Partitioning

Partitioning a hard drive divides the available space into sections that can be accessed independently. An entire drive may be allocated to a single partition, or multiple ones for cases such as dual-booting, maintaining a **swap** partition, or to logically separate data such as audio and video files.

The required information is stored in a **#Partition table** scheme such as MBR or GPT.

Partition tables are created and modified using one of many **#Partitioning tools** which must be compatible to the chosen scheme of partitioning table. Available tools include **fdisk** and **parted**.

Once created, a partition must be formatted with an appropriate **file system** (**swap** excepted) before data can be written to it.

Related articles

- **File systems**
- **fdisk**
- **parted**
- **fstab**
- **LVM**
- **Swap**
- **Arch boot process**
- **Unified Extensible Firmware Interface**

Contents

Partition table

Note: To print/list existing tables (of a specific device), run `parted /dev/sda print` or `fdisk -l /dev/sda`, where `/dev/sda` is a device name.

There are two main types of partition table available: Master Boot Record (MBR), and GUID Partition Table (GPT). These are described below along with a discussion on how to choose between the two. A third, less common alternative is using a partitionless disk, which is also discussed.

Master Boot Record

The **Master Boot Record** (MBR) is the first 512 bytes of a storage device. It contains an operating system bootloader and the storage device's partition table. It plays an important role in the **boot process** under **BIOS** systems. See **Wikipedia:Master boot record#Disk**

partitioning for the MBR structure.

> **Note:** The MBR is not located in a partition; it is located at the first sector of the device (physical offset 0), preceding the first partition. (The boot sector present on a partitionless device or within an individual partition is called a **Volume boot record** instead.)

Master Boot Record (partition table)

There are 3 types of partitions in the MBR scheme:

- Primary
- Extended
  - Logical

Primary partitions can be bootable and are limited to four partitions per disk or RAID volume. If the MBR partition table requires more than four partitions, then one of the primary partitions needs to be replaced by an **extended** partition containing **logical** partitions within it.

Extended partitions can be thought of as containers for logical partitions. A hard disk can contain no more than one extended partition. The extended partition is also counted as a primary partition so if the disk has an extended partition, only three additional primary

partitions are possible (i.e. three primary partitions and one extended partition). The number of logical partitions residing in an extended partition is unlimited. A system that dual boots with Windows will require for Windows to reside in a primary partition.

The customary numbering scheme is to create primary partitions *sda1* through *sda3* followed by an extended partition *sda4*. The logical partitions on *sda4* are numbered *sda5*, *sda6*, etc.

Master Boot Record (bootstrap code)

The first 440 bytes of MBR are **bootstrap code area**. On BIOS systems it usually contains the first stage of the boot loader. The bootstrap code can be backed up, restored from backup or erased **using dd**.

GUID Partition Table

GUID Partition Table (GPT) is a partitioning scheme that is part of the **Unified Extensible Firmware Interface** specification; it uses **globally unique identifiers** (GUIDs), or UUIDs in the Linux world, to define partitions and **partition types**. It is designed to succeed the **#Master Boot Record** partitioning scheme method.

At the start of a GUID Partition Table disk there is a **protective Master Boot Record** to protect against GPT-unaware software. This protective MBR just like a real MBR has a **bootstrap code area** which can be used for BIOS/GPT booting with boot loaders that support

it.

Choosing between GPT and MBR

GUID Partition Table (GPT) is an alternative, contemporary, partitioning style; it is intended to replace the old Master Boot Record (MBR) system. GPT has several advantages over MBR which has quirks dating back to MS-DOS times. With the recent developments to the formatting tools *fdisk* (MBR) and *gdisk* (GPT), it is equally easy to get good dependability and performance for GPT or MBR.

Note: For GRUB to boot from a GPT-partitioned disk on a BIOS-based system, a **BIOS boot partition** is required.

Some points to consider when choosing:

- To dual-boot with Windows (both 32-bit and 64-bit) using Legacy BIOS, the MBR scheme is required.
- To dual-boot Windows 64-bit using **UEFI** mode instead of BIOS, the GPT scheme is required.
- If you are installing on older hardware, especially on old laptops, consider choosing MBR because its BIOS might not support GPT.
- If you are partitioning a disk of 2 TiB or larger, you need to use GPT.
- It is recommended to always use GPT for **UEFI** boot, as some UEFI implementations do not support booting to the MBR while in UEFI mode.

- If none of the above apply, choose freely between GPT and MBR. Since GPT is more modern, it is recommended in this case.

