**Understanding Digital data and storage systems**

Digital data is data that represents other forms of data using specific machine language systems that can be interpreted by various technologies. The most fundamental of these systems is a binary system, which simply stores complex audio, video or text information in a series of binary characters, traditionally ones and zeros, or "on" and "off" values. Digital data is the**electronic representation of information in a format or language that machines can read and understand**. In more technical terms, digital data is a binary format of information that's converted into a machine-readable digital format.

Digital data storage is mostly offline storage for backup and fail-safe data. In recent years, digital data has evolved to include cloud storage. It is a server that hosts all types of uploaded data, including media files. It generally exists to serve companies that have large amounts of data and need it protected and backed up.

In a technical sense, data is stored as code or numbers for a computer to read and control. It’s then guided based on the computer input rules and stored in different locations. Data within files can be stored offline in different drive types, on a physical location like a hard drive and online in the cloud.

Data storage refers to magnetic, optical or mechanical media that records and preserves digital information for ongoing or future operations.

**Different ways to store data**

There are different types of data storage and it’s important to understand how they contrast from one another. RAM, or memory, is temporary data storage so the computer can quickly access it. The data stored here isn’t permanent. It instead allows a computer to read data fast, opposed to the slower alternatives of storage.

The other type of storage is a device like a hard drive, which holds permanent data unlike the RAM storage. This storage is potentially mobile, as the drive can be external as well as internal. Personal data within different types of files including media generally are stored onto things like the hard drive or external drives for future use. It also helps the uploader retrieve data quickly in the future.

**Understanding the Boot Sequence**

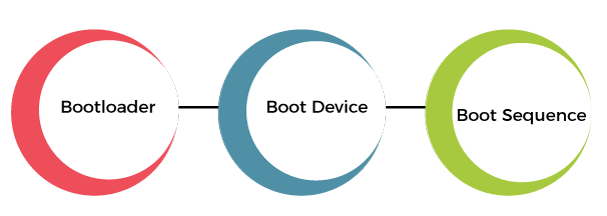
The boot sequence—sometimes called BIOS boot sequence or BIOS boot order—is the order of devices listed in BIOS that the computer will look for an operating system on. Although a hard drive is usually the main device a user may want to boot from, other devices like optical drives, floppy drives, flash drives, and network resources are all typical devices that are listed as boot sequence options in the BIOS.

Typically, a Macintosh structure uses ROM and Windows uses BIOS to start the boot sequence. Once the instructions are found, the CPU takes control and loads the OS into system memory.

The devices that are usually listed as boot order options in the BIOS settings are hard disks, floppy drives, optical drives, flash drives, etc. The user is able to change the boot sequence via the CMOS setup.

### **Sequencing of Booting**

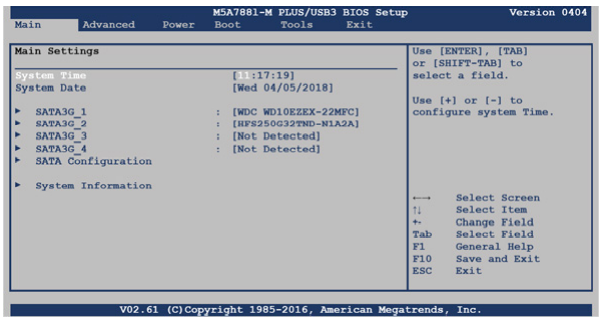
Booting is a start-up sequence that starts the operating system of a computer when it is turned on. A boot sequence is the initial set of operations that the computer performs when it is switched on. Every computer has a boot sequence.



For tablets and smartphones, it’s best to review vendors’ documentation. To ensure that you don’t contaminate or alter data on a suspect’s system, you must know how to access and modify Complementary Metal Oxide Semiconductor (CMOS), BIOS, Extensible Firmware Interface (EFI), and Unified Extensible Firmware Interface (UEFI) settings. A computer stores system configuration and date and time information in the CMOS when power to the system is off. The system BIOS or EFI contains programs that perform input and output at the hardware level. BIOS is designed for x86 computers and typically used on disk drives with Master Boot Records (MBRs). EFI is designed for x64 computers and uses GUID Partition Table (GPT)– formatted disks.

When a subject’s (victim’s) computer starts, you must make sure it boots to a forensically configured CD, DVD, or USB drive, because booting to the hard disk overwrites and changes evidentiary data. To do this, you access the CMOS setup by monitoring the computer during the bootstrap process to identify the correct key or keys to use. The bootstrap process , which is contained in ROM, tells the computer how to proceed. As the computer starts, the screen usually displays the key or keys, such as the Delete key, you press to open the CMOS setup screen. You can also try unhooking the keyboard to force the system to tell you what keys to use. The key you press to access CMOS depends on the computer’s BIOS. Many BIOS manufacturers use the Delete key to access CMOS; other manufacturers use Ctrl+Alt+Insert, Ctrl+A, Ctrl+S, or Ctrl+F1, F2, or F10.

