

Make the most of large drives with GPT and Linux

Preparing for future disk storage with the GUID Partition Table

Skill Level: Introductory

Roderick W. Smith (rodsmith@rodsbooks.com)
Consultant & Writer

28 Jul 2009

Once a faraway problem, an important barrier in disk storage is fast becoming a reality: the venerable master boot record (MBR) partitioning scheme can't fully handle disks larger than 2TB. With 1TB-hard disks now common and 2TB-disks becoming available, forward-looking individuals are thinking about alternatives to the MBR partitioning scheme. The heir apparent is the GUID Partition Table (GPT). Learn how to make sure your Linux® system is fully prepared for the future of disk storage.

Before embarking on a quest to replace your hard disk's partitioning scheme, it's helpful to review the limitations that are about to force this change. Understanding these limits—and the proposed tools for overcoming them—will enable you to judge how quickly you should jump ship from the MBR to GPT. This is particularly true if you're considering adopting GPT before new disk purchases force your hand. GPT offers advantages over MBR even on smaller disks, but you must balance those advantages against the difficulties of a switch.

Understanding the MBR's limits

The MBR partitioning system is a hodge-podge of data structure patches applied to overcome earlier limits. The MBR itself resides entirely on the first sector (512 bytes) of a hard disk. The first 440 bytes of the MBR are devoted to code: the *boot loader*. The BIOS reads this code and executes it when the computer boots.

Following the code area, the MBR stores data on four partitions, known as *primary partitions*. Each partition is described in two ways: using cylinder/head/sector (CHS) notation and using logical block addressing (LBA) notation. The CHS notation is

almost a historical footnote today, because it's a 24-bit number. This means that it's limited to describing 8GB-disk areas. The 32-bit LBA values permit 2TB sizes. This 2TB ceiling is not easily overcome; there simply aren't any unallocated fields left in the MBR that could be used to add more bits to the LBA addresses.

In addition to the looming 2TB problem, the MBR presents other difficulties. Chief among these is the limitation of four primary partitions. To work around this limitation, it's possible to set aside one primary partition as a placeholder (known as an *extended partition*) to hold an arbitrary number of additional partitions, known as *logical partitions*. This is, however, an ugly workaround that creates its own problems, such as difficulties installing multiple operating systems when too many of them want too many primary partitions to themselves.

The MBR has data-integrity problems, as well. It is a single data structure that's vulnerable to damage by carelessness or hardware failure. In addition, because logical partitions are defined in a linked-list structure, damage to one of them can block access to the remaining logical partitions. None of these data structures includes any form of error-detection capability, so damage can be difficult to spot.

GPT's solution

Intel® created the GPT definition as part of its Extensible Firmware Interface (EFI) specification for a BIOS replacement (see [Resources](#) for links to more information). Despite the fact that GPT is part of a standard that's meant to replace the legacy BIOS found on most computers today, it's possible to use GPT even on BIOS-based systems. If your computer already uses EFI, though, this fact is another plus to GPT adoption. Whether your computer uses a legacy BIOS or an EFI, GPT fixes many of the MBR's limitations:

- GPT uses LBA exclusively, so CHS headaches are gone.
- Disk pointers are 64 bits in size, meaning that it can handle disks of up to 9.4×10^{21} bytes (9.4 zettabytes, or 9.4 billion TB) in size.
- Critical GPT data structures are stored twice on the disk: once at the start and again at the end. This behavior improves the odds of successful recovery in case of damage from an accident or a bad sector.
- Cyclic redundancy check (CRC) values are computed for critical data structures, improving the odds of detection of data corruption.
- GPT stores all partitions in a single partition table (with backup), so there's no need for extended or logical partitions. By default, 128 partitions are supported, although you can change the partition table size if the partitioning software supports such changes.

