**RHEL7: List, create, delete partitions on MBR and GPT disks.**

**Presentation**

A **disk** can be used as a simple entity or broken up into one or more **partitions**.

**Disks** are generally called **/dev/sda**, **/dev/sdb**, etc, in **physical** servers (**s** for **scsi** even though they’ve got **IDE**, **SATA** or **SAS** interfaces) and **/dev/vda**, **/dev/vdb**, etc, in **virtual** machines.

**Partitions** get their names from the **disk** name itself and add a number starting at **1** (**/dev/sda1**, **/dev/sda2**, etc or **/dev/vda1**, **/dev/vda2**, etc).

A **partition table** is a special structure containing partitions organization.

Not recent disks use **512-byte sectors** and the **MBR** partition table (**MBR** stands for Master Boot Record). This organization allows for **4** **primary** partitions only. If you want more than that, you need to create an **extended** partition (using one of the 4 primary slots), and then create **logical** partitions inside. More annoying, on disks with capacity greater than **2TB**, space above this limit is not available.

To work around all these limitations, recent disks use **4096-byte sectors** and the **GPT** partition table (**GPT** stands for GUID – Globally Unique IDentifier – Partition Table). More details are available on the [GPT Wikipedia page](http://en.wikipedia.org/wiki/GUID_Partition_Table).

Historically, two commands exist to manipulate disks and partitions: **fdisk** and **parted**.  
As the **fdisk** command doesn’t handle **GPT** partition tables, it is not advisable to use it any more (for your information, some details are given at the end of this page about the **fdisk** command).  
Recently, a new tool called **gdisk** has been created to deal with **GPT** partition tables, offering an alternative to the **parted** command.

**Caution:** In this tutorial, we are dealing with real disks. Any mistake could **entirely destroy** your system.

**The parted Command**

To start the **parted** command, type:

# **parted**

GNU Parted 2.1

Using /dev/sda

Welcome to GNU Parted! Type 'help' to view a list of commands.

(parted)

To list all the disks and partitions, type:

(parted) **print all**

Model: ATA Hitachi HDP72505 (scsi)

Disk /dev/sda: 500GB

Sector size (logical/physical): 512B/512B

Partition Table: msdos

Number Start End Size Type File system Flags

1 1049kB 525MB 524MB primary ext4 boot

2 525MB 500GB 500GB primary lvm

**Model: ATA Hitachi HDP72505 (scsi)**

**Disk /dev/sdb: 500GB**

**Sector size (logical/physical): 512B/512B**

**Partition Table: msdos**

**Number Start End Size Type File system Flags**

Model: Linux device-mapper (linear) (dm)

Disk /dev/mapper/vg\_root-lv\_root: 497GB

Sector size (logical/physical): 512B/512B

Partition Table: loop

Number Start End Size File system Flags

1 0.00B 497GB 497GB ext4

Model: Linux device-mapper (linear) (dm)

Disk /dev/mapper/vg\_root-lv\_swap: 2147MB

Sector size (logical/physical): 512B/512B

Partition Table: loop

Number Start End Size File system Flags

1 0.00B 2147MB 2147MB linux-swap(v1)

Here, we’ve got a disk called **/dev/sdb** without partition but with a **MBR** partition table (Partition Table: msdos).  
To select the **/dev/sdb** disk, type:

(parted) **select /dev/sdb**

Using /dev/sdb

To create a **GPT** partition table on the **/dev/sdb** disk, type:

(parted) **mktable gpt**

Warning: The existing disk label on /dev/sdb will be destroyed and all data on

this disk will be lost. Do you want to continue?

Yes/No? **Y**

(parted) **print**

Model: ATA Hitachi HDP72505 (scsi)

Disk /dev/sdb: 500GB

Sector size (logical/physical): 512B/512B

Partition Table: gpt

Number Start End Size File system Name Flags

Note: Type **mktable msdos** to create a **MBR** partition table.

To create a primary partition with the **ext4** type (here starting at **1MB** and finishing at **400GB**), type:

(parted) **mkpart primary ext4 1MB 400GB**

Note1: Specifying **ext4** doesn’t format the partition in **ext4**, it only tags it as **ext4** partition.  
Note2: The partition doesn’t start at **0** but **1MB** to avoid disk alignment problems.  
Note3: To specify all the remaining space, use **-1** as end position.  
Note4: With **parted**, **1GB**=**1000MB**.

Sometimes when setting up the first partition, you will get a warning: “**Warning: The resulting partition is not properly aligned for best performance.**”  
To solve this problem, type the following command where **400GB** is the size of the partition (you can use **100%** instead of **400GB** if you want to allocate all the disk):

(parted) **mkpart primary ext4 0% 400GB**

To check that the first partition is correctly aligned, type:

(parted) **align-check optimal 1**

1 aligned

To create a **swap** partition with a size of **2GB** (here starting at **400GB** and finishing at **402GB**), type:

(parted) **mkpart primary linux-swap 400GB 402GB**

Note1: **parted** checks that both partitions don’t overlap.  
Note2: If, at a later stage, you want to change the type of partition, don’t drop and recreate the partition: format the partition as you want and **parted** will normally detect the new type.

