



# Partitioning Disks with parted

Author: Yogesh Babar  
Technical Reviewer: Chris Negus  
10/6/2017

Storage devices in Linux (such as hard drives and USB drives) need to be structured in some way before use. In most cases, large storage devices are divided into separate sections, which in Linux are referred to as partitions.

A popular tool for creating, removing and otherwise manipulating disk partitions in Linux is the **parted** command. Procedures in this tech brief step you through different ways of using the **parted** command to work with Linux partitions.

## UNDERSTANDING PARTED

The **parted** command is particularly useful with large disk devices and many disk partitions. Differences between **parted** and the more common **fdisk** and **cdisk** commands include:

- **GPT Format:** The parted command can be used to create Globally Unique Identifiers Partition Tables (GPT), while fdisk and cfdisk are limited to msdos partition tables.
- **Larger disks:** An msdos partition table can only format up to 2TB of disk space (although up to 16TB is possible in some cases). A GPT partition table, however, have the potential to address up to 8 zebibytes of space.
- **More partitions:** Using primary and extended partitions, msdos partition tables allow only 16 partitions. With GPT, you get up to 128 partitions by default and can choose to have many more.
- **Reliability:** Only one copy of the partition table is stored in an msdos partition. GPT keeps two copies of the partition table (at the beginning and end of the disk). The GPT also uses a CRC checksum to check the partition table's integrity (which is not done with msdos partitions).

With today's larger disks and need for more flexibility in working with them, using **parted** to work with disk partitions is recommended practice these days.

**NOTE:** At the time of this writing, experimental support for GPT partition tables is being added to **fdisk** and **cdisk** commands.

## HOW TO MAKE A PARTITION TABLE

Most disk partition tables are created as part of the operating system installation process. Direct use of the **parted** command is most useful when adding an additional storage device



to an existing system. Therefore:

**WARNING:** After you add a new storage device, be sure to identify the correct disk before you begin changing its partition table. If you mistakenly change the disk partition that contains your computer's operating system, you could make your system unbootable! To try the steps in this procedure, we recommend you add a storage device on which you don't mind wiping out the contents.

The following procedure steps you through the process of partitioning a storage device with the **parted** command:

1. **List partitions:** Use **parted -l** to identify the storage device you want to partition. Typically the first hard disk (**/dev/sda** or **/dev/vda**) will contain the operating system so look for another disk to find the one you want (**/dev/sdb**, **/dev/sdc**, **/dev/vdb**, **/dev/vdc**, and so on):

```
# parted -l
```

2. **Open storage device:** Use **parted** to begin working with the selected storage device. In this example, the device is the third disk on a virtual system (**/dev/vdc**):

**NOTE:** It is important to indicate the specific device you want to use. If you just type **parted** with no device name, it will randomly select a storage device to modify.

```
# parted /dev/vdc
(parted)
```

3. **Set partition table:** Set the partition table type to gpt (then type Yes to accept it):

```
(parted) mklabel gpt
```

```
Warning: the existing disk label on /dev/vdc will be destroyed
and all data on this disk will be lost. Do you want to continue?
Yes/No? Yes
```

The **mklabel** and **mktable** parted commands are used for the same purpose. The supported partition tables are:

```
aix, amiga, bsd, dvh, gpt, mac, msdos, pc98, sun, loop
```

Remember 'mklabel' will not make a partition rather it will make a partition table.

4. **Review partition table:** Show information about the storage device:

```
(parted) print
Model: Virtio Block Device (virtblk)
Disk /dev/vdc: 1396MB
Sector size (logical/physical): 512B/512B
Partition Table: gpt
Disk Flags:
Number  Start  End  Size  File system  Name  Flags
```

5. **Get help:** To find how to make a new partition, type:

```
(parted) help mkpart
```



```
mkpart PART-TYPE [FS-TYPE] START END    make a partition
PART-TYPE is one of: primary, logical, extended
FS-TYPE is one of: btrfs, nilfs2, ext4, ext3, ext2, fat32, fat16, hfsx,
hfs+, hfs, jfs, swsusp, linux-swap(v1), linux-swap(v0), ntfs, reiserfs,
hp-ufs, sun-ufs, xfs, apfs2, apfs1, asfs, amufs5, amufs4, amufs3,
amufs2, amufs1, amufs0, amufs, affs7, affs6, affs5, affs4, affs3, affs2,
affs1, affs0, linux-swap, linux-swap(new), linux-swap(old)
START and END are disk locations, such as 4GB or 10%. Negative values
count from the end of the disk. For example, -1s specifies exactly the
last sector.
'mkpart' makes a partition without creating a new file system on the
partition. FS-TYPE may be specified to set an appropriate partition ID.
```