- Whereas MBR provides a 1-byte partition type code, GPT uses a 16-byte GUID (Globally unique identifier) value to identify partition types. In theory, this makes partition-type collisions less likely; however, in practice, this rose already has some thorns, as described shortly.
- GPT adds a field for storing a human-readable partition name. You can use this field to name your Linux /home, /usr, /var, and other partitions for easier identification within partitioning software.

The first sector of the disk is reserved for a *protective MBR*, which is a legal MBR data structure that defines a single partition of type `0xEE` (EFI GPT). On sub-2TB disks, this partition should span the entire disk; on larger disks, it should be 2TB in size. The idea is to protect the GPT disk from damage by GPT-unaware disk utilities. If such tools look at the disk, they'll see an MBR disk with no free space. (Some disk utilities can create a *hybrid MBR*, which defines up to three MBR partitions in addition to the EFI GPT partition. The idea is to enable a GPT-unaware operating system, such as most versions of Windows®, to coexist on a disk along with GPT partitions. This configuration is decidedly non-standard and very kludgy, though.)

Because GPT incorporates a protective MBR, a BIOS-based computer can boot from a GPT disk using a boot loader stored in the protective MBR's code area, but the boot loader and operating system must both be GPT-aware. EFI includes its own boot loader, so you can boot from a GPT disk on an EFI-based system.

The main problem with GPT is one of compatibility: Low-level disk utilities and operating systems must all support GPT. Such support is fairly common for Linux, although you may need to attend to some of these details and change some of the tools you use for low-level disk maintenance. If you multi-boot a computer, you'll have to look into GPT support for all of your operating systems.

Fringe benefits of GPT adoption

The single biggest benefit of GPT is, of course, its shattering of the 2TB barrier. Even if you don't have to deal with 2TB disks, though, you may want to consider adopting GPT. The CRCs and backup data structures are arguably the most important benefits of switching to GPT on sub-2TB disks. Losing the primary-logical distinction may be important for some users, as well. Unfortunately, many of the operating systems that require primary partitions are unable to boot from GPT disks.

If you administer a lot of Linux systems, or if you anticipate adding an over-2TB disk in the not-too-distant future, you may want to consider doing a test installation with GPT. Doing so before you're forced to do it will give you first-hand experience with GPT's features as well as with the quirks of some of the GPT-aware Linux utilities.