To print the result, type:

(parted) **print**

Model: ATA Hitachi HDP72505 (scsi)

Disk /dev/sdb: 500GB

Sector size (logical/physical): 512B/512B

Partition Table: gpt

Number Start End Size File system Name Flags

1 1049kB 400GB 400GB ext4 primary

2 400GB 402GB 2000MB primary

To set the **first** partition as **bootable**, type:

(parted) **set 1 boot on**

(parted) **print**

Model: ATA Hitachi HDP72505 (scsi)

Disk /dev/sdb: 500GB

Sector size (logical/physical): 512B/512B

Partition Table: gpt

Number Start End Size File system Name Flags

1 1049kB 400GB 400GB ext4 primary **boot**

2 400GB 402GB 2000MB primary

Note: Type **set 1 boot off** to remove the **bootable** flag.

To remove the **swap** partition (here partition number **2**), type:

(parted) **rm 2**

To exit the **parted** prompt, type:

(parted) **quit**

To update the disk configuration seen by the kernel, type:

# **partprobe /dev/sdb**

The **parted** command can also be used for **file system** management. However, this usage is deprecated.

**The gdisk Command**

Install the **gdisk** package:

# **yum install -y gdisk**

Execute the **gdisk** command (here with the **/dev/vda** disk as parameter):

# **gdisk /dev/vda**

GPT fdisk (gdisk) version 0.8.6

Partition table scan:

MBR: MBR only

BSD: not present

APM: not present

GPT: not present

Type **?** to display all the options:

Command (? for help): **?**

b back up GPT data to a file

c change a partition's name

d delete a partition

i show detailed information on a partition

l list known partition types

n add a new partition

o create a new empty GUID partition table (GPT)

p print the partition table

q quit without saving changes

r recovery and transformation options (experts only)

s sort partitions

t change a partition's type code

v verify disk

w write table to disk and exit

x extra functionality (experts only)

? print this menu

Type **p** to print the partition table:

Command (? for help): **p**

Disk /dev/vda: 12582912 sectors, 6.0 GiB

Logical sector size: 512 bytes

Disk identifier (GUID): C6F7C323-530D-40B5-A985-241A1B181354

Partition table holds up to 128 entries

First usable sector is 34, last usable sector is 12582878

Partitions will be aligned on 2048-sector boundaries

Total free space is 1318845 sectors (644.0 MiB)

Number Start (sector) End (sector) Size Code Name

1 2048 1026047 500.0 MiB 8300 Linux filesystem

2 1026048 11266047 4.9 GiB 8E00 Linux LVM

Type **n** to create a new partition:

Command (? for help): **n**

Partition number (3-128, default 3): **3**

First sector (34-12582878, default = 11266048) or {+-}size{KMGTP}: **34**

Last sector (34-2047, default = 2047) or {+-}size{KMGTP}: **2047**

Current type is 'Linux filesystem'

Hex code or GUID (L to show codes, Enter = 8300): **8300**

Changed type of partition to 'Linux filesystem'

Type **p** to display the partition table:

Command (? for help): **p**

Disk /dev/vda: 12582912 sectors, 6.0 GiB

Logical sector size: 512 bytes

Disk identifier (GUID): C6F7C323-530D-40B5-A985-241A1B181354

Partition table holds up to 128 entries

First usable sector is 34, last usable sector is 12582878

Partitions will be aligned on 2048-sector boundaries

Total free space is 1316831 sectors (643.0 MiB)

Number Start (sector) End (sector) Size Code Name

1 2048 1026047 500.0 MiB 8300 Linux filesystem

2 1026048 11266047 4.9 GiB 8E00 Linux LVM

3 34 2047 1007.0 KiB 8300 Linux filesystem

Type **w** to write the partition table to disk:

Command (? for help): **w**

Final checks complete. About to write GPT data. THIS WILL OVERWRITE EXISTING

PARTITIONS!!

Do you want to proceed? (Y/N): **y**

OK; writing new GUID partition table (GPT) to /dev/vda.

The operation has completed successfully.

To force the kernel to read the updated **partition table**, type:

# **partprobe**

Source: [Sander van Vugt’s video about gdisk (5min/2014)](https://www.youtube.com/watch?v=V5_d_3oZCLs).

**The fdisk Command**

To list all the partitions, type:

# **fdisk -l**

To create a **primary** partition on a disk (here **/dev/vda**), type:

# **fdisk /dev/vda**

Press ‘**c**‘, ‘**u**‘, then ‘**p**‘ to print the partition table.  
Then press ‘**n**‘ (for new), type the partition number (between **1** and **4**), the first sector and the size.  
Finally, press ‘**w**‘ to save the partition table.

To delete a **primary** partition on a disk (here **/dev/vda**), type:

# **fdisk /dev/vda**

Press ‘**c**‘, ‘**u**‘, then ‘**p**‘ to print the partition table.  
Then press ‘**d**‘ (for delete) and type the partition number (between **1** and **4**).  
Finally, press ‘**w**‘ to save the partition table.

To set the type of a **primary** partition (here **/dev/vda3**), type:

# **fdisk /dev/vda**

Press ‘**c**‘, ‘**u**‘, then ‘**p**‘ to print the partition table.  
Then press ‘**t**‘ (for tag), type the partition number (here ‘**3**‘) and the partition type (**83** for linux, **8e** for Linux LVM, **82** for swap).  
Finally, press ‘**w**‘ to save the partition table.

To force the kernel to read the updated **partition table**, type:

# **partprobe**