GUID Partition Table

From Wikipedia, the free encyclopedia

GUID Partition Table (**GPT**) is a standard for the layout of the partition table on a physical hard disk, using globally unique identifiers (GUID). Although it forms a part of the Unified Extensible Firmware Interface (UEFI) standard (Unified EFI Forum proposed replacement for the PC BIOS), it is also used on some BIOS systems because of the limitations of master boot record (MBR) partition tables, which use 32 bits for storing logical block addresses (LBA) and size information.

As of 2010, most current operating systems support GPT. Some, including OS X and Microsoft Windows on x86, only support booting from GPT partitions on systems with EFI firmware, but FreeBSD and most Linux distributions can boot from GPT partitions on systems with either legacy BIOS firmware interface or EFI.

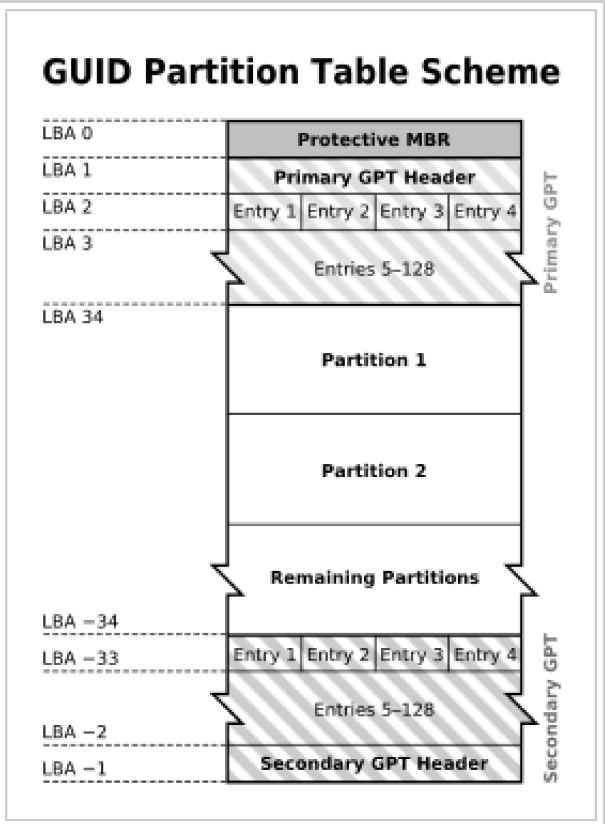

Diagram illustrating the layout of the GPT scheme. In this example, each logical block is 512 bytes in size, and each partition entry is 128 bytes, and the corresponding partition entries are assumed to be located in LBA 2-33, here. LBA addresses that are negative indicate position from the end of the volume, with −1 as the last addressable block.

Contents

History

The widespread MBR partitioning scheme, dating from the early 1980s, imposed limitations that affect the use of modern hardware. One of the main limitations is the usage of 32 bits for storing block addresses and quantity information. For hard disks with 512-byte sectors, the MBR partition table entries allow up to a maximum of 2 TiB (2^{32}×512 Bytes).[1]

Intel therefore developed a new partition table format in the late 1990s as part of what eventually became UEFI. As of 2010, GPT forms a subset of the UEFI specification.[2] GPT allocates 64 bits for logical block addresses, therefore allowing a maximum disk size of 2^{64} sectors. For disks with 512-byte sectors, maximum size is 9.4 ZB (9.4×10^{21} bytes) or 8 ZiB (9,444,732,965,739,290,427,392 bytes, coming from 18,446,744,073,709,551,616 (2^{64}) sectors × 512 (2^{9}) bytes per sector).[1][3]

Features

MBR-based partition table schemes insert the partitioning information for (usually) four "primary" partitions in the master boot record (MBR) (which on a BIOS system is also the container for code that begins the process of booting the system). In a GPT, the first sector of the disk is reserved for a "protective MBR" such that booting a BIOS-based computer from a GPT disk is supported, but the boot loader and operating system must both be GPT-aware. Regardless of the sector size, the GPT header begins on the second logical block of the device.

Like modern MBRs, GPTs use logical block addressing (LBA) in place of the historical cylinder-head-sector (CHS) addressing. The protective MBR is contained in LBA 0, the GPT header is in LBA 1, and the GPT header has a pointer to the partition table, or Partition Entry Array, typically LBA 2. The UEFI specification[4] stipulates that a minimum of 16,384 bytes, regardless of sector size, be allocated for the Partition Entry Array. On a disk having 512-byte sectors, a partition entry array size of 16,384 bytes and the minimum size of 128 bytes for each partition entry, LBA 34 is the first usable sector on the disk.

