MBR, GPT, UUID Information

Find Block Devices

Fist step is to figure out which device is our SD Card. Use the 'lsblk' to list all the block devices currently connected to the system:

```
sudo lsblk -o NAME,FSTYPE,SIZE,MOUNTPOINT,LABEL
```

This should give an output similar to the following, illustrating that the SD card is connected to /dev/sdb:

```
AME   FSTYPE   SIZE MOUNTPOINT             LABEL
sda           232,9G
├─sda1 ntfs     100M                        System Reserved
├─sda2 ntfs    69,9G                        WINDOWS 7
├─sda3 ext4   153,7G /
└─sda4 swap     9,2G [SWAP]
sdb            29,8G
├─sdb1 ext4    28,9G
└─sdb2 swap     948M
```

GPT vs MBR

GPT stands for Globally Unique Identifier (GUID) Partition Table, which is a standard for the layout of the partition table on a physical hard disk. GPT is gradually replacing the old MBR (Master Boot Record) standard, which has some limitations that only supports hard disk up to 2TB in size, and up to four primary partitions.

- GPT supports hard drive lager than 2TB.
- Windows can only boot from GPT on a UEFI-based computers.
- Unlike MBR, GPT hard disk allows you to create primary partitions up to 128 due to system limitations.
- GPT disks only support 64-bit Windows system while MBR disks support both 32-bit and 64-bit Windows

Identify Partition Table type using gdisk (or fdisk):

```
# gdisk -l /dev/sda
```

Output should show something similar to:

```
Partition table scan:
  MBR: protective
  BSD: not present
  APM: not present
  GPT: present
Found valid GPT with protective MBR; using GPT.
```

Or with fdisk:

```
# fdisk -l /dev/sda
```

For MBR partition types, output should show something similar to:

```
Disk /dev/sda: 953,9 GiB, 1024209543168 bytes, 2000409264 sectors
Disk model: Samsung SSD 850
* * *
Disklabel type: dos
```

For GPT partition types, output should show something similar to:

```
Disk /dev/sda: 953,9 GiB, 1024209543168 bytes, 2000409264 sectors
Disk model: Samsung SSD 850
* * *
Disklabel type: gpt
```

GPT Partition Table

Backup GPT

```
sgdisk --backup=filename /dev/sda
```

Restore GPT

```
sgdisk --load-backup=filename /dev/sdb
sgdisk -G /dev/sdb
```

The option G (–randomize-guids)Randomize the disk's GUID and all partitions' unique GUIDs (but not their partition type code GUIDs). This function may be used after cloning a disk in order to render all GUIDs once again unique.

MBR Partition Table

A master boot record (MBR) is the 512-byte boot sector that is the first sector of a partitioned data storage device of a hard disk. The total size is 512 bytes (446 + 64 + 2 = 512). Where,

```
  446 bytes – Bootstrap.
  64 bytes – Partition table.
  2 bytes – Signature.
```

Backup MBR

Next command will copy 512 bytes (MBR) from disk sdX to a backup file:

```
# dd if=/dev/sdX of=mbr.backup.img bs=512 count=1
```

Restore MBR

For disks with identical partitions and size:

Use 512 bytes to overwrite or restore your /dev/sdX the full MBR (which contains both boot code and the drive’s partition table) with the contents of mbr.backup.file.

```
# dd if=/mbr.backup.img of=/dev/sdX bs=512 count=1
```

For disks with different partitions and size:

Use 446 bytes to overwrite or restore your /dev/sdX MBR boot code only with the contents of mbr.backup.file. This will preserve the partitioning schema.

```
# dd if=/mbr.backup.img of=/dev/sdX bs=446 count=1
```

Be aware of UUID

UUIDs are not hardware-specific but stored in the partition's filesystem. That means cloning a disk or partition with dd will result in the same UUID. However recreating the partitions manually on the new disk (e.g. smaller disk), will result in new UUID. This could result in problems when booting the new disk uses UUID in fstab.

```
cat /etc/fstab
# /etc/fstab: static file system information.
# <file system> <mount point>   <type>  <options>       <dump>  <pass>
# / was on /dev/sda3 during installation
UUID=b2fa29ee-670f-4d44-becc-d9ec368d4a41 /               ext4
noatime,nodiratime,errors=remount-ro 0       1
```

This can be solved by either changing the entries in fstab to the old style: /dev/dbX, Or by getting the new UUID and update fstab accordingy:

```
# blkid /dev/sdb1
/dev/sdb1: UUID="34628ffd-58e6-4a58-9b4d-533719305931" TYPE="ext4"
PARTUUID="fa64ccff-01"
```