6. **Make partition:** To make a new partition (1396MB on partition 0), type the following:

```
(parted) mkpart primary 0 1396MB
Warning: The resulting partition is not properly aligned for best performance
Ignore/Cancel? I
(parted) print
Model: Virtio Block Device (virtblk)
Disk /dev/vdc: 1396MB
Sector size (logical/physical): 512B/512B
Partition Table: gpt
Disk Flags:
Number  Start   End     Size   File system  Name      Flags
  1      17.4kB 1396MB 1396MB                primary
```

File system type will not create an ext4 filesystem on /dev/vdc1. Rather it will just set the fstype to it.

**NOTE:** The primary, logical, and extended are the partition types of an **msdos** partition table. In the case of a **gpt** partition table, the partition type is used as the partition name. Providing a partition name under **gpt** is a must. So in the above example, **primary** is the name, not the partition type.

7. **Rename partition:** Set the partition to a new name with the **name** command (GPT only):

```
(parted) print
Model: Virtio Block Device (virtblk)
Disk /dev/vdc: 1395865kB
Sector size (logical/physical): 512B/512B
Partition Table: gpt
Disk Flags:
Number  Start     End        Size      File system  Name      Flags
  1      1049kB   500171kB  499122kB                primary
(parted) name 1 'yogesh'
```



```
(parted) print
Model: Virtio Block Device (virtblk)
Disk /dev/vdc: 1395865kB
Sector size (logical/physical): 512B/512B
Partition Table: gpt
Disk Flags:
Number  Start   End     Size    File system  Name  Flags
  1      1049kB 500171kB 499122kB                yogesh
```

8. **Change unit type:** The default unit size displayed by parted is MB. Use the unit command to change the default unit size. Supported units are s, B, kB, MB, GB, TB, compact, cyl, chs, %, kiB, MiB, GiB, TiB:

```
(parted) print
Model: Virtio Block Device (virtblk)
Disk /dev/vdc: 1396MB
Sector size (logical/physical): 512B/512B
Partition Table: msdos
Disk Flags:
Number  Start   End     Size    Type        File system  Flags
  1      0.00MB 698MB   698MB   primary
```

```
(parted) unit s
```

```
(parted) print
Model: Virtio Block Device (virtblk)
Disk /dev/vdc: 2726298s
Sector size (logical/physical): 512B/512B
Partition Table: msdos
Disk Flags:
Number  Start   End     Size    Type        File system  Flags
  1      1s     1363148s 1363148s primary
```

```
(parted) unit KB
```

```
(parted) print
Model: Virtio Block Device (virtblk)
Disk /dev/vdc: 1395865kB
Sector size (logical/physical): 512B/512B
Partition Table: msdos
Disk Flags:
Number  Start   End     Size    Type        File system  Flags
  1      0.51kB 697932kB 697932kB primary
```

9. **Add partition with unit selector:** To add an ext4 type of partition and set the size in sectors (1363149s in this example): type the following:



```
(parted) mkpart primary ext4 1363149s 100%
Warning: The resulting partition is not properly aligned for best performance
Ignore/Cancel? Ignore
(parted) print
Model: Virtio Block Device (virtblk)
Disk /dev/vdc: 1395865kB
Sector size (logical/physical): 512B/512B
Partition Table: msdos
Disk Flags:
Number  Start      End          Size         Type         File system  Flags
  1      0.51kB    697932kB    697932kB    primary
  2      697932kB  1395865kB   697932kB    primary
```

10. **Save and quit:** Changes you made are automatically saved when you quit **parted**. To quit, type the following:

```
(parted) quit
Information: You may need to update /etc/fstab.
#
```

## COMPARE GPT AND MSDOS PARTITION TABLES

A key difference between msdos partition tables and gpt partition tables is where they are located on a disk. By changing the partition table to msdos on the disk we are using as an example, notice the different starting point of the partition table:

