Forensic Analysis of GPT Disks and GUID Partition Tables

by Bruce J. Nikkel

nikkel@digitalforensics.ch

Originally published by Elsevier in Digital Investigation
The International Journal of Digital Forensics and Incident Response
Vol. 6, No. 1-2 (doi:10.1016/j.diin.2009.07.001)

November 19, 2009

Abstract

Modern computers are beginning to surpass the design limitations of the aging DOS/MBR partition tables and BIOS boot system. As disk sizes begin to exceed two terabytes and hardware vendors begin to transition from BIOS to EFI, understanding GPT disks in forensic examinations becomes useful. This practitioner paper provides an overview of the GUID Partition Table (GPT) scheme from the perspective of the digital forensic investigator. Methods of analysis and acquisition are shown, and artifacts of forensic relevance are identified. The target audience for this paper is digital forensic practitioners and forensic tool developers.

Keywords: GPT, EFI, GUID, MBR, BIOS, Boot Camp, MSR

Contents

1 Introduction to EFI and GPT

Modern computers are beginning to surpass the design limitations of the aging DOS/MBR partition tables and BIOS boot system of the original IBM PC. Although many extensions and enhancements to the original BIOS have been made over the years, these are limited in flexibility and extensibility. The GUID partition scheme and EFI boot framework were developed as a replacement, allowing for continued growth in disk capacity and providing a more flexible and advanced interface between the OS and the underlying firmware/hardware.

This paper provides an introduction to the Extensible Firmware Interface (EFI) and GUID Partition Tables (GPT) for digital forensic investigators. Various tools and methods of analyzing GPT partitions are demonstrated, and possible artifacts of forensic value within the context the EFI/GPT architecture are identified. The use of currently accepted digital forensic methods applied to GPT disks is reviewed.

1.1 History and development

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[1]. The MBR, located in sector zero, contains the initial boot code and a four entry partition table[2].

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[3]. This is now called the Unified EFI (UEFI) specification[4], and managed by the UEFI Forum[5]. A subset of this specification includes GUID (globally unique identification) Partition Tables, or GPT, intended to replace the DOS/MBR partition tables.

1.2 New partition features and functionality

The traditional MBR partitioning scheme supports maximum 2TB sized disk partitions since the MBR partition size field is fixed at 32 bits wide (maximum 0xFFFFFFFF sectors). The GPT architecture supports much larger disks, with a 64 bit wide partition size field for up to 0xFFFFFFFFFFFFFFFF sectors, or 8 Zettabytes[1].

The number of possible partitions has also been increased with GPT disks. While MBR disks supported the definition of four primary partitions, OS vendors typically support up to 128 primary partitions on a GPT disk[2]. Extended partitions on MBR disks provided a workaround to increase the number of usable partitions. Extended partitions are unnecessary and unsupported within the GPT scheme.

[1] Operating System support might be considerably less than this theoretical maximum.

[2] The GPT was designed to be extensible, so this could also be increased.

The original MBR did not allow for unique disk or partition labeling[3], but the GUID scheme provides extensive labeling and unique identification of both disks and partitions.

The GPT scheme provides for backup and integrity checking, while the traditional MBR provides neither. CRC32 checksums of the header and partition table are maintained. A secondary or alternate GPT header and partition table are also maintained in the last sectors of the disk as a backup.

1.3 Current support for EFI and GPT

The most popular hardware vendor using the EFI architecture is Apple Computer, having adopted the EFI specification for use in Intel Macintosh systems. EFI support is rather sparse among other hardware vendors, with usage mainly by the Intel Itanium family of systems. However, as disks surpass the 2TB limit and EFI boot capability becomes better supported by Microsoft, a market for low cost EFI mainboards will likely develop.

Operating system support for GPT disk data access (but without EFI boot functionality) is widespread, with most modern OS vendors supporting access and mounting of filesystems on GUID partitions. The most notable among these are Linux[6], Apple OSX[7], and modern Windows systems[8]. Traditional BIOS based systems are able to mount and access GPT disks, as the EFI system is only needed for GPT booting. Operating system support for EFI booting of GPT disks is still limited, with support by OSX, some Linux distributions[4], and Windows on Itanium platforms.

Common tools for creating, manipulating, and viewing GPT disks exist, and are typically included with the OS supporting GPT disk management. OSX provides the graphical *Disk Utility* as well as the command line *gpt* tool. Windows Vista, Server, and XP-64bit provide both GUI management as well as the command line tool DISKPART. The primary tool for managing GPT disks under Linux is the *parted* command line tool.

Forensic tool support for GPT disk analysis is still maturing. As of this writing forensic tools generally recognize GPT disks and provide access to filesystems on GUID partitions. However, improvements could be made by implementing more detailed descriptions and decoding of the GPT header and GPT entries. This could also include verification of GPT checksums, an overview of the protective MBR, and information about the secondary/backup GPT residing at the end of the disk.