Hard disk manufacturers are transitioning to 4,096-byte sectors. As of 2010, the first such drives continue to present 512-byte physical sectors to the OS, so degraded performance can result when the drive's (hidden) internal 4 KiB sector boundaries do not coincide with the 4 KiB logical blocks, clusters and virtual memory pages common in many operating systems and file systems. This is a particular problem on writes when the drive is forced to perform two read-modify-write operations to satisfy a single misaligned 4 KiB write operation.[5] Such a misalignment occurs *by default* if the first partition is placed immediately after the GPT, as the next block is LBA 34, whereas the next 4 KiB boundary begins with LBA 40.

For backward compatibility with most legacy operating systems like DOS, OS/2 and versions of Windows before Vista, MBR partitions must always start on track boundaries according to the traditional CHS addressing scheme and end on a cylinder boundary. This even holds true for partitions with emulated CHS geometries (as reflected by the BIOS and the CHS sectors entries in the MBR partition table) or partitions accessed only via LBA. Extended partitions always start on cylinder boundaries as well.

This typically causes the first primary partition to start at LBA 63 on disks accessed via LBA, leaving a gap of 62 sectors with MBR-based disks, sometimes called "MBR gap", "boot track", or "embedding area". That otherwise unused disk space is commonly used by boot loaders such as GRUB for storing their second stages.[6] On older computers using alternative LBA/CHS translation schemes or different extended CHS mappings, with smaller LBA-accessed disks, or on disks accessed via CHS only, the gap could be smaller, although not normally less than LBA 16 on normal hard disks.

Since Windows Vista, the first partition usually starts after a gap of 2,047 sectors at LBA 2,048 as part of its new 1 MiB partition alignment policy, so no large-sector misalignment occurs by default, but serious compatibility problems with older operating systems and disk tools exist.

Drives which boot Intel-based Macs are typically formatted with a GPT, rather than with the Apple Partition Map (APM).

GPT also provides redundancy, writing the GPT header and partition table both at the beginning and at the end of the disk.

If the minimum size of 16,384 bytes is allocated for the partition entry array, and the default size of 128 bytes is used for each partition entry, then the maximum number of partitions is limited to 128.

Legacy MBR (LBA 0)

Traditionally, in IBM PC compatible systems the first sector of the disk holds the Master Boot Record (MBR), containing the drive's partitioning information and the code of the first stage boot loader for BIOS-based systems. For limited backward compatibility, this sector is still reserved for an MBR in the GPT specification, but it is now used in a way that prevents MBR-based disk utilities from misrecognizing and possibly overwriting GPT disks. This is referred to as a *protective MBR*.[3]

A single partition type of `EEh`, encompassing the entire GPT drive (where "entire" actually means as much of the drive as can be represented in an MBR), is indicated and identifies it as GPT. Operating systems and tools which cannot read GPT disks will generally recognize the disk as containing one partition of unknown type and no empty space, and will typically refuse to modify the disk unless the user explicitly requests and confirms the deletion of this partition. This minimizes accidental erasures.[3] Furthermore, GPT-aware OSes may check the protective MBR and if the enclosed partition type is not of type `EEh` or if there are multiple partitions defined on the target device, the OS may refuse to manipulate the partition table.[7]

While the MBR layout (and also the protective MBR layout) was defined around a sector size of 512 bytes per sector, the actual sector size can be larger on various media such as MO disks or hard disks with Advanced Format. Extra space in the MBR typically remains unused.

If the actual size of the disk exceeds the maximum partition size representable using the legacy 32-bit LBA entries in the MBR partition table, the recorded size of this partition is clipped at the maximum, thereby ignoring the rest of disk. This amounts to a maximum reported size of 2 TiB, assuming a disk with 512 bytes per sector (see 512e). It would result in 16 TiB with 4 KB sectors (4Kn), but since many older operating systems and tools are hard-wired for a sector size of 512 bytes or are limited to 32-bit calculations, exceeding the 2 TiB limit would cause serious compatibility problems.[3]

In operating systems that support GPT-based boot through BIOS services rather than EFI, the first sector is also still used to store the first stage of the bootloader code, but modified to recognize GPT partitions. The boot loader in the MBR must not assume a fixed sector size of 512 bytes / sector.[3]

Apple's Boot Camp software for Intel-based Apple Macs creates a hybrid partition table to allow the booting of Windows (which at the time of Boot Camp's creation did not support GPT or EFI). In this system the protective partition is reduced in size to cover from sector 1 to the sector before the first regular partition included in the hybrid MBR. Additional MBR partitions are then defined to correspond to the next three GPT partitions.

Partition table header (LBA 1)

The partition table header defines the usable blocks on the disk. It also defines the number and size of the partition entries that make up the partition table. The EFI stipulates a minimum of 16,384 bytes be reserved for the partition table array, so there are 128 partition entries reserved, each 128 bytes long.