1. **Change to msdos table:** Use **mklabel** to change the partition table to msdos as follows:

```
# parted /dev/vdc
(parted) mklabel msdos
Warning: The existing disk label on /dev/vdc will be destroyed and all data
on this disk will be lost. Do you want to continue?
Yes/No? Yes
(parted) print
Model: Virtio Block Device (virtblk)
Disk /dev/vdc: 1396MB
Sector size (logical/physical): 512B/512B
Partition Table: msdos
Disk Flags:
Number  Start  End  Size  Type  File system  Flags
```

2. **Create a partition:** Create a 1000M partition on the msdos partition table as follows:

```
(parted) mkpart primary ext4 0M 1000M
```



Warning: The resulting partition is not properly aligned for best performance  
Ignore/Cancel? **Ignore**

(parted) print

Model: Virtio Block Device (virtblk)

Disk /dev/vdc: 1396MB

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

Partition Table: msdos

Disk Flags:

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

1	512B	1000MB	1000MB	primary		
---	------	--------	--------	---------	--	--

<<<<----- Note the starting point at 512B

### 3. Change to gpt table: Use **mklabel** to change the partition table to gpt as follows:

(parted) **mklabel gpt**

Warning: The existing disk label on /dev/vdc will be destroyed and all data on this disk will be lost. Do you want to continue?

Yes/No? Yes

(parted) **print**

Model: Virtio Block Device (virtblk)

Disk /dev/vdc: 1396MB

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

Partition Table: gpt

Disk Flags:

Number	Start	End	Size	File system	Name	Flags
--------	-------	-----	------	-------------	------	-------

### 4. Create a partition: Create a 1000M partition on the gpt partition table as follows:

(parted) **mkpart primary ext4 0m 1000M**

Warning: The resulting partition is not properly aligned for best performance  
Ignore/Cancel? **Ignore**

(parted) **print**

Model: Virtio Block Device (virtblk)

Disk /dev/vdc: 1396MB

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

Partition Table: gpt

Disk Flags:

Number	Start	End	Size	File system	Name	Flags
--------	-------	-----	------	-------------	------	-------

1	17.4kB	1000MB	1000MB	ext4	primary	<<<--- Note the starting point of partition is 17.4KB
---	--------	--------	--------	------	---------	---



## SET PARTITION FLAGS

By setting a flag on a partition, you can identify the state of a feature on that partition. Flags are assigned using the **set** command in **parted**:

1. **Get help:** To see a description of the set command in parted, type the following:

```
(parted) help set
set NUMBER FLAG STATE          change the FLAG on partition NUMBER
NUMBER is the partition number used by Linux.  On MS-DOS disk labels, the
primary partitions number from 1 to 4, logical partitions from 5 onwards.
FLAG is one of: boot, root, swap, hidden, raid, lvm, lba, hp-service, palo,
prep, msftres, bios_grub, atvrecv, diag, legacy_boot
STATE is one of: on, off
```

2. **Set boot flag:** To make a partition bootable, set the **boot** flag to **1**:

```
(parted) set 1 boot on
(parted) print
Model: Virtio Block Device (virtblk)
Disk /dev/vdc: 1396MB
Sector size (logical/physical): 512B/512B
Partition Table: gpt
Disk Flags:
Number  Start   End     Size   File system  Name      Flags
  1      17.4kB 500MB   500MB                primary  boot
```

3. **Create partitions:** Make several different partitions that you can use to try out setting flags. This example creates an 800M root (*/*) partition, a 200M swap partition, and assigns the rest of the disk space to the boot (*/boot*) partition:

```
(parted) mkpart primary ext4 0 800M          <<<--- root (/) partition
Warning: The resulting partition is not properly aligned for best performance
Ignore/Cancel? Ignore
(parted) print
Model: Virtio Block Device (virtblk)
Disk /dev/vdc: 1396MB
Sector size (logical/physical): 512B/512B
Partition Table: gpt
Disk Flags:
Number  Start   End     Size   File system  Name      Flags
  1      17.4kB 800MB   800MB                primary
(parted) mkpart primary linux-swap 800M 1000M <<<--- swap partition
(parted) print
Model: Virtio Block Device (virtblk)
```

```

Disk /dev/vdc: 1396MB
Sector size (logical/physical): 512B/512B
Partition Table: gpt
Disk Flags:
Number  Start   End     Size   File system  Name      Flags
  1      17.4kB 800MB   800MB                primary
  2      800MB  1000MB  200MB                primary
(parted) mkpart primary ext4 1000M 100%      <<<--- boot (/boot) partition
(parted) print
Model: Virtio Block Device (virtblk)
Disk /dev/vdc: 1396MB
Sector size (logical/physical): 512B/512B
Partition Table: gpt
Disk Flags:
Number  Start   End     Size   File system  Name      Flags
  1      17.4kB 800MB   800MB                primary
  2      800MB  1000MB  200MB                primary
  3     1000MB 1396MB  395MB                primary