Using GPT

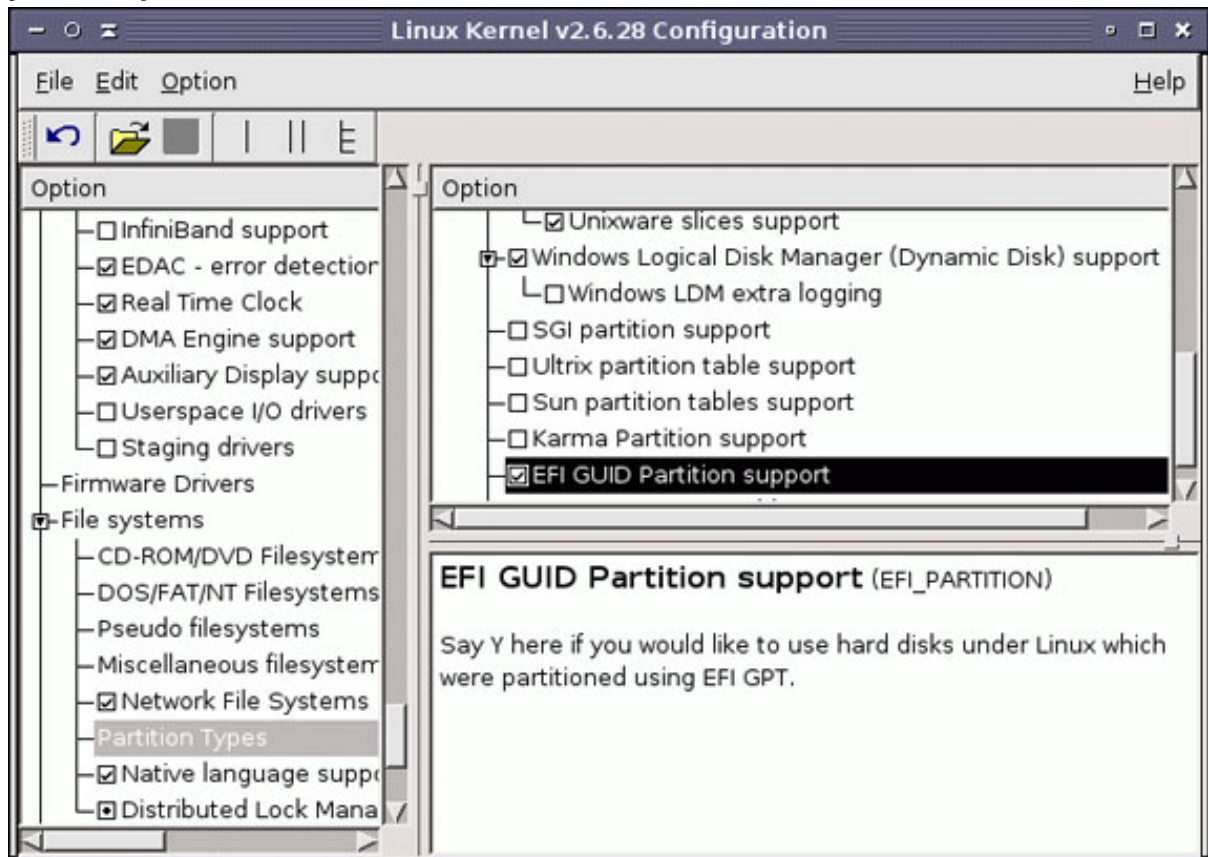
If you want (or need) to take the GPT plunge, you must prepare yourself. Three main classes of software all require GPT support: the kernel, the boot loader, and low-level disk utilities. If you're using GPT because you're setting up a very large RAID array, you may also need to look into file system support for extra-large disks.

Note: If you're installing Linux from scratch and want to use GPT, your installer must provide GPT support in all three of these categories. If it doesn't, you'll need to use the MBR for at least the system boot disk or convert to GPT after installing to the MBR (a potentially risky or tedious proposition).

Kernel support

The Linux kernel must be able to parse GPT data structures in order to provide access to the partitions the disk contains. Fortunately, this support has long been present in Linux. If you compile your own kernel, be sure to check the **EFI GUID Partition Support** item in the **Partition Types** area of the **File Systems** configuration area, as shown in Figure 1.

Figure 1. The Linux kernel provides GPT support, but it must be enabled when you compile a new kernel



If you don't compile your own kernel, you're at the mercy of your distribution provider to enable this support. Fortunately, most do so, but you should verify this detail. If

you're in doubt, you can use a GPT-aware partitioning tool to set up GPT partitions on a spare hard disk or a small removable disk, such as a USB flash drive. If Linux recognizes the partitions, then your kernel is properly configured.

Boot loader support

The boot loader is perhaps the weakest link in the Linux GPT chain. Linux systems generally use either the Linux Loader (LILO) or the Grand Unified Bootloader (GRUB) to boot the kernel. LILO is not GPT-aware, so if your system uses LILO, you'll have to switch to GRUB—and the story with GRUB is variable.

The latest official version of the traditional GRUB is 0.97. This version of GRUB is not GPT-aware; however, patches to this version with GPT support are common, so if you install GRUB version 0.97, you might or might not be able to boot from GPT disks. An entirely new GRUB architecture, known as GRUB2, is now available but considered experimental. GRUB2 does include GPT support.

Overall, if you want to use GPT, your best bet is to install your distribution's GRUB 0.97 and hope that it includes the GPT patches. If it doesn't, all is not lost. You can boot the Linux SystemRescueCd and re-install GRUB from it. This will get your system booting, assuming everything else is configured correctly.