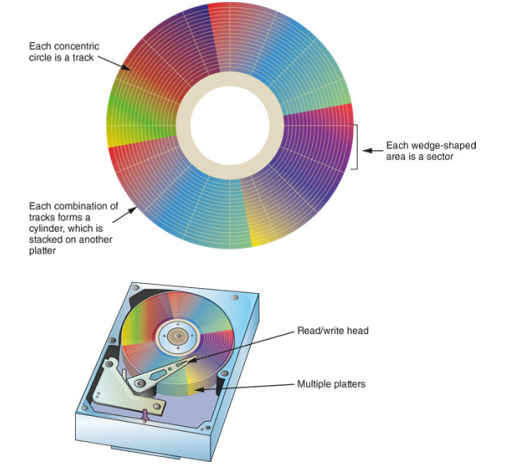
Figure shows a typical CMOS setup screen, where you check a computer’s boot sequence. If necessary, you can change the boot sequence so that the OS accesses the CD/DVD drive, for example, before any other boot device. Each BIOS vendor’s screen is different, but you can refer to the vendor’s documentation or Web site for instructions on changing the boot sequence.



**Understanding Disk Drives**

You should be familiar with disk drives and how data is organized on a disk so that you can find data effectively. Disk drives are made up of one or more platters coated with magnetic material, and data is stored on platters in a particular way. The following disk drive components are illustrated in Figure

* Geometry— Geometry refers to a disk’s logical structure of platters, tracks, and sectors.
* Head—The head is the device that reads and writes data to a drive. There are two heads per platter that read and write the top and bottom sides.
* Tracks— Tracks are concentric circles on a disk platter where data is located.
* Cylinders—A cylinder is a column of tracks on two or more disk platters. Typically, each platter has two surfaces: top and bottom.
* Sectors—A sector is a section on a track, usually made up of 512 bytes



The manufacturer engineers a disk to have a certain number of sectors per track, and a typical disk drive stores 512 bytes per sector. To determine the total number of addressable bytes on a disk, multiply the number of cylinders by the number of heads (actually tracks) and by the number of sectors (groups of 512 or more bytes). Disk drive vendors refer to this formula as a “cylinder, head, and sector (CHS) calculation.” Tracks also follow a numbering scheme starting from 0, which is the first value in computing. If a disk lists 79 tracks, you actually have 80 tracks from 0 to 79.

Other disk properties, such as **zone bit recording (ZBR**) , **track density , areal density , and head and cylinder skew** , are handled at the drive’s hardware or firmware level. ZBR is how most manufacturers deal with a platter’s inner tracks having a smaller circumference (and, therefore, less space to store data) than its outer tracks. Grouping tracks by zones ensures that all tracks hold the same amount of data. Track density is the space between each track. As with old vinyl records, the smaller the space between each track, the more tracks you can place on the platter. Areal density is the number of bits in one square inch of a disk platter. This number includes the unused space between tracks. Head and cylinder skew are used to improve disk performance. As the read-write head moves from one track to another, starting sectors are offset to minimize lag time.

**Solid-State Storage Devices**

Flash memory storage devices used in USB drives, laptops, tablets, and cell phones can be a challenge for digital forensics examiners because if deleted data isn’t recovered immediately, it might be lost forever. The reason is a feature all flash memory devices have: **wear-leveling .**

When data is deleted on a hard drive, only the references to it are removed, which leaves the original data in unallocated disk space. With forensics recovery tools, recovering data from magnetic media is fairly easy; you just copy the unallocated space.

USB drives and other solid state drive systems are different, in that memory cells shift data at the physical level to other cells that have had fewer reads and writes continuously. The purpose of shifting (or rotating) data from one memory cell to another is to make sure all memory cells on the flash drive wear evenly. Memory cells are designed to perform only 10,000 to 100,000 reads/writes, depending on the manufacturer’s design. When they reach their defined limits, they can no longer retain data. When you attempt to connect to the device, you get an access failure message. This process is controlled on the flash device’s firmware.

When dealing with solid-state devices, making a full forensic copy as soon as possible is crucial in case you need to recover data from unallocated disk space. You can test this feature with a USB drive easily by copying data to it, deleting it, and then making a forensic acquisition with any acquisition tool, such as OSForensics or X-Ways Forensics, immediately after the data is deleted. The first acquisition produces recoverable artifacts. If you let the USB drive sit and write no additional data to it, wear-leveling automatically overwrites the unallocated space. All solid-state drives have an internal power source for memory cells (both allocated and unallocated) so that they can preserve data. If you make another acquisition of the USB drive a day or more later, it reveals that the previously recoverable deleted data no longer exists. For mobile device forensics, this feature is extremely important, especially if a suspect deleted relevant messages, for example, just before the device was seized and taken into evidence.

**Exploring Microsoft File Structures**

Because most PCs use Microsoft software products, you should understand Microsoft file systems so that you know how Windows and DOS computers store files. In particular, you need to understand **clusters, File Allocation Table (FAT), and NT File System (NTFS).** The method an OS uses to store files determines where data can be hidden. When you examine a computer for forensic evidence, you need to explore these hiding places to determine whether they contain files or parts of files that might be evidence of a crime or policy violation.