The Sleuthkit opensource forensic tool suite provides support for GPT disks, and at the time of writing, further development work was being done to improve detection of GUID partitions. Carrier has also documented the analysis of GUID partitions in *File System Forensic Analysis*[9].

[3] An optional 32 bit disk identifier is added by some modern operating systems.

[4] Depending on the distribution, EFI booting with Linux may require a certain amount of tweaking to accomplish.

2 GPT Layout

To fully understand the limitations of the DOS/MBR partition tables and advantages of the GUID partition system, it is useful to compare the layouts and features of both. In figure 1 the relative simplicity of the MBR disk layout is observed, with a single initial sector providing the boot code and defining the partition layout of the rest of the disk.

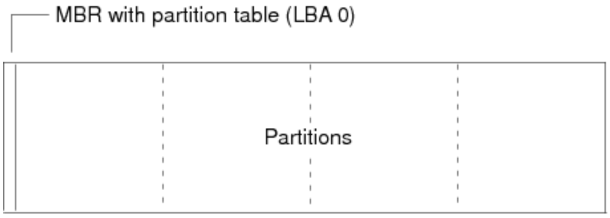


Figure 1: Traditional DOS/MBR disk layout

The GPT layout, seen in figure 2, is much more detailed in comparison. It provides a number of additional redundancy and extensibility features not available in the traditional DOS/MBR scheme.

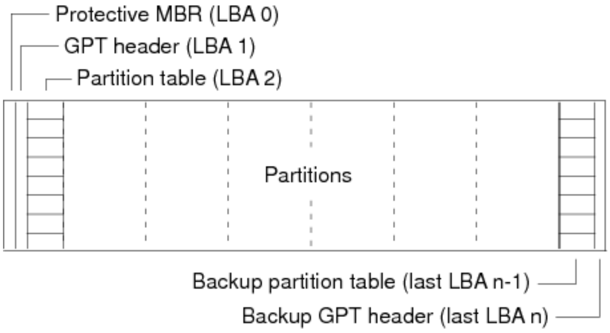


Figure 2: GPT disk layout

2.1 Traditional DOS/MBR partition scheme

The original DOS disk partition information was contained entirely in sector zero, called the Master Boot Record (MBR). This first sector was relatively simple, and contained the initial bootstrap code, four partition table entries, and a signature word. A diagram of the sector zero MBR is shown in figure 3.

Some recent operating systems also include a 32 bit disk signature located before the partition table. This is optional and intended to be unique to the disk. This concept has been extended and formalized in the GPT standard, and

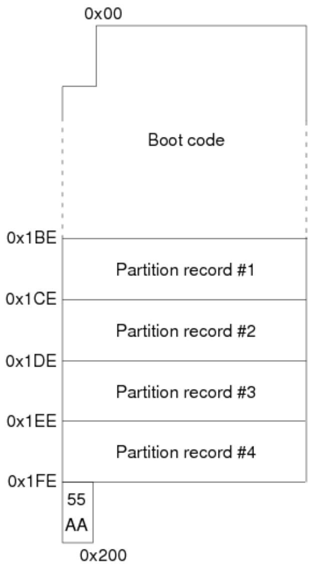


Figure 3: Traditional sector zero MBR

provides unique identifiers both for the entire disk as well as for every individual partition.

The individual 16 byte partition entries in the original MBR scheme provide the basic location, length, type, and bootable status. The basic partition entry can be seen in figure 4.

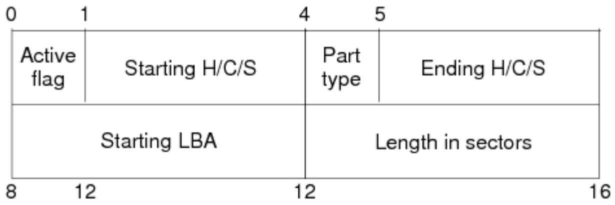


Figure 4: Traditional MBR partition entry

The use of *extended partitions* allows the creation of additional logical or secondary partitions, beyond the four primary partition limit. This workaround is unnecessary under the GPT partitioning scheme.

2.2 GPT header

The GPT partition scheme provides a number of additional features over the traditional DOS MBR partitioning scheme. To provide flexibility, the partition entries are separated from the initial sector. The GPT header layout is created with information about the disk as a whole, with a simple pointer to the location of partition entries. The GPT header, typically found in sector one, can be seen in figure 5. The header contains identifying information such as the "EFI PART" signature and a unique GUID specific to the whole disk which may be useful in a forensic context. Various locations, sizes, and checksums are also provided for integrity, and to locate other relevant GPT areas.