The header contains the disk globally unique identifier (GUID). It records its own size and location (always LBA 1!) and the size and location of the secondary GPT header and table (always the last sectors on the disk). Importantly, it also contains a CRC32 checksum for itself and for the partition table, which may be verified by the firmware, bootloader and/or operating system on boot. Because of this, hex editors should not be used to modify the contents of the GPT. Such modification would render the checksum invalid. In this case, the primary GPT may be overwritten with the secondary one by disk recovery software. If both GPTs contain invalid checksums, the disk would be unusable by software that verifies the checksum.

GPT header format

Offset	Length	Contents
0 (0x00)	8 bytes	Signature ("EFI PART", `45h 46h 49h 20h 50h 41h 52h 54h` or `0x5452415020494645ULL` on little-endian machines)
8 (0x08)	4 bytes	Revision (for GPT version 1.0 (through at least UEFI version 2.3.1), the value is `00h 00h 01h 00h`)
12 (0x0C)	4 bytes	Header size in little endian (in bytes, usually `5Ch 00h 00h 00h` meaning 92 bytes)
16 (0x10)	4 bytes	CRC32 of header (offset +0 up to header size), with this field zeroed during calculation
20 (0x14)	4 bytes	Reserved; must be zero
24 (0x18)	8 bytes	Current LBA (location of this header copy)
32 (0x20)	8 bytes	Backup LBA (location of the other header copy)
40 (0x28)	8 bytes	First usable LBA for partitions (primary partition table last LBA + 1)
48 (0x30)	8 bytes	Last usable LBA (secondary partition table first LBA - 1)
56 (0x38)	16 bytes	Disk GUID (also referred as UUID on UNIXes)
72 (0x48)	8 bytes	Starting LBA of array of partition entries (always 2 in primary copy)
80 (0x50)	4 bytes	Number of partition entries in array
84 (0x54)	4 bytes	Size of a single partition entry (usually 128)
88 (0x58)	4 bytes	CRC32 of partition array
92 (0x5C)	*	Reserved; must be zeroes for the rest of the block (420 bytes for a sector size of 512 bytes; but can be more with larger sector sizes)
LBA size	**Total**	

The values for current and backup LBAs of the primary header should be the second sector of the disk (LBA 1) and the last sector of the disk, respectively. The secondary header at the end of the disk identifies its own table of partition entries, which is located directly before that header.

This table must be referenced relative to LBA 1. Thus, on 4Kn disks with 4 KB sectors it does not follow the 512 bytes of the MBR physically (stored in LBA 0) and thereby becomes *part* of LBA 0 on disks with larger sector sizes. While the described arrangement happens to occur on disks with 512 bytes per sector, there may be "gaps" of unused space between them on disks with larger sector sizes. In other words, with 512-byte sectors it starts at 512th byte from the beginning of the disk, while with 4 KB sectors it starts at 4096th byte. If multi-sector reads are performed, the actual sector size must be included in the calculation when referencing this table.

Partition entries

GUID partition entry format

Offset	Length	Contents
0 (0x00)	16 bytes	Partition type GUID
16 (0x10)	16 bytes	Unique partition GUID
32 (0x20)	8 bytes	First LBA (little endian)
40 (0x28)	8 bytes	Last LBA (inclusive, usually odd)
48 (0x30)	8 bytes	Attribute flags (e.g. bit 60 denotes read-only)
56 (0x38)	72 bytes	Partition name (36 UTF-16LE code units)
	128 bytes total	

The GPT uses simple and straightforward entries to describe partitions. The first 16 bytes designate the partition type globally unique identifier (GUID). For example, the GUID for an EFI System partition is `C12A7328-F81F-11D2-BA4B-00A0C93EC93B`. The second 16 bytes contain a GUID unique to the partition. Then follow the starting and ending 64-bit LBAs, partition attributes and partition names. As is the nature and purpose of GUIDs, no central registry is needed to ensure the uniqueness of the GUID partition type designators. The location of the partition entries array on disk is defined in the GPT header.

The GPT header contains a field that specifies the size of a partition table entry. The minimum required is 128 bytes, but implementations must allow for other values.[8]

Also, the sector size must not be assumed to be hard-wired to 512 bytes per sector in calculations (see Advanced Format), that is, there can be more than four partition entries in a single sector, and (with possible future much larger partition table entries) it is possible to have a sector hold only a fraction of a partition entry. Except for the first two sectors (LBA 0 and LBA 1), the GPT specification just describes the size and organization of a data structure, not in how many sectors it is stored on disk.

The 64-bit partition table attributes are shared between 48-bit common attributes for all partition types, and 16-bit type-specific attributes.