In Microsoft file structures, sectors are grouped to form **clusters** , which are storage allocation units of one or more sectors. Clusters range from 512 bytes up to 32,000 bytes each. Combining sectors minimizes the overhead of writing or reading files to a disk. The OS groups one or more sectors into a cluster. The number of sectors in a cluster varies according to the disk size.

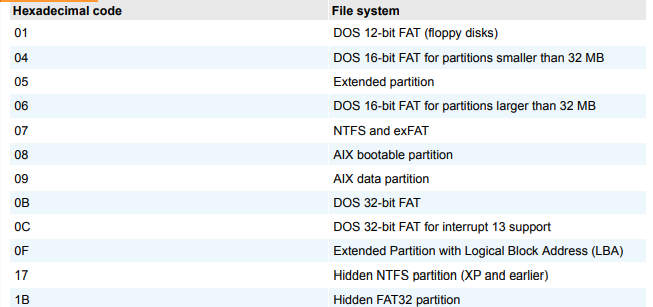
For example, a double-sided floppy disk has one sector per cluster; a hard disk has four or more sectors per cluster. Clusters are numbered sequentially, starting at 0 in NTFS and 2 in FAT. The first sector of all disks contains a system area, the boot record, and a file structure database. The OS assigns these cluster numbers, referred to as logical addresses . They point to relative cluster positions; for example, cluster address 100 is 98 clusters from cluster address 2. Sector numbers, however, are referred to as physical addresses because they reside at the hardware or firmware level and go from address 0 (the first sector on the disk) to the last sector on the disk. Clusters and their addresses are specific to a logical disk drive, which is a disk partition.

**Disk Partitions**

Many hard disks are partitioned, or divided, into two or more sections. A partition is a logical drive. Windows OSs can have **three primary partitions followed by an extended partition** that can contain one or more logical drives. Someone who wants to hide data on a hard disk can create hidden partitions or voids—large unused gaps between partitions on a disk drive. For example, partitions containing unused space can be created between the primary partitions or logical partitions. This unused space between partitions is called the **partition gap** .

It’s possible to create a partition, add data to it, and then remove references to the partition so that it can be hidden in Windows. If data is hidden in this partition gap, a disk editor utility could be used to access it. Another technique is to hide incriminating digital evidence at the end of a disk by declaring a smaller number of bytes than the actual drive size. With disk editing tools, however, you can access these hidden or empty areas of the disk. One way to examine a partition’s physical level is to use a disk editor, such as **WinHex or Hex Workshop**. These tools enable you to view file headers and other critical parts of a file. Both tasks involve analyzing the key hexadecimal codes the OS uses to identify and maintain the file system

Table lists the hexadecimal codes in a partition table and identifies some common file system structures.



The partition table is in the Master Boot Record (MBR) , located at sector 0 of the disk drive. In a hexadecimal editor, such as WinHex, you can find the first partition starting at offset 0x1BE. The second partition starts at 0x1CE, the third partition starts at 0x1DE, and the fourth partition starts at 0x1EE

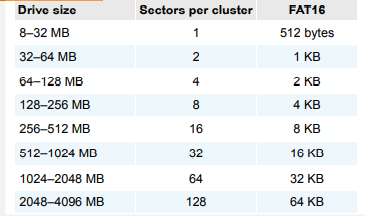
**Examining FAT Disks**

File Allocation Table (FAT) is the file structure database that Microsoft designed for floppy disks. It’s used to organize files on a disk so that the OS can find the files it needs. Since its development, other OSs, such as Linux and Macintosh, can format, read, and write to FAT storage devices such as USB drives and SD cards. The FAT database is typically written to a disk’s outermost track and contains filenames, directory names, date and time stamps, the starting cluster number, and file attributes (archive, hidden, system, and read-only).

There are three current versions of FAT—FAT16, FAT32, and exFAT (used for mobile personal storage devices)—and three older FAT formats, which are FATX, Virtual FAT (VFAT), and FAT12. The FAT version in Microsoft DOS 6.22 had a limitation of eight characters for filenames and three characters for extensions. The following list summarizes the evolution of FAT versions:

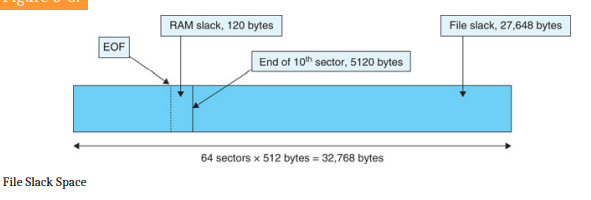
* **FAT12**—This version is used specifically for floppy disks, so it has a limited amount of storage space. It was originally designed for MS-DOS 1.0, the first Microsoft OS, used for floppy disk drives and drives up to 16 MB.
* **FAT16**—To handle larger disks, Microsoft developed FAT16, which is still used on older Microsoft OSs, such as MS-DOS 3.0 through 6.22, Windows 95 (first release), and Windows NT 3.5 and 4.0. FAT16 supports disk partitions with a maximum storage capacity of 4 GB.
* **FAT3**2—When disk technology improved and disks larger than 2 GB were developed, Microsoft released FAT32, which can access larger drives.
* **exFAT**—Developed for mobile personal storage devices, such as flash memory devices, secure digital eXtended capacity (SDCX), and memory sticks. The exFAT file system can store very large files, such as digital images, video, and audio files.
* **VFAT**—Developed to handle files with more than eight-character filenames and threecharacter extensions; introduced with Windows 95. VFAT is an extension of other FAT file systems.

