

Course Book



CYBER SYSTEMS AND NETWORK FORENSICS

DLMCSECSNF01_E

iu

INTERNATIONAL
UNIVERSITY OF
APPLIED SCIENCES

LEARNING OBJECTIVES

Digital evidence can be found in almost every incident that network forensics investigates. The nature of digital evidence means that the investigator must be careful when collecting or analyzing it due to its fragility. A simple zap of static electricity can be enough to alter digital evidence. To use it in an investigation, we must understand how evidence is created and how to associate it with a specific user.

Digital evidence can come from several sources, including networks, mobile devices, and traditional computing platforms (such as desktops or laptop computers). The way that each of these sources creates artifacts can differ greatly. When examining network-based evidence, the investigator must be aware of how easy it is to get overwhelmed by the amount of digital evidence to be examined. An examiner should filter non-pertinent data and focus their efforts on the data set relevant to their investigation.

The course **Cyber Systems and Network Forensics** will give you in-depth experience collecting and examining digital evidence. When analyzing laptop and desktop computers, there must be an understanding of the different operating systems and how they interact with the different file systems and storage devices. An explanation of network protocols will help you understand the tactics used by an attacker. Each unit will address the practical application and provide real-world examples. This will provide an insight into how digital evidence is acquired, analyzed, and utilized in judicial and administrative proceedings.

LEKTION 1

OPERATING SYSTEMS

STUDY GOALS

On completion of this unit, you will be able to...

- explain why operating systems (OSs) are necessary.
- list the functions of an OS.
- explain the purpose of disk management.
- describe the layered structure of input/output (I/O) software.
- use multi-programming systems and multi-tasking systems.

1. OPERATING SYSTEMS

Introduction

In the **twenty-first** century, computer systems affect almost every facet of our everyday lives. Watching television, checking emails, and even starting our cars are done using a computer system. **The use of mobile devices has increased to the point that it is sometimes necessary for a person to have more than one.**

Operating systems (OSs) are used to run these computer systems. The most common operating system in the home computing market is Microsoft Windows (*Desktop operating system market share worldwide*, 2021). As technology changes, so does the operating system. The current Windows version (Windows 10) is far more advanced; computer systems themselves have significantly advanced since the start of the new century.

The operating system acts as a conduit between the system's hardware and the user. It provides a fluid user experience by removing the need for the user to configure all of the attached hardware components. The operating system manages all connected devices in the background to provide an efficient experience.

1.1 Concepts

When we think of the word “computer,” what comes to mind? We might think of a monitor attached to a box with a keyboard and mouse. The user controls the computer using the keyboard and mouse, and their actions are displayed on the monitor. Most users are familiar with their applications, such as word processing documents, web browsers, and email clients. The average user does not know exactly how the “box” takes the input and makes things happen. **So, to begin, let's look inside this box—the computer.** Inside, we find a motherboard. The motherboard holds the central processing unit (CPU), the **random-access memory (RAM)**, the video card, the soundcard, and various other chips and connections. This is all known as hardware. The user applications mentioned earlier (web browsers, email clients, etc.) are all considered software. There is an intermediate layer between the hardware and the software: the operating system. The operating system starts working as soon as the system receives power and manages the hardware resources. Users may have experience with a Windows, macOS, or Linux machine. However, the average user may not know much about the differences between operating systems. The three different operating systems are similar enough **that an average user can figure out how to do the basics.** One reason for this is that users interact with the operating system through a **graphical user interface (GUI)**.

Random access memory (RAM)

This is a type of computer memory that allows data to be read and modified in any order.

Graphical user interface

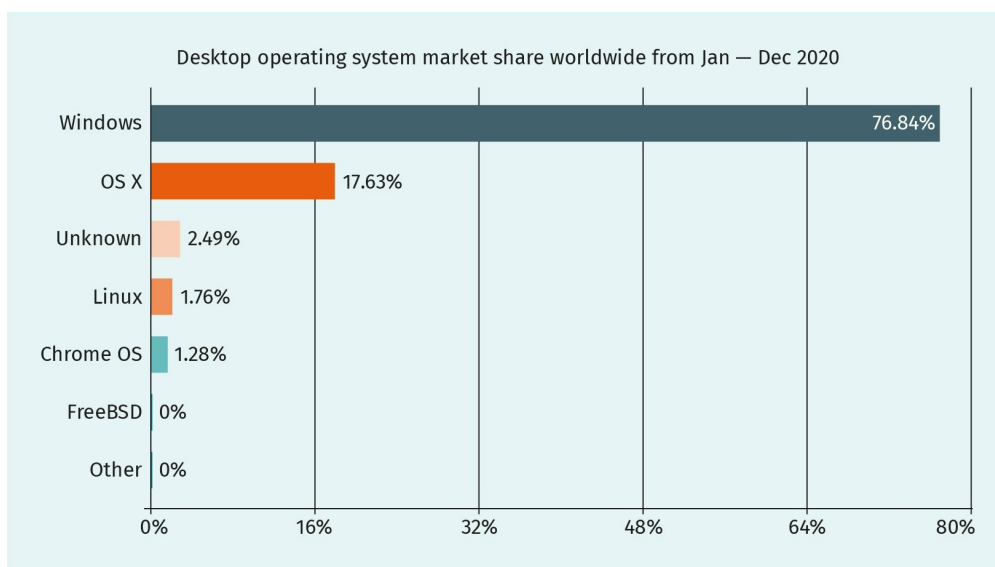
This is a visual way of interacting with a computer.

There is another method used to interact with the **operating system**: the command line interface (CLI). With the CLI, the user types in commands and modifiers using the keyboard. An example of this interface is the old **MS-DOS** operating system or most Linux sys-

tems. The user can either use the GUI or the CLI. Some novice users may feel intimidated by the CLI because it requires a greater knowledge of the operating system and is less intuitive than the GUI.

Microsoft Windows was the most commonly used OS in 2021, with macOS in second place, followed by Linux and other variants (*Desktop operating system market share worldwide, 2021*). Despite the differences between these operating systems, they ultimately all do the same thing: control the hardware connected to the motherboard. Even your mobile device has an operating system, for example, Android or iOS.

Figure 1: 2020 Desktop Operating Systems

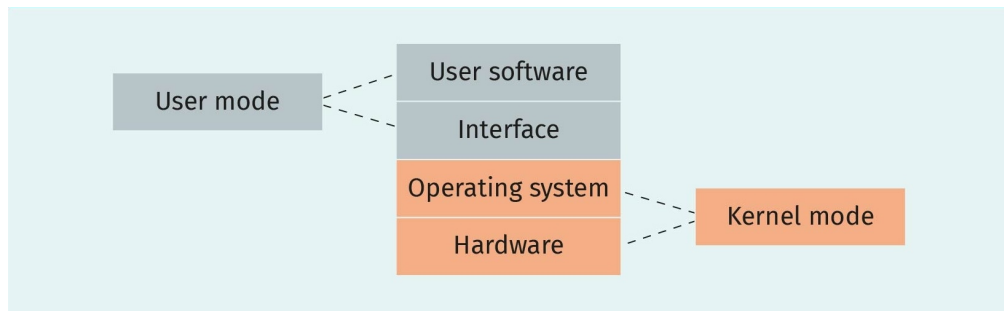


Source: *Desktop Operating System Market Share Worldwide* (2021).

The following figure shows a graphical representation of the computer structure we have discussed. At the very bottom is the hardware layer. The average user will not have access to this level. Above the hardware layer is the operating system, through which the user can access some components of the hardware layer depending on their permissions. Above the operating system is the interface; this is typically how the user will interact with the operating system. “**Super users**” have increased security permissions with which they can perform tasks that the average user cannot. At the very top level is user software or applications. On either side of the stack, we have the user mode and the kernel mode.

Super user
This is a user of a computer system with special privileges.

Figure 2: Operational Modes

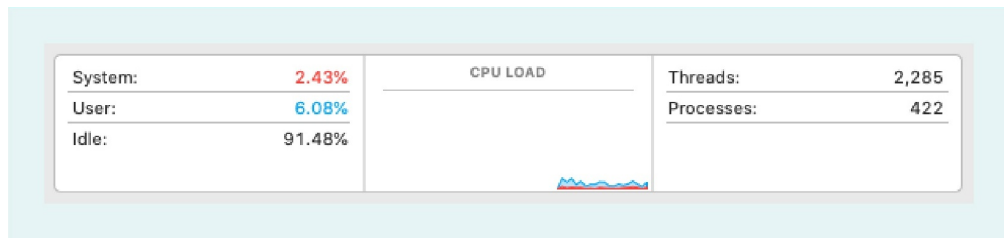


Source: Created on behalf of IU (2022).

Kernel mode
This is the most trusted security mode and provides access to all devices.

There are two security modes available when applications are executed: “**kernel mode**” and “user mode.” In kernel mode, the application has complete access to the system hardware. This mode allows the application to execute CPU instructions and to access all memory spaces. This mode is the highest level of trust for the applications and the operating system. A system crash in this mode can be catastrophic. In user mode, the application has limited access to the hardware and must make requests through a system application programming interface (API). If the application crashes, it will typically not impact the system. In the figure below, taken from a macOS system, the activity monitor has separated the processes into “system” and “user” processes. The activity monitor highlights the system processes in red, while the user processes are set in blue.

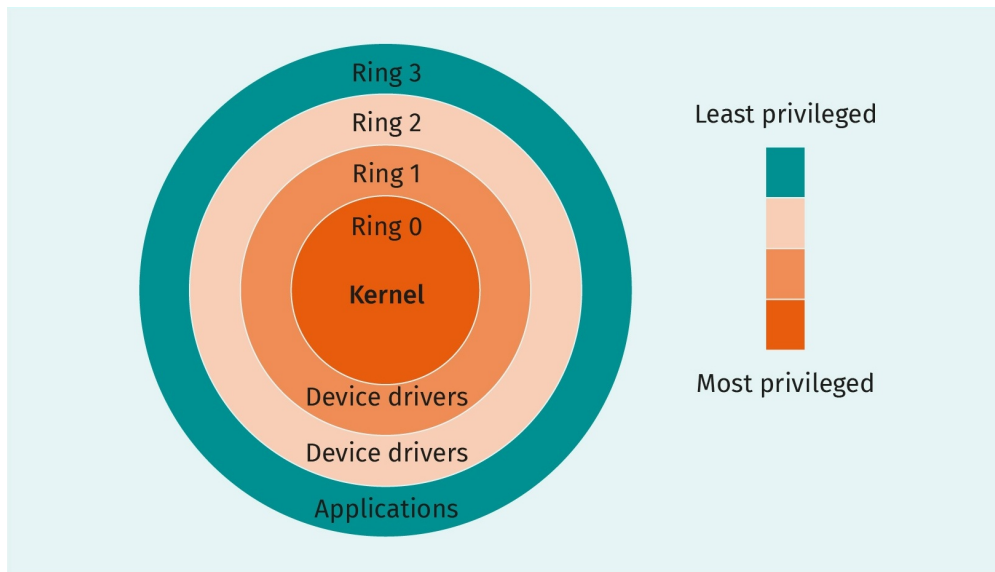
Figure 3: macOS Process



Source: Created on behalf of IU (2022).

This can also be represented by a ring model, as seen in the following figure. Ring zero, at the very center, is the most secure. This is where the trusted applications are executed to grant them full access to CPU and memory addresses. Ring three is the least secure of the four rings. This is typically where user applications that do not directly access the CPU and memory addresses are executed. Rings one and two are not commonly used.

Figure 4: Privilege Rings



Source: Hertzprung (2007). CC BY-SA 3.0.

Beyond the execution of applications, operating systems have many unique features and capabilities, which we will now look at in detail.

Types of Operating System

There are various types of operating system. A single-tasking operating system, for example, allows the execution of one program at a time. A multi-tasking operating system allows more than one task to be performed. The operating system accomplishes this through time sharing, where the operating system monitors the CPU availability and the processes. The operating system will use either preemptive or cooperative multitasking. Preemptive multitasking creates time slices and assigns each process to a specific slice. Cooperative multitasking shares slices among many processes. Other categories of operating system include single-user and multi-user, distributed, embedded, and real-time.

Single-user and multi-user

Single-user operating systems cannot be used by more than one user at a time. A multi-user operating system allows multiple users to log into the system at once. The operating system then allocates resources to each user, such as CPU power and disk space, through time-sharing. The operating system has accounting tools to track, for example, the allocation of computing time, storage, and printing.

Distributed

A distributed operating system manages a group of distinct hosts and makes them appear as a single computer host. The distributed model emerged as networked computer systems became commonplace. With a networked system, the user needed to know the different hosts on the network to carry out tasks. With a distributed operating system, this is not necessary.

Embedded

Embedded operating systems are installed on non-computer devices and do not allow user-installed software. These devices can be complex, having specific functions. Embedded devices can be found everywhere, including in smartwatches, refrigerators, and televisions. Embedded operating systems execute the system's core functions (such as configuration, job management, and memory management) with little user interaction. The fundamental property of embedded systems is that no untrusted software will ever run on them.

Real-time

A real-time operating system will process data to complete tasks within a specific time frame. For example, in an industrial setting, the operating system collects information during production. As the product moves along the assembly line, specific actions must occur at specific times. A rigid real-time system is used to guarantee that a particular action occurs on time. A soft real-time device views missing a deadline as undesirable, but permissible and without long-term consequences. Users cannot install applications on real-time devices.

1.2 Memory Management

Random-access memory (RAM) is a critical resource managed by the operating system. RAM storage is not the same as hard drive storage. Hard drive storage is non-volatile and designed to store information during the lifespan of the device. RAM is volatile storage and requires the system to be powered. Once power is interrupted, the data stored in the chips are no longer available.

The operating system's memory manager tracks which parts of the memory are in use, allocating and de-allocating memory space as needed. Early computer systems (i.e., before the 1980s) had physical memory space (*Timeline of computer history: Memory and storage*, 2021), where each process used the memory space as needed. The downside to this was that only one process could use the memory space at a time. Therefore, the swap file was created. The swap file is space on a non-volatile storage device, such as a hard drive, where the operating system swaps out data between the volatile memory space (RAM) and the non-volatile memory space (hard drive). However, allowing free access to

the physical memory space has some potential security issues. If a process can address every byte of the physical memory space and something goes wrong, this could crash the process (and potentially the entire system).

To achieve better security and greater efficiency, the abstract layer was created. This provided an address space that processes could access. With the abstract layer, each process has its own address space separate from other processes. Examples of address spaces include IPv4 addresses (a 32-bit numeric value) and IPv6 addresses (made up of alphabetical and numeric characters). To address the security concerns, a base register and a limit register were created. These acted as a start address (base register) and an end address (limit register). When processes are executed, the base register records the physical address of the process in RAM. The limit register includes the length of the dataset stored in RAM. Using these registers allows each process to have its own address space and protects it from contamination by other processes.

The abstract layer application allows the use of the unique location identifiers to identify the contents of RAM when conducting forensic analysis. Data stored within the RAM chips are held in “pages,” which are generally four kilobytes in size. When performing forensic analysis of RAM, some of the artifacts found include

- CLI commands,
- encryption keys,
- unencrypted data,
- IP addresses,
- internet history, and
- malware.

Artifacts from any user activity or the operating system may be found within RAM.

Virtual Memory

Often, the sophisticated nature of today’s computer systems calls for more RAM than may be installed. Remember that RAM is **volatile memory**: if the system is shut down, RAM data are not accessible. Other sources that provide access to the information stored in RAM may be available. However, whether these options are available depends on variables such as the type of system and user control settings. Operating systems can now provide virtual address space within RAM through the abstraction layer. The size of the virtual address space depends on how the hardware and operating system are configured. The memory manager assigns the necessary address space to processes and then separates the data into “pages.” A page has a continuous range of addresses. To enable virtual memory, a memory management policy enables “demand paging.” The memory manager identifies the pages containing data that need to be accessed and then moves only what the process requires from the virtual memory to the physical memory. The memory manager can use the following scheduling processes to move data between physical memory and virtual memory:

Volatile memory
This form of memory requires an electric current to retain data.

- round robin process scheduling. The processes are executed based on a temporal criterion; if this criterion expires and the process is not completed, the system will swap it out.
- priority-driven scheduling. A higher-priority process will displace a lower priority process.
- blocked process. Processes waiting for input/output (I/O) can be swapped out.

The following are potential virtual memory data sources similar to those found in RAM.

Hibernation file: hiberfill.sys

Hibernation mode occurs when the system is powered down. In Windows, the RAM is compressed and stored in the hiberfill.sys file. This allows the system to go to sleep. When the system is reactivated, the contents of hiberfill.sys are placed back into RAM. The file header for the file hiberfill.sys can be hibr, HIBR, wake, or WAKE. When power is restored to the system, the header of the hiberfill.sys file is zeroed out. Another option is putting the system into hibernation when a live capture of the RAM is not possible. This causes the operating system to create the hiberfill.sys. When analyzing the hiberfill.sys file, the last modification date (timestamp) is displayed.

Pagefile: pagefile.sys

Paging is a method of storing and retrieving data in RAM using a virtual memory file on a traditional storage device. This technique is not as fast as using RAM alone, but it allows the system to exceed the physical memory capacity. When using paging, the system transfers data in pages. The data stored in the page file are typically the least requested data. When the operating system needs to access the data, they are then placed back into the physical memory. In the Windows operating system, the paging file (pagefile.sys) is generally found at the root of the operating system's volume. Typically, the page file will be one to three times larger than the system's physical memory.

Swap file: swapfile.sys

With the release of Windows 8, Microsoft introduced the swapfile.sys file. This is similar to the page file but has some differences. Microsoft created the swap file for paging operations with suspended Windows applications. When the application is suspended, the system will write the application data in their entirety into the swap file. Space is then created in RAM. When the application is resumed, the operating system places the data in the swap file back into RAM.

With modern operating systems, two separate processes can share memory space. This approach allows for more efficient communication between different processes. This technique also allows the operating system to conserve physical memory space by allocating multiple memory pages to the same data set. For example, the operating system can use this method when various processes need to use the same dynamic libraries. The operating system maps the shared pages as "copy on write." This term means that the memory manager will not make a private copy of the data until the process has modified them. The memory manager then updates the virtual memory for that process. The other

processes continue to access the original shared memory page. Investigators frequently examine shared and virtual memory when investigating a potential system breach or malware activity.

Stacks and Heaps

The operating system segments the memory space into different regions. Two of these regions are known as stack and heap. The data stored within the stack space are temporary data needed to execute processes. Each thread maintains the data in their specific stack. Data are added and removed in line with a “last in” and “first out” methodology. This process differs from how data are stored within the heap space. Once the process has been completed, the system will delete the data in the stack.

The data in the heap space are available to multiple processes, and the data are dynamically available to multiple threads. Instead of being deleted when they are no longer needed, the data remain within the heap space until they are manually deleted. This differs from the data stored within the stack, which are only available for the thread’s specific function. When comparing stack space to heap space, access to the data is faster when using stack space. The stack is faster because the pointer is increased or decreased to allocate or deallocate memory space; the stack stores values in a last-in, first-out sequence. Values for the stack are added in the order they are received and then removed in reverse order. Adding data to stack memory is faster than adding them from heap memory. The system does not need to search for a place to store new data in the stack, as they always goes to the top. Heap serves as a pool of storage space that allocates memory randomly. The system must find space to hold the data, with the additional processing needed to prepare for the allocation. Sometimes, a “stack overflow” occurs: this is an error that happens when the stack runs out of memory in which to store the data.

1.3 Process Management

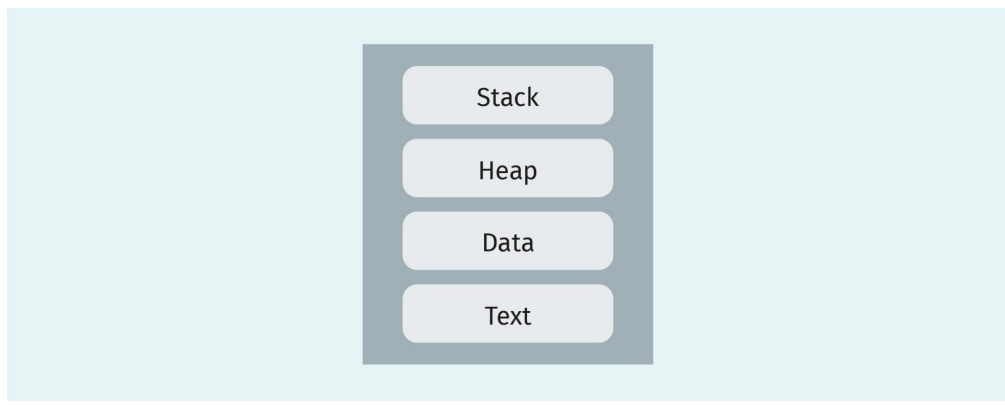
At the very base level of the operating system is the “process.” A process is defined as an executed program within the memory framework. The operating system handles the administration of when processes are executed and monitors the state of the processes if they are suspended or terminated. The operating system can interrupt a process as needed, depending on the process’s priority. When a process is suspended, the operating system will maintain its “state” until it is resumed. The operating system acts as a “traffic control device,” similar to traffic signals encountered on roads. The traffic signals are coordinated to ensure traffic flows efficiently; if it does not, collisions (crashes) occur due to a lack of controlling.

In today’s computing environment, operating systems are capable of “multiprocessing,” which means that they can allow multiple processes to be executed at the same time. To maintain control of the processes, a process identifier and location within the memory is assigned to each process. This location in the memory is referred to as the “process address space” and contains any shared libraries, the code of the application, dynamic

data, and any other associated information. The process address space contains many artifacts relevant to an investigation, such as encryption keys, user passwords, uniform resource locators (URLs), logs, and various data that were used by the operating system.

The following figure shows the process architecture as the system loads the process into memory. At the very top is the “stack” segment, which are the temporary data from the process. Next in line is the “heap” segment from the process. These are the dynamic data and can be accessed by multiple processes. Next is the “data” segment, which contains global and local static variables, such as the variables created by program execution. Finally, the “text” segment contains executable instructions, constants, and macros of the process. This location is read-only and sharable. The data stored here will tell the history of the program’s actions.

Figure 5: Process Architecture



Source: Created on behalf of IU (2022).

As processes are started, the system assigns them a “virtual CPU” and the physical processing unit rapidly switches between virtual processors as needed. The ability of the CPU to switch between virtual processors is called multi-programming. This differs from multi-processing, where the system uses multiple physical CPUs. Multi-processing is faster and can be used for parallel processing.

Process Management Model

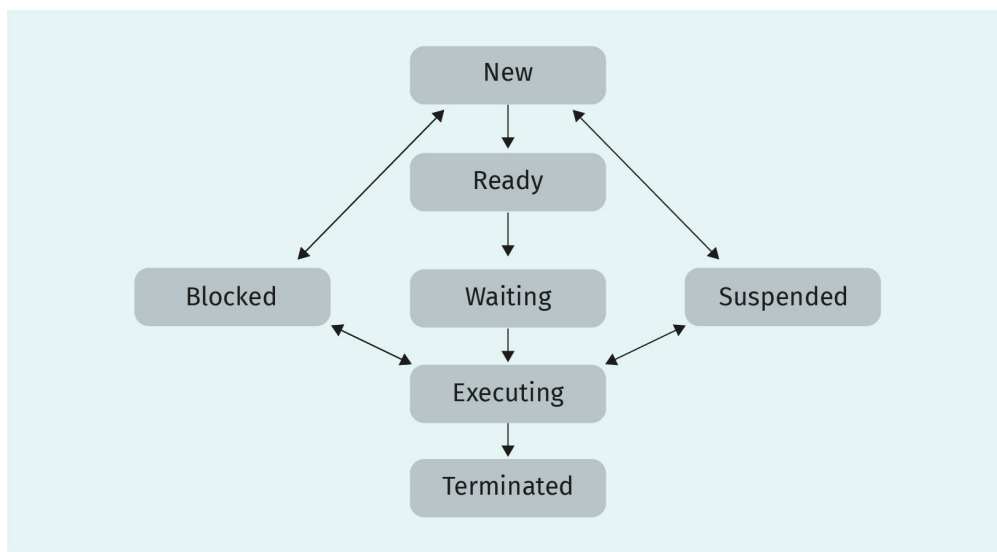
You may encounter two different process management models: two-state process and three-state process management models. In the two-state process management model, the process is considered in the state of “running” or “not running.” When the process is started, it is initially started in the “not running” state and is placed into the queue to wait to be executed. Once the process has been executed, the operating system can interrupt the process and place it back into the “not running” state to allow different processes to be executed.

The three-state process management model has the following states: “running,” “ready,” and “blocked.” The two-state model is less efficient than the three-state model because the “blocked” state is not an option. In the two-state process, the CPU becomes idle while process switches between CPU and input/output (I/O) cycles.

Process Life Cycle

When a process is created, it goes through different states throughout its life cycle. The following are the general states that a process will go through (each operating system may differ in the specific process state and naming convention).

Figure 6: Process States



Source: Created on behalf of IU (2022).

As shown in the previous figure and following table, we encounter various process states.

Table 1: Process State Descriptions

New	This is a newly created process.
Ready	The process is now ready for execution.
Waiting	The process is waiting for resources to become available.
Executing	The process is being executed (running).
Terminated	The process has completed its task and been terminated. All resources are now available for use.
Blocked	The process is waiting for an external event to complete (e.g., an I/O operation).

Suspended

The process is waiting to be placed into the “ready” state.

Source: Created on behalf of IU (2022).

CPU Scheduling

The operating system needs to control access to the CPU and to the many threads being executed. This is known as CPU scheduling. CPU scheduling efficiently switches the threads between CPU utilization and I/O operations. The scheduler determines which threads will be executed and for how long. The switching of threads for CPU utilization is also known as a context switch. The operating system suspends the thread execution and stores it within the RAM. When the operating system activates a suspended thread, it updates the CPU registers with the thread’s state, allowing the thread to resume execution. When examining the RAM, the saved execution information can provide insight into what was being executed.

Deadlocks

Process management involves sharing resources, memory space, and the use of the CPU. Most of the time, this is done without complication, **but it can always happen that** the sharing of resources is not done successfully. This can cause a deadlock or starvation situation. The most recognizable example of a deadlock is a “blue screen of death” (BSOD) on the Windows operating system. This occurs when a combination of circumstances occurs:

- Process A has access to memory space Z in RAM.
- Process B has access to memory space Y in RAM.
- Process C has access to memory space X in RAM.

As the processes complete the tasks, they need to access a different memory space:

- Process A requests to access memory space Y in RAM and is denied because process B is waiting to move.
- Process B requests to access memory space X in RAM and is denied because process C is waiting to move.
- Process C requests to access memory space Z in RAM and is denied because process A is waiting to move.

The system is now deadlocked. Each process is waiting for access to the vital resource, and since neither process can move forward, the system crashes.

1.4 Disk Management

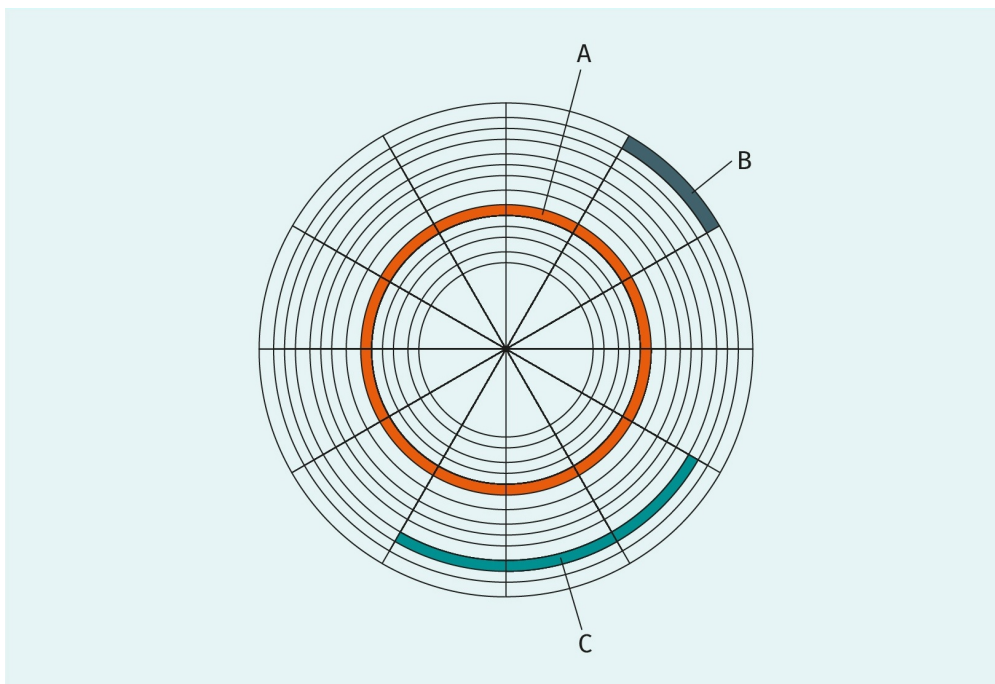
The operating system must use a storage device to store bulk data. Therefore, there must be a way for the operating system to manage the data storage system, which, in most cases, is comprised of hard disk drives (HDD). The disk management system must address the swap space, scheduling, and buffering of the storage device attached to the system.

Disk Structure

A hard disk drive comprises one or more magnetic platters mounted onto a spindle with an actuator arm and a magnetic head. The actuator is a powerful magnet and changes the polarity of specific locations on the platter. In the following figure, the typical structure of hard disk drive is shown. The platter is divided into concentric circles ranging from the interior of the disk to the exterior edge. The area marked as “A” is the track. The system then subdivides the track into **sectors** (the area marked as “B” in the figure). Various sector sizes can be encountered. Originally, sectors were 512 bytes in size; this has increased over time to 4096 bytes (Disk sector, n.d.). The system restricts the file system to track data via clusters. A cluster comprises one or more sectors (shown as “C” in the figure).

Sector
This is the smallest unit that can be accessed on a hard disk.

Figure 7: Disk Structure



Source: Created on behalf of IU (2021), based on Heron2/MistWiz (2008).

Solid-state drives (SSDs) are storage devices with no moving parts. Memory chips are used to store the data, allowing for

- a lighter device,

- higher reliability,
- greater read/write speed, and
- longer battery life.

The firmware of the solid-state device controls several operations:

- wear leveling. This moves data around the chip(s) to ensure the system uses all the chips at the same rate. Data chips have a finite life span. If the system did not use wear leveling, the device's life span would be shortened.
- trim. This command wipes (overwrites) the unallocated space of the device. This action disables collecting data from the unallocated spaces of the device.
- garbage collection. This function scans the memory modules to identify deleted data blocks in the pages of the device. This causes the system to move allocated data to new blocks. After the data are transferred, the system overwrites (wipes) the now-unused data blocks.

These operations reduce our ability to recover data from the unallocated space. These operations are controlled by firmware and cannot be stopped. As soon as the device is powered on, the firmware begins scanning the memory chips, looking for data to move.

Partitions

Partitioning is when the physical device is divided into logical segments called volumes. The master boot record (MBR) partitioning scheme restricts this to four primary partitions. For example, with one physical device, a primary partition can host a Windows operating system, with a second primary partition hosting a Linux operating system. A primary partition is required to boot into an operating system. When the user selects the operating system, that partition becomes the "active" partition.

The MBR is found at sector zero. The MBR contains the information needed to boot the system. Since the formatting includes the MBR in sector zero, it cannot be longer than 512 bytes. The first 446 bytes comprise the boot code, the next 64 bytes are the partition table, and the last two bytes show the boot signature (x55 xAA). The partition table shows which partition is active. Once the starting sector (or active partition) is located, the boot process will continue.

As mentioned, the MBR only allows four primary partitions. This caused the creation of the **extended primary partition**. A singular primary partition is taken and used to create additional logical partitions over the four primary partitions.

The following figure illustrates the replacement of the primary partition with an extended partition.

Extended primary partition

This is a primary partition that has been divided up into logical partitions.

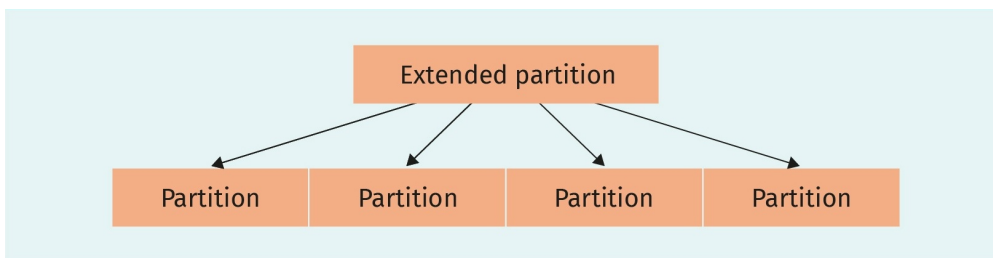
Figure 8: MBR Partition Map



Source: Created on behalf of IU (2022).

The following shows this extended partition and the multiple logical partitions within the extended partition boundary.

Figure 9: Extended Partition Map



Source: Created on behalf of IU (2022).

The extended partition will not have a volume boot record (VBR). It will have an extended boot record (EBR), which points to the first extended logical partition. The first extended logical partition contains information about itself and is a pointer to the next extended logical partition, creating a daisy-chain of pointers from one extended logical partition to the next.

The globally unique identifier (GUID) partition table (GPT) is the partitioning scheme used for newer storage devices and the latest unified extensible firmware interface (UEFI) standard. The UEFI replaces the binary input/output system (BIOS), while the GPT replaces the master boot record (MBR) partitioning scheme. GPT partitioning is found on the physical storage devices used by most systems.

The GPT partitioning scheme uses the logical block addressing (LBA) and uses a protective MBR, which can be found at the physical sector zero. The protective MBR allows for some backward compatibility and helps remove any issues when dealing with legacy utilities that do not recognize the GPT partitioning scheme. There is no boot code available in the protective MBR.

Host protected areas (HPA) and device configuration overlays (DCO) are “hidden” areas on the hard drive created by the manufacturers. The HPA is used by the manufacturer to store recovery and diagnostics tools and cannot be changed or accessed by the user. The DCO is an overlay that allows the manufacturer to use standard parts to build different products. It enables the creation of a standard set of sectors on a component to achieve uniformity. For example, a manufacturer may use one set of parts to create a 500 GB hard drive—the

same components can also be used to make a 600 GB hard drive. Usually, the user does not have access to the DCO. However, some utilities are available that will allow a user to access these locations and store data.

RAIDs

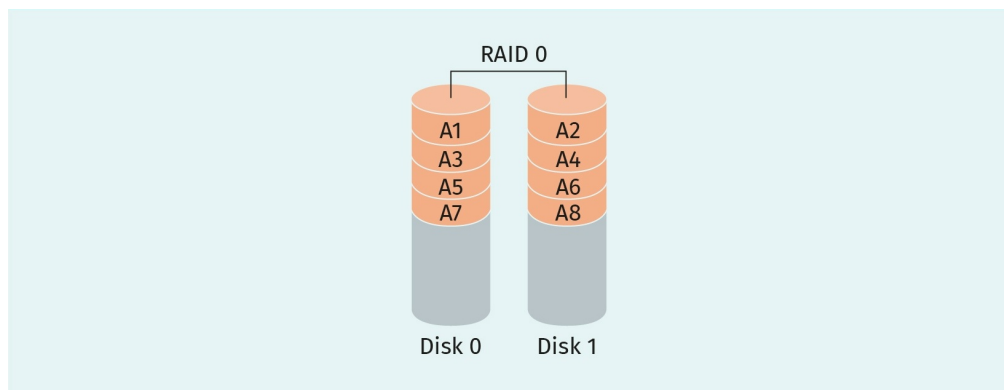
RAID

This is a method of data storage that uses more than one disk.

Another storage option that you may encounter is a redundant array of independent (or inexpensive) disks: **RAID**. A RAID is a method of storage that uses more than one disk. This allows for increased security or performance. A RAID can be used to protect against defective hardware or disk errors, but does not provide protection against user error or catastrophic damage (such as fire or flood). There are various levels of RAID, which we will now discuss.

The first of these, RAID 0, is known as a “striped set.” The data stored in a RAID 0 configuration are segmented into blocks and written across the drives in the set. This method allows for greater I/O (reads/writes) due to the concurrent use of two or more disks. It further enhances performance with the use of multiple disk controllers. This scheme splits the data across two or more disks without parity information (parity is added to the dataset to help achieve redundancy). Some benefits of RAID 0 are the lack of overhead for data storage capacity or parity controls and the relative ease of implementation. The lack of parity results in the absence of fault tolerance or redundancy for the dataset spread across the disks. As a result, if one disk of the stripes fails, access to the data set is lost. A diagram of RAID 0 is shown in the following figure.