Partition attributes

Bit	Content
0	System partition (disk partitioning utilities must preserve the partition as is)
1	EFI firmware should ignore the content of the partition and not try to read from it
2	Legacy BIOS bootable (equivalent to *active flag* (typically bit 7 set) at offset `+0h` in partition entries of the MBR partition table)[9]
3–47	Reserved for future use
48–63	Defined and used by the individual partition type

Microsoft defines the type-specific attributes for Basic data partition according to this TechNet article (http://technet.microsoft.com/en-us/library/cc739412%28WS.10%29.aspx#w2k3tr_basic_how_fgkm) as:

Basic data partition attributes

Bit	Content
60	Read-only
62	Hidden
63	Do not automount (i.e., do not assign drive letter)

Operating systems support

Hybrid MBRs are non-standard and can be interpreted in different ways by different OSes.[10] Unless otherwise noted, OSes provide precedence to the GPT data when a hybrid MBR configuration is encountered.

The term *No native support on this arch and version.* should be understood this way:

> Not supported as data disk,[11] only known legacy partitions found in protective MBR are accessible via the OS. Detachable disks: only support for MBR partitioning; No access with end user applications. GPT contained raw data is accessible with third-party administrator tools for low level disk access. True file system level support in read or read-write form might be subject of software from a third-party vendor.

UNIX and Unix-like operating systems

Details of GPT support on UNIX and Unix-like operating systems

OS family	Version or edition	Platform	Read and write support	Boot support	Note
FreeBSD	Since 7.0	IA-32, x86-64, ARM	Yes	Yes	In a hybrid configuration, both GPT and MBR partition identifiers may be used.
Linux	Most of the x86 Linux distributions Fedora 8+ and Ubuntu 8.04+[12]	IA-32, x86-64	Yes	Yes	New tools such as gdisk,[13] GNU Parted,[14][15] util-linux v2.23+ fdisk,[16][17] Syslinux, GRUB 0.96 + patches and GRUB 2 have been GPT-enabled.
OS X	Since 10.4.0 (some features since 10.4.6)[18]	IA-32, x86-64	Yes	Yes	Only Intel Macintosh computers can boot from GPT.
MidnightBSD	Since 0.4-CURRENT	IA-32, x86-64	Yes	Requires BIOS	In a hybrid configuration, both GPT and MBR partition identifiers may be used.
Solaris	Since Solaris 10	IA-32, x86-64, SPARC	Yes	Yes	[19]
HP-UX	Since HP-UX 11.20	IA-64	Yes	Yes	[20]

Windows: 32-bit versions

Windows 7 and earlier do not support (U)EFI on 32-bit platforms, and therefore do not allow booting from GPT partitions.

Details of GPT support on 32-bit editions of Microsoft Windows[11]

OS version	Release date	Platform	Read or write support	Boot support	Note
Windows XP	2001-10-25	IA-32	No	No	
Windows Server 2003	2003-04-24	IA-32	No	No	
Windows Server 2003 SP1	2005-03-30	IA-32	Yes	No	MBR takes precedence in hybrid configuration[10]
Windows Vista	2006-07-22	IA-32	Yes	No	MBR takes precedence in hybrid configuration[10]
Windows Server 2008	2008-02-27	IA-32	Yes	No	MBR takes precedence in hybrid configuration[10]
Windows 7	2009-10-22	IA-32	Yes	No	MBR takes precedence in hybrid configuration[10]
Windows 8	2012-08-01	IA-32	Yes	Requires UEFI[21]	MBR takes precedence in hybrid configuration[10]

Windows: 64-bit versions

Details of GPT support on 64-bit editions of Microsoft Windows[11]

OS version	Release date	Platform	Read and write support	Boot support	Note
Windows XP Professional x64 Edition Windows Server 2003	2005-04-25[22]	x64	Yes	No	MBR takes precedence in hybrid MBR configuration[10]
Windows Server 2003	2005-04-25	IA-64	Yes	Yes	MBR takes precedence in hybrid MBR configuration[10]
Windows Vista	2006-07-22	x64	Yes	Requires UEFI[a]	MBR takes precedence in hybrid configuration[10]
Windows Server 2008	2008-02-27	x64	Yes	Requires UEFI	MBR takes precedence in hybrid configuration[10]
Windows Server 2008	2008-02-27	IA-64	Yes	Yes	MBR takes precedence in hybrid configuration[10]
Windows 7 Windows Server 2008 R2	2009-10-22	x64	Yes	Requires UEFI[b]	MBR takes precedence in hybrid configuration.[10]
Windows Server 2008 R2	2009-10-22	IA-64	Yes	Yes	MBR takes precedence in hybrid configuration[10]
Windows 8 Windows Server 2012	2012-08-01	x64	Yes	Requires UEFI	MBR takes precedence in hybrid configuration.[10]