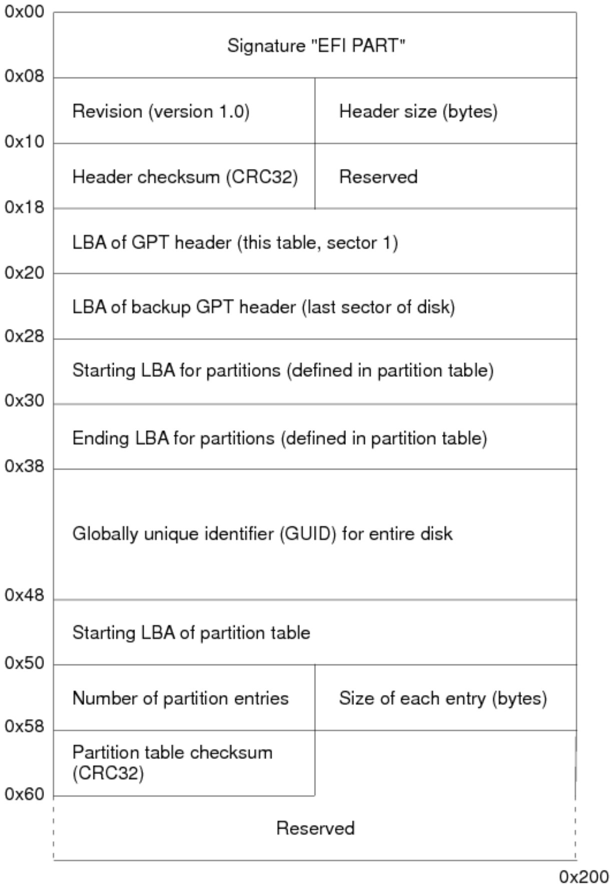

0x00
Signature "EFI PART"
0x08
Revision (version 1.0)
Header size (bytes)
0x10
Header checksum (CRC32)
Reserved
0x18
LBA of GPT header (this table, sector 1)
0x20
LBA of backup GPT header (last sector of disk)
0x28
Starting LBA for partitions (defined in partition table)
0x30
Ending LBA for partitions (defined in partition table)
0x38
Globally unique identifier (GUID) for entire disk
0x48
Starting LBA of partition table
0x50
Number of partition entries
Size of each entry (bytes)
0x58
Partition table checksum (CRC32)
0x60
Reserved
0x200

Figure 5: GPT Header

2.3 GPT entries

The individual partition entries in the GPT contain the location, unique identifying information, the partition type, and type specific attributes. The GPT entry layout can be seen in figure 6.

Of forensic interest here are further unique GUIDs specific to the individual partitions, and a 36 character user-defined partition name.

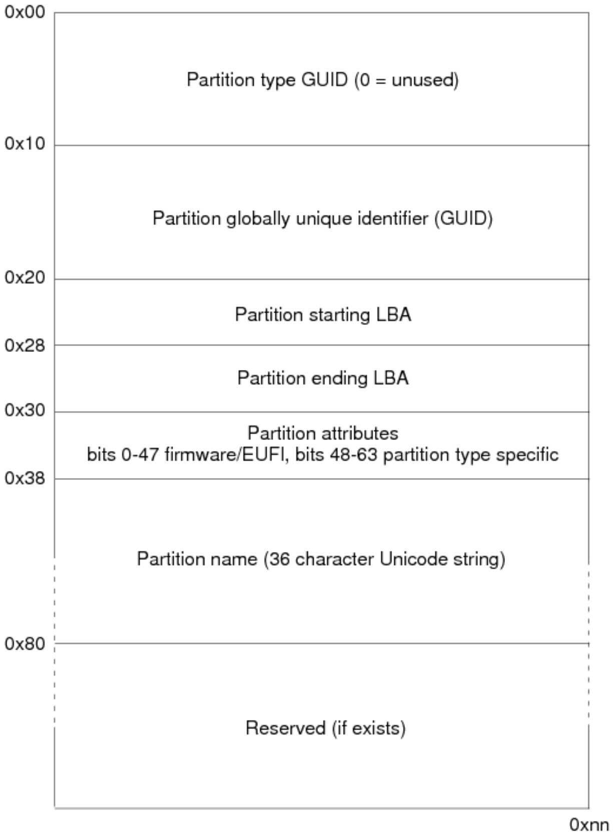

0x00
Partition type GUID (0 = unused)
0x10
Partition globally unique identifier (GUID)
0x20
Partition starting LBA
0x28
Partition ending LBA
0x30
Partition attributes
bits 0-47 firmware/EUFI, bits 48-63 partition type specific
0x38
Partition name (36 character Unicode string)
0x80
Reserved (if exists)
0xnn