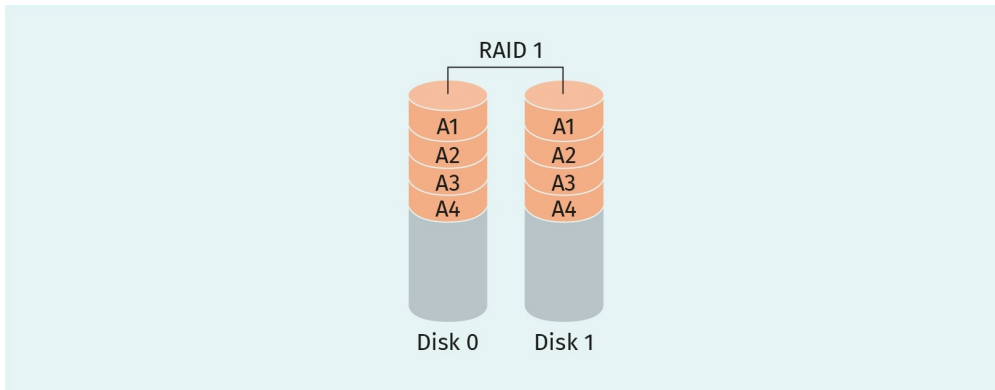
Figure 10: RAID 0



Source: Cburnett (2006a). CC BY-SA 3.0.

RAID 1 is known as a “mirror” and uses two or more hard disks. The data set is placed on one drive and an exact duplicate is placed on the second drive. This RAID scheme also does not use parity. RAID 1 achieves redundancy at a reduced performance rate. If the drives being used differ in capacity, the available space will match the capacity of the smaller drive. RAID 1 is shown in the following figure.

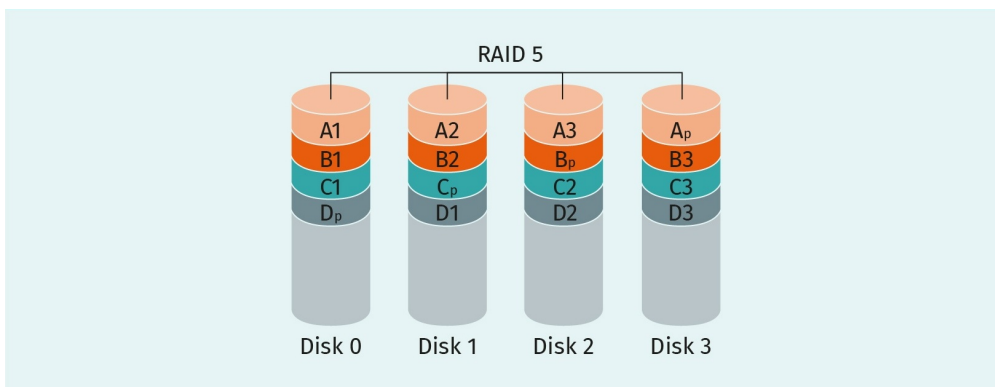
Figure 11: RAID 1



Source: Cburnett (2006b). CC BY-SA 3.0.

RAID 5 requires a minimum of three drives. The dataset is striped across the drives in the array using blocks of data. A parity checksum of the data blocks is also written across the drives. The following figure shows the data blocks A1, A2, and A3 placed across the disks. Block A_p is the parity for the “A” data blocks. This structure is repeated for data blocks “B,” “C,” and “D.” The system rotates through the disks in the array so that the parity blocks are not stored on a single drive. Using parity data allows the system to recalculate the data blocks if a drive is no longer accessible. With a RAID 5 system, a single hard drive can fail without access to the data set being lost. A RAID 5 system can be created using a software or hardware controller. When using a RAID 5 array, read performance is faster than write performance; the system must calculate the parity data when writing the data to the storage space. While you can replace a failed drive in the array, the time to rebuild the array can often exceed 24 hours. If another drive fails in this time, the data set will be lost.

Figure 12: RAID 5

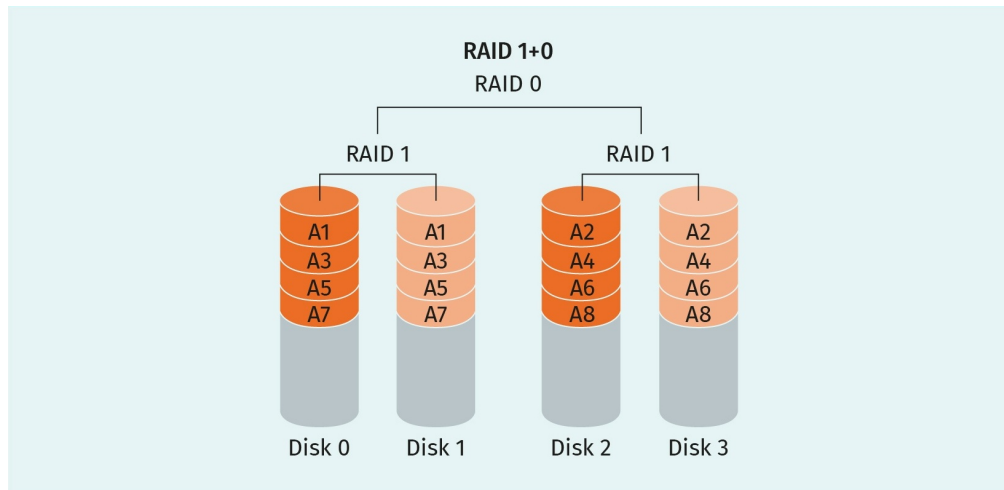


Source: Cburnett (2006c). CC BY-SA 3.0.

There is another set of RAID options you may encounter: the nested (or hybrid) RAID. The nested RAID combines the typical RAID schemes mentioned above to gain additional performance, redundancy, or both. The naming convention for the nested RAIDs is a combi-

nation of the conventional RAID system numbers. For example, RAID 10 is a combination of RAID 1 and RAID 0. An example of RAID 10 is shown in the following figure. RAID 10 is a stripe with a mirror. A minimum of four disks are required to implement a RAID 10 array.

Figure 13: RAID 1+0: Two RAID 1s with Four Disks over One Logical Volume



Source: NudelSuppe (n.d.), based on Cburnett (2006a). CC BY-SA 3.0.

1.5 Input/Output

The operating system controls all the devices attached to the system. We refer to this as input/output (I/O) management. The operating system provides an interface between the I/O devices and the rest of the system, and tracks any errors and issue commands. To help control I/O devices, specific I/O software, known as a device driver, is used.

The devices communicate with the operating system via a physical connection, such as a universal serial bus (USB) port, serial port, or wireless signal. When devices use a physical connection for communication, they connect to the physical computer system through a “bus.” A bus is simply a connection that adheres to a specific set of protocols. Buses you may encounter on a computer system include the following:

- peripheral component interconnect (PCI) bus
- small computer system interface (SCSI) bus
- expansion bus

Input/Output Devices

I/O devices are characterized into three groups: character, block, and network devices. A character device accepts or delivers a data stream of characters (keyboard is an example of a character device). A block device receives or stores information in a fixed size block. All data transfers take place in one or more blocks. Block sizes range from 512 bytes to 64 kB in size. Hard drives and USB sticks are examples of block devices. Network devices are

used to communicate with other computer systems via a communication network. The network device uses a “socket” as the interface connection between two computer systems. This connection allows data to be sent “bi-directionally,” which constitutes a full-duplex connection. The following are different methods the operating system can use to communicate with an I/O device.

Programmed I/O

When processes are executed and have an I/O instruction, the processor issues the command to the device controller. The controller then interfaces with the I/O device and creates a “status of operation” in the status register. The processor continually monitors the status register until the controller marks the operation as complete. This ties up the processor until it has completed the operation. For input operations, the I/O device places the data into the processor register. For output operations, the I/O device reads from the processor register. The processor handles the direct control of the I/O device. This can lead to performance degradation because the processor is not handling any other instructions.

Interrupt-driven I/O

Instead of waiting for the “status of operation” to be completed, an interrupt schema is used. When the I/O device has completed the operation, the device controller will send an interrupt flag to the processor. The processor suspends the process currently being executed and resumes the process that executed the I/O device operation. Most modern operating systems are interrupt-driven.

I/O using DMA

When the processor needs to access a large data set, the interrupt-driven method proves inefficient. Direct memory access (DMA) completes the operation without processor intervention. This method creates an interrupt after a block of data has been received. When the processor executes a process calling for the block transfer, it sends the command to the DMA controller, which then generates the interrupt after receiving the block of data. The processor is responsible for beginning and ending the operation. The DMA controller handles everything in between.

Port-mapped I/O (PMIO)

This method uses a particular class of processor instructions (IN and OUT) that have been designed specifically for handling I/O devices. These instructions copy one to four bytes between the register subdivisions of the CPU and the specified I/O port for the device. The I/O devices have their own segregated address space to be monitored. This is accomplished using an extra I/O pin on the CPU.

Memory-mapped I/O

Like port mapping, the same address space addresses the I/O device and system memory. The registers of the I/O devices and their memory are assigned to the specific address values. When that specified address receives a command from the processor meant for the

I/O device, the device responds. The address space is segregated and is unavailable for use in other processes or purposes. This segregation can be a permanent or temporary assignment.

Channel I/O

Channel I/O is used in mainframe computers and is considered high-performance architecture. Channel processors take the place of the CPU in handling I/O operations. A channel processor is deemed to be a simple self-contained controller specifically designed to handle I/O operations. The channel processor can take over all I/O operations until termination. When the I/O operation has been completed, or an error occurs, the channel processor sends an interrupt flag to the CPU. This technique is also considered to be a version of DMA access.

Input/Output Management

Because of the variety of devices that are available to the consumer, management of I/O devices can be an issue. The following issues have been identified in I/O device management:

- system response. An interrupt-driven I/O or programmed I/O must be implemented to manage the response time to input/output requests.
- transfer of information. Correctly identifying which devices are character or block based can prove to be an issue.
- interface. The operating system must provide a uniform interface between the user and the I/O devices.
- device independence. It is not feasible for the operating system to be updated every time a new device is released. The I/O should be device-independent and layered. The higher layers will allow the user to interface with the device while maintaining the physical details of the device in the lower layers.
- speed throughput. I/O devices communicate at varying speeds. In order to efficiently control I/O devices, a method to buffer the I/O requests must be implemented.
- multitasking. A priority system must be implemented to ensure that a device does not overuse the processor.
- security. The I/O devices must have security protocols implemented to prevent unauthorized or illegal use. Any errors generated must be addressed by the system.

The I/O system should have a user level and a kernel level. Each level will have a specifically designed function so changes in one layer do not affect changes in any other layer. The levels are as follows.

User-level I/O

This level creates a standard I/O library that can be accessed for user-level requests. It links the library to specific programs and becomes a regular part of the I/O system.

Kernel level I/O

The user is provided with an interface to access the I/O devices, but they do not know the specific hardware details of the device. At this level, there is a uniform interface used to communicate with attached devices. The I/O system provides operational control to the devices and monitors the processes that want to interact with the device. The I/O system monitors access to the devices and implements device scheduling when a queue develops. If there is an issue with the speed of requests and responses, a buffering system can also be implemented.

Device driver

With this level, each I/O device has a device driver so the I/O system can communicate with the device controller. The device driver accepts I/O requests from the kernel level subsystem once the device driver receives a request to start an operation, check the device's availability, and validate the request. If the device is busy, a queue is generated; if the device is not available, an error is generated. Once a device driver receives the commands, it writes the commands to the register. After writing each command, the driver checks with the device controller to ensure it is accepted. If the command is accepted, the device driver continues to write commands to the register.



SUMMARY

The operating system works behind the scenes and handles configuration of the connected devices. The operating system can operate in two different modes: user mode (which is more secure and limits access to hardware) and kernel mode (which runs underneath the user mode and has full access to the operating system and the attached hardware). This is comparable to using protection rings for software development, with ring zero being the most trusted and ring three being the least trusted.

Random-access memory (RAM) is a critical resource for the operation of the computer system. It is volatile storage, which means that once the computer system loses power, the data in RAM are lost. Almost every action taken by the user and operating system can leave artifacts within RAM.

A process is created when a program is executed within the memory framework. The process can also comprise threads to increase efficiency. The process life cycle starts with the execution of the process and begins either in the blocked, suspended, or ready state. Once the process is completed, the system ends the operation.

Hard disk drives are nonvolatile storage for the computer system. Two different partitioning schemes are commonly used: master boot record (MBR) and GUID partition table (GPT). Newer operating systems use GPT partitioning, which is also backward compatible (i.e., it can be used with older systems).

The operating system controls access to the devices using an input/output (I/O) system to maintain control. The types of device that can be attached are character and block devices. A character device accepts and sends data in a stream of characters, while a block device receives and sends data using predetermined block sizes.

LEKTION 2

OPERATING SYSTEMS INTERNALS

STUDY GOALS

On completion of this unit, you will be able to ...

- describe the internal workings of the operating system (OS)
- explain how the operating system handles processes.
- list the functions of the Windows registry.
- differentiate between NTFS and FAT file systems.
- understand the structure and information of a Master File Table (MFT).
- analyze the importance of the common attacks.

2. OPERATING SYSTEMS INTERNALS

Introduction

The internal functions of an operating system are essential and drive the need for security. It must be ensured that the operating system functions without an unauthorized third party's interference. In this unit, we will learn about system calls and how an application can use the system call to access resources within the protected ring. When dealing with the Microsoft Windows operating system, the operating system's "heart" is the Windows registry. As users interact with Windows, their actions are recorded within the registry's different facets.

Suppose we are examining a Windows-based computer system. The registry will be one of the first locations we look at to understand what has occurred within that system. We will look at the differences between two common file systems: new technology file system (NTFS) and file allocation table (FAT) 32. Both of these file systems do the same job by tracking the clusters for use within the storage device and tracking which clusters are being used for file storage. The manner in which they accomplish these tasks, however, differs significantly. NTFS is the default file system for the current version of the Windows operating system.

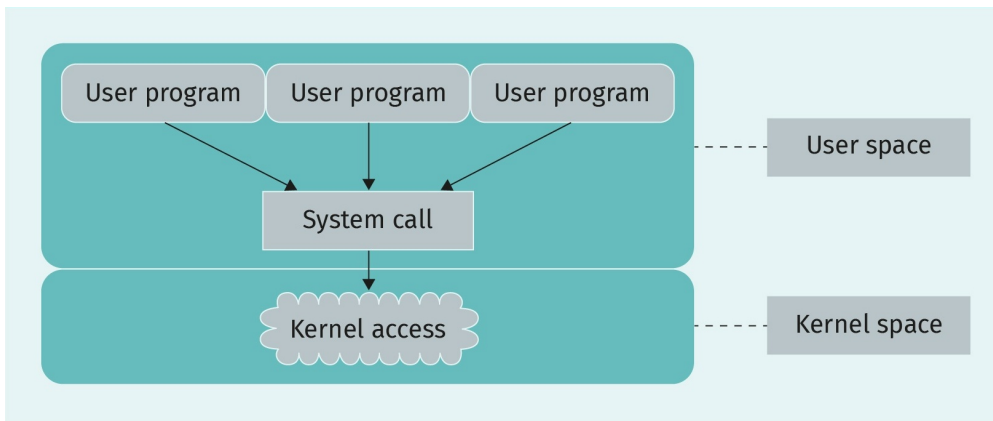
2.1 System Calls

To begin examining operating systems, we must understand the role of system calls (syscalls). There are two modes of operation within an operating system (OS): user and kernel mode. User mode is where all the user processes and operations are executed and does not have full access to the memory space or the hardware. **Kernel mode** is where the system's processes and operations are executed, and has full access to the memory space and hardware. An interface is used to allow user mode to communicate with kernel mode. This interface is a system call. The system call enables the operating system to receive a request to perform kernel functionalities that a user mode operation requires. The system call is not performing the kernel level operation itself but sending an interrupt request to the processor. Upon receipt of this, the operating system takes control of the processor. Once the operating system determines the nature of the system call, it executes the requested interrupt routine. We can visualize the system call as a bridge from the user programs to the operating system. System calls are used because we do not want user programs to have kernel-level access (for security reasons).

Kernel mode

The executing code has complete and unrestricted access to the underlying hardware.

Figure 14: User and Kernel Space with a System Call Acting as a Bridge



Source: Created on behalf of IU (2022).

We can consider a system call to be a management process that facilitates communication between the operating system and user-level application processes at the fundamental level. These processes include file management and process control management. The broader categories of common system calls are as follows:

- Process control monitors the initiation and termination of processes.
- File management monitors the applications' access to file operations, such as “create,” “delete,” “read,” “write,” “open,” and “close.”
- Device management monitors system calls asking to access or manage hardware resources.
- Information maintenance monitors systems calls asking to access or manage information resources.
- Inter-process communication monitors and coordinates communications between different processes, such as opening and closing connections and sending and receiving messages.

A system call and a regular function call are very similar. Both can be programmed using high-level code and are formatted similarly. While a regular function call cannot enter the kernel space, the system call can (however, it does not execute any operation here). To run a system call, a request must be sent to the OS, which passes on the relevant information. The system call can use three methods to send the information to the operating system:

1. Register method. The parameters will be stored in the processor’s registers.
2. Block method. If the parameter size is greater than the size of the registers, the system call uses a memory block in random access memory (RAM) and places the memory block’s address into the register.
3. Stack method. The parameters will be pushed into the stack and then grabbed by the operating system.

There are several factors used to determine which system calls can be implemented within the computer system. These factors include the type of hardware attached to the system, the system architecture, and the operating system. For example, for a Linux system, the

system maintains the system calls within the Linux core in a “system call table.” When the system call is placed into the table, entries are made listing a unique system call identification number and the function to be executed in kernel mode. When it’s time for the system call to be executed, the system call identification number is loaded into the processor’s registers and an interrupt is activated. The operating system then executes the system call.

In a Windows environment, an application programming interface (API) is used by the system call to access the kernel level space. The API provides access to the hardware level and the graphical interface for visual output. Each operating system has a different mechanism to allow the user to monitor the system calls being executed. In Windows, you can use the API monitor. When the user-level application makes a system call, the `CreateProcess` function is used, which identifies and monitors system calls. To watch the application as it interacts with the registry and file system, the “Process Monitor” (ProcMon) application can be used.

With a Linux system, you can use the “ltrace” application. This provides a listing of the library functions used by the application. Another application that can be used is “strace,” which provides a list of the application’s system calls.

We may want to monitor the system calls being used by the operating system in order to determine if a rogue user or malware is being used for dynamic link library (DLL) injection or API hooking. A DLL is a library that multiple processes can use. When an application loads and uses the DLL, it can send out system calls. For DLL injection, malware will create a rogue thread in a running process. The attacker then injects the DLL into the thread. The executing programs search for DLLs in the following order:

1. Executables directory
2. Current directory
3. System directory
4. Windows directory
5. System %PATH%

The windows registry has several keys that will load DLL. Two of these are as follows:

- HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\WindowsNT\CurrentVersion\Windows\AppInit_DLLs
- HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Session Manager\AppCertDLLs

If an attacker can access these locations on the file system or registry, they will be able to launch the compromised DLLs. Hooks allow the attacker to execute code when a specific message is sent to a window. Global hooks allow an attacker to have the malware run in every process of the system. However, this does not overcome the user account control (UAC). When monitoring the system calls, you may see the commands `create_remote_thread`, `virtualallocex`, and `writeprocessmemory`.

DLL forwarding is a method of loading a malicious DLL into a system in order to replace the system DLL with a malware version. The malware version of the DLL only has the capability for a select set of functions. When programs call for the DLL to be activated, the malware DLL acts like a man-in-the-middle attack and intercepts the DLL calls. For any procedures that the malware DLL cannot complete, it will forward the call to the original (but renamed) DLL so the system does not exhibit any suspicious behavior.

The “API hooking” technique monitors the interface between the software’s messages and the active processes. The malware tries to position itself to intercept the messages being broadcast. For example, the malware can listen for messages associated with keyboard events. This could be a keystroke logger, which then records the information (e.g., usernames, passwords, and contents of emails or other documents created on the system) sent in the keyboard event messages.

2.2 Process Table Analysis

The operating system tracks the processes through the use of the process table. As the system creates each process, the operating system creates an entry in the table. Some operating systems will generate the process table entry, which contains the process ID and a pointer to the location of the process control block (PCB). In other operating systems, the process table contains information about the state of the process, memory allocation, scheduling information, and program counter. The process table contains all the information needed to resume the process after it has been suspended or blocked. This allows the process to restart as if it had never stopped. Some fields of the process table deal with three broad functions:

1. Process management. This maintains information about the registers being used, process state, priority, process ID, when the process started, and what the parent process is.
2. Memory management. This maintains pointers to text, data, and stack segment information.
3. File management. This maintains information about root directory, user ID, group ID, and working directory.

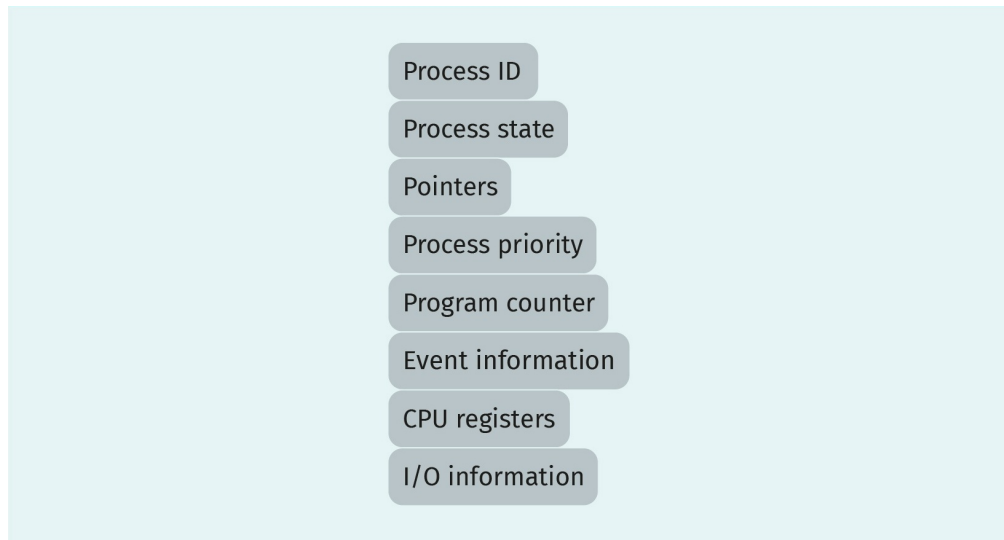
The information in these fields is very operating system-dependent, so additional data or missing data fields may be found during examination.

Process Control Block (PCB)

For the operating system to maintain control of the processes, it creates a process control block (PCB) in RAM for each process as they start. There is no set standard for a process control block, and it can differ greatly between operating systems. The process control block contains information about the state of the process, memory allocation, corners, scheduling information, and any data that would be needed to restart a process from a suspended or blocked state. Once the process has been terminated, the system deletes the process control block.

In the following figure, the PCB structure is laid out. The subsequent table gives a definition of each block.

Figure 15: PCB Information



Source: Created on behalf of IU (2022).

Table 2: PCB Fields

Process ID	Contains unique identification for the process
Process state	Contains the current state of the process (e.g., ready, suspended, or blocked)
Pointers	Contains a number of pointers pointing to, for example, the parent process, child process, or address space
Process priority	Contains the priority number given to the process (typically, a priority number of one is the highest level)
Program counter	Contains pointer to the next executable instruction
Event information	Contains information about specific events in the case that a process is blocked or waiting
Central processing unit (CPU) registers	Contains a list of CPU registers that are needed for the process to be in a running state
Input/output (I/O) information	Contains a list of I/O devices assigned to the process

Source: Created on behalf of IU (2022).

Threads

A thread is a subset of a process and can be found within the same address space. Threads run parallel to the process, and are sometimes referred to as “mini processes.” So, why would you want to have these mini processes operating parallel to primary process? Many applications involve multiple operations (or activities) being executed at the same time. For instance, if an author is creating a document using Microsoft Word, Word is not only recording the keys being pressed on the keyboard, but running multiple operations, such as spelling and grammar, auto-saving, and dictionary usage. If the user has added extensions to Word, they also require additional threads. Threads are slimmed-down when compared to a full process; they can be created and terminated much faster than a whole process. Of course, this also leads to a performance increase, primarily when the threads target I/O operations. While threads and processes share information, threads also have a data set that is not shared with the process. The fields of a thread are as follows:

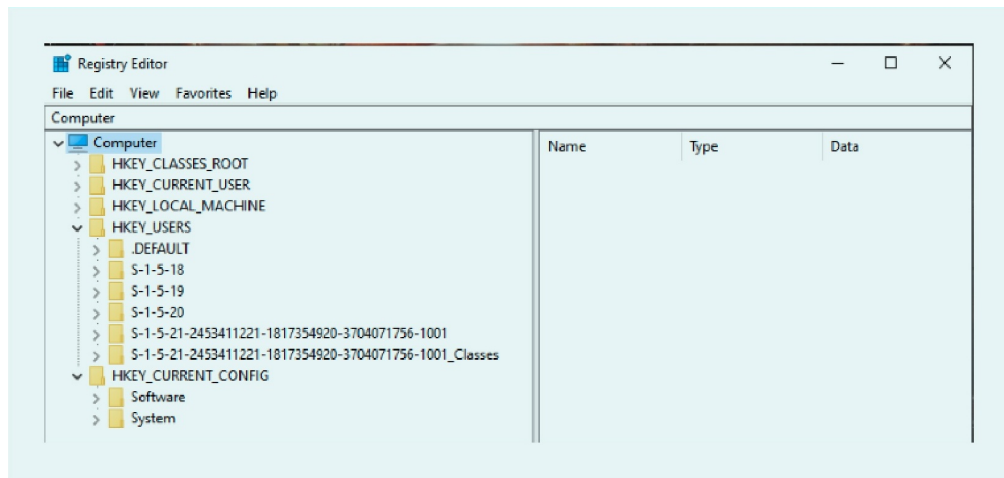
- processor registers. Each thread has a specific address space within the processor.
- stack. The stack contains local variables and the process address, and can be a kernel or user stack.
- program counter. This is unique for each thread. A pointer to the next executable instruction is present here.
- other. Any other necessary information is held here (e.g., prioritization, ID, or current state).

Just like a process with a PCB, each thread has a “thread control block,” which is used in the same context to save or restore the thread. The thread control block holds similar information to the PCB, such as thread ID, CPU registers, the current state of the thread, priority levels, and a pointer to the owner process.

2.3 Windows Registry

Microsoft defines the registry as “a central hierarchical database used in Windows 98, Windows CE, Windows NT, and Windows 2000 used to store information that is necessary to configure the system for one or more users, applications, and hardware devices” (Han et al., 2021, Description of the registry section). The registry is the “heart” of the Windows OS, with most users having little or no knowledge of the registry and how it functions. Every action taken within the operating system can leave an artifact in the registry for an investigator to analyze. Windows comes with a registry editor utility pre-installed.

Figure 16: Registry Editor



Source: Created on behalf of IU (2022).

Many user actions leave footprints within the registry, for example, clicking on the start menu and starting an application. All files listed in the start menu, such as Word documents, Excel spreadsheets, and videos played by the media player are all recorded within the registry. The registry records not only what was executed, but also the date and time the action took place. The keys found in the registry maintain a “last write time” value. When the key is created, modified, or deleted, this value is updated to reflect the new state.

There are five registry hive files that are important to the investigator. Microsoft defines the hive file as “a logical group of keys, sub-keys, and values in the registry that has a set of supporting files loaded into memory when the operating system is started or a user logs in” (Schofield et al., 2021a, para. 1). Whenever a new user logs into the system, a user profile file is created. There are seven registry hive files in the Windows operating system.

Table 3: Table of Registry Hives

Registry hive	Supporting files
HKEY_CURRENT_CONFIG	System, System.alt, System.log, System.sav
HKEY_CURRENT_USER	Ntuser.dat, Ntuser.dat.log
HKEY_LOCAL_MACHINE\SAM	Sam, Sam.log, Sam.sav
HKEY_LOCAL_MACHINE\Security	Security, Security.log, Security.sav
HKEY_LOCAL_MACHINE\Software	Software, Software.log, Software.sav
HKEY_LOCAL_MACHINE\System	System, System.alt, System.log, System.sav
HKEY_USERS\.DEFAULT	Default, Default.log, Default.sav

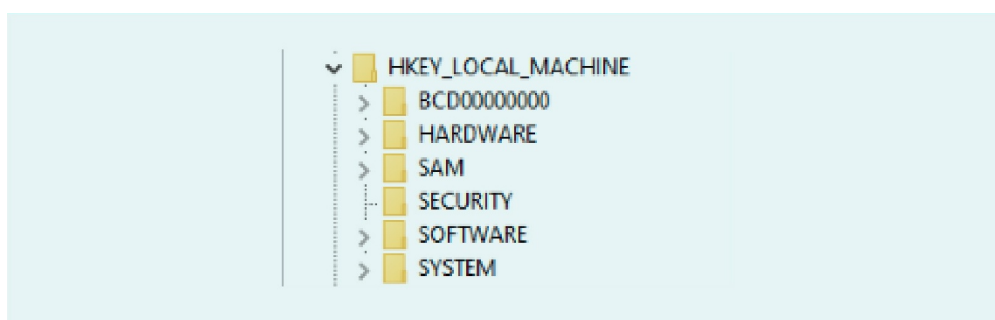
Source: Schofield et al. (2021a).

The system does not store certain portions of the registry hives on the hard drive. The hives are only active while the system is running. For example, HKEY_CURRENT_USER will not be found within a forensic image. When the system is running, the information found within HKEY_CURRENT_USER is populated by the logged-in user's "NTUSER.dat" file. Another setting not found on the file system is HKEY_CURRENT_CONFIG; this key is populated with the information contained in the registry key (HKEY_LOCAL_MACHINE\System\CurrentControlSet\Hardware Profiles\Current).

NTUSER.dat
This is a registry file that contains preferences and settings for each user account on a computer.

We can find the registry's supporting files in the following path: %SystemRoot%\System32\Config. The SAM, SECURITY, SOFTWARE, and SYSTEM hives are found in this location. In Windows 2000, XP, and 2003, the user profiles are found within the "documents and settings" directory. From Windows Vista, the user profiles are found within the "users" directory. The user profile hive can be found at the root of the user profile and has the file name "NTUSER.dat."

Figure 17: HKLM and Registry Hives



Source: Created on behalf of IU (2022).

The registry files can be grouped into "system" and "user" categories based on the type of data encountered.

System settings

"SAM" stands for security accounts manager and contains user login information. Below, you can see a decoded SAM file with two user accounts. The administrator account has a relative identifier (RID) of 500 and has been disabled. The second account shown is a user-created account with username 'jcloudy' and an RID of 1001. The dates and times of the most recent login, account creation, and creation of password hint can be seen here.

Code

User Information

```
-----  
Username      : Administrator [500]  
SID           : S-1-5-21-2734969515-1644526556-1039763013-500
```

Full Name :
User Comment : Built-in account for administering the computer/domain
Account Type :
Account Created : Tue Mar 27 12:13:26 2018 Z
Name :
Last Login Date : Never
Pwd Reset Date : Never
Pwd Fail Date : Never
Login Count : 0
--> Password does not expire
--> Account Disabled
--> Normal user account

Username : jcloudy [1001]
SID : S-1-5-21-2734969515-1644526556-1039763013-1001
Full Name :
User Comment :
Account Type :
Account Created : Tue Mar 27 09:18:58 2018 Z
Name :
Password Hint : It's me!
Last Login Date : Fri Apr 6 12:26:27 2018 Z
Pwd Reset Date : Tue Mar 27 09:18:58 2018 Z
Pwd Fail Date : Fri Apr 6 03:30:52 2018 Z
Login Count : 23
--> Password does not expire
--> Password not required
--> Normal user account

SECURITY is linked to the security database if used in a domain environment. This appears empty unless the user has administrative permissions.

SOFTWARE includes information on the software and system configuration. Below, you can see the Windows version and install date, as well as the current path and default directories.

Code

```
-----  
winver v.20081210  
(Software) Get Windows version  
  
ProductName = Windows 10 Education  
InstallDate = Tue Mar 27 12:13:27 2018  
-----  
win_cv v.20090312  
(Software) Get & display the contents of the Windows\CurrentVersion key  
  
Microsoft\Windows\CurrentVersion
```

LastWrite Time Wed Apr 4 05:45:49 2018 (UTC)

```
SM_GamesName : Games
ProgramFilesPath : %ProgramFiles%
ProgramFilesDir : C:\Program Files
DevicePath : %SystemRoot%\inf
ProgramW6432Dir : C:\Program Files
MediaPathUnexpanded : %SystemRoot%\Media
ProgramFilesDir (x86) : C:\Program Files (x86)
CommonFilesDir : C:\Program Files\Common Files
CommonW6432Dir : C:\Program Files\Common Files
SM_ConfigureProgramsName : Set Program Access and Defaults
CommonFilesDir (x86) : C:\Program Files (x86)\Common Files
```

SYSTEM includes information about the system configuration, including currently mounted devices. Below is an example of some of the information found in the SYSTEM hive.

Code

```
-----
timezone v.20160318
(System) Get TimeZoneInformation key contents

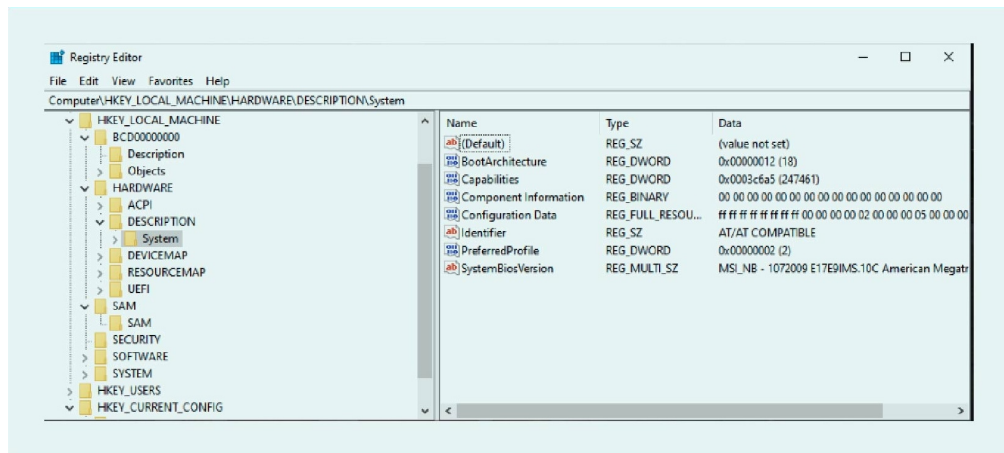
TimeZoneInformation key
ControlSet001\Control\TimeZoneInformation
LastWrite Time Tue Mar 27 09:56:27 2018 (UTC)
    DaylightName -> @tzres.dll,-111
    StandardName -> @tzres.dll,-112
    Bias -> 300 (5 hours)
    ActiveTimeBias -> 240 (4 hours)
    TimeZoneKeyName-> Eastern Standard Time
-----
```

User Settings

NTUser.dat contains information about user behavior and their settings. UsrClass.dat contains information concerning the user access control (UAC) and information about the graphical user interface (GUI) display.

The structure of the registry is that of a “tree format,” as shown in the following figure. The top-level folder is called a key, and can have child directories (referred to as sub-keys). Both the key and sub-key can contain data entries known as values. As shown, the sub-key “system” is highlighted. The full path is HKLM/HARDWARE/DESCRIPTION/System. Under “system” is the “value” of “identifier,” followed by the “type” REG_SZ. In the “data” field is the term “AT/AT compatible.”

Figure 18: The Windows Registry



Source: Created on behalf of IU (2022).

Microsoft provides definitions for each of the different values found in the “type” field, which are listed in the following table.