Partition type GUIDs

Operating system	Partition type	Globally unique identifier (GUID)[c]
(None)	Unused entry	00000000-0000-0000-0000-000000000000
	MBR partition scheme	024DEE41-33E7-11D3-9D69-0008C781F39F
	EFI System partition	C12A7328-F81F-11D2-BA4B-00A0C93EC93B
	BIOS Boot partition[d]	21686148-6449-6E6F-744E-656564454649
	Intel Fast Flash (iFFS) partition (for Intel Rapid Start technology)[23][24]	D3BFE2DE-3DAF-11DF-BA40-E3A556D89593
	Sony boot partition[e]	F4019732-066E-4E12-8273-346C5641494F
	Lenovo boot partition[e]	BFBFAFE7-A34F-448A-9A5B-6213EB736C22
Windows	Microsoft Reserved Partition (MSR)	E3C9E316-0B5C-4DB8-817D-F92DF00215AE
	Basic data partition[f]	EBD0A0A2-B9E5-4433-87C0-68B6B72699C7
	Logical Disk Manager (LDM) metadata partition	5808C8AA-7E8F-42E0-85D2-E1E90434CFB3
	Logical Disk Manager data partition	AF9B60A0-1431-4F62-BC68-3311714A69AD
	Windows Recovery Environment	DE94BBA4-06D1-4D40-A16A-BFD50179D6AC
	IBM General Parallel File System (GPFS) partition	37AFFC90-EF7D-4E96-91C3-2D7AE055B174
	Storage Spaces partition	E75CAF8F-F680-4CEE-AFA3-B001E56EFC2D
HP-UX	Data partition	75894C1E-3AEB-11D3-B7C1-7B03A0000000
	Service Partition	E2A1E728-32E3-11D6-A682-7B03A0000000
	Linux filesystem data[f]	0FC63DAF-8483-4772-8E79-3D69D8477DE4
	RAID partition	A19D880F-05FC-4D3B-A006-743F0F84911E
	Swap partition	0657FD6D-A4AB-43C4-84E5-0933C84B4F4F
		E6D6D379-F507-44C2-A23C-

Linux	Logical Volume Manager (LVM) partition	238F2A3DF928
	`/home` partition[26]	933AC7E1-2EB4-4F13-B844-0E14E2AEF915
	`/srv` (server data) partition[26]	3B8F8425-20E0-4F3B-907F-1A25A76F98E8
	Plain dm-crypt partition[27][28]	7FFEC5C9-2D00-49B7-8941-3EA10A5586B7
	LUKS partition[27][28]	CA7D7CCB-63ED-4C53-861C-1742536059CC
	Reserved	8DA63339-0007-60C0-C436-083AC8230908
FreeBSD	Boot partition	83BD6B9D-7F41-11DC-BE0B-001560B84F0F
	Data partition	516E7CB4-6ECF-11D6-8FF8-00022D09712B
	Swap partition	516E7CB5-6ECF-11D6-8FF8-00022D09712B
	Unix File System (UFS) partition	516E7CB6-6ECF-11D6-8FF8-00022D09712B
	Vinum volume manager partition	516E7CB8-6ECF-11D6-8FF8-00022D09712B
	ZFS partition	516E7CBA-6ECF-11D6-8FF8-00022D09712B
Mac OS X	Hierarchical File System Plus (HFS+) partition	48465300-0000-11AA-AA11-00306543ECAC
	Apple UFS	55465300-0000-11AA-AA11-00306543ECAC
	ZFS[g]	6A898CC3-1DD2-11B2-99A6-080020736631
	Apple RAID partition	52414944-0000-11AA-AA11-00306543ECAC
	Apple RAID partition, offline	52414944-5F4F-11AA-AA11-00306543ECAC
	Apple Boot partition	426F6F74-0000-11AA-AA11-00306543ECAC
	Apple Label	4C616265-6C00-11AA-AA11-00306543ECAC
	Apple TV Recovery partition	5265636F-7665-11AA-AA11-00306543ECAC
	Apple Core Storage (i.e. Lion FileVault) partition	53746F72-6167-11AA-AA11-00306543ECAC
	Boot partition	6A82CB45-1DD2-11B2-99A6-080020736631

Solaris	Root partition	6A85CF4D-1DD2-11B2-99A6-080020736631
	Swap partition	6A87C46F-1DD2-11B2-99A6-080020736631
	Backup partition	6A8B642B-1DD2-11B2-99A6-080020736631
	`/usr` partition[g]	6A898CC3-1DD2-11B2-99A6-080020736631
	`/var` partition	6A8EF2E9-1DD2-11B2-99A6-080020736631
	`/home` partition	6A90BA39-1DD2-11B2-99A6-080020736631
	Alternate sector	6A9283A5-1DD2-11B2-99A6-080020736631
	Reserved partition	6A945A3B-1DD2-11B2-99A6-080020736631
		6A9630D1-1DD2-11B2-99A6-080020736631
		6A980767-1DD2-11B2-99A6-080020736631
		6A96237F-1DD2-11B2-99A6-080020736631
		6A8D2AC7-1DD2-11B2-99A6-080020736631
NetBSD[29][h]	Swap partition	49F48D32-B10E-11DC-B99B-0019D1879648
	FFS partition	49F48D5A-B10E-11DC-B99B-0019D1879648
	LFS partition	49F48D82-B10E-11DC-B99B-0019D1879648
	RAID partition	49F48DAA-B10E-11DC-B99B-0019D1879648
	Concatenated partition	2DB519C4-B10F-11DC-B99B-0019D1879648
	Encrypted partition	2DB519EC-B10F-11DC-B99B-0019D1879648
ChromeOS[30]	ChromeOS kernel	FE3A2A5D-4F32-41A7-B725-ACCC3285A309
	ChromeOS rootfs	3CB8E202-3B7E-47DD-8A3C-7FF2A13CFCEC
	ChromeOS future use	2E0A753D-9E48-43B0-8337-B15192CB1B5E
Haiku[31]	Haiku BFS	42465331-3BA3-10F1-802A-