Figure 6: GPT Entry

2.4 Protective MBR

To prevent older software tools and utilities from accidentally destroying GUID partitions, the *Protective MBR*[5] was created. If a tool doesn't support or recongnize GPT, it will at least think that the entire disk is in use by another (possibly unknown) partition. The protective MBR type is 0xEE and defines a 'placeholder' partition spanning the entire disk. If a GPT disk is larger than 2TB, the protective MBR will simply span the entire 2TB space allowed under the MBR partition scheme. Shown here is a protective MBR on a GPT disk:

```
Disk /dev/md0: 4500.9 GB, 4500905459712 bytes
255 heads, 63 sectors/track, 547203 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
Disk identifier: 0x00000000

    Device Boot Start     End      Blocks   Id  System
/dev/md0p1          1  267350  2147483647+  ee  EFI GPT
```

[5] Some forensic tools refer to this partition as the GPT safety partition.

The protective MBR for this 4.5TB GPT disk[6] contains a single entry, 2TB in size, the maximum size possible with a DOS/MBR partitioning scheme.

2.5 Backup/alternate GPT

One of the useful features of the GPT scheme is the consistency checking and backup capability. All GPT disks are required to maintain a backup of the GPT header and partition table. This backup is located at the end of the disk. Shown here is an example of the last sector on a small 8GB GPT disk, containing the backup GPT header:

```
E6E7FE00   45 46 49 20 50 41 52 54  00 00 01 00 5C 00 00 00 EFI PART....\...
E6E7FE10   17 31 26 C6 00 00 00 00  FF 73 F3 00 00 00 00 00 .1&......s......
E6E7FE20   01 00 00 00 00 00 00 00  22 00 00 00 00 00 00 00 ........".......
E6E7FE30   DE 73 F3 00 00 00 00 00  42 6A C3 63 75 BD 92 40 .s......Bj.cu..@
E6E7FE40   8F AE D0 D5 82 05 11 99  DF 73 F3 00 00 00 00 00 .........s......
E6E7FE50   80 00 00 00 80 00 00 00  29 45 20 2F 00 00 00 00 ........)E /....
E6E7FE60   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 ................
[rest of sector zeros]
```

The partition table array ends in the sector before the GPT header (2nd last sector of the disk). The starting sector of the backup partition table array can be calculated from fields in the GPT header.

There is no backup copy of the protective MBR sector.

3 EFI system booting

The GPT scheme is a subset of a larger standard designed to facilitate the entire boot process. This is the Extended Firmware Interface or EFI, and is a direct replacement for the traditional BIOS (Basic Input/Output System). This is covered very briefly here, and only for completeness. Forensic analysis of EFI artifacts not directly related to the GPT are outside the scope of this paper.

3.1 EFI vs BIOS

The EFI framework provides a bridge between the operating system and the system firmware. During boot, the EFI firmware jumps directly to the EFI system partition to begin the initial bootstrapping[7]. The EFI system is able to find and interpret the system partition, and execute the necessary files needed to initialize and load the OS. This in contrast to the simple way BIOS jumps to sector 0 and immediately begins executing machine code. EFI is intended to replace the BIOS which only provides a basic interface to the hardware.

[6] This is a virtual disk created using multiple disks in a RAID.

[7] This renders a large number of known MBR/boot-sector trojans and virii harmless, since sector zero MBR code is not executed at boot on EFI systems

3.2 EFI system partition

The availability of many partitions provides additional flexibility with the boot process. This can been seen with the use of the EFI System Partition (ESP) to implement enhanced boot processes[8]. In addition to OS boot components, the EFI system partition may also contain EFI shells, or command line interfaces, providing direct access to the EFI subsystem and firmware. A free EFI shell was originally made available by from Intel[10]. The EFI system partition is often a FAT32 filesystem, and can be analyzed with regular forensic tools.

3.3 Apple Boot Camp

Apple's Boot Camp was developed to allow operating systems (primarily windows) to load on Macintosh hardware[9]. To achieve this, Boot Camp provides a BIOS compatibility module to allow native booting without EFI support. This has also been called "BIOS emulation", "hybrid GPT/MBR", "BIOS-based booting" or "legacy OS booting"[11].

3.4 Microsoft MSR

According to Microsoft, the Microsoft Reserved partition (MSR) on a GPT disk is designed to replace the currently used hidden sectors, which no longer function under EFI/GPT[12]. The MSR is for proprietary use by Microsoft, and is required to reside immediately after the EFI system partition. It is either 32MB or 128MB in size[8], depending on the size of the GPT disk.

3.5 Linux EFI support

Linux kernels are able to boot EFI systems natively, without BIOS emulation, but this is largely dependent on the Linux distribution and initial boot loader used. An EFI boot manager rEFIt[13] is available to dual boot Linux on Macs. Several native EFI boot loaders, ELILO[14] and GRUB2[15] also exist for Linux.