Table 4: Microsoft’s Definitions of Values Found in the “Type” Field

Value	Description
REG_BINARY	Binary data in any form
REG_DWORD	A 32-bit number
REG_DWORD_LITTLE_ENDIAN	A 32-bit number in little-endian format
REG_EXPAND_SZ	A null-terminated string that contains unexpanded references to environment variables (e.g., %PATH%)
REG_LINK	A null-terminated Unicode string that contains the target path of a symbolic link
REG_MULTI_SZ	A sequence of null-terminated strings, terminated by an empty string
REG_NONE	No defined value type
REG_QWORD	A 64-bit number
REG_QWORD_LITTLE_ENDIAN	A 64-bit number in little-endian format
REG_SZ	A null-terminated string, either in Unicode or American National Standards Institute (ANSI) encoding

Source: Schofield et al. (2021b).

The maximum number of levels for the registry tree is 512. No two registries will be exactly the same. The data stored in the registry is hardware or configuration specific. Every time the system starts or shuts down, the content of the registry hives changes.

Registry hive files are made up of 4096-byte segments called bins. As the hive file grows, it does so in 4096-byte chunks. Each bin will have the file signature of “regf” and hexadecimal 72656766, as shown in the following figure.

Figure 19: Hive File Header

Offset	00	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	ASCII
00000000	72	65	67	66	21	01	00	00	21	01	00	00	00	00	00	00	regf!.....
00000016	00	00	00	00	01	00	00	00	05	00	00	00	00	00	00	00

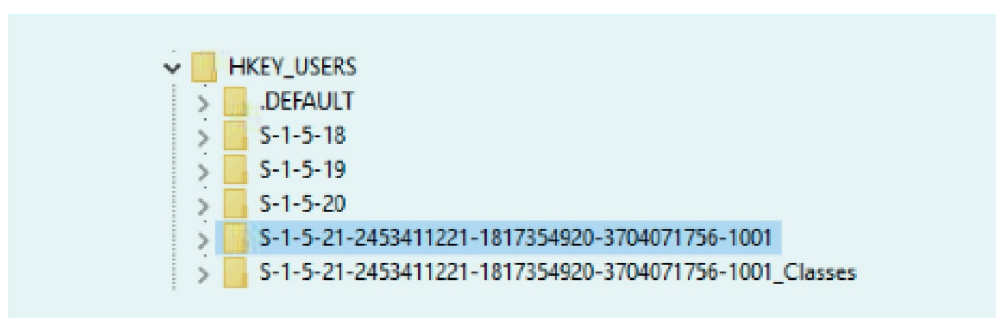
Source: Created on behalf of IU (2022).

When conducting an analysis, it is possible to recover pertinent information in old registry files stored in unallocated space by searching for the file signature.

SID/RID

A security identifier (SID) is used by Windows to identify objects internally. In the following figure, the first three numerical entries are system-created SIDs. S-1-5-18 is a local system service, S-1-5-19 is a local system NT authority (i.e., a powerful local system account), and S-1-5-20 is a network service. The SID also contains the relative identifier (RID) for user accounts. The last three to four digits of the SID are the RID. Each user account receives an RID, with the account’s permissions dictating what those numbers will be. For example, an RID of 500 identifies the account as a system administrator account. A guest account will have an RID of 501. In the following figure, we see the user account has an RID of 1001.

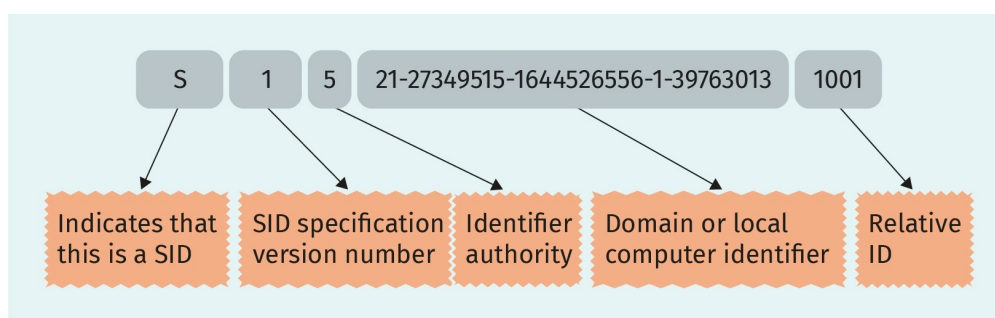
Figure 20: Registry SID and RID Values



Source: Created on behalf of IU (2022).

This user account is a user-created account, not an account created by the system. A system-created account would have an RID under 1001, for example, the default administrator account value of 500. The makeup of the SID is shown in the following figure.

Figure 21: SID with Explanation



Source: Created on behalf of IU (2022).

An investigator can relate an RID to a specific user account. When a user creates a new account, the RID will increase by one digit. Let's say that, during an investigation, you are looking for the account with an RID of 1002. When you examine the Registry, you only find an account with an RID of 1004. Where are the accounts with RIDs 1001—1003? One possibility is that the user deleted these accounts in an attempt to cover their tracks.

Registry Keys of Interest

By default, the registry has four keys for programs to be executed when a user logs into the system:

1. HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Run
2. HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run
3. HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\RunOnce
4. HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\RunOnce

The LOCAL_MACHINE has a RUN and a RUNONCE key, as does the CURRENT_USER. A value is stored within the RUNONCE and is deleted when the executable is activated. If the system was started in safe mode, it ignores any entries within these four keys. These locations are important to examine, especially if a user claims that malware was started automatically or denies knowledge about the program being initiated at startup.

Most Recently Used (MRU)

A “most recently used” (MRU) is a list of recently used files stored in the user's NTUser.dat and SOFTWARE hives. When we open an application and see the history of files used by the application, this is the MRU. There are many MRU lists stored within the registry file.

We will now go over some more common locations. The program RegRipper was used to extract the registry data from a forensic image being used in the “2018 Lone Wolf Scenario” provided by DigitalCorpora. DigitalCorpora (2021) describes itself as “a website of digital corpora for use in computer forensics education research (para. 1). All the disk

images, memory dumps, and network packet captures on this website are freely available and may be used without prior authorization or International Review Board (IRB) approval (Garfinkel et al., 2009).

OpenSavePidLMRU, found in the user's NTUser.dat, tracks the last twenty files that were opened or saved via the Windows Common Dialog (these are the "open/save as" dialog boxes). In the example below, we can see the last twenty files modified by the user.

Code

```
OpenSavePidLMRU\*
LastWrite Time: Fri Apr 6 03:56:31 2018
Note: All value names are listed in MRUListEx order.

My Computer\CLSID_Desktop\Document1.pdf
  My Computer\CLSID_Desktop\APK.docx
  My Computer\CLSID_Desktop\TravelForm.pdf
  My Computer\CLSID_Desktop\IURegistration.pdf
  My Computer\C:\Users\jcloudy\Desktop\Notes (4apr).docx
  My Computer\CLSID_Desktop
  Libraries
  My Computer\CLSID_Desktop\New Rules.pptx
  My Computer\CLSID_Desktop\The Agile Manifesto.docx
  My Computer\C:\Users\jcloudy\Desktop\The Agile Manifesto.docx
  My Computer\CLSID_Desktop\Huckleberry.docx
  My Computer\CLSID_Desktop\IMG_050220.jpg
  My Computer\CLSID_Desktop\IMG_040220.jpg
  My Computer\CLSID_Desktop\download.jpg
  My Computer\CLSID_Desktop\download1.jpg
  My Computer\CLSID_Desktop\map.jpg_large
  My Computer\CLSID_Desktop\Using ATT&CK As a Teacher.html
  My Computer\CLSID_Desktop\8 Best Practices for Data Security in
  Hybrid Environments.html
  My Computer\CLSID_Desktop\Wolf.png
  My Computer\CLSID_Desktop\Sheep.jpg
```

There is a selection of .pdf, .docx, .jpg and .png files to examine. The information contained in the key includes the file path for the files in question. This is a good artifact to see what the user has been accessing.

The SOFTWARE hive contains a large amount of information about the files being accessed by the user. In the next example, we will look at an overview of the RecentDocs key. This key is an MRU for Windows Explorer and displays the history for .csv, .docx, and .html file types, as shown in the following example.

Code

```
• Software\Microsoft\Windows\CurrentVersion\Explorer\RecentDocs\.csv
  LastWrite Time Fri Apr 6 12:27:08 2018 (UTC)
MRUListEx = 0
0 = rootkey.csv
• Software\Microsoft\Windows\CurrentVersion\Explorer\RecentDocs\.docx
  LastWrite Time Thu Apr 5 08:32:48 2018 (UTC)
MRUListEx = 0,3,1,2
0 = Huckleberry.docx
3 = Notes (4apr).docx
1 = APK.docx
2 = The Agile Manifesto.docx
• Software\Microsoft\Windows\CurrentVersion\Explorer\RecentDocs\.html
  LastWrite Time Fri Mar 30 04:32:26 2018 (UTC)
MRUListEx = 1,0
1 = Using ATT&CK As a Teacher.html
0 = 8 Best Practices for Data Security in Hybrid Environments.html
```

In the first entry, the registry has recorded the user accessing a .csv file with the file name rootkey.csv. The following output shows the .docx files accessed by the user. Four Word (.docx) documents have been accessed by the user. In the final output, we see that the user also had access to .html documents. What is listed here is not a history of .html files accessed through the internet, but .html files that the user either created or saved from while browsing. This information can be useful in proving or disproving the allegations being investigated.

The RecentDocs key is also a good source of evidence and is similar to the MRU lists just discussed. This key contains a detailed list of files that were executed or opened by the user through the Windows Explorer application, as well as sub-keys based on file extension that list the files that were executed or opened. The system stores the dates and times that the file was executed or opened by the user in chronological order. There is also an additional sub-key that lists the dates and times that the user opened folders on the system. The time of the last entry or modification of the key corresponds to the last entry in the list.

Code

```
recentdocs v.20100405
(NTUSER.DAT) Gets contents of user's RecentDocs key
```

```
RecentDocs
**All values printed in MRUList\MRUListEx order.
LastWrite Time Fri Apr 6 12:27:08 2018 (UTC)
11 = Downloads
37 = rootkey.csv
36 = Hardware and Sound
35 = ::{025A5937-A6BE-4686-A844-36FE4BEC8B6D}
10 = Wolf.jpg
34 = Document1.pdf
```



```
33 = TravelForm.pdf
12 = Huckleberry.docx
32 = IURegistration.pdf
31 = DailyMenu.pdf
30 = Notes (4apr).docx
29 = Desktop
7 = OneDrive
28 = worldmap.pptx
13 = APK.docx
27 = The Agile Manifesto.docx
16 = Sheep.jpg
```

Time Zone

Time zone information provides a starting point to correlate the user activity recorded on the computer system to when the incident occurred. All of the internal date or time stamps on the computer system are based on the time zone information recorded in the registry. We can find the time zone information within the system hive and in the `TimeZoneInformation` key, located at `SYSTEM\CurrentControlSet\Control\TimeZoneInformation`. RegRipper has been used to create the following output.

Code

```
-----
timezone v.20160318
(System) Get TimeZoneInformation key contents

TimeZoneInformation key
ControlSet001\Control\TimeZoneInformation
LastWrite Time Tue Mar 27 09:56:27 2018 (UTC)
    DaylightName -> @tzres.dll,-111
    StandardName -> @tzres.dll,-112
    Bias -> 300 (5 hours)
    ActiveTimeBias -> 240 (4 hours)
    TimeZoneKeyName-> Eastern Standard Time
-----
```

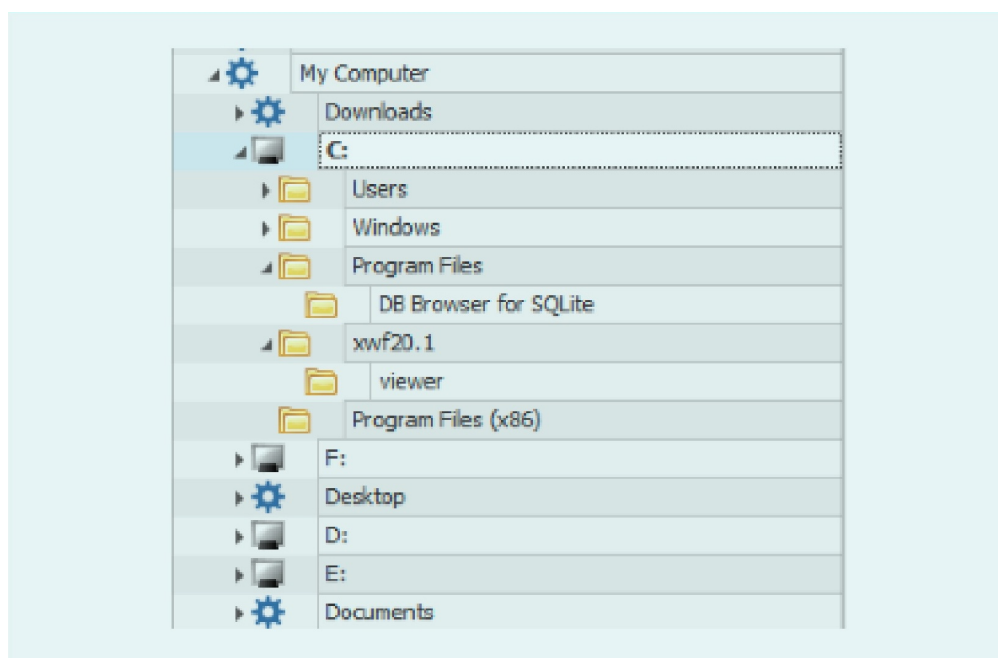
`Tzres.dll` is the time zone resource .dll file for the system. As the output is analyzed, we encounter the fields `BIAS` and `ACTIVE TIME BIAS`. These fields are populated with values of 300 and 240. This is the number of minutes the time zone in question is offset from Greenwich Mean Time (GMT). The following field is the time zone common name, which is Eastern Standard Time (EST). The time zone provides the point of reference for all the dates and timestamps in the system. Suppose we are in GMT ± 0 , the location of the incident is GMT +2, and the subject computer system is set at GMT -4. We need to reconcile the different time stamps into a consistent format to conduct a time analysis of the artifacts.

Shellbags

Shellbags is a set of registry keys used by the operating system to remember the size and location of the folder that the user has accessed. When the user opens the file explorer and the folder and icons are displayed, this information is being recorded. This artifact can display artifacts showing user interaction with network devices, removable media, or encrypted containers. These keys are located in the registry hive `UsrClass.dat` in the following path: `%USERS% AppData\Local\Microsoft\Windows folder`

In the following figure, the free utility Shellbag Explorer is used to examine user activity. The image shows that the user has accessed the “my computer,” “C:,” “D:,” “E:,” and “F:” folders.

Figure 22: ShellBags Explorer Output



Source: Created on behalf of IU (2022).

Within the C: folder, the user opened the following sub-folders: “users,” “Windows,” “program files,” “xwf20.1,” and “program files (x86)”. Finally, the user opened the “DB browser for SQLite” and “viewer” sub-folders. By analyzing this artifact, we can recreate a list of the folders the user viewed.

USB Devices

There are security risks associated with the use of a USB device. These small, portable, high-capacity storage devices can be used to exfiltrate data from the organization or deliver malware into the organization. As digital forensic investigators, we want to know if the user attached any USB devices to the host system being examined. The registry contains several entries that will help us to determine whether a user has attached any USB

devices to the system. In the SYSTEM hive, there are two entries, SYSTEM\CurrentControlSet\Enum\USB and SYSTEM\CurrentControlSet\Enum\USBSTOR, which help to see what devices the user may have attached to the system.

Code

```
usbdevices v.20140416  
(System) Parses Enum\USB key for USB & WPD devices
```

```
VID_0781&PID_5580  
LastWrite: Wed Mar 28 09:22:21 2018  
SN : AA010215170355310594  
LastWrite: Wed Mar 28 12:13:16 2018
```

```
VID_0781&PID_5580  
LastWrite: Tue Mar 27 08:22:21 2018  
SN : AA010603160707470215  
LastWrite: Tue Mar 27 21:45:44 2018
```

The output from RegRipper shows that two USB devices were attached to the system. The two devices can be identified using their serial numbers. Devices that do not have a unique serial number will have an & as the serial number's second character. The vendor ID (VID) and product ID (PID) values are seen here. The user attached the devices on different days, March 27th and March 28th. The first "LastWrite" time is when the USB device was plugged into the system after the most recent boot session. The second "LastWrite" time is the last time the device was connected and disconnected from the system to the system. Let's look at the usbstor registry key and see if we can get more information.

Code

```
usbstor v.20141111  
(System) Get USBStor key info  
USBStor  
ControlSet001\Enum\USBStor
```

```
Disk&Ven_SanDisk&Prod_Extreme&Rev_0001 [Wed Mar 28 09:22:21 2018]  
S/N: AA010215170355310594&0 [Wed Mar 28 12:11:44 2018]  
Device Parameters LastWrite: [Wed Mar 28 12:11:42 2018]  
Properties LastWrite : [Wed Mar 28 09:16:45 2018]  
FriendlyName : SanDisk Extreme USB Device
```

```
S/N: AA010603160707470215&0 [Tue Mar 27 08:22:21 2018]  
Device Parameters LastWrite: [Tue Mar 27 08:22:21 2018]  
Properties LastWrite : [Tue Mar 27 08:23:58 2018]  
FriendlyName : SanDisk Extreme USB Device
```

As we look at the `usbstor` key's output, we can determine the brand name of the devices being used and verify their serial numbers using the USB key information. A third registry key provides information about USB use on the system: `HKLM/SYSTEM/MountedDevices`. The data in this key allow us to map the identified USB device(s) to a drive letter on the system.

Figure 23: Contents of MountedDevices

```
mountdev v.20130530

(System) Return contents of System hive MountedDevices key

MountedDevices

LastWrite time = Wed Mar 28 08:22:21 2018Z

Device: _??
_USBSTOR#Disk&Ven_SanDisk&Prod_Extreme&Rev_0001#AA010215170355310594&0#
{53f56307-b6bf-11d0-94f2-00a0c91efb8b}\DosDevices\E:\??\Volume{5c3108bf-31c0-
11e8-9b10-806e6f6e6963}

LastWrite time = Tue Mar 27 09:22:21 2018Z

Device: _??
_USBSTOR#Disk&Ven_SanDisk&Prod_Extreme&Rev_0001#AA010603160707470215&0#
{53f56307-b6bf-11d0-94f2-00a0c91efb8b}\DosDevices\D:\??\Volume{3869c27a-31b8-
11e8-9b12-ecf4bb487fed}
```

Source: Created on behalf of IU (2022).

The information from this registry key shows that two devices were attached to the system. Device `AA010215170355310594` was connected on March 28, 2018, and was assigned the drive letter `E:`. The second device, `AA010603160707470215`, was connected on March 27, 2018, and was assigned the drive letter `D:`. Can you determine which user account is responsible for attaching the two USB devices? We cannot determine which user account is responsible for attaching the two USB devices yet, but there are additional registry keys we can examine. The last key identified 2 globally unique identifiers (GUIDs): `5c3108bb-31c0-11e8-9b10-806e6f6e6963` and `3869c27a-31b8-11e8-9b12-ecf4bb487fed`.

The system assigned a GUID to each USB device as they were attached. The user's `NTUser.dat` contains the next "breadcrumb" we want to follow. Below is the output from `RegRipper`, which was taken from the `MountPoints2` key in the user's `NTUser.dat` file.

Code

```
mp2 v.20120330  
(NTUSER.DAT) Gets user's MountPoints2 key contents
```

```
MountPoints2  
Software\Microsoft\Windows\CurrentVersion\Explorer\MountPoints2  
LastWrite Time Fri Apr 6 12:35:08 2018 (UTC)
```

Remote Drives:

```
Volumes:  
Fri Apr 6 12:35:08 2018 (UTC)  
    {76d45981-0000-0000-0000-100000000000}  
Tue Mar 27 21:45:54 2018 (UTC)  
    {3869c27a-31b8-11e8-9b12-ecf4bb487fed}  
Tue Mar 28 09:32:09 2018 (UTC)  
    {09931f21-7faf-44a9-81d8-1e73c14b9eaf}  
    {5c3108bb-31c0-11e8-9b10-806e6f6e6963}
```

Both GUIDs match the information we found in the `MountedDevices` key from the `SYSTEM` registry, and the date and time stamps match when the devices were last mounted. We have therefore identified a link between the use of a USB device and a specific user. Now, you can go back and look at the `ShellBags` artifact and see if the user interacted with the attached device.

2.4 File System Forensics

What is a file system? While the operating system controls the interaction between the user, applications, and hardware, a file system maintains control of where data are stored on a storage device. There are many types of file systems. Ultimately, they all do the same job. They identify which clusters are available for use on the storage device and which clusters are occupied. The file system is not operating system dependent; there are multiple file systems that most operating systems can read, and there are file systems that only specific operating systems can only read. For example, **file allocation table 32** (FAT32) is a file system that can be read by Windows, macOS, and Linux operating systems. Microsoft specifically designed the NTFS file system for Windows, but some third-party drivers will allow macOS and Linux to use NTFS. Using this third-party driver is feasible until Microsoft makes an update to NTFS, at which point the user has to wait until the third party updates their NTFS driver.

FAT32
This is the 32-bit version
of the FAT file system

A storage device can contain one or more partitions, with each partition having a given file system (in most cases). The file system monitors the clusters for that partition and tracks the file allocation as files are added, changed, or deleted. This begs the question: What information do we want the file system to be responsible for? All file systems will keep track of the following information points, which you may come across more or less depending on which file system is being used:

- the name of the file
- the owner of the file
- starting cluster and file size
- modified, accessed, or created (MAC) date timestamps
- the access control list (ACL), which provides access control
- the file type, i.e., archive, hidden, or system file
- special attributes

A file system uses the tree concept to present the files to the user. The system organizes files into a directory, which can also be referred to as a folder. The file system presents the filename, the MAC date timestamps, and, depending on the operating system, the security permissions. None of these items are part of the file's content. These are metadata, a term which means "data about data." If any of the information points above are changed, such as the file name or the ACL, the file's content will not be affected. For example, say we have a text file named "great novel.txt." If we use the "save as..." function to change the filename to "really great novel.txt," we now have two files with different names whose content is exactly the same. The changing of the metadata has no impact on the contents of the file.

File naming is one of the more important aspects of the file system. Naming allows the user to determine a file's contents. Each system enforces its own file naming requirements. At one time, the file system restricted a filename to meet the "8.3 requirements." The first eight characters made up the filename, followed by a period and then a three-character file extension. With some file systems, uppercase letters differed from a lowercase letter, and in other file systems, a special character could be used. With most modern-day file systems, a filename can have up to 255 characters. The file extension is still in use with modern file systems, but has increased in size to four characters. With Microsoft Windows, the file extension identifies which application can open the file. For example, a .docx can be opened by a word processing document (e.g., Microsoft Word). In other file systems, the file type is not identified by the file extension, but by the file signature within the file itself.

File types have been standardized and possess unique file signatures. The file system can use the file signature to identify the file type. A file signature is a specific set of bytes (two to four bytes long) at the beginning of the file. For example, a .pdf file has a file signature of $x/25\ x/50\ x/44\ x/46$ which is represented in the American standard code for information interchange (ASCII) as %PDF. A user can easily change the file extension to hide incriminating evidence, but changing the file signature can be more difficult.

FAT File System

The file allocation table (FAT) file system has several versions and has been in use since the late 1970s (File allocation table, n.d.). It is one of the "universal" file systems that most operating systems can read.

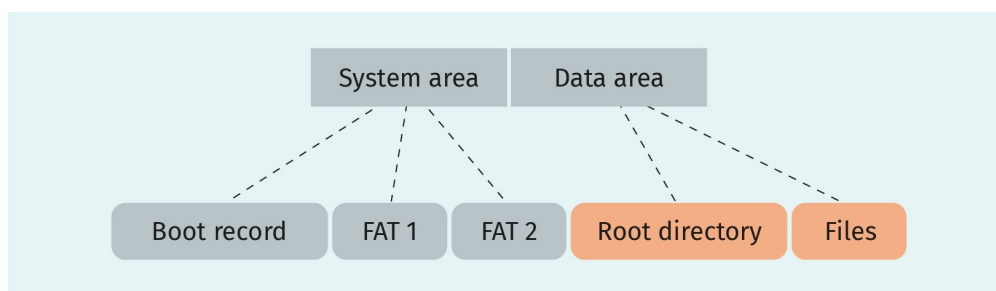
The FAT file system has had many iterations over time:

- FAT 12 uses 12 bits to address the available clusters. Using 12 bits limited the file system to a maximum size of 4096 clusters.
- FAT 16 uses 16 bits to address the available clusters. Using 16 bits limited the file system to a maximum size of 65,536 clusters.
- Virtual FAT (VFAT), introduced with Windows 95, added the virtual file allocation table and the long filename (LFN), and additional timestamps. With 16 bits, the file system remained limited to a maximum size of 65,536 clusters.
- FAT 32 uses 28 bits to address available clusters. Using 28 bits allows for a maximum size of 4,294,967,296 clusters. Microsoft implemented restrictions to limit the volume size to 32 GB, with a maximum file size of 4 GB. This system is still in use today and found on most removable devices.

The FAT file system consists of two areas:

1. System area (volume boot record and the fat tables)
2. Data area (root directory and files)

Figure 24: FAT Areas



Source: Created on behalf of IU (2022).

In the system area, we have the boot record at logical sector 0 (LS 0). This is the first sector within the partition boundaries. The system creates the boot record during the partitioning process, which contains information about volume and boot code. The boot code is used to continue the boot process. The backup boot record is stored at logical sector 6 (LS 6). The beginning and end of the boot sector are depicted in the following figure.

Figure 25: FAT Boot Sector

00010000	EB 58 90 4D 53 44 4F 53	35 2E 30 00 02 08 2A 20	èX.MSDOS5.0...*
00010010	02 00 00 00 00 F8 00 00	3F 00 FF 00 80 00 00 00s...?..ÿ.....
00010020	00 E8 3F 00 EB 0F 00 00	00 00 00 00 02 00 00 00	..è?.è.....
00010030	01 00 06 00 00 00 00 00	00 00 00 00 00 00 00 00
00010040	80 00 29 BD A2 5F E0 4E	4F 20 4E 41 4D 45 20 20	..)¼C_àNO NAME
00010050	20 20 46 41 54 33 32 20	20 20 33 C9 8E D1 BC F4	FAT32 3É.Ñ4é
...			
000101A0	00 00 00 00 00 00 00 00	00 00 00 00 0D 0A 44 69Di
000101B0	73 6B 20 65 72 72 6F 72	FF 0D 0A 50 72 65 73 73	sk errorÿ..Press
000101C0	20 61 6E 79 20 6B 65 79	20 74 6F 20 72 65 73 74	any key to rest
000101D0	61 72 74 0D 0A 00 00 00	00 00 00 00 00 00 00 00	art.....
000101E0	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00
000101F0	00 00 00 00 00 00 00 00	AC 01 B9 01 00 00 55 AA'.....J*

Source: Created on behalf of IU (2022).

In the following figure, the decoded values (along with their offsets in the boot record) are shown. At the start of the FAT boot sector, we find executable instructions for an x86 CPU that occupy the first 3 bytes of the boot sector. These allow the system to skip the bytes that are not executable. Next is the 8 byte “OEM ID,” which identifies the operating system that formatted the volume. The bytes per sector and sectors per cluster are listed here. In this example, there are 512 bytes per sector, with eight sectors per cluster. The serial number of the volume and volume label can also be found at x/43 and x/47.

Figure 26: Map of the Boot Sector

Name	Offset	Value	Copy Value
JMP instruction	000	EB 58 90	33 C0 8E
OEM ID	003	MSDOS5.0	Ð¼
▼ BIOS Parameter Block	00B		
Bytes per sector	00B	512	190
Sectors per cluster	00D	8	124
Reserved sectors	00E	8,234	191
Number of FATs	010	2	6
(unused)	011	00 00	B9 00
(unused)	013	00 00	02 FC
Media descriptor	015	0xF8	0xF3
(unused)	016	00 00	A4 50
Sectors per track	018	63	7,272
Number of heads	01A	255	51,974
Hidden sectors	01C	128	309,755
Total sectors	020	4,188,160	2,147,991,229
Sectors per FAT	024	4,075	2,080,374,910
Extended flags	028	0	3,851
Version	02A	0	3,717
Root cluster	02C	<u>2</u>	<u>281,379,585</u>
System Information sector	030	1	61,922
Backup Boot sector	032	<u>6</u>	<u>6,349</u>
(reserved)	034	00 00 00 00 00 00 0...	88 56 00 55 C6 46
▼ Extended BIOS Parameter Bl...	040		
Physical drive	040	128	180
Reserved	041	0	65
Extended signature	042	41	187
Serial number	043	BD A2 5F E0	AA 55 CD 13
Volume label	047	NO NAME]r. úUªu.÷Á
File system	052	FAT32	.
Bootstrap code	05A	33 C9 8E D1 BC F4...	60 80 7E 10 00 74 2
Signature (55 AA)	1FE	55 AA	55 AA

Source: Created on behalf of IU (2022).

The file allocation table immediately follows the volume boot record. There are two file allocation tables (FAT1 and FAT2), with FAT2 duplicating FAT1.

The purpose of the file allocation table is to track available and occupied clusters and the metadata of the stored files. Each entry in the FAT is made up of 4 bytes (32 bits). The following entries are used to represent the cluster's current status:

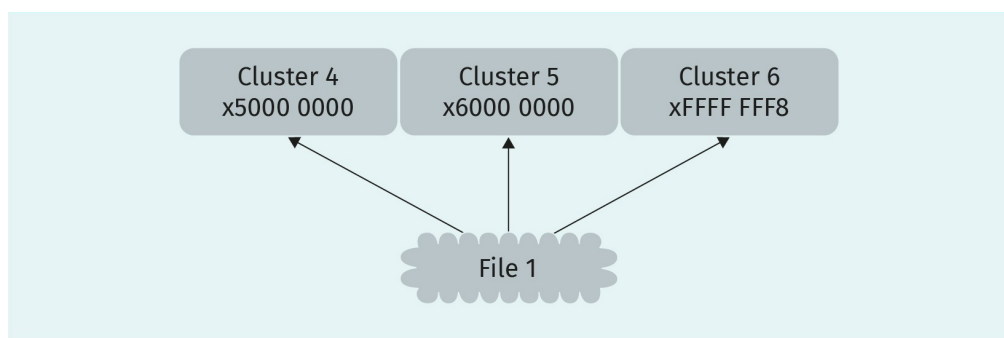
- unallocated (x0000 0000)
- allocated, which is the next cluster used by the file (e.g., cluster 7 is represented as x0700 x0000)

- allocated (EOF), which is the last cluster used by the file (xFFFF FFF8 or xFFFF FF0F)
- bad cluster, which is not available for use (xFFFF xFFF7)

As a reminder, a cluster is the smallest allocation unit the file system can address. A sector is the smallest allocation unit on the disk. A cluster comprises one or more sectors. It is very easy to get confused if these terms are mixed. As users add files to the data area, the system updates the file allocation table. A file may occupy one or more clusters, and the occupied clusters need not be sequential. We refer to this situation as fragmentation; the contents of a file can be in non-continuous physical locations on the disk.

The following figure shows a single file occupying three clusters. The file occupies clusters four, five, and six. The figure shows the hexadecimal values, and demonstrates that cluster four is pointing to cluster five, cluster five is pointing to cluster six, and cluster six is the hexadecimal value for the end of the file (EOF).

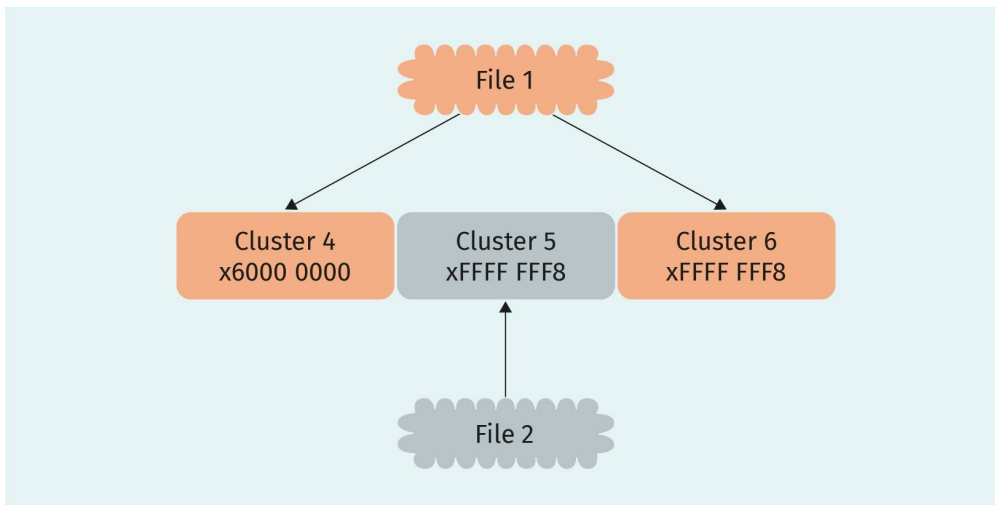
Figure 27: File and Clusters



Source: Created on behalf of IU (2022).

In the following figure, we see a similar representation of the file allocation table. There are two files, with file number one occupying clusters four and six. We can see that cluster four is pointing to the next cluster containing the file data (cluster six). This is an example of file fragmentation. The cluster boundaries of cluster five contain file two. The cluster entry for cluster five will not point to another cluster. This is the same for cluster six, as it is the end of file one.

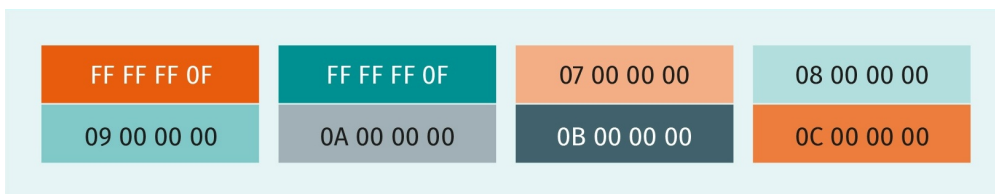
Figure 28: Fragmented File



Source: Created on behalf of IU (2022).

The following figure depicts eight entries in the file allocation table in a hexadecimal format, starting at the top left and moving right. The first two cluster entries show two files, one in each cluster (FF FF FF 0F). With the third entry (07 00 00 00), the pointer is set to cluster seven. Cluster seven (08 00 00 00) is then pointing to cluster eight (09 00 00 00) and will do so until it reaches the end of the file (cluster nine has the value 0A 00 00 00). Depending on the file size, there may be quite a few entries for a single file.

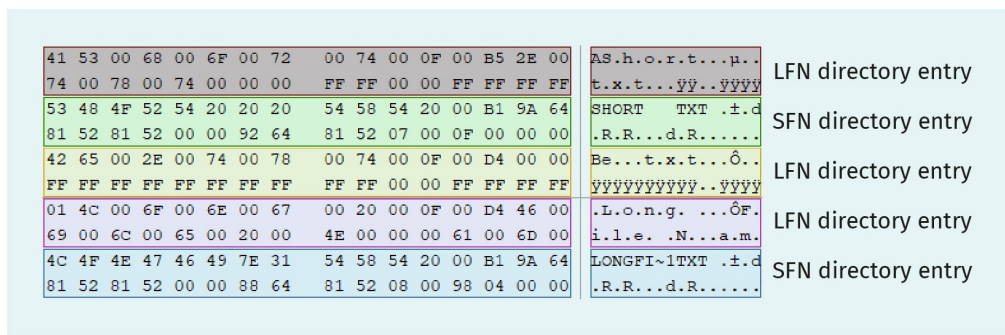
Figure 29: FAT Entry



Source: Created on behalf of IU (2022).

The root directory is in the data area. In prior versions, when the root directory was in the system area, this led to growth issues. The root directory did not have sufficient space to grow to work with larger capacity devices. A critical component of the root directory is the directory entry. If there is a file, directory, or sub-directory, there will be a corresponding directory entry. Each directory entry is 32 bytes in size and tracks the filename, starting cluster, and cluster length.

Figure 30: Directory Entry



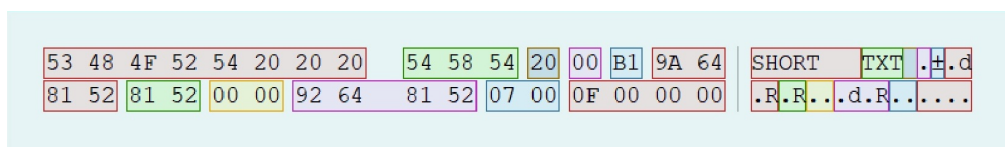
Source: Created on behalf of IU (2022).