		4861696B7521
MidnightBSD[32][h]	Boot partition	85D5E45E-237C-11E1-B4B3-E89A8F7FC3A7
	Data partition	85D5E45A-237C-11E1-B4B3-E89A8F7FC3A7
	Swap partition	85D5E45B-237C-11E1-B4B3-E89A8F7FC3A7
	Unix File System (UFS) partition	0394EF8B-237E-11E1-B4B3-E89A8F7FC3A7
	Vinum volume manager partition	85D5E45C-237C-11E1-B4B3-E89A8F7FC3A7
	ZFS partition	85D5E45D-237C-11E1-B4B3-E89A8F7FC3A7
Ceph	Ceph Journal[i]	BFBFAFE7-A34F-448A-9A5B-6213EB736C22
	Ceph dm-crypt Encrypted Journal[i]	45B0969E-9B03-4F30-B4C6-5EC00CEFF106
	Ceph OSD[i]	4FBD7E29-9D25-41B8-AFD0-062C0CEFF05D
	Ceph dm-crypt OSD[i]	4FBD7E29-9D25-41B8-AFD0-5EC00CEFF05D
	Ceph disk in creation[i]	89C57F98-2FE5-4DC0-89C1-F3AD0CEFF2BE
	Ceph dm-crypt disk in creation[i]	89C57F98-2FE5-4DC0-89C1-5EC00CEFF2BE
OpenBSD	Data partition	824CC7A0-36A8-11E3-890A-952519AD3F61

See also

- Advanced Active Partition (AAP)
- Apple Partition Map (APM)
- BIOS Boot partition
- Boot Engineering Extension Record (BEER)
- BSD disklabel
- Device Configuration Overlay (DCO)
- Disk partitioning
- EFI System partition
- Extended Boot Record (EBR)
- Globally Unique Identifier (GUID)
- Host Protected Area (HPA)
- Master Boot Record (MBR)

- Partition alignment
- Rigid Disk Block (RDB)
- Unified Extensible Firmware Interface (UEFI)

Notes

a. ^ Only if using its service pack 1 or 2.
b. ^ In a multi-disk setup, non-UEFI bootloader (boot drive) requires MBR-based partitioning, while a system drive can use GUID partitioning.
c. ^ The GUIDs in this table are written assuming a little-endian byte order. For example, the GUID for an EFI System partition is written as `C12A7328-F81F-11D2-BA4B-00A0C93EC93B` here, which corresponds to the 16 byte sequence `28h 73h 2Ah C1h 1Fh F8h D2h 11h BAh 4Bh 00h A0h C9h 3Eh C9h 3Bh` – only the first three blocks are byte-swapped.
d. ^ The formation of this GUID does not follow the GUID definition; it is formed by using the ASCII codes for the string "`Hah!IdontNeedEFI`". Such formation of "GUID" value breaks down the guaranteed uniqueness of GUID.
e. ^ *a b* Some computer manufacturers have their own GUIDs for partitions that are analogous to the EFI System Partition, but that hold boot loaders to launch manufacturer-specific recovery tools.[25]
f. ^ *a b* Previously, Linux used the same GUID for the data partitions as Windows (Basic data partition: `EBD0A0A2-B9E5-4433-87C0-68B6B72699C7`). Linux never had a separate unique partition type GUID defined for its data partitions. This created problems when dual-booting Linux and Windows in UEFI-GPT setup. The new GUID (Linux filesystem data: `0FC63DAF-8483-4772-8E79-3D69D8477DE4`) was defined jointly by GPT fdisk and GNU Parted developers. It is identified as type code `0x8300` in GPT fdisk. (See definitions in gdisk's parttypes.cc (http://gptfdisk.git.sourceforge.net/git/gitweb.cgi?p=gptfdisk/gptfdisk;a=blob_plain;f=parttypes.cc;hb=HEAD))
g. ^ *a b* The GUID for `/usr` on Solaris is used as a generic GUID for ZFS by Mac OS X.
h. ^ *a b* NetBSD and MidnightBSD had used the FreeBSD GUIDs before their unique GUIDs were created.
i. ^ *a b c d e f* The Ceph filesystem uses GUIDs to mark the state of preparation a disk is in.[33]