4 Analyzing the GPT header and entries

In this section several examples of tools and techniques are shown for extracting information about GPT partitions. Most operating systems supporting GPT disk access provide a basic partitioning tool which can be used to view details about GPT partition tables.

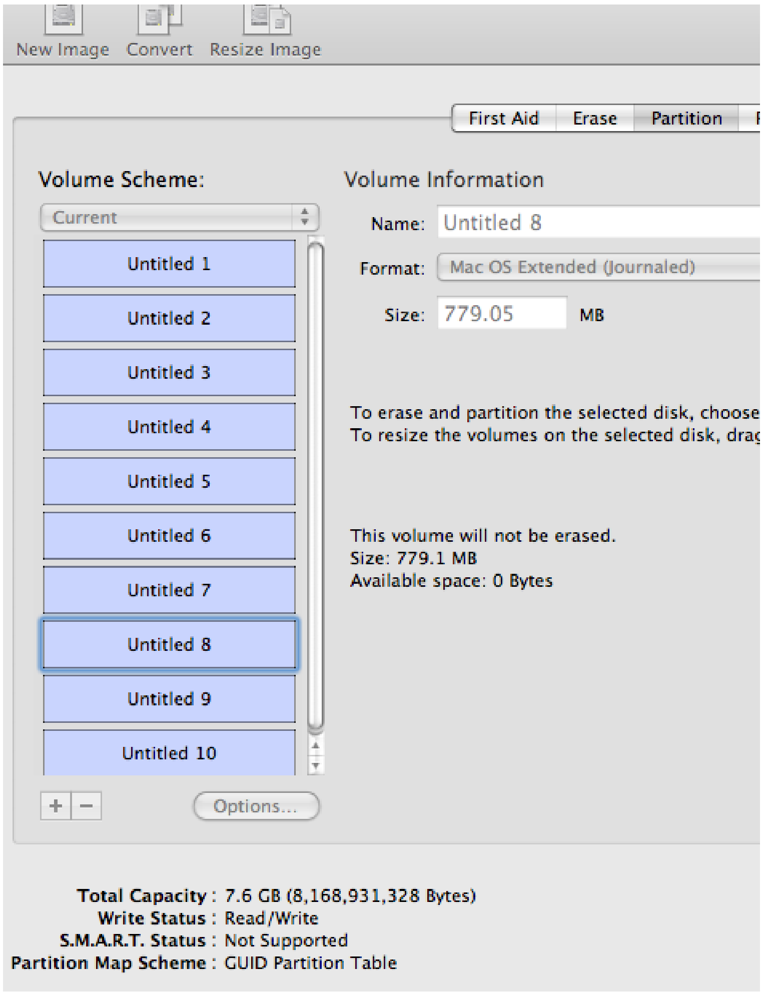
4.1 OSX partition tool

The OSX *Disk Utility* provides a graphical tool for mangaging disk partitions, including the GPT partition scheme. Figure 7 shows the tool viewing a ten partition GPT disk. The OSX Disk Utility provides complete graphical support for GPT disk management as a standard component of the operating system. The command line tool *gpt* is also provided[10].

[8] This system partition could potentially become a Malware target in the future

[9] In addition, boot camp also provides safe repartitioning and device drivers for windows

[10] The gpt command line tool was taken from FreeBSD.



Volume Information

Name: Untitled 8

Format: Mac OS Extended (Journaled)

Size: 779.05 MB

To erase and partition the selected disk, choose
To resize the volumes on the selected disk, drag

This volume will not be erased.
Size: 779.1 MB
Available space: 0 Bytes

Total Capacity : 7.6 GB (8,168,931,328 Bytes)
Write Status : Read/Write
S.M.A.R.T. Status : Not Supported
Partition Map Scheme : GUID Partition Table

Figure 7: Viewing a GPT disk with OSX Disk Utility

4.2 Linux partition tool

The GNU *parted* tool provides gpt partition management for Linux systems. Basic information about GPT disks can be viewed. A simple example of a GPT disk layout analyzed with parted is shown in here.