In the following figure, we see a root directory with multiple directory entries for two different files: Short.txt and LongFileName.txt. The file system will create at least two directory entries for each file, a short file name (SFN) and a long file name (LFN). The file system bases the number of LFN directory entries on the number of characters in the file name. The LFN is stored in Unicode, which means two bytes are used per character in the file name, and the file system uses only thirteen bytes for the file name in each directory entry.

Short File Names

The following figure depicts the directory entry layout and a SFN directory entry. The SFN directory entry has the sub-entries highlighted to help decode it. If the first byte is xE5, the file system will consider that entry to be deleted and available for reuse. The file system will not change the rest of the directory entry until the file system has a new file for it.

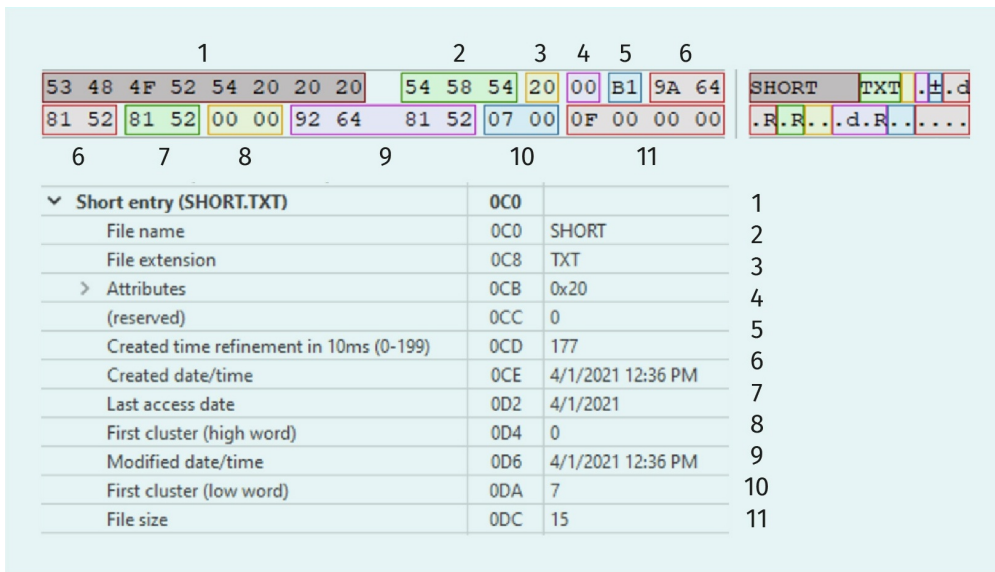
Figure 31: SFN Entry



Source: Created on behalf of IU (2022).

The offsets for the SFN directory entry for the file Short.txt are shown in the following figure.

Figure 32: Directory Map for SFN Entry



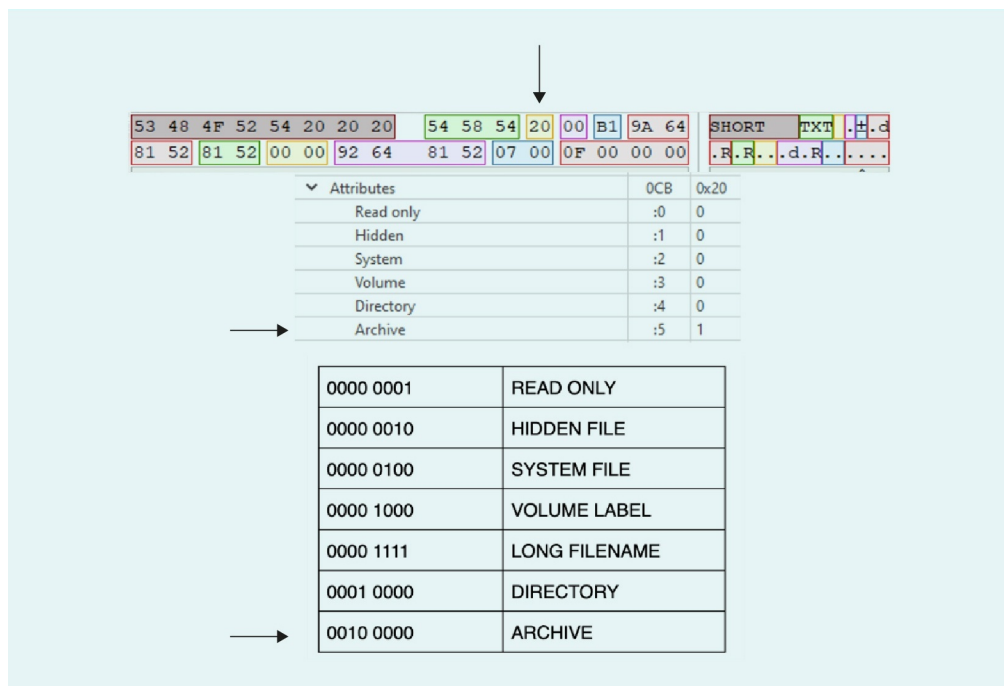
Source: Created on behalf of IU (2022).

The short file name must conform to the following specifications:

- The file name must be one to eight characters in length (if less than eight characters, then the name will be padded with an x20).
- Zero to three characters must be given for the file extension (if less than three characters, then the extension will be padded with x20).
- No spaces are allowed.
- The following characters are not permitted: “ + * , . / : ; < = > ? [\] | ”.

The file system will always store the directory entry in uppercase. The attribute byte (offset x0B) is a “packed byte,” which means the different values have different meanings (see item number three in the previous figure).

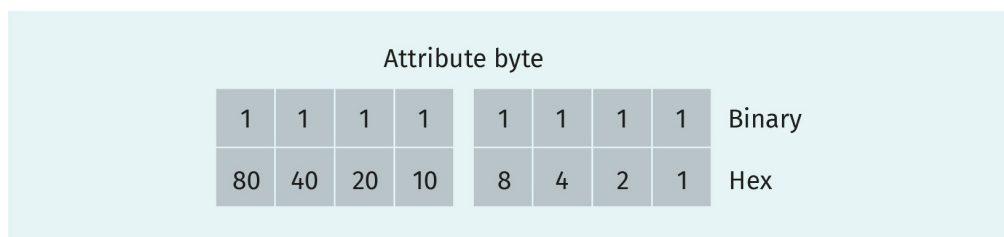
Figure 33: Packed Byte Map for SFN



Source: Created on behalf of IU (2022).

The file system combines the bit values in the attribute flag. The combined value reflects the bit combinations. For example, if a file had the READ-ONLY and the HIDDEN flags, that would give us a binary value of 0000 0011. When that value is converted to hexadecimal, it converts to x03. In the following figure, you can see how the binary values are converted into hexadecimal values found in the directory entry. When we look at this figure, offset x0B has the value x20. x20 is converted to 0010 0000. When compared to the matching value in the figure below, it shows that the file is an archive.

Figure 34: Attribute Byte Map



Source: Created on behalf of IU (2022).

Long File Names

When dealing with a long file name, the file system generates an alias that matches the short file name standard. The file system uses the following format. The first three characters after the file extension dot become the extension; the first six characters are converted to uppercase and used for the SFN standard; a ~ (tilde) and a 1 is added, the latter of which will increase incrementally if there are additional files with the same SFN.

Figure 35: LFN and SFN Directory Entries

42 65 00 2E 00 74 00 78	00 74 00 0F 00 D4 00 00	FF FF FF FF FF FF FF FF	FF FF 00 00 FF FF FF FF	Be...t.x.t...ô..	LFN directory entry
01 4C 00 6F 00 6E 00 67	00 20 00 0F 00 D4 46 00	69 00 6C 00 65 00 20 00	4E 00 00 00 61 00 6D 00	.L.o.n.g. ...ôF.	
4C 4F 4E 47 46 49 7E 31	54 58 54 20 00 B1 9A 64	81 52 81 52 00 00 88 64	81 52 08 00 98 04 00 00	LONGFI~1TXT .t.d	SFN directory entry
				.R.R...d.R.....	

Source: Created on behalf of IU (2022).

The figure above is a directory entry for a file with the LFN “Long File Name.txt.” The file system creates two additional directory entries (depending on the size of the LFN, more entries may be used) after the SFN directory entry. The SFN directory entry is listed first, with the LFN directory entries placed on top. The first byte of the additional directory entries is the sequence byte. The right **nibble** of the byte is the sequence number. This is shown in field one of the directory entry depicted below.

Nibble
This is a four-bit aggregation (or half an octet).

Figure 36: LFN Entry 1

1	2	3	4	5	6	
01	4C 00 6F 00 6E 00 67	00 20 00	0F	00	D4 46 00	.L.o.n.g. ...ôF.
	69 00 6C 00 65 00 20 00	4E 00	00 00	61 00	6D 00	i.l.e. .N...a.m.
	6	7		8		

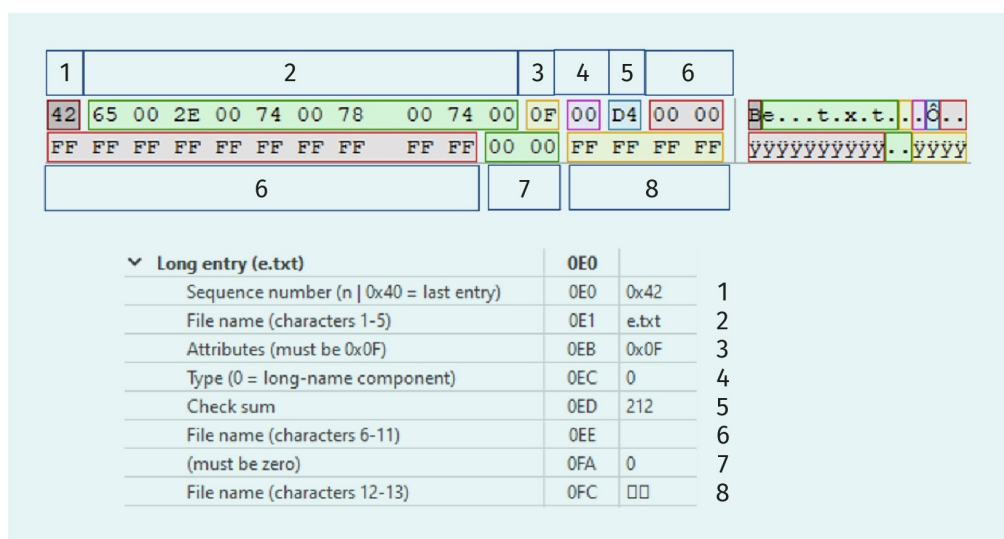
Long entry (Long File Nam)		100	
Sequence number (n 0x40 = last entry)		100	0x01 1
File name (characters 1-5)		101	Long 2
Attributes (must be 0x0F)		10B	0x0F 3
Type (0 = long-name component)		10C	0 4
Check sum		10D	212 5
File name (characters 6-11)		10E	File N 6
(must be zero)		11A	0 7
File name (characters 12-13)		11C	am 8

Source: Created on behalf of IU (2022).

The directory entry above the short file name entry has a hexadecimal value of $\times 01$. The value of 1 tells us that this is the first in the sequence of additional directory entries. The next ten bytes are the first five Unicode characters for the file name “Long.” The attribute byte has a value of $\times 0F$, which indicates that the entry is for an LFN. The checksum is generated from the SFN. The file name continues to be spelled out in Unicode for twelve bytes and then four bytes. However, the file name “Long File Name” still has not been completely spelled out, so the file system must create another directory entry.

At the second directory entry, shown below, there is a value of $\times 42$. Using the right nibble value (2) indicates that this is the second directory entry for this long file name file. The left nibble (4) means that this is the last directory entry for this file.

Figure 37: LFN Entry 2



Source: Created on behalf of IU (2022).

The final portion of the file name, “e.txt,” is listed in the directory entry with the unused character space filled with $\times FF$.

NTFS File System

The new technology file system (NTFS) is the default file system for current Microsoft Windows operating systems. Microsoft initially designed NTFS for server environments, but were soon forced to use it as a replacement for FAT32, which proved unable to address increasing drive capacity.

NTFS is much more complicated than the FAT file system, but its overall purposes remain the same: tracking the metadata of a file, tracking allocated clusters, and tracking unallocated clusters. The following table lists the files at the heart of the NTFS file system (Microsoft, 2009).

Table 5: NTFS Components

\$MFT	Describes all files on the volume, including file names, timestamps, stream names, lists of cluster numbers where data streams reside, indexes, security identifiers, and file attributes
\$MFTMirr	Duplicate of the first vital entries of \$MFT, usually four entries (4 kb)
\$LogFile	Contains transaction log of file system metadata changes
\$Volume	Contains information about the volume, i.e., the volume object identifier, label, flags, and file system version
\$AttrDef	A table of \$MFT attributes that associates numeric identifiers with names
\$(Root file name index)	The root folder
\$Bitmap	Tracks the allocation status of all clusters in the partition
\$Boot	Volume boot record
\$BadClus	A file that contains all the clusters that are marked as having bad sectors
\$Secure	Access control list database
\$UpCase	Converts lowercase characters into Unicode by storing an uppercase version of all Unicode characters
\$Extend	Contains various optional extensions, such as \$Quota, \$ObjId, \$Reparse, and \$UsnJrnl

Source: Microsoft (2009).

The master boot record (MBR) or the GUID partition table (GPT) record will identify the partition parameters and the file system used to format the partition, as shown below.

Figure 38: NTFS MBR

Offset	00 01 02 03 04 05 06 07	08 09 10 11 12 13 14 15	ASCII
0000000000	EB 52 90 4E 54 46 53 20	20 20 20 00 02 08 00 00	NTFS
0000000016	00 00 00 00 00 00 FB 00 00	3F 00 FF 00 80 00 00 00	...?..y.....
0000000032	00 00 00 00 80 00 80 00	FF E7 3F 00 00 00 00 00	...ÿç?.....
0000000048	AA A9 02 00 00 00 00 00	02 00 00 00 00 00 00 00	*@.....
0000000064	F6 00 00 00 01 00 00 00	66 20 92 02 61 92 02 7C	ö.....f..a..l
0000000080	00 00 00 00 FA 33 C0 8E	D0 BC 00 7C FB 68 C0 07	...ú3À.Ð%. ûhÀ.

Source: Created on behalf of IU (2022).

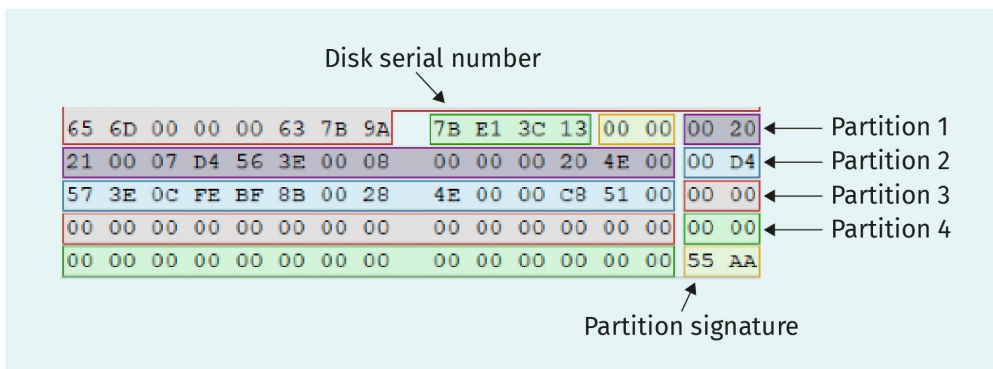
Figure 39: Map of NTFS MBR

Name	Offset	Value	Copy Value
JMP instruction	000	EB 52 90	EB 52 90
OEM ID	003	NTFS	NTFS
▼ BIOS Parameter Block	011		
Bytes per sector	011	512	512
Sectors per cluster	013	8	8
Reserved sectors (always zero)	014	0	0
(unused)	016	00 00 00	00 00 00
(unused)	019	00 00	00 00
Media descriptor	021	248	248
(unused)	022	00 00	00 00
Sectors per track	024	63	63
Number of heads	026	255	255
Hidden sectors	028	128	128
(unused)	032	00 00 00 00	00 00 00 00
Signature	036	80 00 80 00	80 00 80 00
Total sectors	040	4,188,159	4,188,159
\$MFT cluster number	048	174,506	174,506
\$MFTMirr cluster number	056	2	2
Clusters per File Record Se...	064	246	246
Clusters per Index Block	068	1	1
Volume serial number	072	66 20 92 02 ...	66 20 92 02 ...
Checksum	080	0	0
Bootstrap code	084	FA 33 C0 8E...	FA 33 C0 8E...
Signature (55 AA)	510	55 AA	55 AA

Source: Created on behalf of IU (2022).

In the following figure, the partition tables are identified. There are partition tables for four primary partitions; this disk has two partitions. Each partition table is sixteen bytes in length. At the end of the partition tables is the end-of-partition signature x55 xAA.

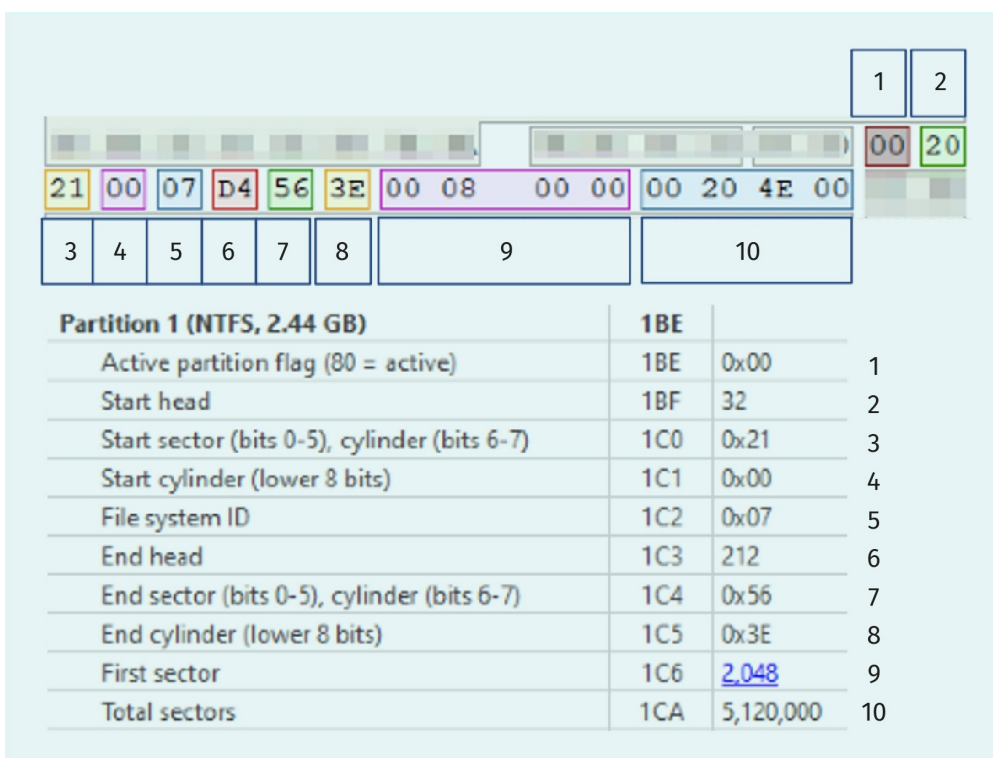
Figure 40: Map of NTFS Partition Table 1



Source: Created on behalf of IU (2022).

The first partition table is laid out in the following figure. The first byte indicates whether the partition is bootable. This partition has a value of 00, which indicates that it is not bootable (a value of 80 indicates a bootable partition). The partition table then lists the starting and total number of sectors and the file system ID (item 5 in the figure below). The partition has a value of x07, which indicates that this is an NTFS-formatted partition.

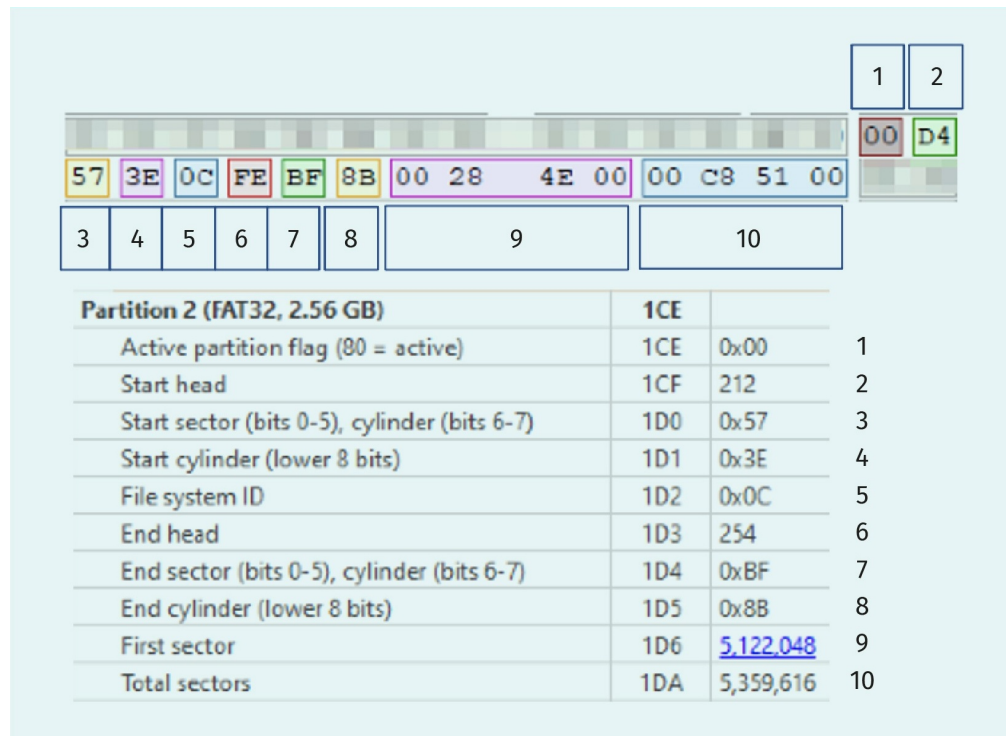
Figure 41: Map of NTFS Partition Table 2



Source: Created on behalf of IU (2022).

The figure below shows the second partition table. This partition is also not bootable and has a file system ID of x0C, which indicates that the partition is formatted as a FAT32 partition.

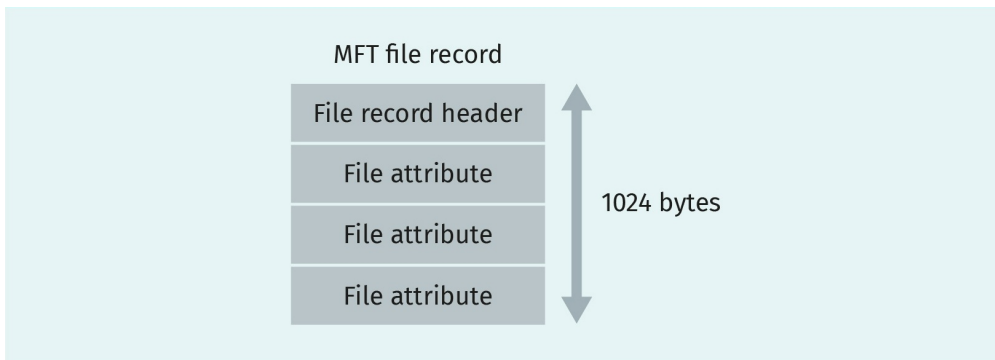
Figure 42: Map of FAT Partition Table



Source: Created on behalf of IU (2022).

Unlike with a FAT-formatted partition, an NTFS-formatted partition has no “system” or “data” areas. In NTFS, everything is considered to be a “file,” including the system data. The information in the volume boot record (VBR) comes from the file \$BOOT. \$MFT is the most critical system file in the NTFS file system. The \$MFT tracks all the files in the volume, including itself. The file system accomplishes this by using file entries in the \$MFT, creating a file record.

Figure 43: MFT File Record



Source: Created on behalf of IU (2022).

Each file record is uniquely numbered and does not exceed 1024 bytes in size. A file record consists of a file record header and File Attributes. Each file record starts with a header with the ASCII text “FILE” and has an end-of-file marker of hexadecimal FF FF FF FF. As the user adds files to the volume, a new file record is created. If the user deletes a file, the file system zeroes out the file record. Once the file record has been zeroed out, it is now available to be reused. The \$MFT looks for an empty file record and uses it before creating a new record. The file system can quickly reuse the file record, overwriting the previous data.

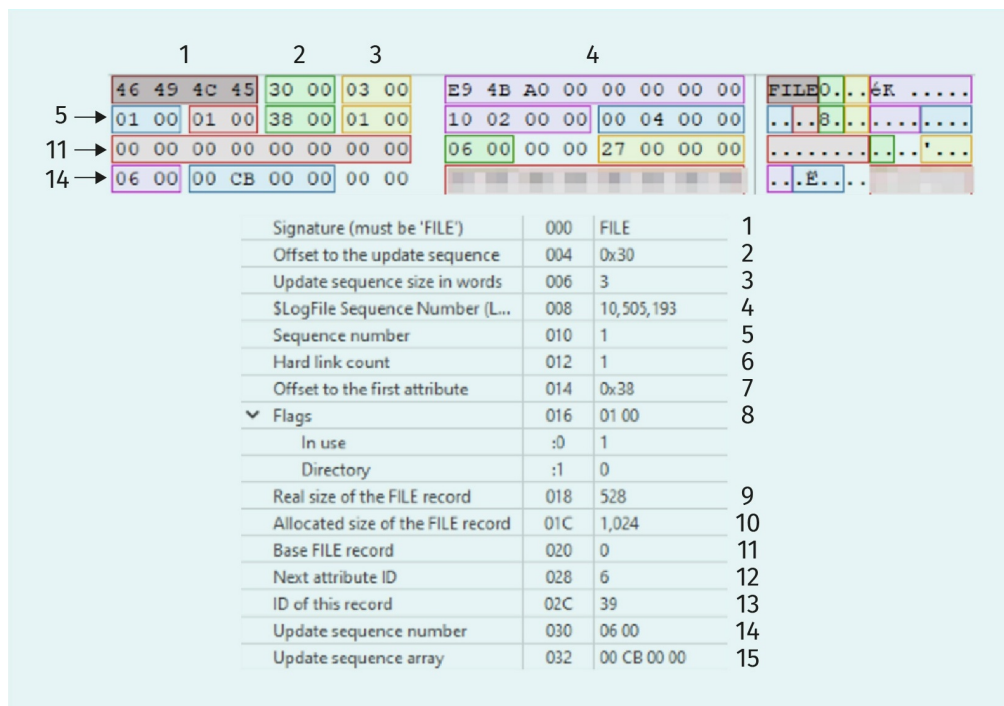
The following figure shows a file record header. The file header begins with the ASCII values of FILE (x/46 49 4C 45). If the record had been corrupted, you would see the ASCII value BAAD. The file header for the file record is 56 bytes in size. The data structure of the file record header includes the following attributes.

Table 6: Attributes of the Data Structure Belonging to the File Record Header

\$LogFile sequence number(item 4)	This is a number that identifies the records written to the \$LogFile.
Sequence number (item 5)	This is the number of times a file record entry has been used. When the file record is created, the sequence value is set to x01. This value increases when the file record is marked as unused.
Allocation status flags (item 8)	This has a value of 0 (in use/allocated) or 1 (directory).
Base file record (item 11)	There may be times when a single file record is not enough to contain all the information about a single file (e.g., when a file becomes fragmented). The first record is designated as the “base record.” The remaining records are known as “extension records.” An extension record has the \$MFT number of the base record in its header.
Next attribute ID (item 12)	This is the number of attributes in the file record (including deleted attributes).

Source: Created on behalf of IU (2022).

Figure 44: File Record Header



Source: Created on behalf of IU (2022).

The file record also contains defined data blocks called file attributes which store specific types of information. The following table shows common file attributes encountered in almost every record (Microsoft, 2009).

Table 7: File Attributes

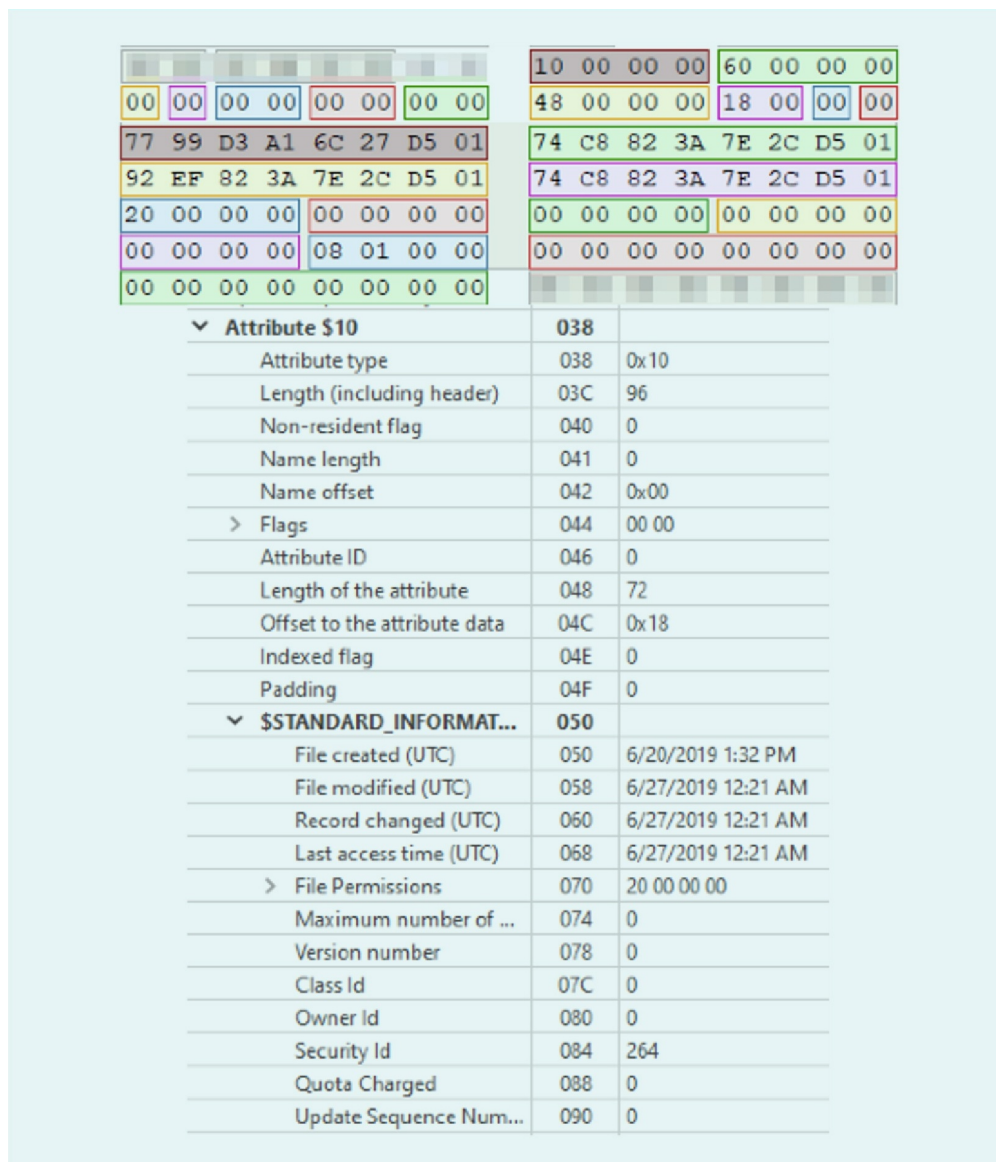
\$Standard_Information - 0x10	Includes information such as timestamp and link count
\$Attribute list - 0x20	Lists the location of all attribute records that do not fit in the \$MFT record
\$File_Name - 0x30	Repeatable attribute for both long and short file names; the long name of the file can be up to 255 Unicode characters in length; the short name is the 8.3-compliant, case-insensitive name for the file
\$Security_descriptor - 0x50	Describes who owns the file and who can access it
\$Data - 0x80	Contains file data: NTFS allows multiple data attributes per file; each file type has one unnamed data attribute and can have one or more named data attributes

Source: Microsoft (2009).

\$Standard_Information (x10) attribute

The file attributes follow the file header and contain information about the file and sometimes the file itself. The following figure depicts a file attribute. The first four bytes indicate the attribute type. The \$10 Standard_Information attribute contains general information, flags, accessed, write, and create times. Owner and security ID are also found here. The hexadecimal header x/10 00 00 00 identifies this attribute as the \$Standard_Information attribute.

Figure 45: Standard Information Attribute and Map

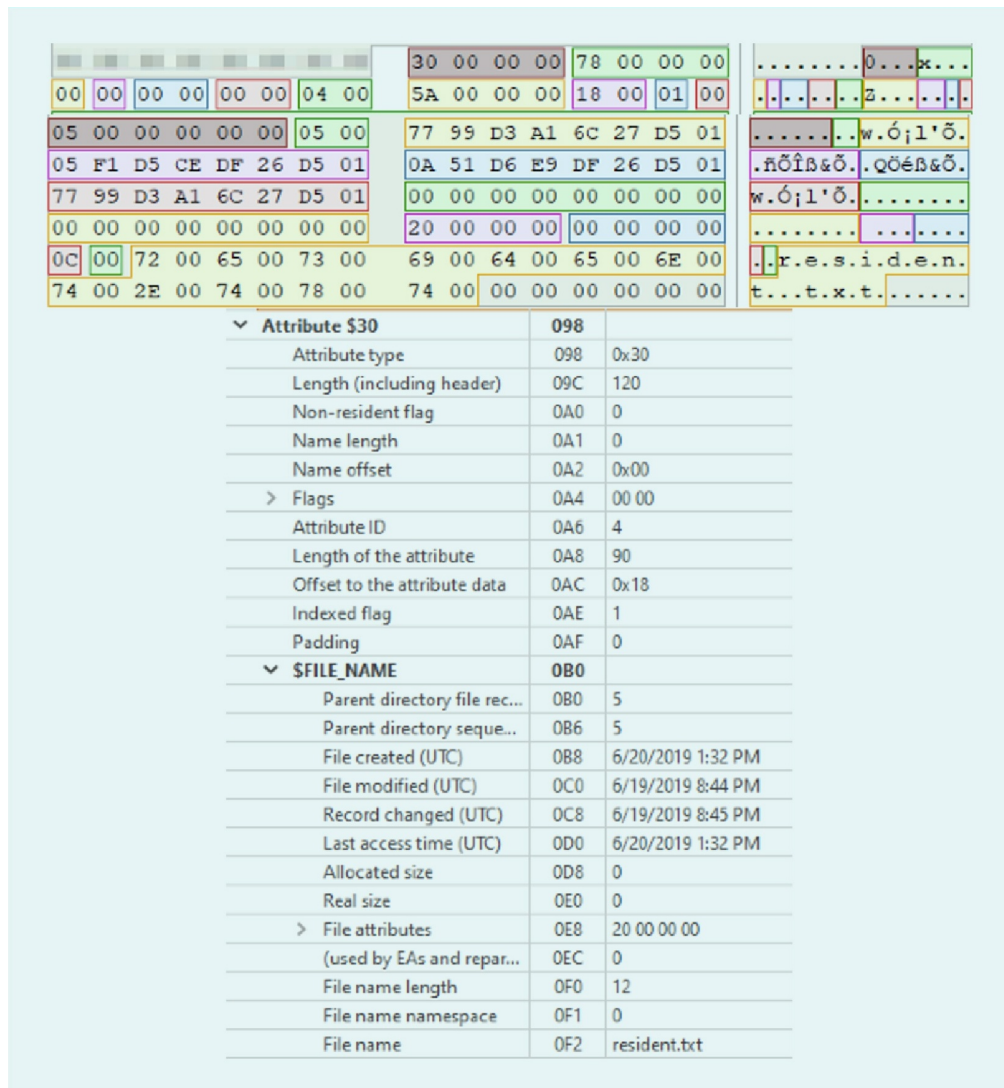


Source: Created on behalf of IU (2022).

\$File_Name (x30) attribute

The \$30 File_Name attribute stores the file's name. This attribute is always resident and enforces a maximum filename length of 255 Unicode characters. The hexadecimal header x/ 30 00 00 00 identifies it as the \$File_Name attribute. The attribute is shown in the following figure.