References

1. ^ *a b* "FAQ: Drive Partition Limits" (http://www.uefi.org/sites/default/files/resources/UEFI_Drive_Partition_Limits_Fact_Sheet.pdf) (PDF). UEFI Forum. Retrieved 2013-11-04.
2. ^ Nikkel, Bruce J. (September 2009). "Forensic analysis of GPT disks and GUID partition tables". *Digital Investigation* **6** (1-2): 39–47. doi:10.1016/j.diin.2009.07.001 (https://dx.doi.org/10.1016%2Fj.diin.2009.07.001). "The current popular BIOS and MBR partitioning scheme was originally developed in the early 1980s for the IBM Personal Computer using IBM PC-DOS or MS-DOS. The Basic Input/Output System (BIOS) provides an interface to the hardware and initiates the boot process (IBM, 1983). The MBR, located in sector zero, contains the initial boot code and a four entry partition table (Microsoft, 1983). Intended to solve booting and partitioning limitations with newer hardware, a replacement for both the BIOS and the MBR partition table was developed by Intel in the late 1990s (Intel,

2000). This is now called the Unified EFI (UEFI, 2008 UEFI Forum. Unified extensible firmware interface specification version 2.2 2008.UEFI, 2008) specification, and managed by the UEFI Forum (UEFI, 2009). A subset of this specification includes GPT, intended to replace the DOS/MBR partition tables."

3. ^ *a b c d e* Roderick W. Smith (2012-07-03). "Make the most of large drives with GPT and Linux" (http://www.ibm.com/developerworks/library/l-gpt/). IBM. Retrieved 2013-05-29.
4. ^ UEFI specification (http://www.uefi.org/specifications)
5. ^ "Western Digital's Advanced Format: The 4K Sector Transition Begins" (http://anandtech.com/storage/showdoc.aspx?i=3691). Anandtech.
6. ^ "Installation" (http://www.gnu.org/software/grub/manual/html_node/BIOS-installation.html). *3.4 BIOS installation*. GNU GRUB. Retrieved 2013-09-25.
7. ^ "Technical Note TN2166: Secrets of the GPT" (https://developer.apple.com/library/mac/technotes/tn2166/_index.html#//apple_ref/doc/uid/DTS10003927-CH1-SUBSECTION11). Developer.apple.com. 2006-11-06. Retrieved 2014-04-16.
8. ^ "Mac Developer Library" (http://developer.apple.com/mac/library/technotes/tn2006/tn2166.html#SECGPTOVERVIEW). Developer.apple.com. Retrieved 2014-07-13.
9. ^ "e09127r3 EDD-4 Hybrid MBR boot code annex" (http://t13.org./documents/UploadedDocuments/docs2010/e09127r3-EDD-4_Hybrid_MBR_boot_code_annex.pdf).
10. ^ *a b c d e f g h i j k l m n* Smith, Rod. "Hybrid MBRs: The Good, the Bad, and the So Ugly You'll Tear Your Eyes Out" (http://www.rodsbooks.com/gdisk/hybrid.html).
11. ^ *a b c* "Windows and GPT FAQ" (http://www.microsoft.com/whdc/device/storage/GPT_FAQ.mspx#ELD). Microsoft.
12. ^ "Ubuntu on MacBook" (https://help.ubuntu.com/community/MacBook). Ubuntu Community Documentation.
13. ^ Smith, Rod. "GPT fdisk for Linux" (http://www.rodsbooks.com/gdisk).
14. ^ "GNU Parted FAQ" (https://www.gnu.org/software/parted/faq.shtml#features).
15. ^ "mklabel - GNU Parted Manual" (https://www.gnu.org/software/parted/manual/html_node/mklabel.html).
16. ^ "fdisk: add GPT support" (http://git.kernel.org/cgit/utils/util-linux/util-linux.git/commit/?id=766d5156c43b784700d28d1c1141008b2bf35ed7). kernel.org. 2013-09-27. Retrieved 2013-10-18.
17. ^ Davidlohr Bueso (2013-09-28). "fdisk updates and GPT support" (http://blog.stgolabs.net/2012/09/fdisk-updates-and-gpt-support.html). Retrieved 2013-10-18.
18. ^ "Myths and Facts About Intel Macs" (http://refit.sourceforge.net/myths/). rEFIt.
19. ^ "Booting From a ZFS Root File System" (http://docs.oracle.com/cd/E19963-01/html/821-1448/ggpco.html).
20. ^ "idisk(1M)" (http://h20000.www2.hp.com/bc/docs/support/SupportManual/c02256075/c02256075.pdf). Hewlett-Packard Co.
21. ^ Windows 8 32-bit supports booting from UEFI-based PC using GPT-based disks. (http://technet.microsoft.com/en-us/library/hh824898.aspx)
22. ^ Microsoft raises the speed limit with the availability of 64-bit editions of Windows Server 2003 and Windows XP Professional [1] (http://www.microsoft.com/presspass/press/2005/apr05/04-25Winx64LaunchPR.mspx)
23. ^ ftp://download.gigabyte.ru/manual/mb_manual_intel-ui_e.pdf
24. ^ "F6F: Funtoo Linux and Intel Rapid Start Technology" (http://blog.adios.tw/2012/10/funtoo-linux-and-intel-rapid-start.html). Blog.adios.tw. 2012-10-30. Retrieved 2014-01-29.
25. ^ GPT fdisk partitioning program 1.0.0 (http://www.rodsbooks.com/gdisk/)