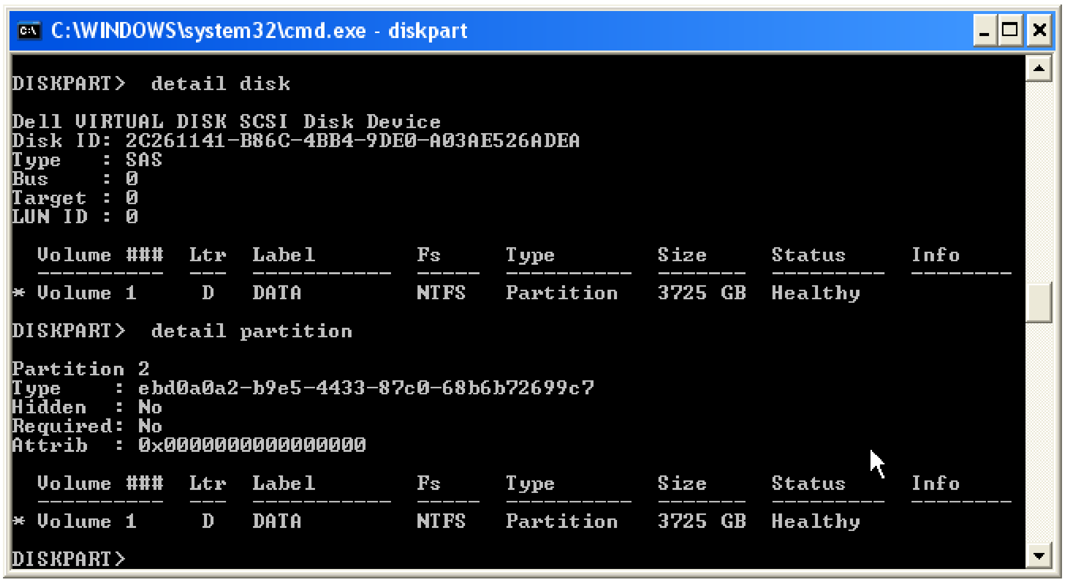
```
# parted /dev/sdg print

Disk /dev/sdg: 120GB
Sector size (logical/physical): 512B/512B
Partition Table: gpt

Number  Start   End     Size    File system  Name                  Flags
 1      20.5kB  210MB   210MB   fat32        EFI System Partition  boot
 2      210MB   86.0GB  85.8GB  hfs+         Untitled
 3      86.1GB  120GB   33.9GB  ntfs         Untitled
```

4.3 Microsoft Windows partition tool

The Windows DISKPART tool provides a command line tool for managing GPT disks under windows. In addition to viewing the partitions on the disk, the DISKPART tool also provides information about the GUID's of the disk and the partition types. An example can be seen in figure 8.



```
DISKPART> detail disk

Dell VIRTUAL DISK SCSI Disk Device
Disk ID: 2C261141-B86C-4BB4-9DE0-A03AE526ADEA
Type   : SAS
Bus    : 0
Target : 0
LUN ID : 0

  Volume ###  Ltr  Label        Fs     Type        Size     Status     Info
  ----------  ---  -----------  -----  ----------  -------  ---------  --------
* Volume 1     D   DATA         NTFS   Partition   3725 GB  Healthy

DISKPART> detail partition

Partition 2
Type     : ebd0a0a2-b9e5-4433-87c0-68b6b72699c7
Hidden   : No
Required: No
Attrib   : 0x0000000000000000

  Volume ###  Ltr  Label        Fs     Type        Size     Status     Info
  ----------  ---  -----------  -----  ----------  -------  ---------  --------
* Volume 1     D   DATA         NTFS   Partition   3725 GB  Healthy

DISKPART>
```

Figure 8: Viewing GPT disks with DISKPART

4.4 Sleuthkit mmls

Shown here is example output from the Sleuthkit mmls command. A detailed partition layout is shown, and includes the protective MBR (Safety Table), partitions, and inter-partition gaps[11]. The mmls tool provides a view of the start/end sectors of each partition.

```
# mmls -t gpt /dev/sdg
GUID Partition Table (EFI)
Offset Sector: 0
Units are in 512-byte sectors

     Slot    Start        End          Length       Description
00:  Meta    0000000000   0000000000   0000000001   Safety Table
01:  -----   0000000000   0000000039   0000000040   Unallocated
02:  Meta    0000000001   0000000001   0000000001   GPT Header
03:  Meta    0000000002   0000000033   0000000032   Partition Table
```

[11] The backup header and backup partition table are not currently shown by mmls.

```
04:  00      0000000040   0000409639   0000409600   EFI System Partition
05:  01      0000409640   0167919655   0167510016   Untitled
06:  -----   0167919656   0168183807   0000264152   Unallocated
07:  02      0168183808   0234440703   0066256896   Untitled
08:  -----   0234440704   0234441647   0000000944   Unallocated
```

4.5 Hex dumpers/editors

In cases where tools do not support GPT disks, or don't provide enough information, details can be extracted through a manual analysis of the GPT header and partition entries. An example of this is shown here using a standard hex editor.

Pointing the hex editor to sector one (not sector zero, as it is the protective MBR) of the disk provides a dump of the GPT header:

```
00000200   45 46 49 20 50 41 52 54  00 00 01 00 5C 00 00 00  EFI PART....\...
00000210   9B 87 49 72 00 00 00 00  01 00 00 00 00 00 00 00  ..Ir............
00000220   AF 4B F9 0D 00 00 00 00  22 00 00 00 00 00 00 00  .K......".......
00000230   8E 4B F9 0D 00 00 00 00  A9 CD 82 8B 1F F3 A6 47  .K.............G
00000240   B4 58 64 D6 71 B4 60 57  02 00 00 00 00 00 00 00  .Xd.q.‘W........
00000250   80 00 00 00 80 00 00 00  AC E2 28 41 00 00 00 00  ..........(A....
```

This hex dump of the GPT header can be decoded using the table in figure 5. The subsequent partition table can also be manually decoded using the table in figure 6.