Figure 46: File Name Attribute and Map

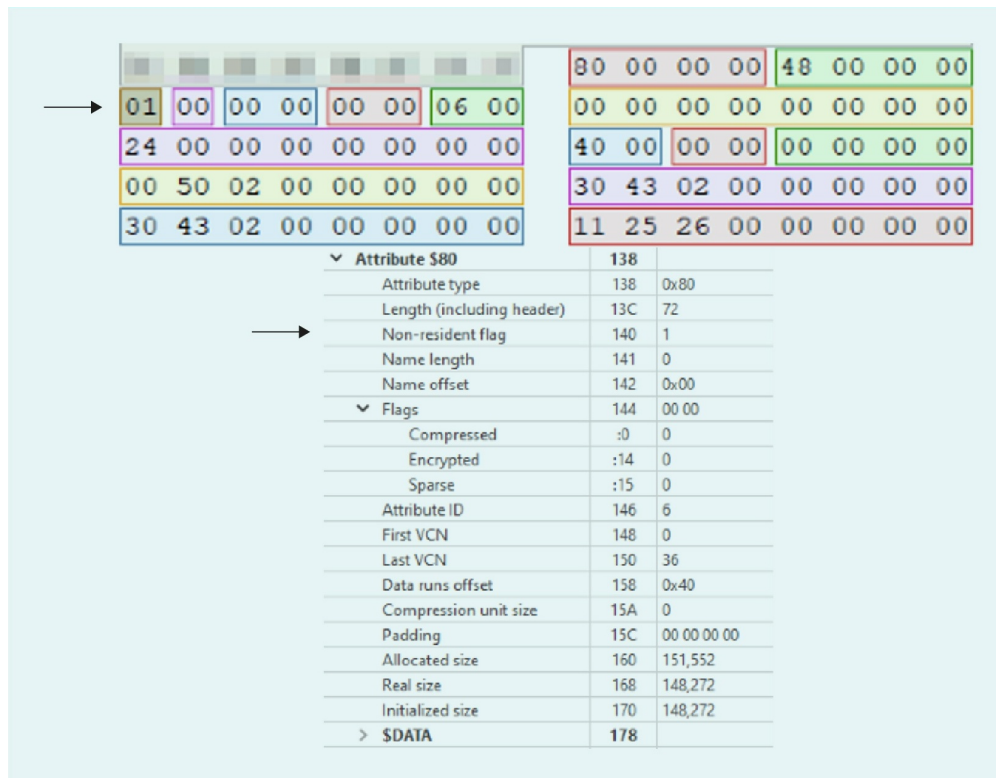


Source: Created on behalf of IU (2022).

\$Data (x80) attribute

The \$Data attribute contains the file contents or pointers to where the file is on the disk. If the file is a resident file, the attribute will contain the file's contents. The file must be smaller than 1024 bytes in size to be resident data. The following figure shows the \$Data attribute for a non-resident file. We know this because the "non-resident" flag has a value of 1.

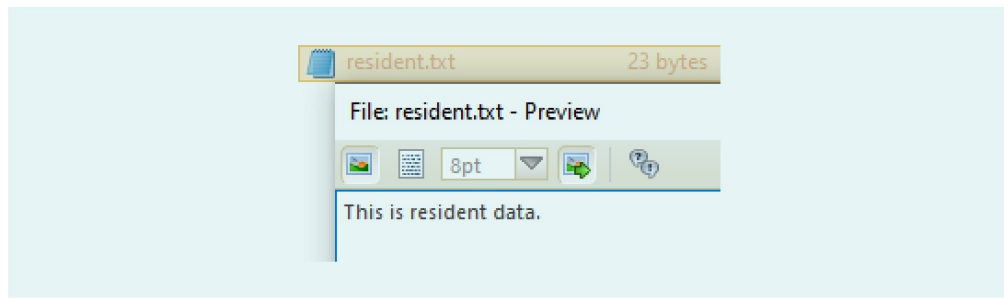
Figure 47: Data Attribute and Map



Source: Created on behalf of IU (2022).

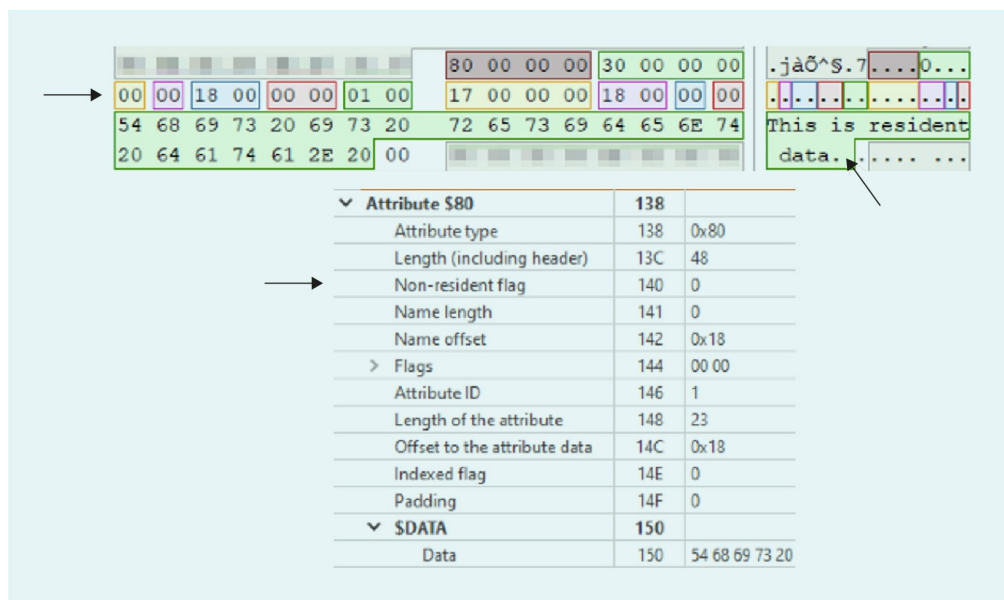
Let's examine another \$Data attribute, 0x80, where the file system is storing the file's contents within the MFT file record itself. When the file's data content fits within the file record, they are called "resident" data. The following figure depicts a text file titled "resident.txt." The contents of the file are a single sentence: "These are resident data." This file is much smaller than the 1024 bytes of the size of the file record.

Figure 48: Resident Data File



Source: Created on behalf of IU (2022).

Figure 49: Resident Data File MFT Entry

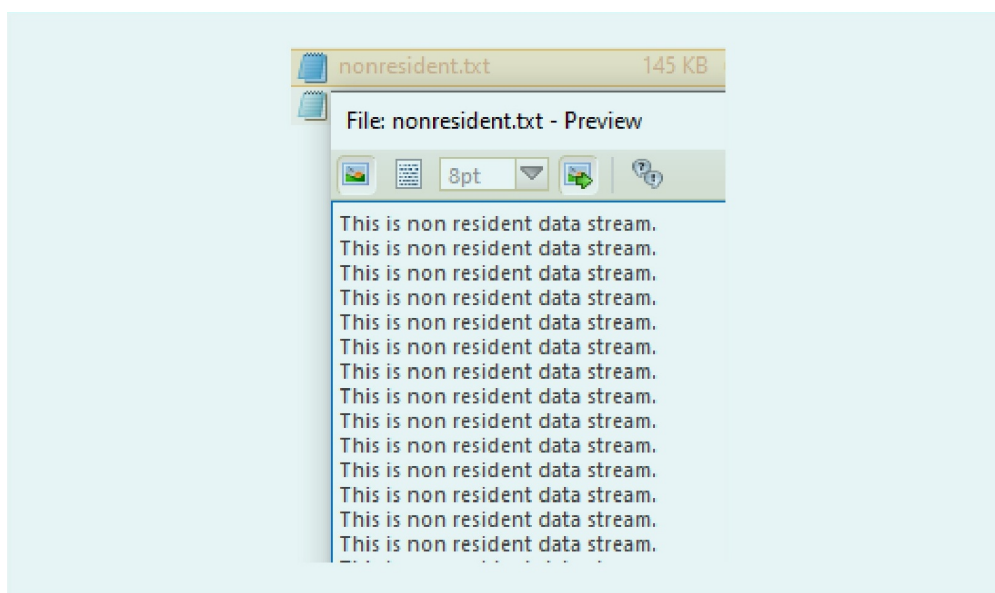


Source: Created on behalf of IU (2022).

The figure above depicts the \$Data attribute of resident.txt. The \$Data attribute has a value of x0 for the non-resident flag, which means the file will be resident in the \$MFT. When we examine the \$Data attribute, we see the contents of the file: “These are resident data.”

Next, we have the file “nonresident.txt,” which is 145KB in size. This is larger than the 1024byte file record.

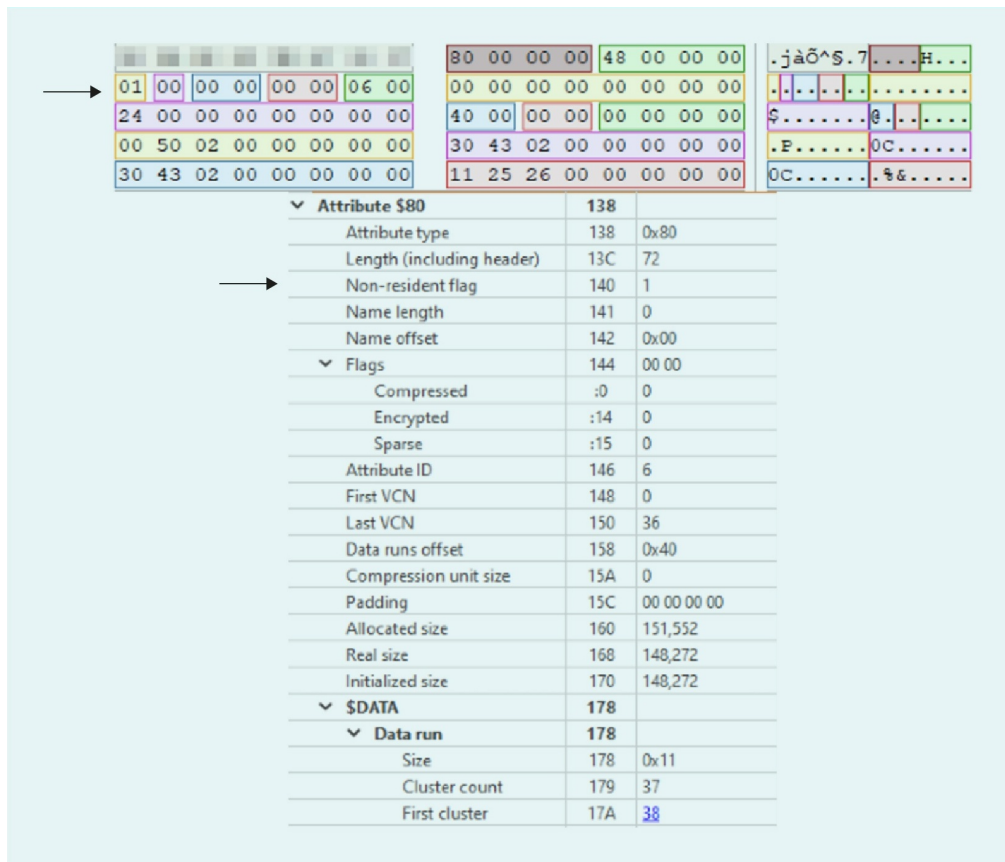
Figure 50: Non-Resident Data File



Source: Created on behalf of IU (2022).

In the \$Data attribute 0x80 of the file, we do not see the file's contents. Instead, there are pointers to the location of the file. This is non-resident content. Once the content of the attribute becomes non-resident, it cannot become resident again. The pointers in the attribute file record are a "run list" for the data run locations of the non-resident data. The following figure shows the \$Data attribute for a non-resident file; the "non-resident" flag has a value of 1.

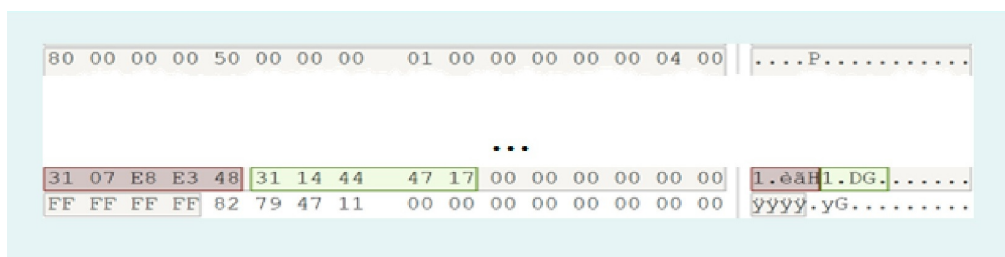
Figure 51: Non-Resident Data File MFT Entry



Source: Created on behalf of IU (2022).

With non-resident data, there can be one or more data runs within the \$Data attribute. Decoding the run list for the data runs can be tricky. In the following figure, a \$Data attribute 0x80 with two data run lists is shown.

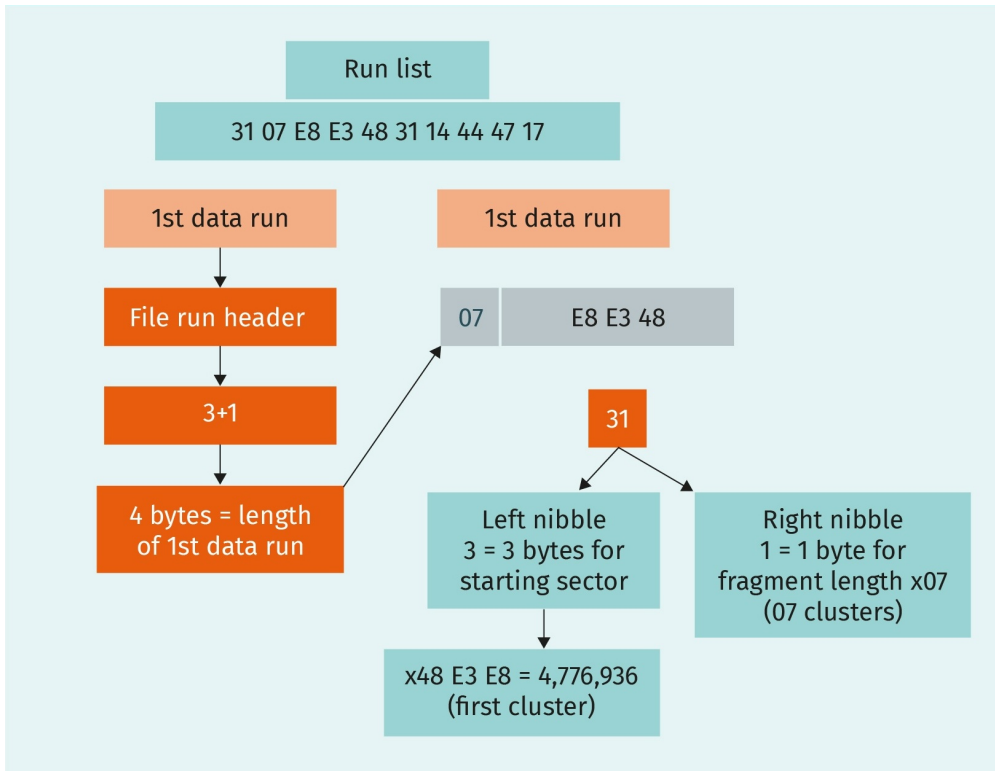
Figure 52: Data Run in Non-Resident Data File MFT Entry



Source: Created on behalf of IU (2022).

When the file is not fragmented, we encounter a single-run list. If the file is fragmented (which is very common), we encounter multiple run lists. Each run list provides information about the starting cluster for each fragment. The two run lists are highlighted in the previous figure and decoded in the following two figures.

Figure 53: First Decoded Data Run



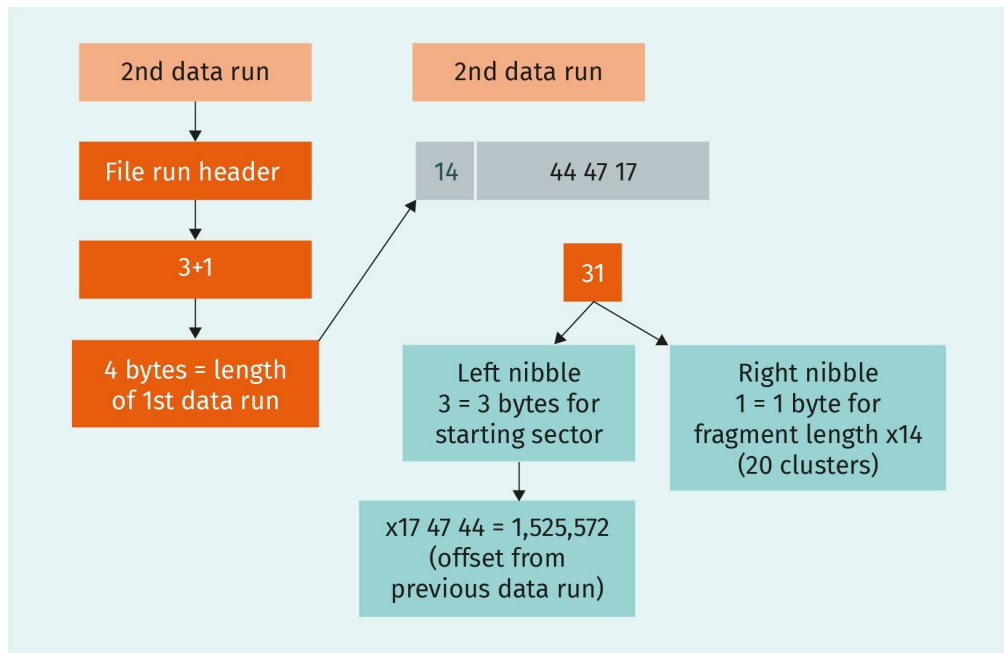
Source: Created on behalf of IU (2022).

The first run list comprises the hexadecimal values 31 07 E8 E3 48. We take the first byte of the header ($\times/31$) and add the left and right nibble ($3 + 1 = 4$). 4 is the number of bytes in the run list entry, i.e., $\times/07\ E8\ E3\ 48$.

The right nibble ($\times/1$) tells us that one byte is being used to represent the number of clusters being used for this fragment. We find a value of $\times/07$ in the length field, which represents seven clusters for this fragment.

The left nibble ($\times/3$) informs us that three bytes ($\times/48\ E3\ E8$) represent the logical starting cluster of the fragment when converted to decimal (the order of the bytes are reversed in order to convert them). At the end of the first run, we have a second run list of $\times/31\ 14\ 44\ 47\ 17$.

Figure 54: Second Decoded Data Run



Source: Created on behalf of IU (2022).

We take the first byte of the header ($x/31$) and add the left and right nibble ($3 + 1 = 4$). 4 is the number of bytes in the run list entry ($x/14 \ 44 \ 47 \ 17$).

The right nibble ($x/1$) tells us that one byte is being used to represent the number of clusters being used for this fragment. We find a value of $x/14$ in the length field, which represents twenty clusters for this fragment. The left nibble ($x/3$) informs us that three bytes ($x/17 \ 47 \ 44$) represent the offset from the previous run list cluster. This process continues until the system reaches $x/ \ 00 \ 00 \ 00 \ 00$.

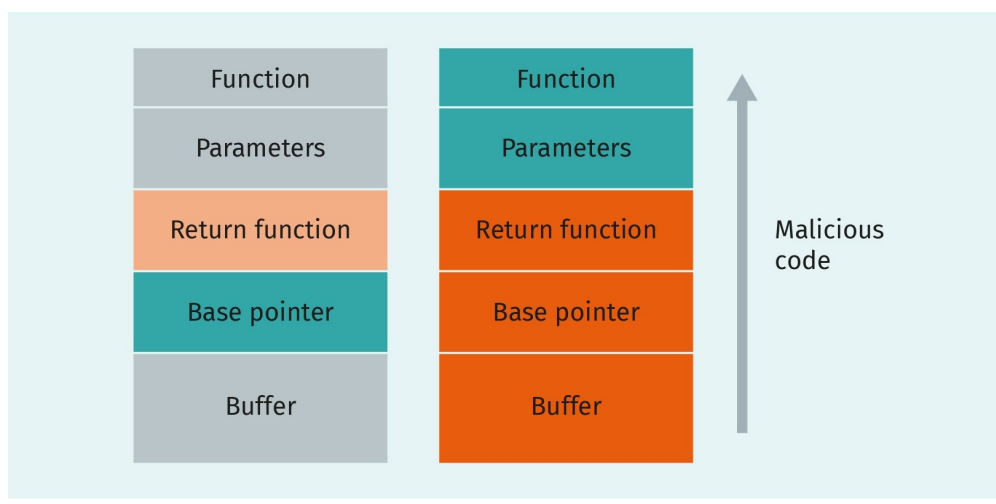
2.5 Common Attacks

As computer systems become increasingly ubiquitous in the modern environment, with e-commerce transactions, governmental data, and private-sector data being generated, it follows that attacks also increase in number. Indeed, the likelihood of attacks grows despite an increase in mitigation efforts. Attackers target the system's memory to create a buffer overflow, such as return-to-libc attacks or malware execution within memory. Thus, the need to protect these data becomes greater every day; what seems to be a secure computer system one day can be invalidated the next. We will now look at some common attacks.

Buffer Overflow

C and C++ are programming languages used on most operating systems. The downside of using C or C++ when using the library function “gets” is that there is no array-bound checking, meaning there is no check performed on the input to ensure that it meets and does not exceed the fixed-size buffer. Once the user inputs their response, any data that exceed the buffer size will overwrite the memory location (or the return address in memory). An attacker can craft a specific input and use the buffer overflow to execute malicious code, as shown in the following figure. The code being executed maintains the privileges of the original process, which can have a considerable impact on the system.

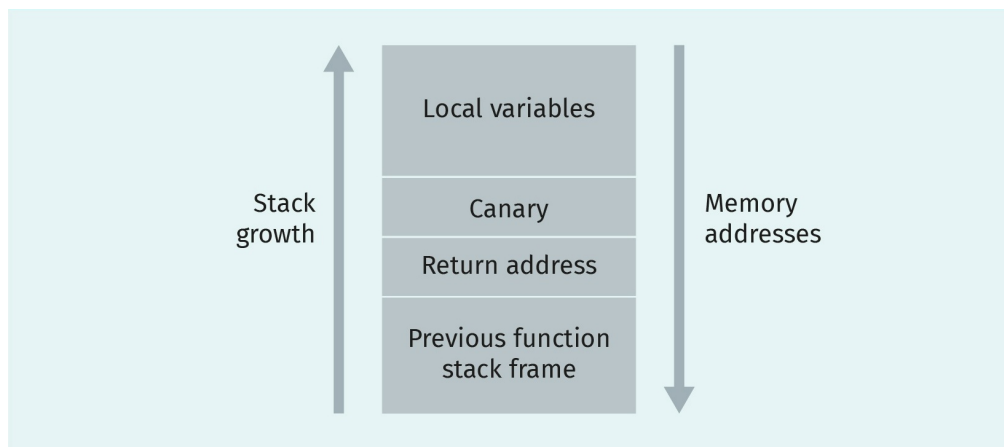
Figure 55: Buffer Overflow



Source: Created on behalf of IU (2022).

Instead of causing a crash when the system examines the memory location, the attacker has crafted a set of instructions that the system will not execute. The instructions will open a shell terminal, which then gives the attacker the ability to execute commands via the terminal. A common defense against this attack is to use “stack canaries,” as depicted in the following figure.

Figure 56: Stack Protection (Canaries)



Source: Created on behalf of IU (2022).

Similar to how coal miners used canaries to identify toxic gas in tunnels, the operating system can create an early warning system. When the code calls for a function call, the compiler will add a canary value to the memory stack. On the return function, there will be a code to verify the canary value. If the canary value has changed, there has been unauthorized access to the system.

Memory Corruption

Memory corruption occurs when the contents of memory are changed in a way that the programmer did not intend. Typically, memory corruption happens due to a programming error. The contents of the corrupted memory are then used by the program, which causes the program to either crash or exhibit strange and unexpected behavior. This type of corruption can be hard to correct because of the delay in the cause and effect. The program might crash an indeterminate amount of time after the memory has been corrupted. Another factor that could lead to a delay in correction is the conditions that cause memory corruption. This makes it hard to reproduce the error while running a diagnostic operation.

Code injection is an attack that causes the victim system to process invalid data. The attacker “injects” code into a system, which causes the system to execute the attacker’s malicious instructions. Code injection vulnerabilities are found in a variety of sources, such as databases, operating system (OS) commands, or Extensible Markup Language (XML) parsers (an interface for working with XML documents). There have been incidents of code injection where the attacker has successfully taken over the victim system, and others where the attack resulted in data loss or a denial of access attack.

A control flow hijacking attack will cause data structures to be overwritten, allowing the attacker to take the control of the victim host. In addition to buffer overflow attacks, examples of this type of attack include:

- Integer overflow attacks. This is when an attacker corrupts an arithmetic operation, causing the result to exceed the maximum size of the expected field. An integer overflow occurs when the interpreted value appears to have exceeded the maximum value and starts again at the minimum value.
- Format string attack. This is an attack that occurs when the attacker submits data of an input string and the system treats it as a command by the application. This allows the attacker to execute malicious code or read data stored in memory.

Return-Oriented Programming

Return-oriented programming (ROP) is an attack that bypasses security protections, such as executable space protection and code signing. In 2007, Hovav Shacham presented on the technique, stating that new techniques had been introduced that allow a return-into-libc attack “to be mounted on x86 executables that calls no functions at all” (Abstract section). The attack combined many short instruction sequences to build “gadgets” that “allow arbitrary computation.” The discovery of these instruction sequences was achieved by means of static analysis: “we make use, in an essential way, of the properties of the x86 instruction set” (Shacham, 2007, Abstract section).

The attacker does not use malware or malicious code in this type of attack. Instead, they use the system’s own code against it by changing the return addresses. The attacker interrupts the program control flow by gaining control of the call stack. Once the attacker controls the call stack, they can deploy “gadgets,” which are computer code that is already in memory. The codes are typically subroutines of programs being executed on the system or a shared library. In either case, the attacker includes a return instruction that allows the attacker to link multiple “gadgets.” This process enables the attacker to control the system and carry out further program executions on the system. Some memory regions that are classified by the operating system as non-executable include the stack and heap. Rendering the stack and heap as non-executable can help prevent buffer overflow exploits from being executed. Using address space layout randomization (ASLR) helps reduce this type of attack by randomly using different address spaces for the process’s data areas. If the attacker does not know which data carriers are being used, it is harder for the attacker to execute the malicious code.

Return-to-libc Attack

“libc” is a shorthand for “C standard library,” the standard library of the programming language C. Return-to-libc is a buffer overflow attack on a system that has enabled a non-executable memory stack. A buffer overflow involves the attacker attempting to overflow the buffer with malicious code or meaningless data. This causes the program executing the code to jump to the shellcode in the stack. The attacker inserts a new return address to a location where the attacker can execute the code. libc does not access an executable stack, nor does it need access to the shellcode. The attacker exploits a program, causing it to jump to an existing piece of code via the libc library. The system has already loaded the libc library into the process’s memory space. This allows the library to be responsible for the code execution. Some of the mitigation methods include ASLR, randomizing the address within the stack, randomizing the library protocols, and data execution prevention (DEP) by removing executable memory regions. As operating systems evolve, and

attackers become more sophisticated, new attacks and vulnerabilities are emerging. To help determine if the attacker has compromised the system using a libc attack, detective controls need to be implemented, such as monitoring the operating system's logs.

Rowhammer Bug

Rowhammer is a hardware-based bug. An attacker starts rapidly accessing random access memory (RAM) memory rows, trying to break the memory isolation of the processes. When this happens, it allows the attacker to cause bit flips in adjacent rows of the RAM chips. Google's Project Zero reported that, "when run on a machine vulnerable to the Rowhammer problem, the process was able to induce bit flips in page table entries (PTEs)" (Seaborn & Dullien, 2015, para. 1). This was then used to gain write access to the page table, granting read and write access to all of the physical memory.

Double data rate 3 (DDR3) memory, which is made up of high-density memory cells, is susceptible to a rowhammer attack. The increased row activations cause the voltage to fluctuate, which causes adjacent capacitors to discharge at abnormal rates. These adjacent rows are considered "victim rows," and errors in the memory are created when the capacitors are discharged to lower levels before the system can refresh the rows' contents. This then causes the memory protection protocols to short circuit, allowing the attacker to increase their privileges on the system and execute kernel level code within the system.

Side-Channel Attack

A side-channel attack is an attack based on information gathered about how the system operates. This attack is not based on exploiting a vulnerability, but rather on exploiting the system set up on information learned through reconnaissance. This requires the attacker to have significant technical knowledge about the operation of the system. Attacks of this type include the following:

- cache-based attack. The attacker monitors the victim's system cache on shared systems.
- electromagnetic attack. The attacker monitors electromagnetic radiation to determine which keystrokes are being used or what content is being displayed on screen.
- acoustic cryptanalysis. The audio or sounds being generated by a computer system are monitored, analyzed, and exploited. For example, different RSA keys create different sound patterns; these can be analyzed to identify the key being used.
- **data remanence**. This exploit is used to access the data set after it has been deleted from the system.

Data remanence
This is the residual representation of digital data that remains even after the data are removed or erased.

Deterrence controls focus on isolating the system to prevent surveillance. This can include jamming the audio and electromagnetic frequencies being emitted by the system, or ensuring that the power being used by the system has been conditioned and filtered.



SUMMARY

We have looked at the operating system's internal functions, the purpose of system calls, and how an application accesses the protected resources. Operating systems use the process table to manage processes, memory, and files. The process control block (PCB) contains information about the state of the process, memory allocation, and scheduling information, as well as any information needed to resume the process from a suspended or blocked state. Processes have sub-processes called threads, which are miniature processes that can be created and terminated much faster than a complete process.

The Microsoft Windows operating system uses the registry to store information about the operating system's configuration, users, applications, and hardware devices. Nearly every action taken within the operating system will leave a footprint in the registry. The registry comprises hive files such as SAM, SECURITY, SOFTWARE, and SYSTEM. The default file system for the Windows operating system is the NTFS file system, in which every object is considered a file. NTFS tracks the clusters using a master file table (MFT). The \$MFT file contains file records of the files stored within the file system. The file records are no larger than 1024 bytes and can sometimes include the actual file data. Each file entry comprises a series of attributes: the \$Standard_Information attribute, \$File_Name attribute, and the \$Data attribute.

FAT32 is another common file system used by multiple operating systems. The FAT file system has two distinct areas: system area and data area. The system area contains the file allocation tables and the boot record, while the data area contains the root directory and the files. Short and long file names are found in the directory entries. Some common attacks used against the operating system include the buffer overflow attack and memory corruption. Address space layout randomization (ASLR) helps reduce this type of attack by randomly using different address spaces for the process's data areas. While data execution prevention (DEP) stops a process being executable in memory regions.

LEKTION 3

THE NETWORK STACK

STUDY GOALS

On completion of this unit, you will be able to ...

- explain the need for a communication standard.
- describe the goals of the open systems interconnecting (OSI) model.
- define the functions of the transmission control protocol/internet protocol (TCP/IP) model.
- collect digital evidence.
- understand the issues that mobile devices present.

3. THE NETWORK STACK

Introduction

This unit will focus on computer networks and how they function. Our ability to investigate is based on our knowledge of how a network functions and how an attacker may exploit network vulnerabilities. The popularity of network-connected devices has grown exponentially in recent times; desktops, laptops, mobile phones, refrigerators, light bulbs, speakers, and automobiles are all connected to networks.

The transmission control protocol/internet protocol (TCP/IP) is the protocol used in commercial and consumer networks. An understanding of the layers of the TCP/IP protocol is essential for forensic analysis. The open systems interconnecting (OSI) model is a conceptual model upon which the networking environment is based. TCP, user datagram protocol (UDP), IPv4, IPv6, and address resolution protocol (ARP) are relevant terms and protocols in this context. Encryption is also fundamental to the security of the network. Most users are unaware of the encryption protocols that are deployed on their behalf. We can deploy encryption to protect data at rest and data in motion. This unit will provide an understanding of the encryption methods being used to protect our data.

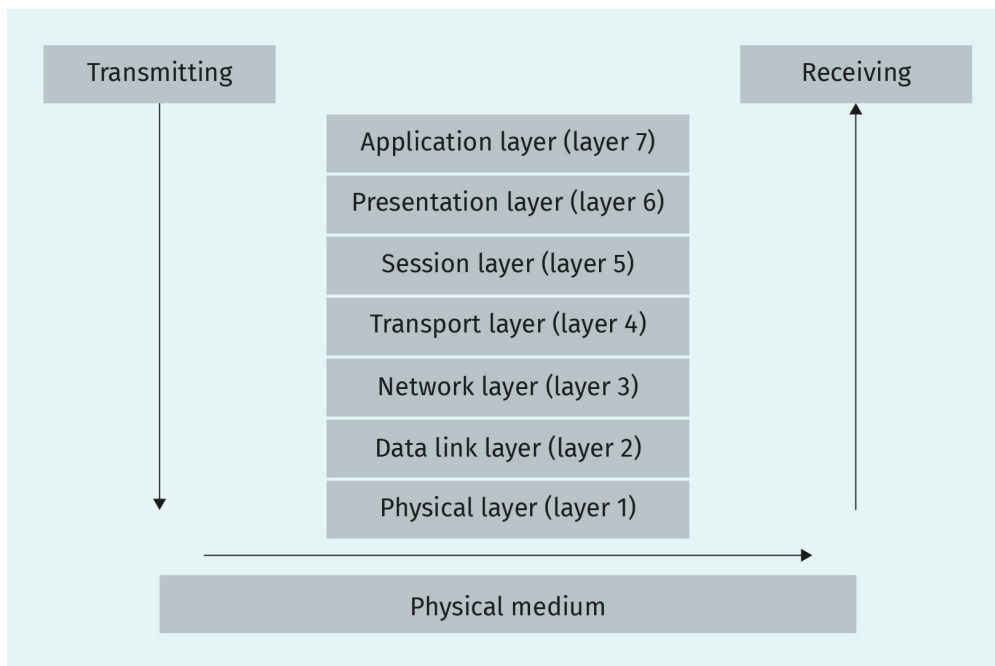
Finally, we will look at common attacks and how they can be mitigated. Not all security efforts take place in the digital realm; we also need to be aware of the security requirements of the physical world. Sometimes, keeping a door locked is enough to protect your servers from an outside physical attack.

3.1 TCP/IP and the OSI Model

OSI Model

In the early days of computer networks, it was uncommon for products from different manufacturers to be able to communicate. A standard reference model needed to be created for this purpose. The open systems interconnection (OSI) model helped the vendors create compatible networks (Davies & Bressan, 2010). The OSI model is used to create different layers, which facilitate how application data are transported within the system. Data pass through the network medium and arrive at the destination host, sending the data to the correct application. The OSI model is not a physical device or application; it is a compilation of guidelines. Developers use these guidelines to build applications that run on a network. The guidelines also serve as a theoretical basis for network standards.

Figure 57: OSI Layers



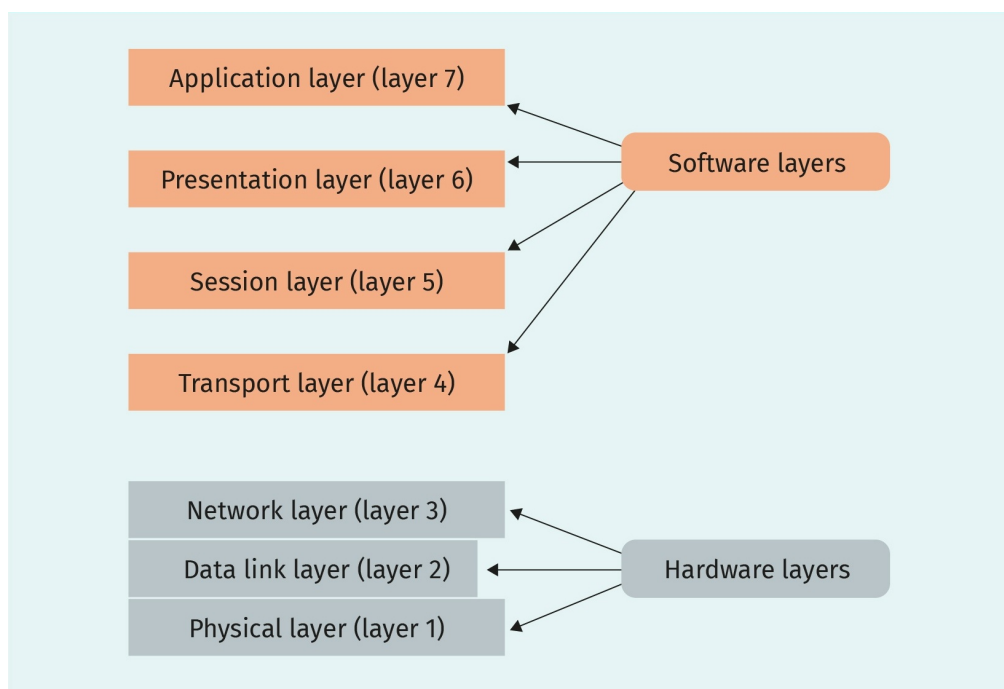
Source: Created on behalf of IU (2022).