```

4. **Set bootable partition:** To make the third partition (/boot) bootable, type the following:

```

(parted) set 3 boot on
(parted) print
Model: Virtio Block Device (virtblk)
Disk /dev/vdc: 1396MB
Sector size (logical/physical): 512B/512B
Partition Table: gpt
Disk Flags:
Number  Start   End     Size   File system  Name      Flags
  1      17.4kB 800MB   800MB                primary
  2      800MB  1000MB  200MB                primary
  3     1000MB 1396MB  395MB                primary boot
(parted) quit

```

## CREATE AN ESP (EFI SYSTEM PARTITION)

The following example shows how to create an ESP partition and set it to be bootable:

1. **Create ESP** : Check the current partitions, then create an ESP as follows:

```

(parted) print
Model: Virtio Block Device (virtblk)
Disk /dev/vdc: 1396MB
Sector size (logical/physical): 512B/512B

```

```

Partition Table: gpt
Disk Flags:
Number Start End Size File system Name Flags
(parted) mkpart ESP fat32 0M 200M
Warning: The resulting partition is not properly aligned for best performance
Ignore/Cancel? Ignore
(parted) print
Model: Virtio Block Device (virtblk)
Disk /dev/vdc: 1396MB
Sector size (logical/physical): 512B/512B
Partition Table: gpt
Disk Flags:
Number Start End Size File system Name Flags
 1      17.4kB 200MB 200MB          ESP

```

2. **Make ESP bootable:** Type the following to make the ESP bootable:

```

(parted) set 1 boot on
(parted) print
Model: Virtio Block Device (virtblk)
Disk /dev/vdc: 1396MB
Sector size (logical/physical): 512B/512B
Partition Table: gpt
Disk Flags:
Number Start End Size File system Name Flags
 1      17.4kB 200MB 200MB          ESP boot

```

## RESIZE A PARTITION

Resizing a partition allows you expand the space available on the partition while leaving existing data in place. Because partition resizing can result in data loss, some rules have to be followed:

- Partition should not be mounted or in use.
- Partition can only grow or shrink from the end.
- If growing, the partition has to be resized first then filesystem.
- If reducing, filesystem has to be resized first then the partition.

**NOTE:** The option to resize partitions (`resizepart`) is not available in Red Hat Enterprise Linux 7. It is scheduled to be added in RHEL 7.5 (see [Bug 1423367](#)). In the mean time, you can try this feature using Fedora 26 or later, using the procedure contained in this section.

## Grow a partition

Use the following procedure (on a Fedora 26 or later system) to grow a partition.

1. **Grow partition:** Type the following to list partition information, then grow the size of the first partition from 500M to 700M and exit from **parted**.

```
[root@fedora ~]# parted /dev/vdb
(parted) print
Model: Virtio Block Device (virtblk)
Disk /dev/vdb: 1074MB
Sector size (logical/physical): 512B/512B
Partition Table: gpt
Disk Flags:
Number  Start   End     Size   File system  Name      Flags
  1      17.4kB 500MB   500MB  ext4         primary
(parted) resizepart 1 700M
(parted) print
Model: Virtio Block Device (virtblk)
Disk /dev/vdb: 1074MB
Sector size (logical/physical): 512B/512B
Partition Table: gpt
Disk Flags:
Number  Start   End     Size   File system  Name      Flags
  1      17.4kB 700MB   700MB  ext4         primary
(parted) quit
Information: You may need to update /etc/fstab.
```

2. **Grow filesystem:** To resize the file system to use space on the expanded partition, type the following from the shell:

```
[root@fedora ~]# resize2fs /dev/vdb1
resize2fs 1.42.13 (17-May-2015)
Resizing the filesystem on /dev/vdb1 to 683576 (1k) blocks.
The filesystem on /dev/vdb1 is now 683576 (1k) blocks long.
```

## Shrink a partition

Use the following procedure (on a Fedora 26 or later system) to shrink a partition.