Some advantages of GPT over MBR are:

- Provides a unique disk GUID and unique **partition GUID** (`PARTUUID`) for each partition - A good filesystem-independent way of referencing partitions and disks.
- Provides a filesystem-independent **partition name** (`PARTLABEL`).
- Arbitrary number of partitions - depends on space allocated for the partition table - No need for extended and logical partitions. By default the GPT table contains space for defining 128 partitions. However if you want to define more partitions, you can allocate more space to the partition table (currently only *gdisk* is known to support this feature).
- Uses 64-bit LBA for storing Sector numbers - maximum addressable disk size is 2 ZiB. MBR is limited to addressing 2 TiB of space per drive.
- Stores a backup header and partition table at the end of the disk that aids in **recovery** in case the primary ones are damaged.
- CRC32 checksums to detect errors and corruption of the header and partition table.

The section on **#Partitioning tools** contains a table indicating which tools are available for creating and modifying GPT and MBR tables.

Partitionless disk

Partitionless disk a.k.a. **superfloppy (https://docs.microsoft.com/en-us/windows-hardware/manufacture/desktop/windows-and-gpt-faq#what-is-a-superfloppy)** refers to using a storage device without using a partition table, having one file system occupying the whole storage device. The boot sector present on a partitionless device is called a **volume boot record (VBR)**.

Btrfs partitioning

Btrfs can occupy an entire data storage device and replace the **MBR** or **GPT** partitioning schemes. See the **Btrfs#Partitionless Btrfs disk** instructions for details.

Backup

See **fdisk#Backup and restore partition table**.

Recover

It may be possible to recover a destroyed MBR partition table with `gpart (https://www.archlinux.org/packages/?name=gpart)`. See `gpart(8) (https://jlk.fjfi.cvut.cz/arch/manpages/man/gpart.8)` for instructions.

For GPT it is possible to restore the primary GPT header (located at the start of the partition table) from the secondary GPT header (located at the end of the partition table) or vice versa. See **gdisk#Recover GPT header**.

Partition scheme

There are no strict rules for partitioning a hard drive, although one may follow the general guidance given below. A disk partitioning scheme is determined by various issues such as desired flexibility, speed, security, as well as the limitations imposed by available disk space. It is essentially personal preference. If you would like to dual boot Arch Linux and a Windows operating system please see **Dual boot with Windows**.

Note:

- **UEFI** systems require an **EFI System Partition**.
- BIOS systems that are partitioned with **GPT** require a **BIOS boot partition** if **GRUB** is used as the bootloader.

Tip: If using **Btrfs**, subvolumes can be used to imitate partitions. See the **Btrfs#Mounting subvolumes** section.

Single root partition

This scheme is the simplest and should be enough for most use cases. A **swapfile** can be created and easily resized as needed. It usually makes sense to start by considering a single `/` partition and then separate out others based on specific use cases like RAID, encryption, a shared media partition, etc.

Discrete partitions

Separating out a path as a partition allows for the choice of a different filesystem and mount options. In some cases like a media partition, they can also be shared between operating systems.

Below are some example layouts that can be used when partitioning, and the following subsections detail a few of the directories which can be placed on their own separate partition and then **mounted** at mount points under `/`. See **`file-hierarchy(7) (https://jlk.fjfi.cvut.cz/arch/manpages/man/file-hierarchy.7)`** for a full description of the contents of these directories.

/

The root directory is the top of the hierarchy, the point where the primary filesystem is mounted and from which all other filesystems stem. All files and directories appear under the root directory `/`, even if they are stored on different physical devices. The contents of the

root filesystem must be adequate to boot, restore, recover, and/or repair the system. Therefore, certain directories under `/` are not candidates for separate partitions.

The `/` partition or root partition is necessary and it is the most important. The other partitions can be replaced by it.

Warning: Directories essential for booting (except for `/boot`) **must** be on the same partition as `/` or mounted in early userspace by the **initramfs**. These essential directories are: `/etc` and `/usr` **[1] (https://freedesktop.org/wiki/Software/systemd/separate-usr-is-broken)**.

`/` traditionally contains the `/usr` directory, which can grow significantly depending upon how much software is installed. 15–20 GiB should be sufficient for most users with modern hard disks. If you plan to store a swap file here, you might need a larger partition size.