In the previous figure, we can see the conceptual framework of the OSI model. At the very top is the application layer, and at the bottom is the physical layer. The user application is found in the application layer. The data being transmitted moves downwards towards the physical layer. The information is then sent towards its destination, and the process reverses itself. Some mnemonic phrases to help you remember the layers of the OSI model are as follows:

- “Please Do Not Throw Sausage Pizza Away” from bottom to top
- “All People Seem To Need Data Processing” from top to bottom

Each layer of the OSI model handles different functions. Layers four through seven are the “software layers,” and layers one through three are the “hardware layers.” The following figure depicts the separation of these layers.

Figure 58: Function OSI Layers



Source: Created on behalf of IU (2022).

In the following figure, you can see the generalized functions of each layer. The software layers create the infrastructure for the applications to communicate. The hardware layers create the standard on how the data will be transmitted from the source to the destination.

Figure 59: Separate OSI Layers

Application layer (layer 7)	User services
Presentation layer (layer 6)	Translation, encryption, and compression
Session layer (layer 5)	Initiate, maintain, and terminate connections
Transport layer (layer 4)	Delivery and end-to-end connection control
Network layer (layer 3)	Routing and moving packets
Data link layer (layer 2)	Data frame
Physical layer (layer 1)	Physical topology

Source: Created on behalf of IU (2022).

The software layers do not deal with anything concerning the network or network addresses. Their sole function is facilitating communication between applications. At the same time, the hardware layers know nothing about applications, as their function is to define how the data are sent across the wire. The term “wire” encompasses the network’s physical topology to include the switches and routers. Let’s start at the top and look at the application layer.

Application Layer

At the top of the stack, we have the application layer. This is the layer where users typically interface with applications, and the system to connect to network services. Software applications such as email clients, directory services, file transfer services, and web browsers, can interface here when dealing with different host resources. Attacks at this layer include

- cross-site scripting,
- buffer overflow attack, and
- structured query language (SQL) injection.

The application layer identifies the application’s availability on the remote host and the communication channel’s resources to be successful. Some protocols found at this level include hypertext transfer protocol (HTTP), simple mail transfer protocol (SMTP), file transfer protocol (FTP), and post office protocol 3 (POP3).

Presentation Layer

The presentation layer handles data translation, formatting, and the presenting of data to the application layer. The presentation layer also handles data encryption and compression. When the data are transmitted, this layer converts the application data into a compatible transmission format. This layer translates the data from a format used for transfers in the network into a format that can be used on the host computer. An example of this is converting the American Standard Code for Information Interchange (ASCII) text and Extended Binary Coded Decimal Interchange Code (EBCDIC) text. Some protocols found at this level include transport layer security (TLS), secure sockets layer (SSL), and moving picture experts group (MPEG). Elements of SSL and TLS can be found at different layers of the stack. There is a bidirectional data stream, which allows it to be placed at layer four or above. SSL and TLS also organize data as records containing handshake messages, which point to layer five or above. Depending on which aspects of SSL or TLS are being examined, elements that belong in layer six or seven can be found. The OSI model is a guideline, and, sometimes, not everything fits cleanly into a clear-cut label. Be it data compression, encryption, or decryption, all functions have to meet the standards of the presentation layer.

Session Layer

The session layer opens, closes, and manages the communication channel between user application processes. It is common to find remote procedure calls (RPCs) made by user applications at this level. The different modes of communication available at this layer

Full-duplex

This is a communication channel where both sides can communicate at the same time.

include simplex, half-duplex, and **full-duplex**. This layer maintains each application's communication separately to other applications, while managing the service requests and responses between the different network device applications. This layer can use security protocols using username recognition and by logging user access and activity while allowing the various devices to communicate over the network. Some protocols encountered at this layer include point-to-point tunneling protocol (PPTP), layer 2 tunneling protocol (L2TP), network basic input/output system (NetBIOS), and remote procedure call (RPC).

Transport Layer

The transport layer handles host-to-host communication and can create a logical connection between hosts, sending and receiving data, and conducting reliable data transmissions. At this layer, connection-oriented protocols and connectionless oriented protocols are possible. This layer takes on the responsibility of the data transfer from higher-level layers. Examples of attacks at this level include scanning from NMap as a reconnaissance of the network. An attacker can also use an internet control message protocol (ICMP) sweep by pinging all the hosts on the network and mapping their responses. The transport layer uses segmentation, taking long messages from the higher layers and dividing them into smaller messages. This layer (and the ones above) is considered an "end-to-end" layer. This layer communicates directly from source to destination, and does not concern itself with the transmission details in the layers below. Some protocols encountered at this level include TCP and UDP.

Network Layer

The network layer manages logical addressing, device location, and data transportation, which allows it to choose the best path for data transmission. Considerations are made regarding the condition of the network and priority of service. At this layer, controls to monitor network congestion, routing, and logical addressing are present. The router sends the traffic to destination devices that are not on the local network. There are two types of packets you will encounter at the network layer:

1. Data packet. This packet is used to transport user data and is sent with routed protocols, such as IPv4 or IPv6.
2. Route update packet. This packet is used to update nearby routers regarding the current state of the network. This information is then used to update the routing tables. Some protocols used to send these packets include routing information protocol (RIP), RIPv2, and open shortest path first (OSPF).

This layer is responsible for the network address, which is based on the protocol being used. For example, an IPv4 network will have an address that looks like this: 192.168.1.1.

An example of a layer 3 device is a router. When a router receives a packet, it checks the destination IP address. If the router does not host the destination address, the routing table determines the network address for the next step in the packet's journey. Once the

destination has been determined, the router sends the packet to the next network device. If the router cannot determine or find an entry for the destination in the routing table, it drops the packet. Some things to remember about routers are as follows:

- Routers will not forward broadcast or multicast packets.
- Routers will use the logical address to determine the path to the destination.
- Routers can use an access list to approve or deny access.
- Routers have the ability to provide connections to virtual local area networks (LANs).
- Routers have the ability to monitor traffic for quality of service (QoS).
- Some router devices can function as a layer two device (a bridge).

Data Link Layer

The data link layer handles the transfer of data across the physical network. This layer handles its level's flow regulation and error control. The hardware addresses (MAC) on host network adapters are found at this layer. It takes the packet and converts it into a "data frame" or "frame." This frame contains the destination and source hardware addresses. The difference between the data link layer and the networking layer is that the data link layer is only concerned with the local network hosts. The network layer is not concerned with the hosts on the local network segment, but wants to know where the network devices are located. The network frame will contain a checksum, source or destination address, and the data being transported. The data link layer can define network topology, physical addressing, sequencing, and flow control. Some protocols encountered at this layer include point-to-point protocol (PPP), asynchronous transfer mode (ATM), and frame relay. The data link layer has two sub-layers:

1. Media Access Control (MAC). This sub-layer defines how the system will place the packets on the network media. It takes a "first come, first served" approach when packets are being delivered. This layer defines the physical addressing and logical topologies at this level.
2. Logical Link Control (LLC). This sub-layer identifies network layer protocols and then encapsulates them. The data link layer uses the logical link control header to determine how to treat the frame when it is received.

Physical Layer

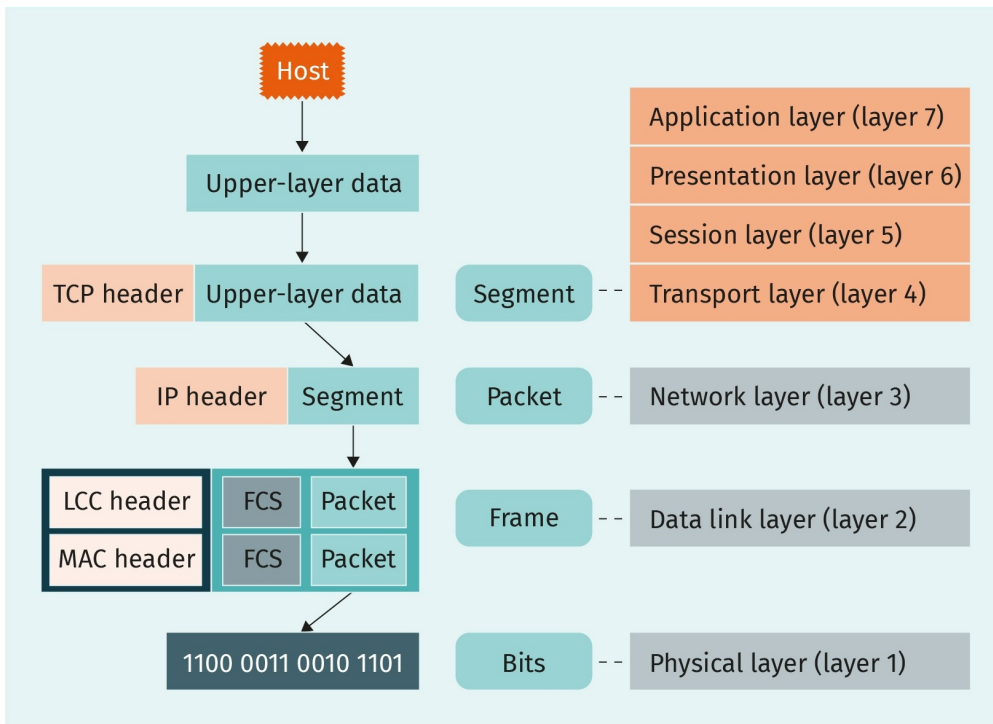
This is the lowest layer of the OSI stack and handles the physical transmission medium used to send and receive the raw bitstream. This level deals with bits which are represented with either a 1 or a 0. The physical medium can consist of electrical, optical, and mechanical interfaces used to transmit data. Once received at the destination, the data are sent to the data link layer (layer two). Topologies encountered here include mesh, bus, ring, and star networks. The specifications can define the media, voltage levels, timing, data rates, distances, and physical connectors of a transmission. This level may experience attacks including deliberate interruption of power, physical destruction of the network infrastructure, and electromagnetic pulse effects.

Encapsulation

When the host wants to send data across the network, the data are “encapsulated”: At each layer, the data receive a wrapper that contains protocol information for that layer. Each layer communicates with the corresponding layer at the destination host. As the data starts at the top, they traverse each layer, which encapsulates data using protocol data units (PDUs). The PDUs contain control information to be placed into the header or the end of the data field. The destination’s corresponding layer reads this information in the PDU and removes it as the data travel up the stack. The following figure shows how data travel down the OSI stack to be sent across the network. The transport layer sends a “sync” packet to the destination host to notify it that traffic is about to be sent. The layer breaks the data traffic into smaller pieces called segments and adds a transport layer header to each segment. The segment headers have a unique sequence number used to correctly assemble the received data set at the destination.

The transport layer then sends the segments to the network layer, where a network layer PDU is added. The segment then becomes a packet. The network layer handles the logical addressing and is responsible for sending the packet to the network (this occurs for destinations outside of the local network). The data link layer encapsulates the packet and adds the source and destination hosts’ physical addresses. When data are transmitted, bits and bytes may become corrupted. To detect possible errors, a frame check sequence (FCS) number is used. Within the frame, there is an FCS field. This field contains a number based on the data in the frame. When the frame arrives at its destination, the system compares the number in the FCS field to an FCS number generated by the destination host. If the numbers are different, the system assumes the data have been corrupted, and the frame is deleted. The packet is now referred to as a frame. At the physical layer, the system converts the frame into bits to be sent to the physical medium.

Figure 60: Encapsulation



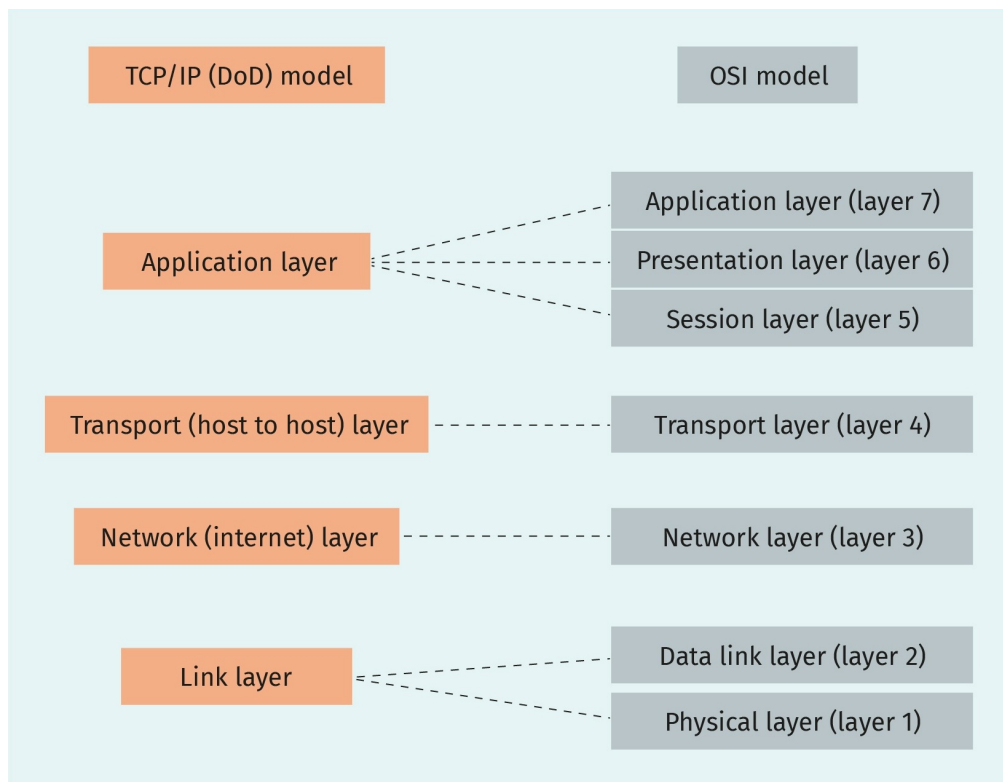
Source: Created on behalf of IU (2022).

Once the data arrive at their destination, the process is reversed, with each layer removing the corresponding PDU before sending the data up to the next level.

TCP/IP

Transmission control protocol and internet protocol (TCP/IP) is a communication protocol that was developed by the United States Department of Defense in the early 70s, eventually replacing the network control protocol (NCP) in 1983 (Cole, 2011). TCP/IP is the backbone of the internet, as it provides a solid, end-to-end connection between many hardware and software platforms. Like the OSI model, the TCP/IP model (also known as the DoD model) also has layers. Instead of dealing with seven layers (as with the OSI model), the TCP/IP model has four layers, as shown in the following figure.

Figure 61: TCP/IP Model and OSI Model



Source: Created on behalf of IU (2022).

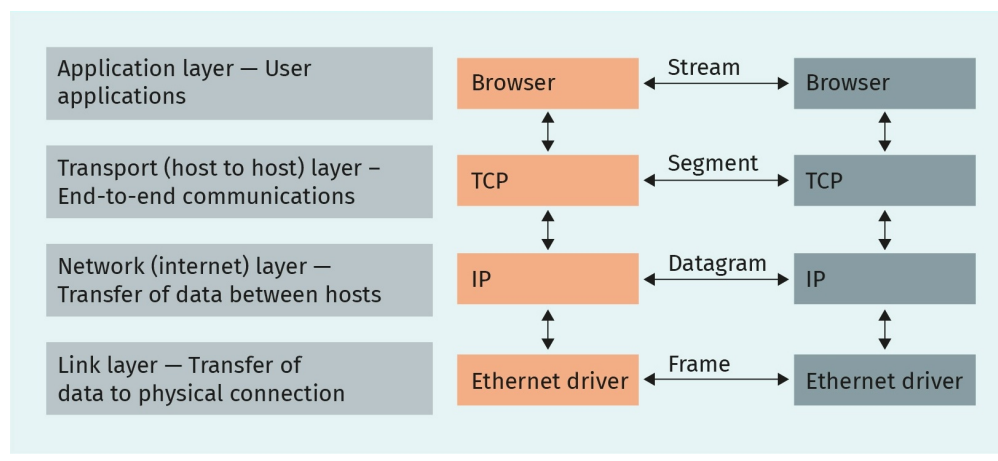
Applications used for information exchange are found at the application layer. Protocols that operate at this layer include hypertext transfer protocol (HTTP), file transfer protocol (FTP), simple mail transfer protocol (SMTP), Telnet, and dynamic host configuration protocol (DHCP). The protocols encapsulate the data from the application layer at the transport layer. The application layer corresponds to the application, presentation, and session layers of the OSI model.

The transport (or host-to-host) layer is the layer below the application layer. This layer handles the communication between the two hosts. The two transport layer protocols found here are TCP and user datagram protocol (UDP). TCP is a connection-oriented protocol that guarantees that all data will be delivered to the destination. UDP is a connectionless protocol that does not guarantee that all the data will be delivered to the destination host. The transport layer directly corresponds to the transport layer of the OSI model.

The network (or Internet) layer, also referred to as the network access or network interface layer, handles the protocols dealing with logical addressing and routing. This layer identifies the path the data will take to the destination host. The internet protocol (IP) is the principal component used at this level. The logical address can be a 32-bit IP address (IPv4) or a 128-bit address (IPv6). This layer directly corresponds to the network layer of the OSI model. Protocols found operating at this layer include IP and internet control message protocol (ICMP).

The link layer directly corresponds to the data link and physical layers of the OSI model. This layer addresses the receiving and sending of data from the host over a physical interface. The mapping of IP addresses (logical addressing) to media access control (MAC) addresses (physical addressing) occurs at this level. The datagrams from the network level are encapsulated and referred to as frames.

Figure 62: Data Flow in the TCP/IP Model



Source: Created on behalf of IU (2022).

The previous figure shows an example of the data stream flowing through the TCP/IP model. At the top of the stack, we have a web browser requesting content from the server. This request travels from the top of the stack, passing through the layers (shown by the vertical arrows) of the requesting host, arriving at the destination host, and finally being sent back to the original host. The horizontal arrows between the different layers signify the communication between the hosts. A direct channel is not being used between the two hosts; rather, encapsulation is used when data are being sent back and forth between hosts.

Protocols

The TCP/IP protocol suite is found in most networks and comprises two different protocols: internet protocol (IP) and transmission control protocol (TCP). IP is a protocol found at the network layer, meaning that it transports data between the source and destination hosts. As mentioned, IP is a connectionless protocol (it does not guarantee that the data will reach the destination host). IP is also responsible for the logical addressing of the host on the network. Currently, there are two versions of IP addressing available to networks: IPv4 and IPv6.

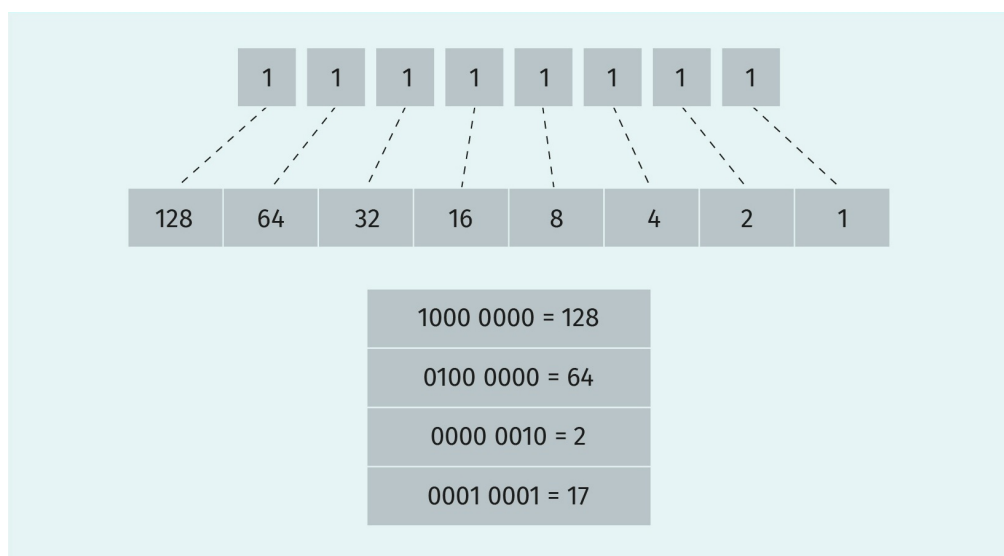
IP Addressing

A host on a network must have a logical address to receive traffic. This is comparable to having a telephone number. When connecting to the telephone network, one must have an address (the telephone number). The IP address tells the system which network the host is on and the location within the network. A telephone number is often composed of a country code, area code, and local number.

IPv4 and IPv6

An IPv4 address is made up of 32 bits arranged in four sets of octets (eight bits), and is represented as xxx.xxx.xxx.xxx. Each bit of the address is assigned a decimal value, with the leftmost value being 128 and the values decreasing in the order 64, 32, 16, 8, 4, 2, 1. In the following figure, we see an octet with all the binary values set to 1 and the corresponding decimal value shown below. With all the bits set to 1, the value of the octet is 255 ($128 + 64 + 32 + 16 + 8 + 4 + 2 + 1 = 255$).

Figure 63: Binary to Decimal Conversion

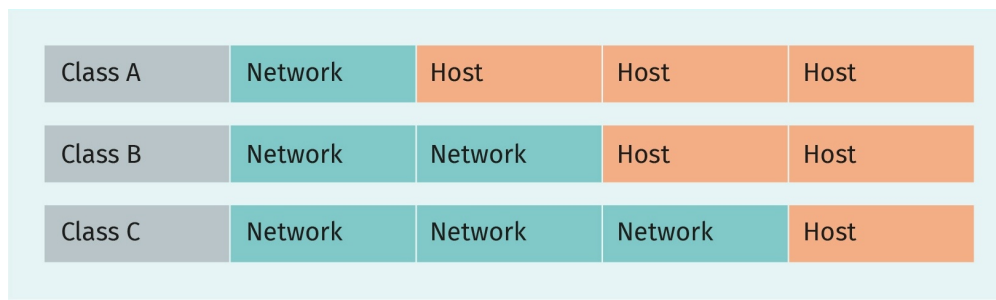


Source: Created on behalf of IU (2022).

Below the decimal representation are some examples of additional binary-to-decimal conversions, for example, in the first example, a binary value of 1000 0000. With the far-left bit having a binary value of 1, which converts to a decimal value of 128, the rest of the bits have a value of zero, so this octet value is 128. In the second example, the second bit from the left is turned on (i.e., has the binary value of 1), and all other bits are turned off (i.e., have a binary value of zero) this will convert to the decimal value of 64.

The logical address uniquely identifies each host on the network. This standard breaks the logical address up into different classes. Classes A through C use a portion of the network address to identify the specific network, with a part of the address being used for the hosts.

Figure 64: IPv4 Network Classes



Source: Created on behalf of IU (2022).

The previous figure shows how this is broken down. With a class A address, the first octet of the network address identifies the network segment. The next three octets are used to identify the hosts on the network segment. Using a class A address, we can have more than sixteen million hosts on the network. With a class B network address, the first two octets are used to identify the network segment. The last two octets are used to identify the hosts on the network segment. This allows for over sixty-five thousand hosts on a class B network.

The class C network uses the first three octets to identify the network segment, with the last octet used to identify the host on the network. 254 hosts are possible on a class C network. Two of the addresses $x.x.x.0$ and $x.x.x.255$ are “broadcast addresses,” and cannot be used as a host address. A broadcast address is an address used to transmit a message to all the hosts on a network and does not require a response. There are also class D and class E networks, but they are reserved for other uses (multicast and research).

Table 8: IPv4 IP Address Classes and Ranges

Class	Address range	Max number of hosts	Private IP range
A	0.0.0.0– 127.255.255.255	16,777,214	10.0.0.0– 10.255.255.255
B	128.0.0.0– 191.255.255.255	65,532	176.16.0.0– 172.31.255.255
C	192.0.0.0– 223.255.255.255	256	192.168.0.0– 192.168.255.255
D	224.0.0.0– 239.255.255.255	Reserved for multicast	-
E	240.0.0.0– 255.255.255.255	Reserved for research	-

Source: Created on behalf of IU (2022).

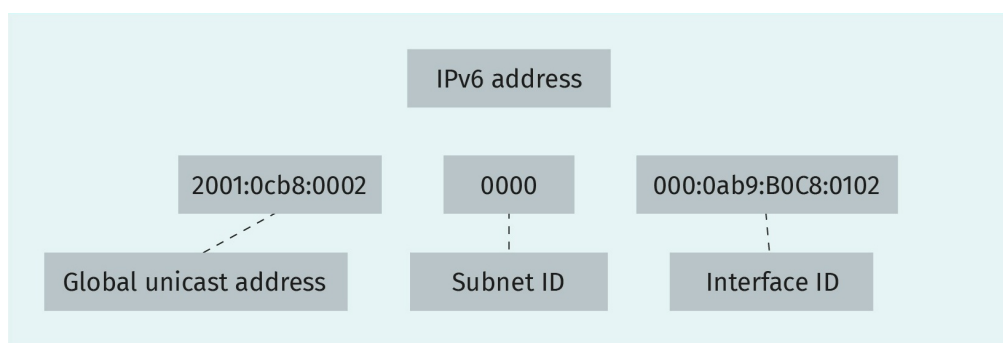
The previous table shows the different classes of an IPv4 network. Each network class also has private (or reserved) IP addresses. These ranges of IP addresses are meant for use in a closed local network. Therefore, it is not possible to communicate outside the local net-

work using an IP address from these ranges. We can typically find local networks using this range of IP addresses behind a router using network address translation (NAT). NAT allows the router to convert the traffic from the hosts on the private network and then relay the traffic using a public IP address. When the traffic is returned, the router then sends the traffic back to the host. With this method, the only IP address that is exposed is the public IP address.

IPv6 can be considered as the next generation of logical network addressing. It is predicted that, ultimately, there will not be enough IPv4 addresses available as more and more devices connect to the internet. An IPv6 is 128 bits in size, versus the 32 bits of an IPv4 address. IPv6 addresses are also written using hexadecimal. This means that an IPv6 address comprises 32 hexadecimal numbers that are broken up into eight different segments separated by a colon symbol (:). Each segment can have a hexadecimal value of 0 through FFFF. An example of an IPv6 address is 2001:0CB8:0002:0000:0000:0AB9:B0C8:0102. We can also shorten this address by removing the leading zeros: 2001:CB8:2::AB9:B0C8:102

Just like with IPv4, the IPv6 address is separated into network and host identification, as shown in the following figure. IPv6 uses the first 64 bits to identify the network and the next 64 bits to identify the hosts. The system bases the host identification on the network adapter's physical address (MAC address), which is identified as the interface identification (ID) in the following figure.

Figure 65: IPv6 IP Address



Source: Created on behalf of IU (2022).

The upper 64 bits are further broken down into a 48-bit global network address and a 16-bit subnet ID. The subnet ID describes the private topology of the network. The system administrator will configure this value. There are three different classes of IPv6 addresses:

1. Global unicast address. This address is routable on the internet and starts with 2001.
2. Unique local address. This is used for internal networks and is not routable on the internet. The unique local address can begin with FC00 or FD00 and is followed by a 40-bit hexadecimal string of random characters. FD00 indicates the address is locally assigned, while FC00 is globally assigned. The globally-assigned specification has not been defined by the Internet Engineering Task Force (Haberman & Hinden, 2005).
3. Link-local address. This address is not routable locally or on the internet. It is expressed as FE80:0000:0000 followed by 54 zeroes.

3.2 Core Internet Services

The internet is a worldwide network that has become a fundamental resource for business and consumer resources. Most consumers are only familiar with email or web browsing as components of the internet. However, there are many more options available to the user, which we will now discuss.

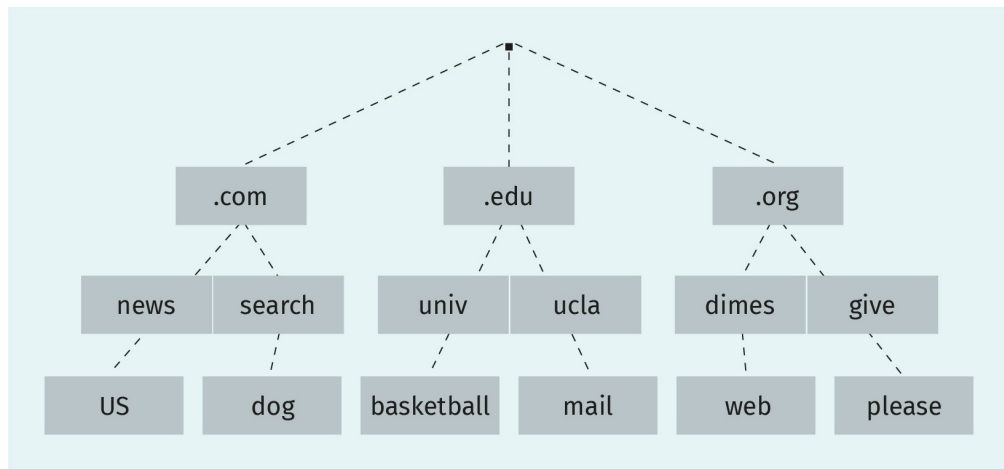
Domain Name System (DNS)

The internet is made up of large, interconnected networks, the highest tier of which is network service providers. A network service provider is an organization that may own, operate, or sell access to their backbone infrastructure. Regional and local internet service providers then provide access to their users by connecting to the network service providers. Every computer connected to the network must have an IP address. IP addresses are difficult to memorize and inconvenient to write out every time you want to visit a network location, but, if you do not know the IP address, how can you connect to another organization's server? The answer is the domain name service (DNS). DNS is a service that maps IP addresses to domain names. DNS is a distributed database and can be used by any operating system. A hierarchy is used with DNS to resolve a fully-qualified domain name (FQDN). Firstly, we have the top-level domains (TLD), examples of which are

- .com, commercial organization,
- .edu, an educational organization,
- .gov, a US government organization,
- .int, an international government organization,
- .mil, a US military organization,
- .net, a network organization,
- .org, a nonprofit organization,
- .ca, a Canada-based organization,
- .jp, a Japan-based organization, and
- .de, a Germany-based organization.

Of over 190 countries in the world, most have created a top-level domain (WorldStandards, 2021). In addition, new TLDs are being authorized at regular intervals. The following figure is an example of the domain hierarchy.

Figure 66: Domain Hierarchy



Source: Created on behalf of IU (2022).

At the very top, we have the “period.” This shows the root of the domain namespace. This is the root name server for the top-level domains and is the first step in the DNS process. It answers requests and provides the authoritative name server list for the TLD. The TLD name server provides the IP address of the authoritative name server for that top-level domain. The authoritative name server takes the domain name and sub-domain name and returns the IP address. The following are types of DNS process:

- Request of information. The local computer host checks the local DNS cache to see if the domain name has been resolved before. If the local host does not contain that information, it will make a DNS query to the recursive DNS server.
- Recursive DNS server. The server checks its cache to see the name resolution has been requested before. If the cache does not have an entry for the domain name resolution, the next step will be to ask the authoritative name server.
- Authoritative DNS server. This server maintains the records for the address resolution. It accesses the name record and identifies the IP address. It then sends this information to the recursive DNS server. The recursive DNS server stores this information in its cache and forwards it to the local host, which stores the information in its cache.

The DNS record has a time-to-live (TTL) value. This value shows when the information will expire. An additional request will have to be made once it has expired in the cache.

DNS Record

There are many types of DNS record. The key types will now be described.

Mail exchanger (MX)

This record contains the IP address to the server hosting the domain's email accounts. Multiple mail exchange records can be found for the domain. The servers can have a different priority value. When the high-priority server is unavailable, the system will use the next server in line.

Text (TXT) record

This record contains human or machine-readable text. Administrators can use it for a variety of purposes, such as providing information for verification.

Canonical name (CNAME) record

This record contains links of alias names to another domain name. For example, `www.google.com` can link to `google.com`. This record does not include an IP address.

Address (A) record

This record contains information about the physical IP address hosting the domain's services.

Name server (NS) record

This record contains information about the authoritative DNS servers for a domain. Typically, the internet service provider (ISP) provides information about the primary and secondary name server they want us to use.

Start of authority (SOA) record

The SOA record contains information about the DNS records within that zone. The following information may be found within the SOA record:

- the primary DNS server name.
- the email address of the person responsible for the zone.
- the serial number used by the secondary DNS.
- the time-to-live (TTL) interval.
- the retry interval. This determines how often the secondary DNS server should attempt to refresh after failure.
- the expire interval. This determines how long the TTL will be valid after a refresh.

Service (SRV) record

This record contains information used to establish connections between host names and services. We use it when an application is trying to locate a specific service using the SRV. Information found here includes port numbers, IP addresses, host name, and priority.

Pointer (PTR) record

This record contains the reverse DNS information by resolving an IP address to a domain name.

Routing

Routing is the moving of packets between networks using routers. This does not automatically occur when you plug a router into your network. Routing protocols are implemented to help find distant networks and make sure all routers are using the same up-to-date routing table. For a router to send and receive packets, it must have the necessary information, such as

- a destination address,
- nearby routers,
- knowledge of potential routes to remote networks,
- knowledge of the best route to remote networks, and
- the ability to verify and maintain routing information.

When a router is set up on the network, the administrator ensures that an up-to-date routing table is available. The routing table tracks the available remote networks. There are two ways for the routing table to be populated: static routing, where someone enters all the data by hand, and dynamic routing, where all the routers use the same protocol and update the neighborhood routers as changes occur.

Distance Vector Routing

The routing information protocol (RIP) is a commonly used routing protocol that has two versions: one for the TCP/IP network, and one for the IPX/SPX network protocols. RIP uses “hop counts” to determine along which routes to send the packet. A hop is essentially a step towards the packet’s destination; each step is a router that the packet will make contact with as it travels towards the destination. The maximum number of hops when using RIP is fifteen. If the destination is a distance of over fifteen hops away, the network considers the destination to be unreachable. RIP operates by having all routers continuously send updates regarding the routers they have information about to other directly-connected routers. This information is used to create the routing tables.

The downside to using this procedure is that updates can be slow and create additional traffic on the network. As the packet is being sent to the destination, the router is not aware of every specific hop. The router is only aware of the next hop that is in line with the destination. If incorrect information is being propagated between the routers, this can lead to a routing loop. To combat this, various strategies have been developed. The “split horizon” approach prevents a router from taking a route back to a router from which it had previously received an update. The “poison reverse” approach handles this a bit differently, propagating a way back to the router from which it received an update but listing the number of hops as 16. This would make it an unreachable destination, allowing for loops to be avoided.

Link State Routing

Link state routing takes a different approach than distance vector-based routing. Instead of broadcasting the information to each router, the router will send a special type of packet out. These packets are called “link state advertisements” (LSA) and contain information about the specific router. The routers use the LSA packet to create the routing table and create a map of the entire network. The system sends the packet to every router on the network. Two conditions will cause an LSA packet to be sent: when a new router is brought onto the network, or when there is a change in the network topology. The link-state protocol found on a TCP/IP network is open shortest path first (OSPF). This is a routing protocol that allows routers to share information about the network topology and the state of the network. The routers then use the information to determine the shortest path to the destination.

3.3 The World Wide Web

The World Wide Web is sometimes referred to as the internet, which is not an accurate description. In reality, the World Wide Web is one aspect or feature of the internet. The World Wide Web is a series of connected servers hosting public-facing web pages that users can access via a browser. There are several components necessary for a user to access content on the World Wide Web:

- hypertext transfer protocol (HTTP), the network protocol. Found at port 80 (or 443 for HTTPS), HTTP is used to transfer the data between a web server and browser.
- uniform resource locator (URL). This is the web address (typically in the form `http://www.iu.org`) of the content being hosted by the web server. The web browser will display the URL in the address bar. DNS is used to translate the URL into the server's IP address. The system accomplishes this in the background, with the average user having no idea what is taking place.
- hypertext markup language (HTML). This is a standard format used to create the web pages hosted on the web server.
- web server. This is a server that hosts web pages to be accessed by a user through a web browser.
- web browser. This is software used to access content being hosted on a web server.