25. ^ GPT fdisk: parttypes.cc, line 198 (http://sourceforge.net/p/gptfdisk/code/ci/master/tree/parttypes.cc)
26. ^ *a b* The Discoverable Partitions Specification (http://www.freedesktop.org/wiki/Specifications/DiscoverablePartitionsSpec/)
27. ^ *a b* "[dm-crypt] LUKS GPT GUID" (http://www.saout.de/pipermail/dm-crypt/2014-January/003855.html). Saout.de. Retrieved 2014-01-29.
28. ^ *a b* "[dm-crypt] LUKS GPT GUID" (http://www.saout.de/pipermail/dm-crypt/2014-January/003859.html). Saout.de. Retrieved 2014-01-29.
29. ^ "CVS log for src/sys/sys/disklabel_gpt.h" (http://cvsweb.netbsd.org/bsdweb.cgi/src/sys/sys/disklabel_gpt.h?only_with_tag=MAIN). Cvsweb.netbsd.org. Retrieved 2014-01-29.
30. ^ "Disk Format - The Chromium Projects" (http://www.chromium.org/chromium-os/chromiumos-design-docs/disk-format). Chromium.org. Retrieved 2014-01-29.
31. ^ src/add-ons/kernel/partitioning_systems/gpt/efi_gpt.cpp (http://cgit.haiku-os.org/haiku/tree/src/add-ons/kernel/partitioning_systems/gpt/efi_gpt.cpp)
32. ^ http://www.midnightbsd.org/cgi-bin/cvsweb.cgi/src/sys/sys/gpt.h.diff?r1=1.4;r2=1.5 src/sys/sys/gpt.h
33. ^ Script to set up a ceph disk: ceph-disk, lines 76-81 (https://github.com/ceph/ceph/blob/9bcc42a3e6b08521694b5c0228b2c6ed7b3d312e/src/ceph-disk#L76-L81)

External links

- Microsoft TechNet: Disk Sectors on GPT Disks (Archived page) (http://web.archive.org/web/20080321063028/http://technet2.microsoft.com/windowsserver/en/library/bdeda920-1f08-4683-9ffb-7b4b50df0b5a1033.mspx?mfr=true)
- Microsoft TechNet: Troubleshooting Disks and File Systems (http://technet.microsoft.com/en-us/library/bb457122.aspx)
- Microsoft TechNet: Using GPT Drives (http://www.microsoft.com/whdc/device/storage/GPT-on-x64.mspx)
- Microsoft: FAQs on Using GPT disks in Windows (http://www.microsoft.com/whdc/device/storage/GPT_FAQ.mspx)
- Apple Developer Connection: Secrets of the GPT (http://developer.apple.com/technotes/tn2006/tn2166.html)
- Make the most of large drives with GPT and Linux (http://www.ibm.com/developerworks/linux/library/l-gpt/)
- GPT fdisk: Information on Hybrid GPT-MBR, Converting MBR and BSD disklabels to GPT and Booting from GPT disks (http://rodsbooks.com/gdisk/)
- Convert Windows Vista SP1+ or 7 x86_64 boot from BIOS-MBR mode to UEFI-GPT mode without Reinstall (https://gitorious.org/tianocore_uefi_duet_builds/pages/Windows_x64_BIOS_to_UEFI)
- Rod Smith - A BIOS to UEFI Transformation (http://rodsbooks.com/bios2uefi/)
- Support for GPT (Partition scheme) and HDD greater than 2.19 TB in Microsoft Windows XP (http://www.ghacks.net/2010/11/04/how-to-use-3tb-hard-drives-on-windows-xp/)

- Setting up a RAID volume in Linux with >2TB disks (http://www.technotes.se/?p=1732)

Retrieved from "http://en.wikipedia.org/w/index.php?title=GUID_Partition_Table&oldid=646356533"

Categories: BIOS | Booting