Cluster sizes vary according to the hard disk size and file system. Table lists the number of sectors and bytes assigned to a cluster on FAT16 disk according to hard disk size. For FAT32 file systems, cluster sizes are determined by the OS. Clusters can range from 1 sector consisting of 512 bytes to 128 sectors of 64 KB.



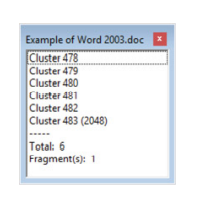
Microsoft OSs allocate disk space for files by clusters. This practice results in drive slack , composed of the unused space in a cluster between the end of an active file’s content and the end of the cluster. Drive slack includes RAM slack (found mainly in older Microsoft OSs) and file slack .

For example, suppose you create a text document containing 5000 characters—that is, 5000 bytes of data. If you save this file on a FAT16 1.6 GB disk, a Microsoft OS reserves one cluster for it automatically. For a 1.6 GB disk, the OS allocates about 32,000 bytes, or 64 sectors (512 bytes per sector), for your file. The unused space, 27,000 bytes, is the file slack (see Figure 5- 8). That is, RAM slack is the portion of the last sector used in the last assigned cluster, and the remaining sectors are referred to as “file slack.” The 5000-byte text document uses up 10 sectors, or 5120 bytes, so 120 bytes of a sector aren’t used; however, DOS must write in full 512-byte chunks of data (sectors). The data to fill the 120-byte void is pulled from RAM and placed in the area between the end of the file (EOF) and the end of the last sector used by the active file in the cluster. Any information in RAM at that point, such as logon IDs or passwords, is placed in RAM slack on older Microsoft OSs when you save a file. File fragments, deleted emails, and passwords are often found in RAM and file slack.



An unintentional side effect of FAT16 allowing large clusters was that it reduced fragmentation as cluster size increased. The OS added extra data to the end of the file and allowed the file to expand to this assigned cluster until it consumed the remaining reserved 27,000 bytes. This increased cluster size resulted in inefficient use of disk space. Because of this inefficient allocation of sectors to clusters, when nearly full FAT16 drives were converted to FAT32, users discovered they had a lot of extra free disk space because the files wasted less space.

When you run out of room for an allocated cluster, the OS allocates another cluster for your file. As files grow and require more disk space, assigned clusters are chained together. Typically, chained clusters are contiguous on the disk. However, as some files are created and deleted and other files are expanded, the chain can be broken or fragmented. With a tool such as WinHex, you can view the cluster-chaining sequence and see how FAT addresses linking clusters to one another



When the OS stores data in a FAT file system, it assigns a starting cluster position to a file. Data for the file is written to the first sector of the first assigned cluster. When this first assigned cluster is filled and runs out of room, FAT assigns the next available cluster to the file. If the next available cluster isn’t contiguous to the current cluster, the file becomes fragmented.

In the FAT for each cluster on the volume (the partitioned disk), the OS writes the address of the next assigned cluster. Think of clusters as buckets that can hold a specific number of bytes. When a cluster (or bucket) fills up, the OS allocates another cluster to collect the extra data.

On rare occasions, such as a system failure or sabotage, these cluster chains can break. If they do, data can be lost because it’s no longer associated with the previous chained cluster. FAT looks forward for the next cluster assignment but doesn’t provide pointers to the previous cluster. Rebuilding these broken chains can be difficult.

**Deleting FAT Files**

When a file is deleted in Windows Explorer or with the MS-DOS delete command, the OS inserts a HEX E5 (0xE5) in the filename’s first letter position in the associated directory entry. This value tells the OS that the file is no longer available and a new file can be written to the same cluster location.

In the FAT file system, when a file is deleted, the only modifications made are that the directory entry is marked as a deleted file, with the HEX E5 character replacing the first letter of the filename, and the FAT chain for that file is set to 0. The data in the file remains on the disk drive. The area of the disk where the deleted file resides becomes unallocated disk space (also called “free disk space”). The unallocated disk space is now available to receive new data from newly created files or other files needing more space as they grow. Most forensics tools can recover data still residing in this area.

**Examining NTFS Disks**

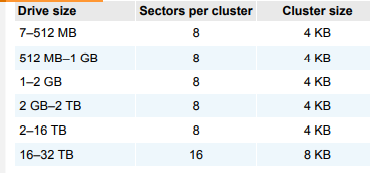
NT File System (NTFS) was introduced when Microsoft created Windows NT and is still the main file system in Windows 10. Each generation of Windows since NT has included minor changes in NTFS configuration and features. The NTFS design was partially based on, and incorporated many features from, Microsoft’s project for IBM with the OS/2 operating system; in this OS, the file system was High Performance File System (HPFS). When Microsoft created Windows NT, it provided backward-compatibility so that NT could read OS/2 HPFS disk drives. Since the release of Windows 2000, this backward-compatibility is no longer available.