Web pages can be interlinked using “hyperlinks,” which allow the user to access additional content hosted on the web server. At the fundamental level, the World Wide Web is nothing more than a series of numerous client computers accessing content hosted on servers. Let's go into some detail about the software you will use to access the World Wide Web, specifically the browser or web browser.

Dark Web

There is a segment of the World Wide Web that the average user does not have access to. A special browser must be used to access this “dark web.” The dark web is a small portion of the World Wide Web that is considered to be part of the more extensive “deep web.” The

deep web is a portion of the World Wide Web that is ignored (not indexed) by search engines. There may be additional security precautions, such as a username and password, to access the “hidden” content. The dark web is considered the underbelly of the World Wide Web. The user can find many illegal activities, such as the purchase of narcotics, firearms, and many other illicit commodities. Cryptocurrency, such as Bitcoin, is often used to pay for these illegal products. This helps maintain anonymity between the vendor and the buyer.

To access the content on the dark web, the user must access the Tor network. This network is also referred to as the “onion network” or the “anonymity network.” Tor is an open-source, free software that allows anonymous communication by navigating the internet through relays. Volunteers manage over seven thousand Tor network relays. These relays create an overlay network over the internet and help conceal the user’s identity, location, and internet activity. Tor’s intended purpose is to protect the user’s privacy by keeping their communication channels confidential. It also has unintended consequences of being used for illegal activities.

The Tor browser utilizes the onion network, also known as “onion routing,” to access both dark and legitimate websites. Onion routing uses encryption that encapsulates the data in layers, similar to the layers in an onion. The encryption is implemented at the application layer. The data, including the IP address for the next hop, are encrypted multiple times as they are sent to the virtual relay network, with each relay being randomly selected. As the data arrive at the relay, it decrypts the encrypted layer, revealing the next destination. On arriving at the final relay, the last layer of encryption is decrypted, and it sends the data to the final destination. It does this in a way where the starting IP address is not revealed. Each relay only has knowledge of the previous relay and the next relay. With each relay, a separate layer of encryption is used, making it more difficult for an attacker to intercept the traffic. When the data leave the Tor network and reaches the ultimate destination, from the destination’s perspective, the traffic originated from the last Tor relay.

The US government originally developed onion routing to protect communications relating to US intelligence. The US government released the code for the Tor network under a free and open license. This led to the creation of a nonprofit, “the Tor project,” which now maintains the Tor browser (Tor Project, n.d.). The Tor network allows access to an underground marketplace of illegal activities. To exist on the Tor network, the website must have the URL that ends with “.onion.” Currently, there are hundreds of thousands of URLs that can be found on the Tor network, with a sharp increase to over two hundred thousand unique “.onion” addresses from less than one hundred thousand unique onion addresses in early 2020 (Tor Project, 2021).

Governments around the world have not ignored this illegal activity. The United States Federal Bureau of Investigation helped to take down the “Silk Road” marketplace for selling illicit drugs. The US Government convicted Silk Road founder Ross Ulbricht in a US federal court of money laundering, narcotics trafficking, and computer hacking. The US government further alleged that Ulbricht engaged the services of an assassin to murder five people, but they did not present any evidence to prove the allegations (Weiser, 2015).

Web Browsers

The Tor browser is a specialized version of a web browser. A web browser or browser software is used to access content that is hosted on Web servers. A user can install a web browser on a desktop computer, server, laptop computer, and mobile device. In 2021, the most popular browser was Google's Chrome browser, which has captured almost sixty-five percent of the marketplace. Safari was the next most popular browser, and has not captured even twenty percent of the market (StatCounter, 2021).

The most advanced browsers are not necessary to access the World Wide Web. There are browsers designed to be minimal in size and features, browsers that are text-based only, and browsers with every feature a developer can think of. Some additional features a browser may have are the ability to access email, access USENET (a distributed bulletin board system that was one of the original components of the internet), or access the internet relay chat (IRC) system. Some of the standard features found in web browsers are

- pop-up blockers. This is a utility that prevents unwanted browser windows from being created and "popping up," blocking the user's view of the web page.
- bookmarks. These provide the ability to save websites (URLs).
- navigation buttons. The back, forward, refresh, stop, and home buttons allow the user to go backward/forwards, stop a web page from loading, or refresh the content being viewed with the browser. The user can also designate a homepage, which will be automatically visited whenever the user launches the browser.
- address bar. This is a field into which the user can enter the URL of a website. It can also be used as a method to access the search engine.
- tabbed browsing. Tabs allow the user to have multiple web pages open in a single browser window. The user can select the desired page by clicking on a tab to change the display to that web page.

Browser Security

Browser security is consistently being updated as attackers find new ways to deploy their attacks. It is not uncommon for an attacker to create a clone of a legitimate website. For example, `http://bankofamerica.com` is the legitimate URL for an international financial institution. An attacker can create a fraudulent version of the website and make a minor change to the URL, for example, `http://bank0famerica.com`. Here, the attacker has changed the letter "o" to a zero. If the user is not paying particular attention to the URL, they may not realize they are receiving data from a fraudulent site. Web browsers can now check whether the URL in the taskbar is valid. Another option is using different colored fonts to accentuate the URL. In the example, the text `bankofamerica` is presented to the user using a dark color font, while the `http://` and the `.com` portion of the URL is presented to the user using a lighter color font.

A standard warning issued to users is to look at the padlock symbol displayed in the taskbar. The padlock symbol will either depict an open padlock or a closed padlock. An open padlock shows that the protocol HTTP is being used. This means that the user's traffic is being sent to the web server, and the web traffic being sent to the user is not encrypted. This allows an attacker to set up an eavesdropping device between the user and the server

(i.e., a man-in-the-middle attack) and read the contents of the communication channel. The closed padlock indicates the secure protocol HTTPS is being used. HTTPS is based on the HTTP protocol and secures the communication channel between the user and the server. The current HTTPS version uses transport layer security (TLS) to create an encrypted tunnel between the user and the web server. This effort reduces the risk of an attacker successfully carrying out a man-in-the-middle attack. The Cybersecurity and Infrastructure Security Agency (2015) of the United States government has identified that the attacks directed at web browsers are made worse because of the following factors:

- Users click on links without considering the risk
- Webpage addresses are disguised
- Security is given less consideration than functionality
- Third-party add-ons may not have the ability to obtain a security update
- Users do not know how to configure the security features of their web browser
- Users are unwilling to disable functionality to increase security

Some attackers have also compromised a user's web browser. This is known as a "man-in-the-browser" attack. The browser can be compromised if the user activates malware, which is most commonly deployed as a Trojan (an application that looks legitimate but is malicious). The attacker has then compromised the communication channel before TLS is implemented between the user and the server. To mitigate this type of attack, the user should only install browser extensions or add-ons from trusted sources, the operating system and browser should be kept up-to-date, and users should install antivirus software.

Another attack on the browser is "session hijacking." When the user interacts with the server via the browser or another application, the server needs to track which requests are coming from which users. A session token is generated to uniquely identify the user and the user's requests. To prevent an attacker from taking over the connection, the session token must remain confidential. If the session token is compromised, the attacker inserts themselves into the communication channel and takes the user's place.

Anonymous/Private Browsing

Another feature that has become popular is the user's ability to use private, "incognito," or anonymous viewing modes within their browser. Private browsing is a feature that allows the user to keep their browsing history, search records, or cookies from being saved on the computer they are using. This may be, for example, because they are using a public computer and do not want their browsing history or data to be accessible to an unknown third party.

What private browsing does not allow the user to do, however, is browse the web anonymously. The website, internet service provider, or employer can still see your identity, location, and viewing history. The function of private browsing (and each browser implements this option differently) is to simply not store your internet history, cookies, passwords, and usernames. Each time the private browser mode is activated, the user is presented with a new browser window. When the user closes the private browser window, any artifacts of the user's actions are deleted from the local system.

3.4 Transport Layer Encryption

Cryptography is a process that can protect the confidentiality of your data while they are at rest or in transit. The goal of using cryptography is to prevent the dataset from being accessed and read by an unauthorized party. Different implementations of cryptography can provide either powerful or very weak protection, depending on which resources the attacker has access to. There is no version of cryptography that is 100 percent secure; instead, the goal is to make the attack too time-intensive or work-intensive to be worthwhile for the attacker. Encryption is the process of converting “plaintext” (text that can be read) into a format that cannot be read or understood, known as ciphertext. Only users who possess the key will be able to decrypt the ciphertext. The process of converting ciphertext into plaintext is called decryption.

Encryption differs from encoding, and is used when data need to be transferred between systems or applications in a new format. Encoding changes the format of the data and can be easily reversed using the same encoding scheme. Encoding does not require a key and does not protect the confidentiality of the encoded data.

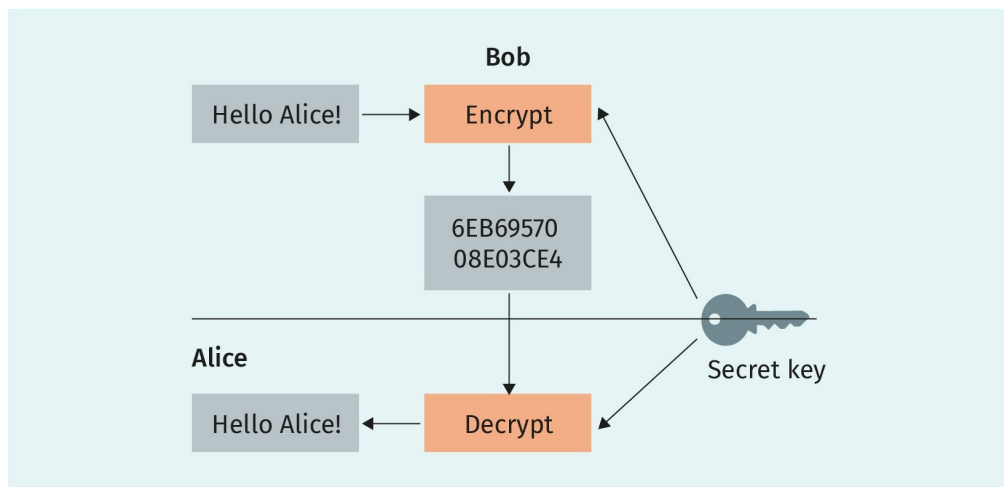
The system used to encrypt and decrypt is called a cryptosystem. The algorithm used for the encryption and decryption is called a cipher. The final piece in the cryptosystem is the key. The key is the “crypto variable” and is used with the algorithm to encrypt or decrypt the data set. The key can be a user-generated artifact or be generated by the algorithm. The key space determines the length and complexity of the key. The ability to create a random key using a large key space makes it difficult for an attacker to decrypt the ciphertext.

Encryption Methods

Cryptography algorithms come in two different applications: asymmetric and symmetric. Symmetric cryptography (also known as secret-key cryptography) is a type of cryptography in which the same key is used to encrypt and decrypt the data set. Using symmetric cryptography requires the user to maintain the secrecy of the key. If the key becomes compromised, then the attacker can gain full access to any data that have been encrypted with that key. Some common applications of symmetric cryptography include stream and block ciphers.

A stream cipher will encrypt or decrypt one character at a time. A block cipher will take a group of characters and encrypt or decrypt them as a block. If the dataset has a shortfall and cannot fill the block size, the algorithm pads the block with random characters and proceeds with the encryption. In the following figure, we see an example of symmetric cryptography. The user Bob creates the plaintext “Hello Alice!” Bob then encrypts the plaintext using the secret key and creates the ciphertext. Bob sends the encrypted message to Alice. When Alice receives the encrypted message, she then decrypts the message using the same secret key. Alice can now read the decrypted message.

Figure 67: Symmetric Cryptography

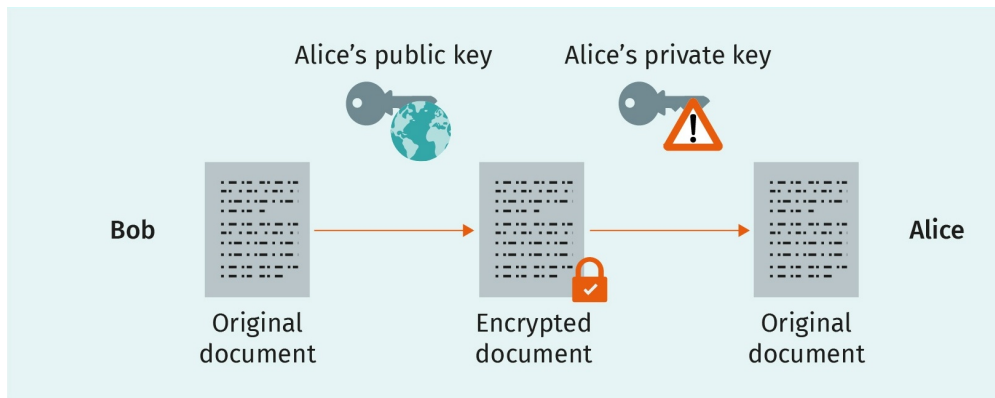


Source: Phayzfaustyn (2014), based on Davidgothburg (2006). CC 0.

When using symmetric cryptography, confidentiality is possible, but authentication and non-repudiation are not. There is no way to determine who originally sent a message, because multiple people have access to the same security key. Though symmetric cryptography is much faster than asymmetric cryptography, and can be very secure when using a large key, there is a lack of a secure delivery process for the security key(s). Every message sent requires the sender and recipient to use the same key to encrypt and decrypt messages.

As the scope of communication grows, it necessitates a very complicated key management system. Asymmetric cryptography uses a public/private key system. A public key is a key that is available to all users. A private key is only available to the owner. When the key pair is generated, the owner will keep the private key secret and then provide access to the public key to the public. The two keys are mathematically related, but they are not interchangeable. If user Z wanted to send an encrypted message to user A, user Z would encrypt that message using user A's public key. The only key that can decrypt a message encrypted with the public key is the private key. If user A wishes to respond to user Z, they will encrypt his response using the public key of user Z. The only key that can decrypt a message encrypted with a user's public key is the user's private key. It is not possible to encrypt or decrypt a data set using only a single key.

Figure 68: Asymmetric Cryptography



Source: Fleshgrinder (2014). Public domain.

In the previous figure, you can see that Bob has created a document that he wants to send to Alice. Bob has Alice's public key and creates the ciphertext. Bob sends the encrypted document to Alice, who decrypts the ciphertext using her private key. If Alice wanted to respond to Bob and make sure that Bob could authenticate her message, she would digitally sign her response using her private key. By using her private key, the confidentiality of the message is unprotected, but the message can be authenticated, as Alice is (in theory) the only person with her private key. If Alice had encrypted her response with Bob's public key, there would be no way to determine who sent the message.

Suppose Alice was concerned about the confidentiality of the message. In that case, she could encrypt her response using her private key and then encrypt the ciphertext a second time using Bob's public key. Alice has now authenticated the message and provided a layer of confidentiality. Bob would need to use his private key to decrypt the first layer and then use Alice's public key to decrypt the second layer.

An asymmetric algorithm uses more complex mathematical functions than symmetric algorithms, making them much slower than symmetric encryption. The key management system is much easier with asymmetric cryptography. The organization only needs to provide a key pair (public and private keys) to each member.

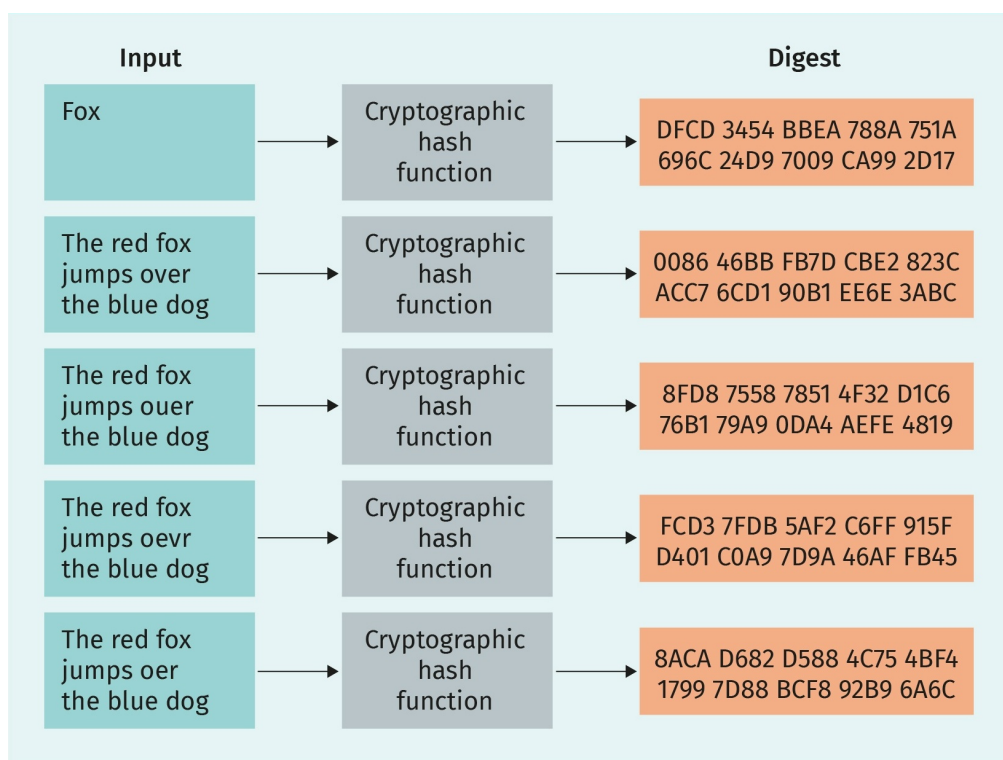
Initialization Vectors

To help avoid creating patterns in the ciphertext, initialization vectors (IVs) are used to create random values with the encryption key. For example, if message A is created and encrypted with encryption key Z, the same ciphertext is created every time the message is encrypted. An attacker can then use the pattern in the ciphertext to break the encryption. By adding the initialization vector with the encryption key, the randomness of the ciphertext is increased, making it harder to attack.

Hashing Algorithms

Hashing algorithms are a form of encryption without a key. This is a one-way process where the user takes a variable-length input and receives a fixed-length output (referred to as a hash value). This hash value is unique and will not be repeated with a different input. This function is used when trying to determine whether a file has been altered. If we use a hashing algorithm on a file, we receive the standardized output as a hash value. We refer to this as the “fingerprint” of a file. If anyone has changed a single bit in the file, we will receive a different hash value. The following figure is an example of this process.

Figure 69: Hash Functions



Source: Stolfi (2008). Public domain.

If the hashing algorithm provides the same value for two different inputs, then we have what is called a “collision.” The following are characteristics of a robust hashing algorithm:

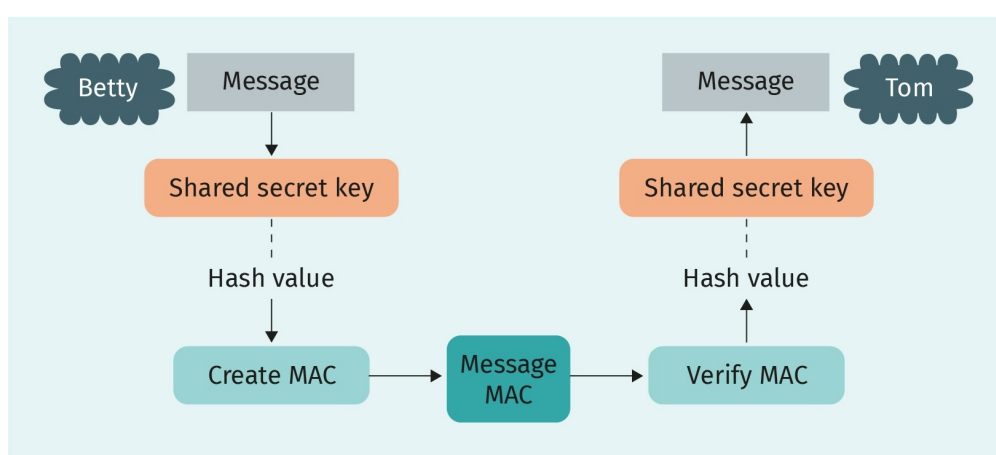
- The algorithm should compute the hash value based on the entire input.
- It should be a one-way function where it is impossible to re-create the input based on the hash value.
- Collisions should not occur.

Some standard hashing algorithms in use today are the MD5 (message-digest 5) algorithm, which produces a 128-bit hash value, and the SHA-1 (secure hash algorithm), which creates a 160-bit hash value. Both MD5 and SHA-1 have identified collision avoidance issues.

Alternatives are SHA-256 and SHA-512, both of which are part of the SHA-2 family. MD5 and SHA-1 are still widely used in the forensic environment today to determine file integrity.

Another example of using a hashing algorithm is that user Betty creates a message for user Tom. After Betty has composed her message, she can calculate the hash value of the message and include that value with the message. An attacker, Carl, intercepts Betty's message and changes it. Carl then rehashes the message and replaces the old hash value with the new one. Once Tom receives the message, there is no way for him to know that someone has tampered with the message. Betty would need to use a message authentication code (MAC), such as hash-based MAC (HMAC). In this scenario (as shown in the following figure), Betty creates her message and uses a hashing algorithm to create the hash value. Next, she encrypts the message and hash value with a secret key, creating the MAC value. If Carl intercepts this message, he will not be able to re-create the MAC value because he did not have access to the secret key.

Figure 70: HMAC Process



Source: Created on behalf of IU (2022).

IPsec

Internet protocol security (IPsec) is a set of communication protocols that provide a cryptographic function to IPv4 and IPv6. This allows for creating a virtual private network (VPN), which enables the sending of information over a public or insecure network. IPsec uses two protocols to achieve this functionality: authentication header (AH) and encapsulating security payload (ESP). AH does not provide confidentiality for the data stream; it does, however, allow for the authentication of the packets. ESP provides for confidentiality of the data stream because it encrypts the data. Two modes can be used with IPsec: tunnel and transport. When using ESP tunnel mode, the entire packet is encrypted to include the packet headers. ESP transport mode will encrypt the packet except for the original headers. This mode is used with AH so that the original packet headers can be authenticated.

Transport Layer Security (TLS)

The secure sockets layer (SSL) is a legacy session-orientated protocol used to create secure web sessions, email traffic, Telnet traffic, and file transfer protocol (FTP) traffic. Transport layer security (TLS) has, for the most part, replaced SSL. Both protocols operate in the same general manner. At the beginning of the session, they perform a handshake or negotiation to set up the communication parameters. The handshake uses an asymmetric key (the server's public key) and determines which version of the protocol will be used and which cipher suite will encrypt the communication channel. The client and the server then create and use a symmetric key. Once the tunnel is created, the data are encrypted or decrypted with the newly-created, shared symmetric key. This secure session provides a confidential communication channel.

3.5 Common Attacks

Man-in-the-Middle Attack

A man-in-the-middle attack is where a third party places themselves in the middle of the communication channel between two other parties. This can be as simple as eavesdropping on a conversation between two people or intercepting the signal between a user and an open wireless router. In either case, the attacker has breached the confidentiality of the communication. Sometimes, the attacker can receive information (e.g., from user Bob), change the message's content, and then send the fraudulent message to the user Alice. The control that organizations can implement is requiring endpoint encryption. This that, when Bob creates a communication channel to Alice, each side of the channel must verify the identity of either party using strong encryption and digital certificates for authentication. Using encryption will remove the third party's ability to breach the confidentiality of the communication channel. Another control that organizations can implement is authenticating the users on the communication channel. The most common limitation of this control is the use of transport layer security (TLS) when connecting to an e-commerce site via a web browser.

DNS Spoofing

Domain name system (DNS) spoofing is an attack where the attacker inserts fraudulent DNS information into a DNS server's cache. When a user's computer system makes a requests to the DNS server to resolve a URL into an IP address, the manipulated information sends a fraudulent location to the user's computer system. For example, the user wants to go to their banking site, "www.mybank.com," the computer system sends a request to the DNS server and asks for the IP address associated with "www.mybank.com." If the DNS cache has not been corrupted, then the user will be directed to the correct site. If the DNS cache has been corrupted, however, the user will be directed to a fraudulent site. Attackers have been known to create a clone of the authentic site to deceive the user. Once the user enters their credentials into the fraudulent site, the attacker then bounces the user back to the authentic site. The user would never know they had entered their credentials into the fraudulent site and would think that it was

some sort of glitch that caused them to re-enter their credentials on the authentic site. To mitigate this type of attack, the organization can deploy secure DNS, which uses digital signatures and public-key certificates to authenticate the DNS response's integrity.

ARP Poisoning

Address resolution protocol (ARP) poisoning can be a form of the man-in-the-middle attack, a denial-of-service attack, or a session hijacking attack. ARP is a communication protocol used on the local network that creates a table associating the logical address (IP address) with the physical address (MAC address) of the computer systems on that local network segment. The attacker can “poison” the ARP cache to have the network traffic sent to the attacker's computer system. The attacker can then intercept the communication channel, change the traffic on the communication channel, or stop all traffic on the communication channel. This type of attack requires the attacker to have direct access to the local network segment. The organization can deploy several controls to mitigate the ARP attack. For example, the organization can create a static ARP cache, which cannot be altered by an attacker unless the request comes from a certified source. The organization can also monitor all ARP traffic, and receive alerts when the ARP cache entries are changed.

Distributed Denial-Of-Service Attack (DDoS Attack)

A distributed denial of service (DDoS) attack occurs when the attacker floods the target system with requests, overloading the targeted system or communication channel. The attack typically originates from multiple hosts that are infected with malware. It is not uncommon for thousands of compromised hosts to be used in the attack. This is hard to mitigate because the attack is coming from multiple locations, and an attempt to block IP addresses may also block legitimate traffic. Recently, there have been vendors who have been advertising the “denial of service” as a service. This would allow attackers to pay via a website and use a point-and-click interface for their attack. The hosts used in the distributed denial of service attack are usually compromised by malware and become a “zombie” within a botnet. A botnet is a group of compromised hosts controlled by a single attacker or an attacker organization. There are many versions of the distributed denial of service attack, such as the “nuke” attack. A nuke attack sends out invalid internet control message protocol (ICMP) packets to the target host until the target host can no longer respond, causing it to crash.

Insider Threat

The insider threat can be the most insidious and hardest to protect. An insider starts with knowledge of the organization that an outsider lacks. The insider may be aware of the physical defenses, network topology, and planned response to an attack. One of the mitigation controls an organization can deploy is the concept of “least privilege.” A user should only have access to the resources that are required to accomplish their assigned tasks. Sensitive tasks (or processes) should adhere to another concept: “separation of duties.” This mitigation effort requires sensitive tasks (or processes) to be separated into two or more parts and completed by different people. Through this approach, the organization is creating a set of “checks and balances” on the employees completing the tasks.

The insider would have to compromise another employee before they could initiate the attack. Another mitigation effort the organization can deploy is that, when the decision is made to terminate an employee, the employee's network credentials are revoked. This will remove the employee's ability to access to any of the organization's resources.

Software Development Security

As application environments become more complex with technological progress, so does application security. While software, environmental, and hardware controls are essential, they cannot prevent problems caused by poor programming practices. Developers can improve the quality of user input by using limit and sequence tests to verify user feedback. Even if programmers follow best practices, an application can still fail due to unforeseen circumstances. To resolve unexpected errors effectively, programmers should first log all the details to prepare for auditing. As the level of protection rises, so does the cost and administrative burden. Some common attack vectors include

- cross-site scripting (XSS). This occurs when an application does not validate or sanitize data inputted into a web page or database. The attacker can then use the browser or database to read and execute the foreign data. This allows them to execute scripts in the victim's browser, compromising web pages, redirecting the victim to a malicious site, or hijacking their session.
- structured query language (SQL) injection. This occurs when an SQL query is added to a user's input and is then submitted to a database for processing.

To combat these attacks, developers need to deploy some of the following mitigation efforts:

- validate input. The system must check all data being accepted by the application to ensure they are in the form and format expected. For example, if an application has a field for receiving telephone numbers, there should be a fixed length (depending on the country) of 10–15 characters. Numbers should consist of the digits 0 to 9, with the + symbol being accepted for international numbers. The application should not accept any other characters, nor should the received input exceed the maximum character limit.
- design with security as a feature. The design process should begin with the goal of security in mind.
- principle of least privilege. As discussed earlier, access should only be given to resources necessary to accomplish tasks.
- data sanitization. This occurs before sending the data set to another system. The developer must ensure that the system checks whether the data set meets the form and format requirements of the secondary system. Data sanitization can also ensure the data set meets security-related requirements. This can help to address data leakage as the data set crosses the boundary to the secondary system.



SUMMARY

The open systems interconnection (OSI) model is the conceptual diagram used to create the networking environment we have grown used to. The transmission control protocol and internet protocol (TCP/IP) model is then used to transmit our data. TCP and IP are both protocols. The IP protocol makes sure the data gets to their destination, while TCP comprises two protocols: TCP and user datagram protocol (UDP). TCP is a connection-oriented protocol, and UDP is a connectionless protocol. The domain name system (DNS) handles the translation of the URL address and the IP address of the hosts and servers on the network. With DNS, users can remember the address to their favorite website, such as the FC Bayern team, because it is easier to remember `fcbayern.com` than the IP address of `23.51.180.8`.

Browsers are a significant aspect of the internet and are used on multiple devices. A specific browser is required to browse the dark web: the Tor browser. The Tor network is considered a criminal version of the internet due to its anonymity, being used to sell illegal items.

Insider threat should always be considered, as insiders already have detailed knowledge of the physical and digital realm in which an organization operates. To combat insider threat, there are several mitigation efforts that can be deployed, such as the separation of duties.

LEKTION 4

COMPUTER FORENSICS

STUDY GOALS

On completion of this unit, you will be able to ...

- classify digital evidence.
- locate potential artifacts.
- analyze malware.
- identify the common methods used to exfiltrate data.
- understand the concept of encryption when used by the attacker.

4. COMPUTER FORENSICS

Introduction

Computer forensics is a division of forensics involving the recovery and analysis of data from digital devices. Its goal is to identify, collect, examine, and analyze digital data while preserving their integrity. However, computer forensics is not always about proving a suspect guilty; as a forensic examiner, there is an ethical obligation to find exculpatory evidence that will prove the subject's innocence. A forensic examiner is an unbiased third party in an investigation. In a criminal examination, the findings could result in depriving someone of their liberty. In a corporate investigation, the findings may lead to a criminal investigation or cost someone their livelihood. The conclusions found in the report can have an extraordinary impact on the subject of the investigation.

Computer forensics is not just about finding artifacts. As a member of the information technology (IT) field, an investigator must have knowledge of networking and computer operating systems. Evidence must be preserved, maintaining the chain of custody and presenting it in criminal or administrative proceeding. To be an effective digital forensic examiner, both legal and technical knowledge is required. Understanding how data are created, shared, and saved in the digital realm, preserving the evidence generated in a forensically sound manner, and testifying to explain your findings are key tasks. Sometimes, the ability to talk in front of a large group while answering hard questions posed by attorneys from both sides is the role's most difficult challenge.

4.1 Evidence

There are multiple scenarios where collecting digital evidence for the investigation is important. A law enforcement officer may respond to the scene of a crime, identify potential sources of digital forensic evidence, and then seize those items. A private sector or corporate investigator may be called upon to take an employee's workstation or to respond to the server room (either physically or remotely) to collect the data to be analyzed.

An examiner will respond to the scene and ensure it is secure. They will verify that the crime scene has not been tampered with and document it with photographs. Then, it is time to collect digital evidence. A source of potential evidence is volatile memory. In the past, the data contained within volatile memory were disregarded with a "pull the plug" mentality. As volatile memory is only available while the system is up and running, when the plug was pulled, this was the best practice for criminals, as all these data were lost. However, as the field of digital forensics matured, we learned that this was untrue. To collect volatile evidence, we should start from the most volatile to the least volatile. This is called the order of volatility, and is as follows:

1. Live system
2. Running

3. Network
4. Virtual
5. Physical

The approach to the collection of volatile data assumes the same mindset as for the collection of forensic images. The steps taken in the collection of volatile data are documented because the collection will alter the evidence on the local machine. The investigator interacts with the machine to collect the volatile data, and this interaction will change the evidence. Typically, these changes do not affect the “what” of the investigation. The investigator may be asked questions about these changes while testifying at administrative or judicial proceedings. If the investigator doesn’t know the answer, it could be professionally embarrassing. The changes will be found in random access memory (RAM) and, for some processes, in date and time stamps. Some examples of volatile data to be collected are as follows:

- current time of the system
- networking information (the ARP table, connections, routing table, name cache)
- logged on users
- running services
- running processes
- shared drives
- remote activity
- open encrypted containers

The term “forensically sound” means leaving the smallest footprint of changed data during data collection. The order of the collection of the volatile data is significant because, if the volatile data are collected in the wrong order, pertinent data may be destroyed. RAM is the most volatile, which is why it should be collected first.

However, this may not always be possible depending on the specific set of circumstances you encounter on the scene. If we find there is a destructive process running on the machine and the information we want to collect is being altered or overwritten, we may not want to take the time to collect the RAM, as evidence is already being manipulated. If a remote connection is causing this destructive process to take place, we may want to document and sever the connection and then collect the RAM data. If the attacker is connected remotely and is accessing highly sensitive data, we should decide whether we allow the attacker to maintain access while we collect the RAM or should we interrupt the connection. Questions must be asked here: what if it is not critical information? Should we allow the attacker to maintain access while we process evidence?

Ultimately, the goal of digital forensics is to create a forensic image for analysis, and, under normal circumstances, it is not appropriate to change the digital evidence during the collection process. In today’s environment, this is not always possible; users have easy access to full disk or volume encryption, and it is no longer acceptable to “pull the plug” on computer systems when encryption could be running. At the basic level, encryption is changing the presentation of the data to protect the confidentiality of the information, allowing only the person with the decryption key to access it. All forms of encryption can be broken if the attacker has enough time. With today’s equipment, that time factor is

measured in hundreds of years. As technology advances, with notable increases in processing power, the time factor in decrypting today's top-level encryption decreases. What was considered strong encryption in the 1990s is now regarded as quite weak. That is why it is imperative not to pull the plug on a system when it is possible encryption is being used. Without access to the decryption key, we cannot get at the data.

Chain of Custody

Maintaining the chain of custody is critical for evidence preservation and authentication. The chain of custody establishes who had access to the evidence, when they accessed it, and why. The National Institute of Standards and Technology (NIST) offers a generic chain of custody form to be used and adjusted as needed (National Institute of Standards and Technology, n.d.). The goal of this form is to track digital evidence and maintain control of the evidence so that the investigator or third party can authenticate it later. An additional option is to make a permanent mark on the items being seized and examined, but only if this can be done without reducing the value of the item. Hard drives can be marked as "hard disk drive" (HDD) with the date and the initials of the examiner seizing the device (HDD-XXX). The examiner will refer to the device as HDD-XXX for the rest of the proceedings. If the device cannot be marked without permanently reducing its value, such as an iPad, a permanent marker is used to write "mobile device" (MD) on an adhesive label that is attached to the device (MD-XXX).

The investigation is not performed on the original evidence; a copy is created for examination purposes. This is to ensure that the investigators do not make changes to the original evidence. There are three choices for making that working copy:

1. Forensic copy. This is a straight bit-for-bit copy of the source to the destination. This is not common in today's environment. It is important to ensure that the destination device does not have old data from previous investigations. This eliminates the chance of cross-contamination between the current investigation and a past investigation. With this method, the examiner can recover deleted files, file slack, and partition slack.
2. Forensic image. This involves creating a bit-for-bit copy of the source device with the data stored in a forensic image format. The source data is wrapped in a protective digital wrapper. With this method, examiner can recover deleted files, file slack, and partition slack.
3. Logical forensic image. Sometimes, there are restrictions regarding the specific datasets the examiner can access. These restrictions prevent access to the entire container and prevent them from creating a bit-for-bit copy. This can occur when extracting data from a server, as the server cannot be shut down to create a forensic image from the source hard drives. Logical copies of the files and folders pertinent to the investigation are then made instead. With this method, the examiner will not be able to recover deleted files, file slack, and partition slack.