1. **Shrink file system:** Before you can shrink the partition for an existing file system, you must shrink the file system on that partition. Type the following to shrink a partition from 700M to 500M

```
[root@fedora ~]# resize2fs /dev/vdb1 500M
resize2fs 1.42.13 (17-May-2015)
```

```
Resizing the filesystem on /dev/vdb1 to 512000 (1k) blocks.
The filesystem on /dev/vdb1 is now 512000 (1k) blocks long.
[root@fedora ~]#
```

2. **Shrink partition:** Type the following to list partition information, then shrink the size of the first partition from 700M to 500M and exit from **parted**.

```
[root@fedora ~]# parted /dev/vdb
(parted) print
Model: Virtio Block Device (virtblk)
Disk /dev/vdb: 1074MB
Sector size (logical/physical): 512B/512B
Partition Table: gpt
Disk Flags:
Number  Start   End     Size   File system  Name      Flags
  1      17.4kB  700MB   700MB   ext4         primary
(parted) resizepart 1 500M
Warning: Shrinking a partition can cause data loss, are you sure you want to
continue?
Yes/No? Yes
(parted) print
Model: Virtio Block Device (virtblk)
Disk /dev/vdb: 1074MB
Sector size (logical/physical): 512B/512B
Partition Table: gpt
Disk Flags:
Number  Start   End     Size   File system  Name      Flags
  1      17.4kB  500MB   500MB   ext4         primary
```

## REMOVE A PARTITION

Removing a partition can be accomplished with the **rm** command in **parted**.

1. **List partition table:** To open a partition table and list its contents, type the following:

```
# parted /dev/vdb
(parted) print
Model: Virtio Block Device (virtblk)
Disk /dev/vdc: 1396MB
Sector size (logical/physical): 512B/512B
Partition Table: msdos
Disk Flags:
Number  Start   End     Size   Type        File system  Flags
  1      1049kB  500MB   499MB  primary    ext4
  2      500MB   700MB   200MB  primary
```

3	700MB	1396MB	695MB	extended		lba
5	700MB	800MB	99.6MB	logical		
6	800MB	900MB	99.6MB	logical		
7	900MB	1000MB	100MB	logical		

2. **Remove a partition:** To remove the fifth partition from a partition table, type the following:

```
(parted) rm 5
```

```
(parted) print
```

```
Model: Virtio Block Device (virtblk)
```

```
Disk /dev/vdc: 1396MB
```

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

```
Partition Table: msdos
```

```
Disk Flags:
```

Number	Start	End	Size	Type	File system	Flags
1	1049kB	500MB	499MB	primary	ext4	
2	500MB	700MB	200MB	primary		
3	700MB	1396MB	695MB	extended		lba
5	800MB	900MB	99.6MB	logical		
6	900MB	1000MB	100MB	logical		

Note that the logical partitions numbers have changed. If any of these partitions are mounted, you need to check your **/etc/fstab** file to see if you need to change the device names of those partitions. Note also that the primary partition's numbers remain intact

3. **Remove another partition:** To remove the second partition from a partition table, type the following:

```
(parted) print
```

```
Model: Virtio Block Device (virtblk)
```

```
Disk /dev/vdc: 1396MB
```

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

```
Partition Table: msdos
```

```
Disk Flags:
```

Number	Start	End	Size	Type	File system	Flags
1	1049kB	500MB	499MB	primary	ext4	
2	500MB	700MB	200MB	primary		
3	700MB	1396MB	695MB	extended		lba
5	800MB	900MB	99.6MB	logical		
6	900MB	1000MB	100MB	logical		

```
(parted) rm 2
```

```
(parted) print
```

```
Model: Virtio Block Device (virtblk)
```

```
Disk /dev/vdc: 1396MB
```

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

Partition Table: msdos

Disk Flags:

Number	Start	End	Size	Type	File system	Flags
1	1049kB	500MB	499MB	primary	ext4	
3	700MB	1396MB	695MB	extended		lba
5	800MB	900MB	99.6MB	logical		
6	900MB	1000MB	100MB	logical		

## RESCUE A PARTITION

To attempt to recover a partition, you can try the **rescue** command in **parted**. But there is no guarantee it will rescue the deleted partition.