Utilities support

The third area of GPT support is system utilities. Linux provides a wide range of tools for partitioning MBR disks, including `fdisk`, `cdisk`, `sfdisk`, GNU Parted, the GNOME Partition Editor (GParted), and more. Many of these tools are GPT-unaware, so if you want to use GPT, you'll have to use those that can handle the task.

Of the common tools, only those based on `libparted`—such as the text-mode GNU Parted or the GUI GParted—can handle GPT. The `fdisk` program and its variants will show you the protective MBR data, possibly along with a warning that GPT data has been detected. If you want to create fresh GPT partitions on a disk using GNU Parted, you should launch the program, then use its `mklabel` command:

Listing 1. Using Parted to create GPT disk partitions

```
# parted /dev/sdc
GNU Parted 1.7.1
Using /dev/sdc
Welcome to GNU Parted! Type 'help' to view a list of commands.
(parted) mklabel
New disk label type? gpt
(parted)
```

At this point, you can begin creating partitions using GNU Parted's `mkpart` or `mkpartfs` commands or otherwise manipulate partitions and file systems. When

you're done, you can use the normal Linux file system-management tools, such as `mkfs`, to create file systems on your disk. You can also create logical volume management (LVM) and RAID configurations much as you would on MBR disks.

If you prefer a GUI tool, GParted will do the job. Click **Device > Set Disklabel** to create a new GPT data structure. Click **Advanced**, then select **gpt** from the **Select new labeltype** list, as shown in Figure 2. Click **Create** to create your new GPT data structures. You can then create new partitions in the same way you would if you were manipulating an MBR disk.

Figure 2. You must explicitly set the GPT label type to create GPT partitions in the GNOME Partition Editor



The GPT creation tools of both GNU Parted and GParted are inherently destructive—if you've got an MBR disk, the only way to turn it into a GPT disk with these tools is to destroy your existing MBR partitions. If you want to convert an MBR disk in place, you could look into GPT fdisk, which is an `fdisk`-like tool for manipulating GPT partitions. GPT fdisk is very new and immature, however, so you should use it with caution.

A few quirks of GNU Parted, GParted, and other `libparted`-using tools deserve mention. Most importantly, these tools usually create entirely new protective MBRs

whenever they manipulate the GPT data structures. This means that they destroy the boot loader, so you may need to re-install GRUB after you make partition changes. Partition type codes are handled a bit strangely; some of them are referred to as *flags*. Most notably, if you want to set up a partition for use in an LVM or RAID configuration, you'll need to do so using the flag-manipulation tools. One of these flags, Microsoft Reserved (`msftres`), is incorrectly set on all FAT partitions, which makes the partition inaccessible from Windows or Mac OS X. (GPT fdisk enables you to correct this problem.)

As you're probably aware, Linux employs a handful of MBR partition type codes, such as `0x82` and `0x83`, to identify its MBR partitions. Similar GUID codes exist to identify Linux GPT partitions. One important caveat is that Linux and Windows use the same GUID for their data partitions. Thus, it's impossible to differentiate Linux file systems and NTFS or FAT file systems from their partition table GUIDs alone. GNU Parted and GParted look into the file system itself to identify the file system, but not all tools do this (GPT fdisk doesn't, for instance). You should be aware of this fact and perhaps use the GPT partition name field to help differentiate partitions.

Large file systems support

If you're switching to GPT because you're using a very large RAID configuration, you may need to investigate support for large file system sizes in the file systems you deploy. Table 1 summarizes these limits. (Note that some values vary with partitioning options.) Some of these values are quite large and use suffixes that may be unfamiliar. One terabyte is 1024GB; 1 petabyte (PB) is 1024TB; 1 exabyte (EB) is 1024 PB; and 1 zettabyte (ZB) is 1024PB.