NTFS offers substantial improvements over FAT file systems. It provides more information about a file, including security features, file ownership, and other file attributes. With NTFS, you also have more control over files and folders (directories) than with FAT file systems.

NTFS was Microsoft’s move toward a **journaling file system**. The system keeps track of transactions such as file deleting or saving. This journaling feature is helpful because it records a transaction before the system carries it out. That way, in a power failure or other interruption, the system can complete the transaction or go back to the last good setting.

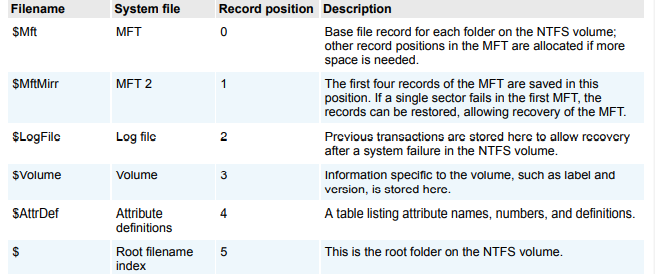
In NTFS, everything written to the disk is considered a file. On an NTFS disk, the first data set is the Partition Boot Sector , which starts at sector [o] of the disk and can expand to 16 sectors. Immediately after the Partition Boot Sector is the Master File Table (MFT) . The MFT, similar to FAT in earlier Microsoft OSs, is the first file on the disk. An MFT file is created at the same time a disk partition is formatted as an NTFS volume and usually consumes about 12.5% of the disk when it’s created. As data is added, the MFT can expand to take up 50% of the disk.

An important advantage of NTFS over FAT is that it results in much less file slack space. Compare the cluster sizes, which showed FAT cluster sizes. Clusters are smaller for smaller disk drives. This feature saves more space on all disks using NTFS.



**NTFS System Files**

Because everything on an NTFS disk is a file, the first file, the MFT, contains information about all files on the disk, including the system files the OS uses. In the MFT, the first 15 records are reserved for system files. Records in the MFT are called metadata. Table lists some metadata records you find in the MFT.



**MFT and File Attributes**

When Microsoft introduced NTFS, the way the OS stores data on disks changed substantially. In the NTFS MFT, all files and folders are stored in separate records of 1024 bytes each. Each record contains file or folder information. This information is divided into record fields containing metadata about the file or folder and the file’s data or links to the file’s data. A record field is referred to as **an attribute ID**

File or folder information is typically stored in one of two ways in an MFT record: **resident and nonresident**. For very small files, about 512 bytes or less, all file metadata and data are stored in the MFT record. These types of records are called resident files because all their information is stored in the MFT record.

Files larger than 512 bytes are stored outside the MFT. The file or folder’s MFT record provides cluster addresses where the file is stored on the drive’s partition. These cluster addresses are called **data runs** . This type of MFT record is referred to as **“nonresident**” because the file’s data is stored in its own separate file outside the MFT.

Each MFT record starts with a header identifying it as a resident or nonresident attribute. The first 4 bytes (characters) for all MFT records are FILE. The header information contains additional data specifying where the first attribute ID starts, which is typically at offset 0x14 from the beginning of the record. Each attribute ID has a length value in hexadecimal defining where it ends and where the next attribute starts. The length value is located 4 bytes from the attribute ID.

To understand how data runs are assigned for nonresident MFT records, you should know that when a disk is created as an NTFS file structure, the OS assigns logical clusters to the entire disk partition. These assigned clusters, called logical cluster numbers (LCNs) , are sequentially numbered from the beginning of the disk partition, starting with the value 0. LCNs become the addresses that allow the MFT to link to nonresident files (files outside the MFT) on the disk’s partition.

When data is first written to nonresident files, an LCN address is assigned to the file in the attribute 0x80 field of the MFT. This LCN becomes the file’s virtual cluster number (VCN) , which is listed as zero: VCN(0). If there’s not enough space at VCN(0)’s location because of excessive disk fragmentation, another data run is added. More VCNs are added as needed, and each additional VCN is sequentially numbered as VCN(1), VCN(2), and so on until all data is written to the drive.

The value in VCN(0) is the first cluster for the file; this value is the cluster’s actual LCN. VCN(1) and other VCNs are the offset of the cluster’s number from the previous VCN cluster position in the data run.

**MFT Structures for File Data**

When you’re viewing an MFT record with a hexadecimal editor, the data is displayed in reverse order ,read from right to left. For example, the hexadecimal value 400 is displayed as 00 04 00 00, and the number 0x40000 is displayed as 00 00 04 00.

The first section of an MFT record is the header that defines the size and starting position of the first attribute. Following the header are attributes that are specific for the file type, such as an application file or a data file. MFT records for directories and system files have additional attributes that don’t appear in a file MFT record.