/boot

The `/boot` directory contains the kernel and ramdisk images as well as the bootloader configuration file and bootloader stages. It also stores data that is used before the kernel begins executing user-space programs. `/boot` is not required for normal system operation, but only during boot and kernel upgrades (when regenerating the initial ramdisk).

A separate `/boot` partition is needed if installing a software RAID0 (stripe) system.

Note: If booting using UEFI **boot loaders** that do not have drivers for other file systems it is recommended to mount **EFI System Partition** to `/boot`. See **EFI System Partition#Mount the partition** for more information.

A suggested size for `/boot` is 200 MiB unless you are using EFI System Partition as `/boot`, in which case refer to **EFI System Partition#Create the partition** for the recommended size.

/home

The `/home` directory contains user-specific configuration files, caches, application data and media files.

Separating out `/home` allows `/` to be re-partitioned separately, but note that you can still reinstall Arch with `/home` untouched even if it is not separate—the other top-level directories just need to be removed, and then pacstrap can be run.

You should not share home directories between users on different distributions, because they use incompatible software versions and patches. Instead, consider sharing a media partition or at least using different home directories on the same `/home` partition. The size of this partition varies.

/var

The `/var` directory stores variable data such as spool directories and files, administrative and logging data, **pacman**'s cache, the **ABS** tree, etc. It is used, for example, for caching and logging, and hence frequently read or written. Keeping it in a separate partition avoids running out of disk space due to flunky logs, etc.

It exists to make it possible to mount `/usr` as read-only. Everything that historically went into `/usr` that is written to during system operation (as opposed to installation and software maintenance) must reside under `/var`.

Note: `/var` contains many small files. The choice of file system type should consider this fact if a separate partition is used.

`/var` will contain, among other data, the **ABS** tree and the **pacman** cache. Retaining these packages is helpful in case a package upgrade causes instability, requiring a **downgrade** to an older, archived package. The pacman cache in particular will grow as the system is expanded and updated, but it can be safely cleared if space becomes an issue. 8–12 GiB on a desktop system should be sufficient for `/var`, depending on how much software will be installed.

/data

One can consider mounting a "data" partition to cover various files to be shared by all users. Using the `/home` partition for this purpose is fine as well. The size of this partition varies.

Swap

A **swap** partition provides memory that can be used as virtual RAM. A **swap file** should be considered too, as they don't have any performance overhead compared to a partition but are much easier to resize as needed. A swap partition can *potentially* be shared between operating systems, but not if hibernation is used.

Historically, the general rule for swap partition size was to allocate twice the amount of physical RAM. As computers have gained ever larger memory capacities, this rule is outdated. For example, on average desktop machines with up to 512MiB RAM, the 2x rule is usually adequate; if a sufficient amount of RAM (more than 1024MiB) is available, it may be possible to have a smaller swap partition. See **Suspend and hibernate** to hibernate into a swap partition or file.

Example layouts

Note: UEFI/GPT does not have a "boot" flag, booting relies on files in **EFI System Partition**. **Parted** and its front-ends use a `boot` flag on GPT to indicate an EFI System Partition.

UEFI/GPT example layout

Mount point	Partition	Partition type (GUID)	Partition attributes	Suggested size
/boot	/dev/sda1	C12A7328-F81F-11D2-BA4B-00A0C93EC93B : EFI System Partition		550 MiB
/	/dev/sda2	4F68BCE3-E8CD-4DB1-96E7-FBCAF984B709 : Linux x86-64 root (/)		23 - 32 GiB
[SWAP]	/dev/sda3	0657FD6D-A4AB-43C4-84E5-0933C84B4F4F : Linux swap		More than 512 MiB
/home	/dev/sda4	933AC7E1-2EB4-4F13-B844-0E14E2AEF915 : Linux /home		Remainder of the device

BIOS/MBR example layout

Mount point	Partition	Partition type (ID)	Boot flag	Suggested size
/	/dev/sda1	83 : Linux	Yes	23 - 32 GiB
[SWAP]	/dev/sda2	82 : Linux swap	No	More than 512 MiB
/home	/dev/sda3	83 : Linux	No	Remainder of the device

Partitioning tools

The following programs are used to create and/or manipulate device partition tables and partitions. See the linked articles for the exact commands to be used.