The backup GPT header can be viewed by pointing the hexeditor to the last sector on the disk. The backup partition table can be viewed by working backwards from the last sector on the disk.

5 Acquiring GPT disks and partitions

5.1 GPT disk acquisition

Performing a full disk acquisition is no different than acquiring disks with any other partitioning scheme (DOS, BSD, older Macs, etc.). The GPT scheme only refers to how the partitions are defined and organized across a disk. Acquiring the disk starting with sector zero and proceeding to the last sector on the disk will produce a complete disk image which can be hashed for integrity/preservation. Cloning or duplicating a GPT disk is also accomplished the same way[12].

5.2 Extracting individual GPT partitions

The filesystems residing on GPT disks are independent of the partitioning scheme. It is possible to extract individual GUID partitions, and analyze them in the same way as traditional DOS/MBR partitions. Some examples of extracting a partition from a GPT disk with Linux are shown here.

[12] When duplicating a disk at the sector level, the GUID of the disk is no longer unique as defined in the standards

Using dcfldd to extract the 13th partition via the device file[13]:

```
# dcfldd if=/dev/sdb13 of=partition.dd
```

Using Sleuthkit's mmcat[14] tool to extract a partition via the raw disk device:

```
# mmcat -t gpt /dev/sdb 18 > partition.dd
```

As of this writing, Sleuthkit tools may need to be explicitly told (-t gpt) if a disk uses a GPT partition scheme. The number 18 in this last example is not the 18th partition, but corresponds to the Sleuthkit mmls slot number.

5.3 HPA and DCO

Disks containing a Host Protected Area (HPA) or Device Configuration Overlay (DCO) can also be forensically imaged in the normal way, but this may have an effect on the analysis of GPT disks. Removing a HPA or DCO will extend the user accessible area, changing the location of the last sector. The GPT system expects the last sectors of the disk to contain the backup/secondary copy of the partition header and partition entries. However, a GPT disk with a removed HPA or DCO will no longer have the backup partition header and table at the end of the disk. This could potentially confuse some GPT analysis tools and provide unexpected/incorrect results.

6 GPT artifacts and reconstruction

This section describes some of the GPT specific artifacts which are of forensic interest. It also discusses the reconstruction of deleted/overwritten partitions and data hiding.

6.1 Deleted and overwritten GUID partitions

An MBR disk repartitioned or converted to GPT will typically have sector zero overwritten with a protective MBR, leaving no evidence of the previous partition table. Standard forensic methods of exhaustively searching for filesystems still apply to a converted GPT disk, and data from the previous MBR partitioned volumes might be recovered.

Depending on the tool used, a GPT disk repartitioned or converted to MBR style partitioning may leave the GPT header and tables intact (either the primary, alternate, or both). These can then be recovered or analyzed. A suspected individual using common tools to delete partitions on an incriminating GPT disk might simply be deleting the protective MBR, leaving the GPT information untouched. Reconstructing a disk in such a case is a simple matter of recreating the protective MBR.

[13] The OS kernel must recognize GPT disks and provide the relevant device files.

[14] The mmcat tool is available in Sleuthkit v3

The UEFI specification states that if all fields are zeroed out in a partition entry, then the entry is not in use. This prevents recovery of useful information about deleted GUID partition entries.

6.2 Analyzing the GUID identifiers

The GPT scheme provides several globally unique identifiers (GUIDs) which uniquely identify both the entire disk, and each individual partition. These GUIDs are of forensic value because of their uniqueness, and for the potential information encoded within. EFI GUID's follow the format defined in RFC 4122[16].

If an operating system has a log, history, or other artifacts containing disk GUIDs or partitions GUIDs used, the unique nature of the GUID provides strong evidence linking the use of a particular disk to a system.

In some cases, GUIDs may contain information encoded into the GUID string itself. For example, a version 1 GUID contains a timestamp and possibly a network MAC address. According to the EFI standard, the GUIDv1 timestamp represents the creation timestamps of the GPT header and each partition entry. However, for privacy reasons, many vendors are now using randomly generated UUID's, reducing the amount of useful information that can be extracted.