**MFT Header Fields**

For the header of all MFT records, the record fields of interest are as follows:

* At offset 0x00—The MFT record identifier FILE; the letter F is at offset 0.
* At offset 0x1C to 0x1F—Size of the MFT record; the default is 0x400 (1024) bytes, or two sectors.
* At offset 0x14—Length of the header, which indicates where the next attribute starts; it’s typically 0x38 bytes.
* At offset 0x32 and 0x33—The update sequence array, which stores the last 2 bytes of the first sector of the MFT record. It’s used only when MFT data exceeds 512 bytes. The update sequence array is used as a checksum for record integrity validation.

**NTFS Compressed Files**

To improve data storage on disk drives, NTFS provides compression similar to FAT DriveSpace 3, a Windows 98 compression utility. With NTFS, you can compress files, folders, or entire volumes. With FAT16, you can compress only a volume. On a Windows NT or later system, compressed data is displayed normally when you view it in Windows Explorer or applications such as Microsoft Word.

During an investigation, typically you work from an image of a compressed disk, folder, or file. Most forensics tools can uncompress and analyze compressed Windows data, including data compressed with the Lempel-Ziv-Huffman (LZH) algorithm and in formats such as PKZip, WinZip, and GNU gzip. However, forensics tools might have difficulty with third-party compression utilities, such as the .rar format. If you identify third-party compressed data, you need to uncompress it with the utility that created it.

**NTFS Encrypting File System**

When Microsoft introduced Windows 2000, it added optional built-in encryption to NTFS called Encrypting File System (EFS). EFS uses public key and private key methods of encrypting files, folders, or disk volumes (partitions). Only the owner or user who encrypted the data can access encrypted files. The owner holds the private key, and the public key is held 1. by a certification authority, such as a global registry, network server, or company such as VeriSign.

When EFS is used in Windows 2000 and later, a recovery certificate is generated and sent to the local Windows administrator account. The purpose of the recovery certificate is to provide a mechanism for recovering files encrypted with EFS if there’s a problem with the user’s original private key. The recovery key is stored in one of two places. When a network user initiates EFS, the recovery key is sent to the local domain server’s administrator account. On a stand-alone workstation, the recovery key is sent to the local administrator account.

Users can apply EFS to files stored on their local workstations or a remote server. Windows 2000 and later decrypt the data automatically when a user or an application accesses an EFS file, folder, or disk volume. In Windows Server 2003 and later, users can grant other users access to their EFS data. If a user copies a file encrypted with EFS to a folder that isn’t encrypted, the copied data is saved in unencrypted format.

**Deleting NTFS Files**

Typically, you use File Explorer to delete files from a disk. When a file is deleted in Windows NT and later, the OS renames it and moves it to the Recycle Bin. Another method is using the del (delete) MS-DOS command. This method doesn’t rename and move the file to the Recycle Bin, but it eliminates the file from the MFT listing in the same way FAT does.

When you delete a file in Windows or File Explorer, you can restore it from the Recycle Bin. The OS takes the following steps when you delete a file or a folder in Windows or File Explorer

* Windows changes the filename and moves the file to a subdirectory with a unique identity in the Recycle Bin.
* Windows stores information about the original path and filename in the Info2 file , which is the control file for the Recycle Bin. It contains ASCII data, Unicode data, and the date and time of deletion for each file or folder.

The following steps also apply when a user empties the Recycle Bin.) The OS performs the following tasks:

* The associated clusters are designated as free—that is, marked as available for new data.
* The $Bitmap file attribute in the MFT is updated to reflect the file’s deletion, showing that this space is available.
* The file’s record in the MFT is marked as being available.
* VCN/LCN cluster locations linked to deleted nonresident files are then removed from the original MFT record.
* A run list is maintained in the MFT of all cluster locations on the disk for nonresident files. When the list of links is deleted, any reference to the links is lost.

**Resilient File System (ReFS)**

With the release of Windows Server 2012, Microsoft created a new file system: Resilient File System (ReFS) . ReFS is designed to address very large data storage needs, such as the cloud. The following features are incorporated into ReFS’s design:

* Maximized data availability
* Improved data integrity
* Designed for scalability

ReFS is an outgrowth of NTFS designed to provide a large-scale data storage access capability. It’s intended only for data storage, so as of this writing, it can’t be used as a boot drive. Windows 8/8.1 and later and Windows Server 2012 and later are the only Windows OSs that can access ReFS disk drives.

ReFS uses disk structures similar to the MFT in NTFS. Its storage engine uses a B+-tree sort method for fast access to large data sets. It also uses a method called “allocate-on-write” that copies updates of data files to new locations; similar to shadow paging, it prevents overwriting the original data files. The purpose of writing updates to new locations is to ensure that the original data can be recovered easily if a failure occurs in the update write to disk.

**Understanding Whole Disk Encryption(WDE)**

Loss of personal identity information (PII) and trade secrets caused by computer theft has become more of a concern. Company PII might consist of employees’ full names, home addresses, and Social Security numbers. With this information, criminals could easily apply for credit card accounts in these employees’ names. Trade secrets are any information a business keeps confidential because it provides a competitive edge over other companies. The inadvertent public release of this information could devastate a business’s competitive edge.