Table 1. File system volume and size limits

File system	Maximum volume size	Maximum file size
Second extended file system (ext2) and third extended file system (ext3)	16TB	2TB
Fourth extended file system (ext4)	1EB	16TB
ReiserFS	16TB	8TB
Journalized file system (JFS)	32PB	4PB
XFS	16EB	8EB
Btrfs (under development)	16EB	16EB

Beyond the file- and volume-size limits, there are file system performance differences. This topic is extremely complex, so you may need to consult with others who run setups similar to the one you're planning.

GPT partitioning advice

Some special concerns crop up for GPT partitioning, particularly if your computer uses EFI or if you run in a multi-boot environment:

- EFI requires that a partition known as an *EFI System Partition* (ESP) be present. GNU Parted identifies the ESP as having the boot flag set. The ESP is normally about 200MB in size and is formatted for FAT-32. It holds drivers that EFI can use during the boot process. This partition is not required if your computer uses BIOS to boot.
- Many GPT partitioning tools create gaps of about 128MB after each partition (the ESP is an exception to this rule). The intention is that disk utilities can use this space to help with their jobs.
- On Mac OS X systems, partitions are created in sizes that are multiples of 4KB (normally, 8 sectors). This feature relates to limitations of the HFS Plus file system that's used by most modern Macs.

You can follow these partitioning rules or ignore them as you see fit. Linux is flexible enough that it won't be bothered by a disregard for these rules, unless your computer requires an ESP to boot.

It's possible to run a system with a mixed MBR-GPT configuration—that is, you can have one disk partitioned with the MBR and another partitioned with GPT. Thus, if you add a new over-2TB disk to your system, you don't need to touch your original sub-2TB disk's partitions. In fact, it's probably safest if you don't.

Conclusion

GPT is poised to become the standard for hard disk partitioning because of the size limitations of the MBR. Fortunately, Linux is well prepared for this transition. Although Linux users may have to give up certain tools (LILO and `fdisk`, for instance), other tools are available to take their places (patched versions of GRUB and GNU Parted, for example). Understanding the requirements will help you make the transition easily when the time comes to do so. You'll need to attend to your kernel configuration, your boot loader configuration, and the utilities you use to create and manage partitions.

Resources

Learn

- The [Wikipedia article on GPT](#) provides details about GPT's layout and features.
- Apple's [Technical Note TN2166](#) provides information on Apple's recommended partitioning rules.
- Read the [EFI specification](#) from Intel.
- "[Inside the Linux boot process](#)" (developerWorks, May 2006) describes how the BIOS, the boot loader, and the Linux kernel work together to start your computer.
- In the [developerWorks Linux zone](#), find more resources for Linux developers, and scan our [most popular articles and tutorials](#).
- See all [Linux tips](#) and [Linux tutorials](#) on developerWorks.
- Stay current with [developerWorks technical events and Webcasts](#).

Get products and technologies

- The [GNU Parted program](#) is a mature text-mode MBR and GPT partitioning tool.
- The [GNOME Partition Editor \(GParted\)](#) is a GUI MBR and GPT partitioning tool that's based on `libparted`.
- The [GPT fdisk](#) program is a GPT-only partitioning program modeled after Linux `fdisk`.
- The [GRUB Web page](#) provides information and resources for both GRUB 0.97 (legacy GRUB) and GRUB2.
- The [Linux SystemRescueCd](#) is a useful utility for emergency maintenance. It includes a GPT-aware version of GRUB that you can install from the CD boot.
- With [IBM trial software](#), available for download directly from developerWorks, build your next development project on Linux.

Discuss

- Get involved in the [My developerWorks community](#); with your personal profile and custom home page, you can tailor developerWorks to your interests and interact with other developerWorks users.

About the author

Roderick W. Smith

Roderick W. Smith is a consultant and author of over a dozen books on UNIX and Linux, including *The Definitive Guide to Samba 3*, *Linux in a Windows World*, and *Linux Professional Institute Certification Study Guide*. He currently resides in Woonsocket, Rhode Island, and you can reach him at rodsmith@rodsbooks.com.