A useful tool for decoding various versions of GUID/UUID is the *uuid* tool[15]. Shown here is an example of a decoded version 1 GUID, providing a timestamp and the computer's network MAC address:

```
$ uuid -d 1d49cc1e-2937-11de-a42e-001d7d479397
encode: STR:     1d49cc1e-2937-11de-a42e-001d7d479397
        SIV:     38930789549595083809781482993362637719
decode: variant: DCE 1.1, ISO/IEC 11578:1996
        version: 1 (time and node based)
        content: time:  2009-04-14 20:59:15.900931.0 UTC
                 clock: 9262 (usually random)
                 node:  00:1d:7d:47:93:97 (global unicast)
```

6.3 Hiding information on GPT disks

A suspected individual with a GPT disk has the possibility of hiding data in similar ways as with traditional MBR disks. Having a flexible and extensible disk partitioning scheme like the GPT provides additional opportunities to create areas for data hiding.

Such data hiding could include leaving inter-partition gaps (spaces in between defined partitions), or putting data in unpartitioned space towards the end of the disk (but without destroying the backup GPT).

[15] This is an optional software package which can be installed for free on Linux

Depending on the vendor implementation of the GPT standard, it might also be possible to manipulate the GPT header[16] to create places for data hiding. For example, moving the starting/ending LBAs for the partition area, or increasing the partition entry number/size to create possible data hiding places. Also, any areas designated as reserved by the specification could be used as potential hiding places (some of these areas might not be practical, as they may not provide a lot of space).

7 Conclusion

The GPT is a partitioning scheme which is poised to become more widely used in the near future. Most modern operating systems currently support GPT data partitions. Apple is using GPT as the default partitioning scheme with EFI for booting OSX on Intel Macs[17]. All operating systems must support GPT to use disks greater than 2TB. Having a basic understanding of the GUID partitioning scheme is becoming important for digital forensic investigators.

This paper has shown that while many new concepts are introduced with the GPT, the residing file systems are still the same and can be analyzed using familiar methods and techniques. The process of forensic acquisition also remains unchanged for acquiring GPT disks, and extracting/accessing individual GUID partitions is relatively straight forward.

The forensic analysis possibilities for extracting useful information are much greater than previously possible with MBR/DOS style partition tables. The GPT provides unique identifying information for both disks and individual partitions. It provides additional possibilities for reconstructing deleted partitions. The possibility of data hiding among various parts of a GPT disk are also greater than with a simple MBR/DOS partitions.

Current forensic tools and methods for analyzing the GPT are still in the process of maturing. Not all forensic tools have full support for GPT analysis, especially for verifying the backup/secondary GPT in the final sectors of the disk, or decoding the GPT header and partition entries.

7.1 Further work

Further work in this area could include a study of OS specific artifacts related to GPT mounted disks. If an operating system stores GUID information about previously attached GPT disks or mounted GUID partitions, this could be useful in strongly linking a particular disk to a particular system.

A more comprehensive feasibility study of data hiding on GPT disks would be useful, especially through manipulation of the GPT header.

[16] It should be noted that this is not trivial manipulation, and requires the checksum be recalculated on both the primary and alternate GPT headers.

[17] Apple PPC systems use Open Firmware, not EFI.

A survey of forensic tools which describes the extent of support for GPT disks, shortcomings, etc. would also be interesting for the digital forensics community.

References

[1] IBM, Technical Reference, Personal Computer XT Hardware Reference Library, 1983

[2] Microsoft, MS-DOS 2.0, http://support.microsoft.com/kb/69912, 1983

[3] Intel, Extensible Firmware Interface Specification 1.02, 2000

[4] UEFI Forum, Unified Extensible Firmware Interface Specification Version 2.2, 2008

[5] UEFI Forum, http://www.uefi.org/

[6] GNU, GNU Parted, http://www.gnu.org/software/parted, version 1.8.8, released 2007

[7] Apple, Technical Note TN2166, Secrets of the GPT, 2006

[8] Microsoft, Windows and GPT FAQ, Version 1.1, http://www.microsoft.com/whdc/device/storage/GPT_FAQ.mspx, 2006

[9] Brian Carrier, File System Forensic Analysis, Addison Wesley, 2005

[10] efi-shell Project home, https://efi-shell.tianocore.org/, 2009

[11] rEFIt homepage, http://refit.sourceforge.net/myths, Myths and Facts About Intel Macs, 2006

[12] Mark Kieffer, Microsoft Windows & EFI, Microsoft, Slides from Intel Developer Forum, 2000

[13] The rEFIt Project, http://refit.sourceforge.net, 2009

[14] ELILO: EFI Linux Boot Loader, http://sourceforge.net/projects/elilo, 2009

[15] GRUB2, http://www.gnu.org/software/grub/grub-2.en.html, 2009

[16] RFC4122, A Universally Unique IDentifier (UUID) URN Namespace, Network Working Group, 2005

Document History

Feb 2009 - April 2009: Created original paper
April 2009 Submitted to Elsevier for review
Jul 2009 Accepted for publication
Aug 2009 Published by Elsevier
Nov 2009 My version made available on my personal website