Of particular concern is the theft of laptops and handheld devices, such as smartphones. If data on these devices isn’t secured correctly, the owners could be liable for any damages incurred, such as stolen identities, credit card fraud, or loss of business caused by the release of trade secrets to the competition. Because of the PII problem, many states have enacted laws requiring any person or business to notify potential victims of the loss as soon as possible. To help prevent loss of information, software vendors, including Microsoft, now provide whole disk encryption

This feature creates new challenges in examining and recovering data from drives. Whole disk encryption tools offer the following features that forensics examiners should be aware of:

* Preboot authentication, such as a single sign-on password, fingerprint scan, or token (USB device)
* Full or partial disk encryption with secure hibernation, such as activating a passwordprotected screen saver
* Advanced encryption algorithms, such as Advanced Encryption Standard (AES) and International Data Encryption Algorithm (IDEA)
* Key management function that uses a challenge-and-response method to reset passwords or passphrases

WDE tools encrypt each sector of a drive separately. Many of these tools encrypt the drive’s boot sector to prevent any efforts to bypass the secured drive’s partition. To examine an encrypted drive, you must decrypt it first. An encryption tool’s key management function typically uses a challenge-and-response method for decryption, which means you must run a vendor-specific program to decrypt the drive. Many vendors use a bootable CD or USB drive that prompts for a one-time passphrase generated by the key management function. If you need to decrypt the same computer a second time, you need a new one-time passphrase.

The biggest drawback to decrypting a drive is the several hours required to read, decrypt, and write each sector. The larger the drive, the longer decryption takes. After you’ve decrypted the drive, however, you can use standard acquisition methods to retrieve data.

**Microsoft BitLocker**

BitLocker is Microsoft’s utility for protecting drive data. It’s available in Windows Vista Enterprise and Ultimate editions, Windows 7, 8, and 10 Professional and Enterprise editions, and Windows Server 2008 and later.

BitLocker Drive Encryption is a data protection feature that integrates with the operating system and addresses the threats of data theft or exposure from lost, stolen, or inappropriately decommissioned computers.

BitLocker provides the maximum protection when used with a Trusted Platform Module (TPM) version 1.2 or later versions. The TPM is a hardware component installed in many newer computers by the computer manufacturers. It works with BitLocker to help protect user data and to ensure that a computer has not been tampered with while the system was offline.

Data on a lost or stolen computer is vulnerable to unauthorized access, either by running a software-attack tool against it or by transferring the computer's hard disk to a different computer. BitLocker helps mitigate unauthorized data access by enhancing file and system protections. BitLocker also helps render data inaccessible when BitLocker-protected computers are decommissioned or recycled.

BitLocker’s current hardware and software requirements are as follows:

* A computer capable of running Windows Vista or later (non-home editions)
* The Trusted Platform Module (TPM) microchip, version 1.2 or newer
* A computer BIOS compliant with Trusted Computing Group (TCG)
* Two NTFS partitions for the OS and an active system volume with available space
* The BIOS configured so that the hard drive boots first before checking the CD/DVD drive

or other bootable peripherals

Several vendors offer third-party WDE utilities that often have more features than BitLocker. For example, BitLocker can encrypt only NTFS drives. If you want to encrypt a FAT drive, you need a third-party solution.

**Understanding the Windows Registry**

When Microsoft created Windows 95, it consolidated initialization (.ini) files into the Registry , a database that stores hardware and software configuration information, network connections, user preferences (including usernames and passwords), and setup information. The Registry has been updated and is still used in Windows Vista and later.

For investigative purposes, the Registry can contain valuable evidence. To view the Registry, you can use the Regedit (Registry Editor) program for Windows 9x and Regedt32 for Windows 2000, XP, and Vista. For Windows 7 and 8, both Regedit and Regedt32 are available.

In general, you can use the Edit, Find menu command in Registry Editor to locate entries that might contain trace evidence, such as information identifying the last person who logged on to the computer, which is usually stored in user account information. You can also use the Registry to determine the most recently accessed files and peripheral devices. In addition, all installed programs store information in the Registry, such as Web sites accessed, recent files, and even chat rooms accessed

As a digital forensics investigator, you should explore the Registry of all Windows systems. On a live system, be careful not to alter any Registry setting to avoid corrupting the system and possibly making it unbootable.

**Exploring the Organization of the Windows Registry**

The Windows Registry is organized in a specific way that has changed slightly with each new version of Windows. However, the major Registry sections have been consistent, with some minor changes, since Windows 2000; they’re slightly different in Windows 9x/Me. Before proceeding, review the following list of Registry terminology:

* Registry—A hierarchical database containing system and user information.
* Registry Editor—A Windows utility for viewing and modifying data in the Registry. There

are two Registry Editors: Regedit and Regedt32 (introduced in Windows 2000).

* HKEY—Windows splits the Registry into categories with the prefix HKEY\_. Windows 9x

systems have six HKEY categories and Windows 2000 and later have five. Windows

programmers refer to the “H” as the handle for the key.

* Key—Each HKEY contains folders referred to as keys. Keys can contain other key folders

or values.

* Subkey—A key displayed under another key is a subkey, similar to a subfolder in Windows

or File Explorer.

