 ..
crw-r--r--   1 root root     10, 235 сен  9 20:06 autofs
drwxr-xr-x   2 root root         280 сен  9 20:06 block
drwxr-xr-x   2 root root          80 сен  9 20:06 bsg
crw-------   1 root root     10, 234 сен  9 20:06 btrfs-control
drwxr-xr-x   3 root root          60 сен  9 20:06 bus
lrwxrwxrwx   1 root root           3 сен  9 20:06 cdrom -> sr0
drwxr-xr-x   2 root root        3,6K сен  9 20:06 char
crw--w----   1 root tty       5,   1 сен  9 20:06 console
lrwxrwxrwx   1 root root          11 сен  9 20:06 core -> /proc/kcore
drwxr-xr-x   3 root root          60 сен  9 20:06 cpu
crw-------   1 root root     10, 124 сен  9 20:06 cpu_dma_latency
crw-------   1 root root     10, 203 сен  9 20:06 cuse
drwxr-xr-x   7 root root         140 сен  9 20:06 disk
drwxr-xr-x   2 root root          60 сен  9 20:06 dma_heap
drwxr-xr-x   3 root root         100 сен  9 20:06 dri
lrwxrwxrwx   1 root root           3 сен  9 20:06 dvd -> sr0
crw-------   1 root root     10, 126 сен  9 20:06 ecryptfs
crw-rw----   1 root video    29,   0 сен  9 20:06 fb0
```

```
kuro@kuro-VirtualBoxZorinOS:/boot/grub$ cat /proc/meminfo
MemTotal:        2010608 kB
MemFree:          824312 kB
MemAvailable:    1305100 kB
Buffers:           42944 kB
Cached:           556040 kB
SwapCached:            0 kB
Active:           309120 kB
Inactive:         692768 kB
Active(anon):       1196 kB
Inactive(anon):   409968 kB
Active(file):     307924 kB
Inactive(file):   282800 kB
Unevictable:          32 kB
Mlocked:              32 kB
SwapTotal:       1190388 kB
SwapFree:        1190388 kB
Dirty:                 0 kB
Writeback:             0 kB
AnonPages:        402968 kB
Mapped:           160360 kB
Shmem:              8260 kB
```

Questions to answer:

1. A lot.

```
kuro@kuro-VirtualBoxZorinOS:/boot/grub$ df -i
Filesystem      Inodes  IUsed   IFree IUse% Mounted on
udev            235807    456  235351    1% /dev
tmpfs           251326    686  250640    1% /run
/dev/sda2      1605632 273784 1331848   18% /
tmpfs           251326      1  251325    1% /dev/shm
tmpfs           251326      2  251324    1% /run/lock
tmpfs           251326     19  251307    1% /sys/fs/cgroup
/dev/sda1            0      0       0     - /boot/efi
tmpfs           251326     48  251278    1% /run/user/1000
```

2. The file system of my EFI partition is of the type VFAT, an extension of the FAT file system.

```
kuro@kuro-VirtualBoxZorinOS:/boot/grub$ df -Th
Filesystem     Type      Size  Used Avail Use% Mounted on
udev           devtmpfs  922M     0  922M   0% /dev
tmpfs          tmpfs     197M  1,1M  196M   1% /run
/dev/sda2      ext4       24G  8,5G   15G  38% /
tmpfs          tmpfs     982M     0  982M   0% /dev/shm
tmpfs          tmpfs     5,0M     0  5,0M   0% /run/lock
tmpfs          tmpfs     982M     0  982M   0% /sys/fs/cgroup
/dev/sda1      vfat      511M  5,3M  506M   2% /boot/efi
tmpfs          tmpfs     197M   12K  197M   1% /run/user/1000
```

3. Mounted at my / directory is /dev/sda2 which belongs to my virtual box harddisk.

```
kuro@kuro-VirtualBoxZorinOS:/boot/grub$ sudo fdisk -l
Disk /dev/sda: 25 GiB, 26843545600 bytes, 52428800 sectors
Disk model: VBOX HARDDISK
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: gpt
Disk identifier: 0B5BE532-4874-4961-BD78-43E910A12FFA

Device       Start      End  Sectors  Size Type
/dev/sda1     2048  1050623  1048576  512M EFI System
/dev/sda2  1050624 52426751 51376128 24,5G Linux filesystem
kuro@kuro-VirtualBoxZorinOS:/boot/grub$ df -h /mnt
Filesystem      Size  Used Avail Use% Mounted on
/dev/sda2        24G  8,5G   15G  38% /
```

4. Can be checked using `blkid | grep UUID=`, the `grep` command filters out results containing `UUID=`.

```
kuro@kuro-VirtualBoxZorinOS:/boot/grub$ sudo blkid | grep UUID=
/dev/sda2: UUID="866d1769-2b9f-417e-82f0-d7a56166781d" TYPE="ext4" PARTUUID="8c9420cc-7def-47d8-854d
-09fd8ae8309f"
/dev/sda1: UUID="3BB0-DDBB" TYPE="vfat" PARTLABEL="EFI System Partition" PARTUUID="5598ae77-deae-4bf
4-8d44-b0beff389de9"
```

5. Some of the methods are:

   - Using `blkid`
   - Using `lsblk -f`
   - Using `ls -l /dev/disk/by-uuid`

```
kuro@kuro-VirtualBoxZorinOS:/boot/grub$ sudo blkid | grep UUID=
/dev/sda2: UUID="866d1769-2b9f-417e-82f0-d7a56166781d" TYPE="ext4" PARTUUID="8c9420cc-7def-47d8-854d
-09fd8ae8309f"
/dev/sda1: UUID="3BB0-DDBB" TYPE="vfat" PARTLABEL="EFI System Partition" PARTUUID="5598ae77-deae-4bf
4-8d44-b6beff389de9"
```

```
kuro@kuro-VirtualBoxZorinOS:/boot/grub$ lsblk -f
NAME   FSTYPE LABEL UUID                                 FSAVAIL FSUSE% MOUNTPOINT
sda
├─sda1 vfat         3BB0-DDBB                            505,8M     1% /boot/efi
└─sda2 ext4         866d1769-2b9f-417e-82f0-d7a56166781d   14,3G    35% /
sr0
```

```
kuro@kuro-VirtualBoxZorinOS:/boot/grub$ ls -l /dev/disk/by-uuid
total 0
lrwxrwxrwx 1 root root 10 сен  9 20:06 3BB0-DDBB -> ../../sda1
lrwxrwxrwx 1 root root 10 сен  9 20:06 866d1769-2b9f-417e-82f0-d7a56166781d -> ../../sda2
```

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

End of Exercises