This table will help you to choose utility for your needs:

	MBR	GPT
Dialog	fdisk parted	fdisk gdisk parted
Pseudo-graphics	cfdisk	cfdisk cgdisk
Non-interactive	sfdisk parted	sfdisk sgdisk parted
Graphical	GParted gnome-disk-utility partitionmanager	GParted gnome-disk-utility partitionmanager

Warning: To partition devices, use a partitioning tool compatible to the chosen type of partition table. Incompatible tools may result in the destruction of that table, along with existing partitions or data.

fdisk/gdisk

These group of tools fall under *fdisk* or *gdisk* and are described in the **fdisk** article.

- **fdisk** — Dialog-driven program for creation and manipulation of partition tables.

  https://www.kernel.org/pub/linux/utils/util-linux/ || **util-linux (https://www.archlinux.org/packages/?name=util-linux)**

- **cfdisk** — Curses-based variant of fdisk.

https://www.kernel.org/pub/linux/utils/util-linux/ || util-linux (https://www.archlinux.org/packages/?name=util-linux)

- sfdisk — Scriptable variant of fdisk.

  https://www.kernel.org/pub/linux/utils/util-linux/ || util-linux (https://www.archlinux.org/packages/?name=util-linux)

- gdisk — GPT alternative to fdisk.

  http://www.rodsbooks.com/gdisk/ || gptfdisk (https://www.archlinux.org/packages/?name=gptfdisk)

- cgdisk — Curses-based variant of gdisk.

  http://www.rodsbooks.com/gdisk/ || gptfdisk (https://www.archlinux.org/packages/?name=gptfdisk)

- sgdisk — Scriptable variant of gdisk.

  http://www.rodsbooks.com/gdisk/sgdisk-walkthrough.html || gptfdisk (https://www.archlinux.org/packages/?name=gptfdisk)

GNU Parted

These group of tools are described in the **GNU Parted** article.

- **GNU Parted** — Terminal partitioning tool.

  https://www.gnu.org/software/parted/parted.html || **parted (https://www.archlinux.org/packages/?name=parted)**

- **GParted** — Graphical tool written in GTK.

  https://gparted.sourceforge.io/ || **gparted (https://www.archlinux.org/packages/?name=gparted)**

- **GNOME Disks** — Graphical tool written in GTK.

  https://wiki.gnome.org/Apps/Disks || **gnome-disk-utility (https://www.archlinux.org/packages/?name=gnome-disk-utility)**

- **KDE Partition Manager** — Graphical tool written in Qt.

  https://www.kde.org/applications/system/kdepartitionmanager/ || **partitionmanager (https://www.archlinux.org/packages/?name=partitionmanager)**

Partition alignment

fdisk/gdisk and **parted** handle alignment automatically. See **GNU Parted#Check alignment** if you want to verify your alignment after partitioning.

For certain drives **Advanced Format** might be able to provide a better-performing alignment.

Tips and tricks

Converting MBR to GPT

See **fdisk#Convert between MBR and GPT**.

GPT Kernel Support

The `CONFIG_EFI_PARTITION` option in the kernel config enables GPT support in the kernel (despite the name, EFI PARTITION). This option must be built in the kernel and not compiled as a loadable module. This option is required even if GPT disks are used only for data storage and not for booting. This option is enabled by default in Arch's `linux (https://www.archlinux.org/packages/?name=linux)` and `linux-lts (https://www.archlinux.org/packages/?name=linux-lts)` kernels in the [core] repo. In case of a custom kernel, enable this option by doing `CONFIG_EFI_PARTITION=y` .

See also

- **Wikipedia:Disk partitioning**
- **Wikipedia:Binary prefix**
- **Understanding Disk Drive Terminology (http://thestarman.pcministry.com/asm/mbr/DiskTerms.htm)**
- **What is a Master Boot Record (MBR)? (https://kb.iu.edu/d/aijw)**
- Rod Smith's page on **What's a GPT? (http://www.rodsbooks.com/gdisk/whatsgpt.html)** and **Booting OSes from GPT (http://rodsbooks.com/gdisk/booting.html)**
- **Make the most of large drives with GPT and Linux - IBM Developer Works (https://www.ibm.com/developerworks/linux/library/l-gpt/index.html?ca=dgr-lnxw07GPT-Storagedth-lx&S_TACT=105AGY83&S_CMP=grlnxw07)**
- **Microsoft's Windows and GPT FAQ (https://docs.microsoft.com/en-us/windows-hardware/manufacture/desktop/windows-and-gpt-faq)**
- **Partition Alignment (https://www.thomas-krenn.com/en/wiki/Partition_Alignment)** (with examples)

Retrieved from "https://wiki.archlinux.org/index.php?title=Partitioning&oldid=509701"