* Branch—A key and its contents, including subkeys, make up a branch in the Registry.
* Value—A name and value in a key; it’s similar to a file and its data content.
* Default value—All keys have a default value that may or may not contain data.
* Hives—Hives are specific branches in HKEY\_USER and HKEY\_LOCAL\_MACHINE. Hive

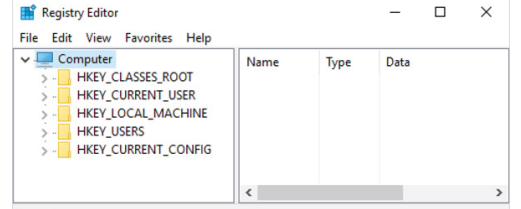
branches in HKEY\_LOCAL\_MACHINE\Software are SAM, Security, Components, and

System. For HKEY\_USER, each user account has its own hive link to Ntuser.dat.

There are two additional tools in the Remote Server Administration Tools which you can use to manage BitLocker.

**BitLocker Recovery Password Viewer**. The BitLocker Recovery Password Viewer enables you to locate and view BitLocker Drive Encryption recovery passwords that have been backed up to Active Directory Domain Services (AD DS). You can use this tool to help recover data that is stored on a drive that has been encrypted by using BitLocker. To view recovery passwords, you must be a domain administrator, or you must have been delegated permissions by a domain administrator.

**BitLocker Drive Encryption Tools**. BitLocker Drive Encryption Tools include the command-line tools, manage-bde and repair-bde, and the BitLocker cmdlets for Windows PowerShell. Both manage-bde and the BitLocker cmdlets can be used to perform any task that can be accomplished through the BitLocker control panel, and they are appropriate to be used for automated deployments and other scripting scenarios.



**Examining the Windows Registry**

Some forensics tools, such as X-Ways Forensics, OSForensics, Forensic Explorer, and FTK, have built-in or add-on Registry viewers. For this next activity, your company’s Legal Department has asked you to search for any references to any e-mail addresses containing the name Denise or Robinson with the domain name outlook.com. A paralegal gives you a raw (dd) image file containing InCh05.img, a forensic image of a Windows 8 computer’s hard drive used by Superior Bicycle employee Denise Robinson.

For this activity, you use OSForensics to examine Denise Robinson’s NTUser.dat file. If you find any items of interest, add them to an OSForensics case report that you can give to the paralegal. The following steps explain how to generate a case report in OSForensics.

**Understanding Microsoft Startup Tasks**

You should have a good understanding of what happens to disk data at startup. In some investigations, you must preserve data on the disk exactly as the suspect last used it. Any access to a computer system after it was used for illicit purposes alters your disk evidence.

Altering disk data lessens its evidentiary quality considerably. In some instances, accessing a suspect computer incorrectly could make the digital evidence corrupt and less credible for litigation. In the following sections, you learn what files are accessed when Windows starts. This information helps you determine when a suspect’s computer was last accessed, which is particularly important with computers that might have been used after an incident was reported.

Any computer using NTFS performs the following steps when the computer is turned on:

* Power-on self test (POST)
* Initial startup
* Boot loader
* Hardware detection and configuration
* Kernel loading
* User logon

**Startup Files for Windows XP**

Unless otherwise specified, most startup files for Windows XP are in the **root folder** of the system partition.

* NT Loader (Ntldr) loads the OS. When the system is powered on,
* Ntldr reads the Boot.ini file, which displays a boot menu.
* After you select the mode to boot to, Boot.ini runs Ntoskrnl.exe and reads Bootvid.dll, Hal.dll, and startup device drivers. Boot.ini specifies the Windows XP path installation and contains options for selecting the Windows version.

If a system has multiple boot OSs, including older ones such as Windows 9x or DOS, Ntldr reads BootSect.dos (a hidden file), which contains the address (boot sector location) of each OS.

* When the boot selection is made, Ntldr runs NTDetect.com , a 16-bit real-mode program that queries the system for device and configuration data, and then passes its findings to Ntldr.

This program identifies components and values on the computer system, such as the following:

* CMOS time and date value
* Buses attached to the motherboard, such as Industry Standard Architecture (ISA) or
* Peripheral Component Interconnect (PCI)
* Disk drives connected to the system
* Mouse input devices connected to the system
* Parallel ports connected to the system

**NTBootdd.sys** is the device driver that allows the OS to communicate with SCSI or ATA drives that aren’t related to the BIOS. (On some workstations, a SCSI disk is used as the primary boot disk.) Controllers that don’t use Interrupt 13 (INT-13) use NTBootdd.sys. It runs in privileged processor mode with direct access to hardware and system data.

**Ntoskrnl.exe** is the Windows XP OS kernel, located in the systemroot\Windows\System32 folder.

**Hal.dll** is the Hardware Abstraction Layer (HAL) dynamic link library, located in the systemroot\Windows\System32 folder. The HAL allows the OS kernel to communicate with the computer’s hardware.

At startup, data and instruction code are moved in and out of the Pagefile.sys file to optimize the amount of physical RAM available

**Device drivers** contain instructions for the OS for hardware devices, such as the keyboard, mouse, and video card, and are stored in the systemroot\Windows\System32\Drivers folder.