Write Blocking

Write blocking is at the core of the forensic examination environment; we want to ensure we do not change a single bit of data on the source device. Evidence handling is an essential function of the examination process and must ensure that we meet all the requirements to avoid altering or damaging the evidence. If a device is connected to a Windows-based computer system, for example, the operating system will scan and write to that device. To prevent this, and the alteration of the data on the source device, the examiner must use a write blocker. The examiner has a choice of utilizing a hardware-based write blocker or a software-based write blocker.

Hardware Write Blocker

As the operating system issues commands, it will read and write from the source device. A hardware write blocker is a device that intercepts and prevents any modification to the source device. It is a physical connection between the computer and the source device. There are also standalone and self-contained hardware write blockers that allow the examiner to connect to the source and destination device and then create the forensic image. NIST has created the Computer Forensics Tool Testing Program (CFTTP), where they list the testing results for hardware write blockers (National Institute of Standards and Technology, 2019).

Software Write Blocker

Software write blocking is where a change is made in the operating system as a result of writes made to the device. For a Windows-based system, there is a registry change that can be made to prevent modification rights for attached universal serial bus (USB) devices. This requires a hard drive dock to be connected to the system. Another software-based option is to utilize a bootable operating system, such as Paladin or the Windows Forensic Environment.

Forensic Imaging

An examiner never wants to change the source device of digital evidence, so a digital forensic examination is not carried out on the original device. The examiner should perform the analysis on a copy. The forensic copy made will also be evidence and has the same legal status as the source device. For the forensic copy and the forensic image, the examiner will acquire every bit on the source device. If there are restrictions, the examiner will only be able to copy the logical files. The examiner will put the logical files into a forensic container (such as an e01 or dd image); this encapsulates the data set in a protective format to prevent any alteration to the data.

This is not a backup, as might be seen in the corporate environment. In the corporate environment, the system does not create backups in a forensically sound manner. The corporate backup lacks information about file slack, unallocated space, deleted files, or any data that are not maintained by the file system. There are two common formats of forensic images used by government and corporate digital forensic investigators: dd image and expert witness format (e01).

dd Image

`dd` is a Unix command and is considered the oldest imaging tool available. While `dd` started as part of the Unix operating system, it has now migrated to other operating systems. There are versions of `dd` that work on Linux, Windows, and macOS, all in relatively the same manner. The command is designed to copy data from a source device to a destination device. The raw image format is a bit-for-bit copy of the source device. It may store the data as single or multiple files. No metadata are stored in the image file(s). Depending on the tool being used for imaging, it may create a text file containing details about the source device and the creation of the image file. This file can include information about the hardware and software used, source and destination information, and the pre- and post-hash values.

A raw image format will contain solely unmodified source data. The raw format is a fundamental image file that all forensic and non-forensic tools can read. A point of consideration is that the raw format does not use compression. When using this format, we must ensure that our destination device is large enough to store the contents of the source device. Raw images may have the following file extension: `.dd`, `.raw`, and `.img`. Some of the tools that specifically make a raw image are `dcfldd` and `dc3dd`.

`dcfldd` is a version of the `dd` command that has incorporated additional features such as on-the-fly hashing, status output, disk wiping, verifying an image or wipe, multiple outputs, splitting outputs, and pipe output and logs. `dcfldd` has an issue with imaging faulty drives. NIST has reported that `dcfldd` will misalign the data in the image after it encounters a faulty sector on the source device (Homeland Security Science and Technology, 2013).

`dc3dd` is another version of the `dd` command. While `dcfldd` is a fork of the `dd` command, `dc3dd` is a patch of the `dd` command. While both options are similar, they have slightly different code bases and feature sets. When the `dd` command is updated, `dc3dd` is also automatically updated. Some features available on `dc3dd` include on-the-fly hashing with multiple algorithms, the ability to write errors directly to the file, combined error log pattern wiping, verify mode, progress reports, and split output.

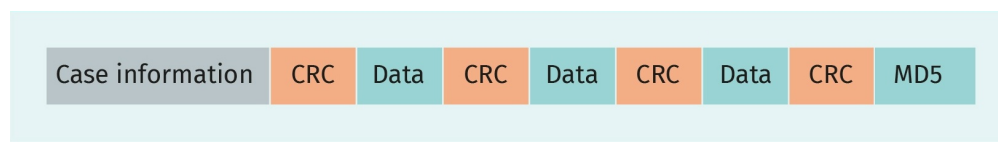
Expert Witness Format

Another forensic image format is the EnCase evidence file, commonly referred to as `e01` or the expert witness file format. Just like the raw image format, the `e01` format is a bit-for-bit copy and includes metadata within the file format. EnCase Forensics uses the forensic image file known as the `e01` format, expert witness format (EWF), or the EnCase image file format.

The `e01` file format is a forensic image and encapsulates the data from the source device to prevent them from being altered. The `dd` image will only contain data from the source device; the `e01` forensic image contains “header information,” or metadata. This information includes evidence names and numbers, acquisition dates and times, investigator notes, and information about the forensic tool used to create the forensic image. There are also security features incorporated into the forensic image file. As the file is created,

there will be a cyclic redundancy check (CRC) calculation in every 64 sectors. The CRC is used to detect changes in the data stream. The CRC creates a fixed-length value (like the output of a hashing algorithm) from a variable input. The CRC values are stored within the forensic image. When the forensic image is loaded into forensic software, the CRC values are checked and validated.

Figure 71: e01 File Format



Source: Created on behalf of IU (2022).

The figure above shows the format of the e01 file format. The case information is placed at the beginning of the forensic image. The process creates a CRC value for the header information. The process receives the following 64-sector block, which is added to the image file. A CRC value is created for that 64-sector block and added to the forensic image. A data block is added, then a corresponding CRC value is created and added; this repeats until the source device has been acquired. After the process has reached the end of the source device, an MD5 hash value is generated from the data blocks, and the process adds the value to the end of the forensic image. With this format, compression is possible, but significantly more time is needed to create the forensic image.

Mobile Devices

Mobile forensics is an offshoot of traditional computer forensics. It is much more challenging to collect and analyze evidence from a mobile device than from a conventional hard disk drive or an solid-state disk (SSD). Part of the challenge with mobile devices is that everything classified under the user experience is proprietary information. The operating system and the file system are based on the vendor's vision for the user experience. For example, let's say someone purchased an iPhone 10 when it was first released, and then another iPhone 10 at the very end of the product life cycle. While the devices are visually similar, and provide the same user experience, they can differ significantly in how they save data and integrate with the hardware.

Numerous commercial tools can be used to carry out mobile device extraction and analyze the data set. The vendors of these commercial tools spend thousands of hours on reverse-engineering the mobile device operating in file systems. A single mobile device forensic tool may not extract or process all the artifacts. One of the current best practices in mobile forensics is to use multiple tools to extract and analyze the data. This way, we can validate the data that have been acquired and potentially identify artifacts that one or more of the mobile device forensic tools might had missed.

Mobile devices are designed to communicate using a variety of network media. A mobile device can use cellular phone signals, **Bluetooth**, or Wi-Fi to communicate. When a mobile device is taken into custody, one of the first considerations is how to isolate the device from these many network media. The goal of isolating of the mobile device is to prevent a

Bluetooth

This is a short-range wireless technology used for transferring and exchanging data.

user from accessing the content remotely and changing or deleting evidence. Mobile devices also allow a remote wipe to be initiated, which would destroy the data. Some options for isolating the mobile device include using a container that will block the reception and transmission of the network signals. Another option is for the examiner to place the device into airplane mode, which will turn off the network communication channels. To do this, the examiner must have access to the device's settings. If the screen is locked and the passcode is not known, airplane mode will not be an option.

Once the device has been isolated from the different network media, the examiner can begin with data acquisition and processing. Due to the nature of mobile devices, and the security protocols enabled by the mobile device vendor, it is not possible to create the type of forensic image that is commonplace in computer forensics. User encryption and the vendor's security protocol isolate applications and the data that the different applications can access. Generally, the examiner can complete a "logical extraction." A logical extraction is a collection of the logical storage objects of the device. Deleted data cannot be recovered. It is possible, however, when examining the data stored in an SQL lite database, for the examiner to retrieve data that were deleted in the cells and tables and have not been overwritten yet. The other option is a "physical extraction," which will include the data available in a logical extraction and data that are not typically available to the user.

Mobile devices vendors are constantly increasing the security protocols on mobile devices and enabling them by default. This is a positive for the device user, as their data have a greater level of protection in the event the mobile device is lost or stolen. This is a negative for the mobile forensic examiner, though, as it restricts their ability to access the user's data and carry out an analysis. Security features such as encryption are enabled without a "backdoor." This means that, once the security feature has been enabled, it cannot be bypassed unless the proper security key is provided to the mobile device's operating system. Some tools exploit either physical or software vulnerabilities to gain a greater level of access than the mobile device vendor has granted. One such organization is Gray Shift and their tool GrayKey. GrayKey was deployed in 2018 and is sold exclusively to law enforcement organizations (Brewster, 2018).

4.2 Malware

Malware is a general term for any software that has created an unwanted condition or damaged a computer system. Within this broad term, we find, for example, applications designed to steal credit card numbers and applications that steal and encrypt the user's data, holding their files ransom. Categories that can be used for more specificity when describing malware include

- backdoor. This is a piece of code used to infect the system, which then allows the attacker to take control of it.
- botnet. This is a collection of computer systems that an attacker has compromised and controlled, commonly used in a distributed denial-of-service (DDoS) attack.

- downloader. This is part of a multi-stage malware code used to bypass security controls and then download additional code. It is considered to be a sophisticated attack.
- file wipers. These are programs designed to delete files or change the master boot record (MBR).
- keylogger. This is code designed to record the user's keystrokes.
- ransomware. This is code that encrypts the contents of the user's system. The attacker then demands payment to decrypt the contents.
- rootkit. This is used to camouflage other malicious code, for example, a **remote access Trojan** (RAT).
- trojan. This is malware that appears to be a legitimate file.
- virus. This is a piece of code that infects existing programs. It cannot self-replicate.
- worm. This can be a component of a virus. It can self-replicate and attack multiple systems.

Remote access Trojan
This is a malware program used to create a backdoor for administrative control over the victim host.

Analyzing malware is a branch of network and computer forensics that can be highly technical and require specialized knowledge. Generally, there are two different methods in malware analysis: static and dynamic. Like anything dealing with digital evidence, there are advantages and disadvantages to both types of analysis. The following seven steps can be a guide when analyzing malware:

1. The analysis must be done in a controlled environment.
2. The malware's behavior should be observed when activated in the operating system.
3. A static code analysis of the malware should be conducted.
4. A dynamic code analysis of the malware should be conducted to determine whether there are any differences from the static analysis.
5. If the malware is "packed," the analysis should be continued while the malware is "unpacked" (packed malware has been modified by being compressed or encoded).
6. The analysis objectives should be clearly defined, and analysis must continue until the objectives have been met.
7. An analysis report must be completed (which includes return the controlled environment to the same state it was before the analysis).

Static Malware Analysis

Static malware analysis is the examination of the malware without executing it on a system. The analysis can be accomplished using different techniques, such as

- fingerprinting, which generates a hash value for the file and compares the hash value to a database of known malware.
- antivirus scanning, which involves scanning the malware using multiple antivirus programs. It is possible that a vendor had encountered the malware before and created a signature for their antivirus program.
- string analysis, which involves examining the code for data strings. Programmers will leave artifacts of interest within the code, such as data encoded in clear text, IP addresses, and error messages. The data in the strings may lead the analysis to other locations on a network or to the command-and-control server.
- packer analysis, which investigates whether compression or encoding was used by the attacker to hide the malware signature.

- code disassembly, where, by parsing through the malware code, the analysts can determine what the malware does.

The advantage of static malware analysis is that the analysis does not execute the malware. The disadvantage is that the exam can be laborious, tedious, and time consuming.

Dynamic Malware Analysis

In static malware analysis, the focus is on examining the code or file attributes to identify the malware and compare it to other databases. In dynamic malware analysis, the focus is on executing the code in a controlled environment, which allows the analyst to monitor the code and observe how the code will compromise the host. An advantage of dynamic malware analysis is that it is much faster than static malware analysis. Additionally, if the malware is “packed,” the execution of the code removes the requirement of the analyst trying to remove the compression or encryption. In this sense, the malware naturally sheds the encryption or compression, allowing access to the hidden code. The analyst has two different types of analysis that can be deployed: defined point analysis and run time behavior analysis.

With defined point analysis, a testing operating system is created with the express purpose of being compromised by malware. The operating system is the baseline for this analysis. The malware is then introduced to the operating system and executed. Once the malware has been executed and has completed its programmed tasks, the analyst compares the current state of the operating system to the pre-malware state of the operating system. The analyst can then compare the two states of the operating system and identify any changes that were caused by the malware. This can be done using traditional forensics, where the volatile memory is captured before the analyst forensically images the storage devices. With run time behavior analysis, a testing operating system is created for the express purpose of being compromised by the malware. The analyst uses utilities to monitor the behavior of the malware in real time.

When conducting malware analysis, remember that it should never be done in a “production” environment. There should be specific hosts used for malware analysis that are never connected to the production network. This controlled environment is also known as a sandbox. Depending on the size of the organization and the operating systems, hosts, and servers deployed in the organization, the sandbox environment can grow rather quickly. The sandbox environment should reflect the equipment being used by the organization.

Memory Forensics

Random access memory (RAM) contains various artifacts relating to how the computer is configured and how it was being used before an investigation. This information includes (but is not limited to)

- recently typed commands,
- passwords and encryption keys,
- unencrypted data,

- IP addresses,
- internet history,
- instant messages,
- emails, and
- malware.

An analyst can capture the information from RAM in its entirety. There are several tools that aid proficiency and accuracy. These tools can capture the contents of RAM and store them in a binary file for later analysis. Several tools can import this binary file and extract information relevant to the case you are working on. Deciding which tool to use depends on the specifics of the investigation. One of the primary considerations is which tool leaves the least significant footprint on the system.

A memory dump is a file that contains the complete contents of the system's RAM. The dump file will be the same size as the RAM installed on the system. A system with 8 GB of RAM will have an 8 GB memory file. To image RAM, three things are required:

1. An external storage device
2. An interactive login for access to the desktop
3. Administrator privileges.

Three standard tools for acquiring RAM are as follows:

1. **Dumplt.** This is a combination of Win32dd and Win64dd in one executable. The user executes the file, which creates a copy of the physical memory. Dumplt can be deployed on a USB key for incident response operations. Dumplt leaves the most diminutive significant footprint in RAM.
2. **wine/wine64.** This is a command-line utility that allows the investigator to dump the memory from a system. This leaves a small footprint in RAM. However, running the wrong command may cause the system to lock up.
3. **FTK Imager Lite.** This is a graphical user interface (GUI)-based utility that allows a user to dump the memory of a computer system running either a 32-bit or 64-bit Windows operating system. This tool is easy to use and deployable on a thumb drive. This tool also allows the mounting of binary dump files for viewing. It leaves the most significant footprint in RAM.

Once we have acquired the system memory, one of the factors we may encounter is that a system is a virtual machine. If it is a virtual machine, we have some additional options to consider before starting the process of collecting the RAM. If the virtual machine is powered down or hibernating, then the data in RAM are not volatile. We may find that the contents of the RAM have been saved as a hibernation file (a virtual machine memory file). If the virtual machine is up and running, then we can create a snapshot, clone the virtual machine, or clone the virtual machine hard drive and memory files.

Ideally, an investigator should also consider the potential impact on the system before collecting RAM. There is a risk when this process is started because most operating systems do not natively support a mechanism for acquiring the contents of RAM. The act of collecting from RAM may destabilize the system and cause it to behave in an unpredictable man-

ner. If the system is mission-critical, where the system can only be shut down or rebooted as an extreme measure, the analyst will need to justify assuming that risk. If the system is responsible for providing life-saving services, the risk of destabilizing the system (and potentially crashing it) is not justified. Information sharing is critical in this situation. The analyst must inform the person or persons responsible for assuming that risk of the potential danger so that they can make a well-informed decision.

Deciding when to start the collection of memory can depend on many factors, for example, whether we are collecting from a suspect computer system. If we are accessing a suspect machine, we can plan the collection for when the attacker is logged into the system. This will allow us to collect artifacts about the login session, any cloud services, or remote storage, meaning the potential for finding encryption keys increases. For a victim host, the best approach is to wait until the suspect is not logged into the system. If the suspect is accessing the system when the data collection begins, they will notice our activity. The last thing we want is for the suspect to gain knowledge about the investigation and start destroying evidence.

When we decide to collect the memory, we want to ensure that the dump file is saved to an external device. This way, we will not overwrite any unallocated data on the systems attached storage devices. We must ensure that your destination device has a greater capacity than the amount of installed RAM. Depending on the operating system, we will want to make sure that we format the external device with a file system that can accept very large files. For example, if the collection program is from a Windows-based operating system with 16 GB of RAM, we will need to format the device as an NTFS partition. We should not use the same external device to collect RAM from multiple systems (during the same incident). Some malware can be spread through external media; we do not want to be responsible for infecting systems that were not infected before. With that in mind, we cannot connect the external device directly into our examination system. Instead, we create an intermediary system that we can connect to the external device. This gives us time to go over the external device to determine whether there are any abnormalities. The intermediary system should then be connected to the exam machine via an isolated, non-switched network. After the memory dump has been preserved, the analyst should sterilize the external device to remove any artifacts on the device.

Software-based memory collection tools follow the same process where a kernel module is used to map the physical addresses of the desired spaces into the virtual address space of a running task on the system. The collection tool can then access the data set via the virtual address space and send it to external storage. Most tools can extract relevant information for a case. While a tool like Volatility has around 60 plug-ins available, we only need to focus on a handful of them (Volatile Community, 2019). Forensic tools like Internet Evidence Finder make locating cached internet history items very easy.

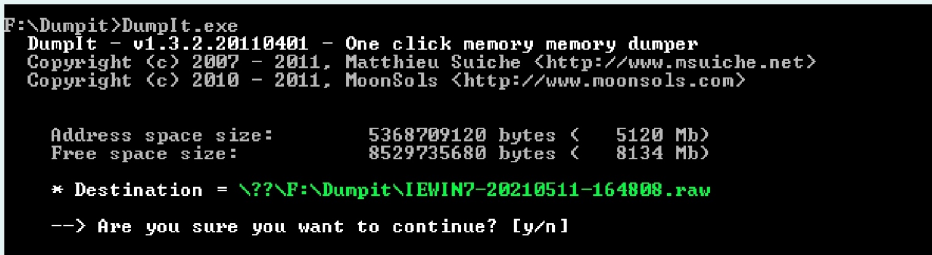
Memory Dump File

To collect RAM, the program must have administrator access. The system must be unlocked or run from an account with administrative privileges. If we cannot gain the proper privileges to run the software, we can try to use the hibernate function, which is part of Windows 8 and later. If we only have access to a user account, hibernate can be

forced at the command line using any account. This option must be used on a case-by-case basis, as it can change the system's internal files (registry and event logs). To do this, we can press the Windows key and "S." Enter cmd . exe. At the command prompt, we enter shutdown-h. The computer will write the contents of RAM to the hard disk and enter hibernation.

To create a memory dump file, the analyst should use a tested and validated tool. In the following figure, the DumpIt utility is ready for the collection process. This version of DumpIt is a straightforward utility. Once the user selects "yes" to continue, the utility collects the memory and saves it in the same directory in which it resides. This translates into a tiny footprint used by the operating system in the collection of RAM. The execution of this program will overwrite only a small portion of RAM. On the other end of the spectrum, if the analyst were to use a utility with a graphical user interface (GUI), that would create a larger footprint, and the system would overwrite more data in RAM. When DumpIt has completed, it displays a "success" message.

Figure 72: DumpIt Utility



```
F:\DumpIt>DumpIt.exe
DumpIt - v1.3.2.20110401 - One click memory memory dumper
Copyright (c) 2007 - 2011, Matthieu Suiche <http://www.msuiche.net>
Copyright (c) 2010 - 2011, MoonSols <http://www.moonsols.com>



Address space size:      5368709120 bytes <  5120 Mb>
Free space size:        8529735680 bytes <  8134 Mb>

* Destination = \??\F:\DumpIt\IEWIN7-20210511-164808.raw
--> Are you sure you want to continue? [y/n]
```

Source: Created on behalf of IU (2022).

In the following figure, the directory containing the raw memory dump is depicted. DumpIt has used the naming convention of the hostname (IEWIN7), followed by the date (yyyy-mm-dd) and, finally, the Greenwich mean time (GMT) that the analyst collected the memory. The memory dump will be a little larger than the size of the installed RAM. In this collection, a machine with 4 GB RAM will produce a file roughly 5 GB in size.

Figure 73: File Directory Containing the Memory Image

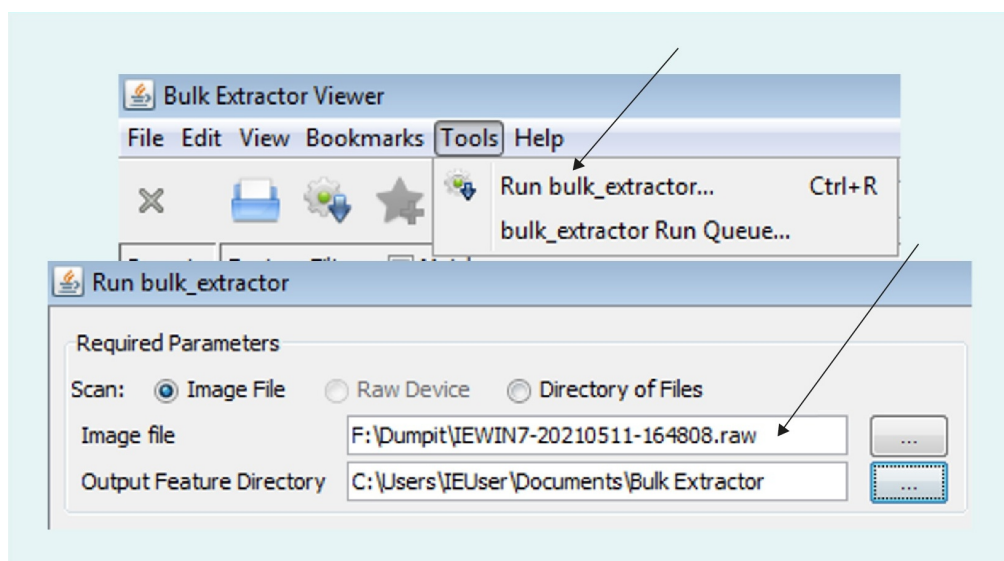
Name	Date modified	Type	Size
 DumpIt.exe	5/3/2011 5:41 AM	Application	203 KB
 IEWIN7-20210511-164808.raw	5/11/2021 9:52 AM	RAW File	5,242,880 KB

Source: Created on behalf of IU (2022).

Bulk extractor is a tool that scans a standard input (such as a memory file) and creates histograms of the artifacts it finds. This tool will ignore any structure of the file and process the dataset in parallel. An advantage of ignoring file systems is that the Bulk Extractor can process any image type file.

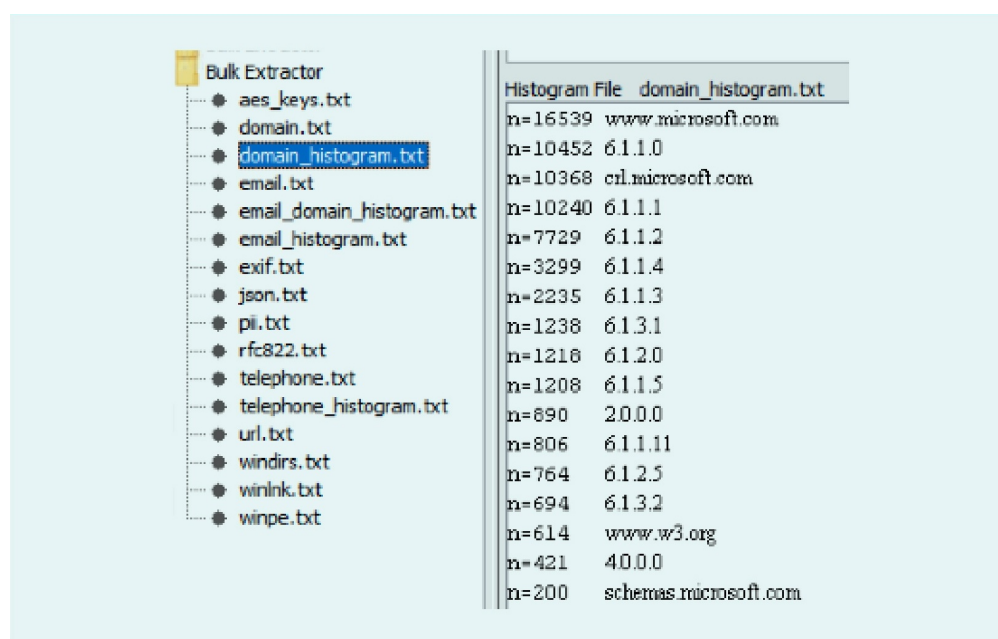
From the bulk extractor viewer, the user will need to access the “tools” menu and select the “run bulk extractor...” option. The following window will ask for the source to be examined; in this case, it is the memory file that was just created. The user will also need to specify the output location so the bulk extractor can place the files created while processing the raw memory file. This is shown in the following figure.

Figure 74: Bulk Extractor Settings



Source: Created on behalf of IU (2022).

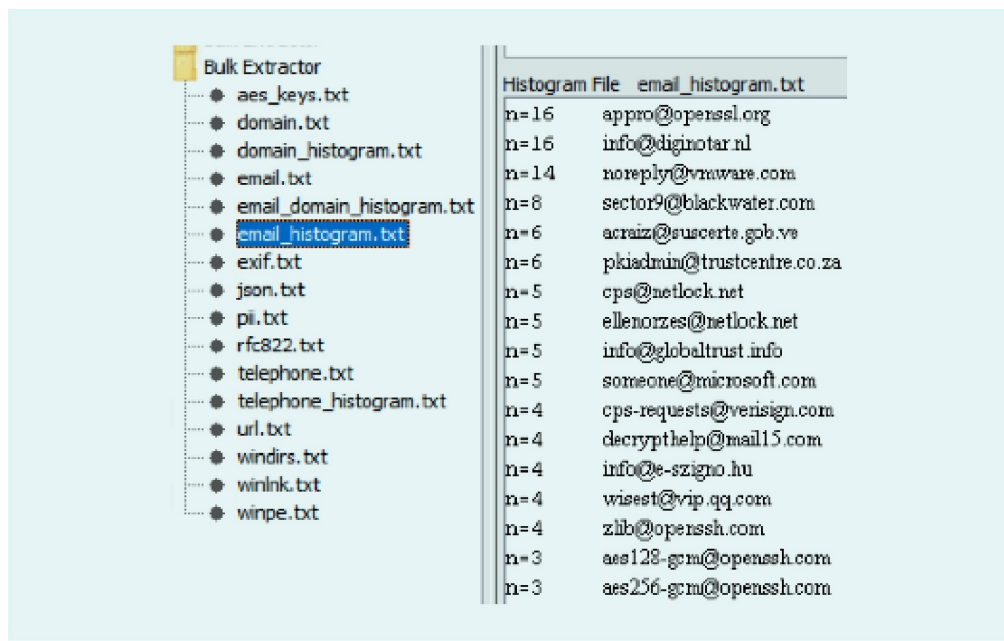
Figure 75: Bulk Extractor Results: Domain Histogram



Source: Created on behalf of IU (2022).

Once the bulk extractor has finished parsing the raw memory image, we can view the results. In the previous figure, the domain histogram has been selected. The right-hand column is a list of all the IP addresses and URLs that were found in the RAM. In the following figure, the email histogram category has been selected on the left, with the results listed on the right. These are the email addresses found in the RAM, which also lists the number of times the program found a specific email address in the memory file. The top entry shows that the bulk extractor found the email address “appro@openssl.org” 16 times. These findings do not indicate any characterization of the user’s behavior; the artifacts still need to be analyzed and placed into context.

Figure 76: Bulk Extractor Results: Email Histogram

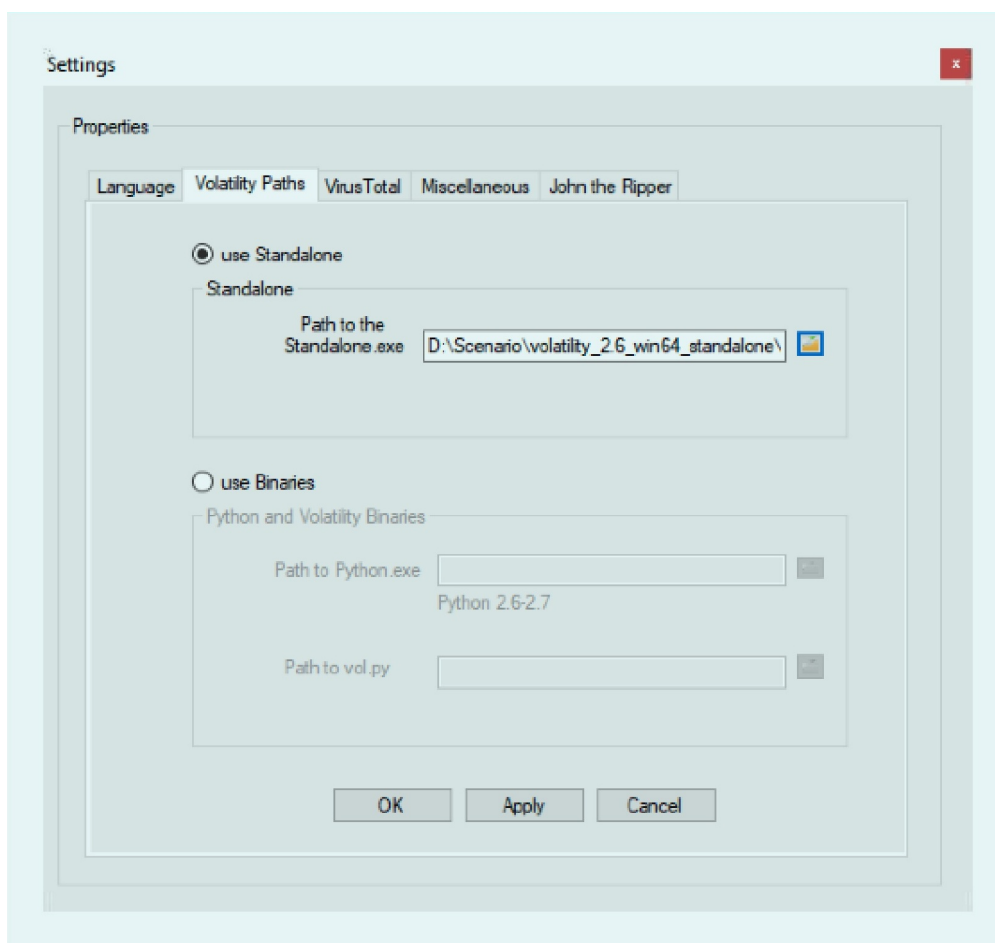


Source: Created on behalf of IU (2022).

The Volatility framework is a collection of open-source tools used to analyze the contents of RAM. It can be installed and used on all operating systems that support Python. Volatility is not used to collect the contents in RAM; it is an analyze-only tool. By default, Volatility is a command-line tool and does not have a GUI frontend. Some third parties have created a frontend for the Volatility framework, such as VOLIX II v2. Several “cheat sheets” are available for download from organizations such as SANS. Volatility also comes as a standalone executable for the Windows operating system. The standalone version includes the Python interpreter and the dependencies Volatility requires. An analyst can also run Volatility from an external device, and third-party scripts are available to help automate the analysis of the memory dump file.

Several options must be specified when using VOLIX II as a GUI frontend for Volatility. As shown in the following figure, the most important task is to identify the location of Volatility. There is an option to use the standalone version or to use the separate Python and Volatility binaries. Here, the user selected the standalone version, and the path to the location has been chosen.

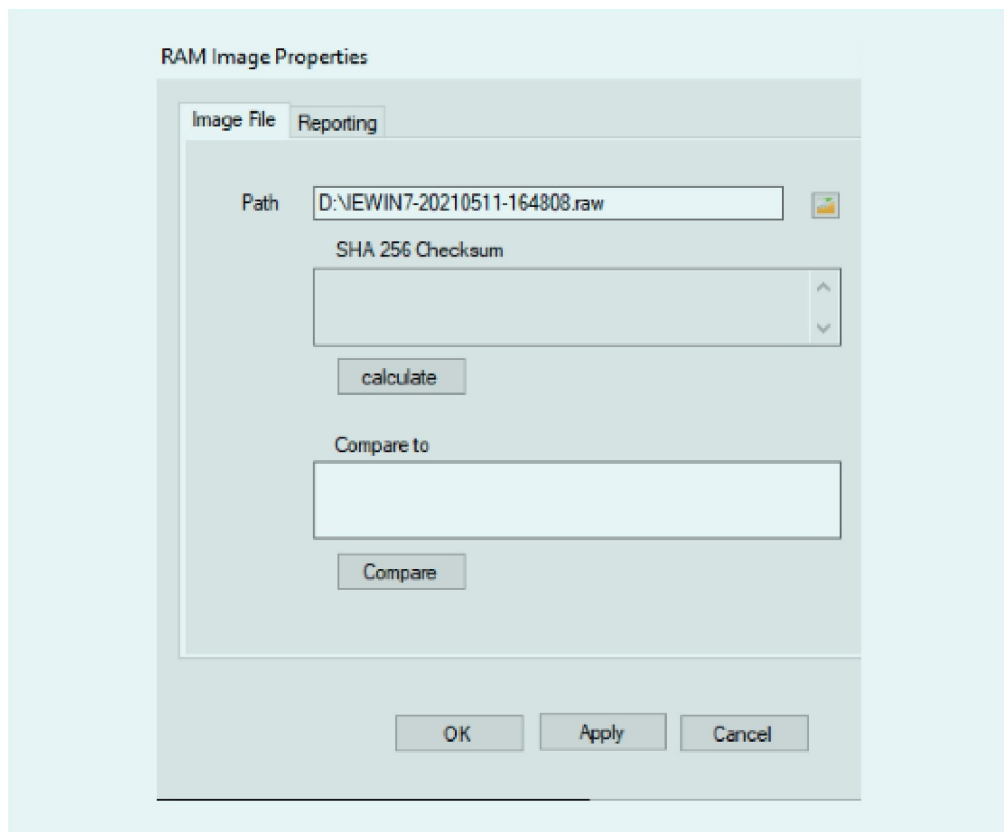
Figure 77: VOLIX II Setup Screen



Source: Created on behalf of IU (2022).

The next option (shown in the following figure) is to complete the path to the memory image that is to be examined. There is also an option to create a secure hash algorithm (SHA) 256 checksum and to compare it to another file. This is to ensure that no one has changed the contents of the memory file.

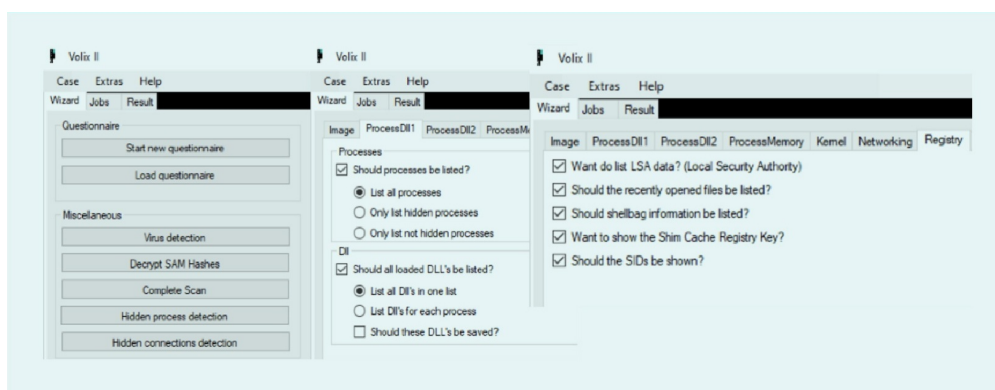
Figure 78: VOLIX II Setup Screen: Image Location



Source: Created on behalf of IU (2022).

The next step is to tell Volatility what to look for when analyzing the memory image. Using the GUI, the user can check the box for which options they want to enable. As shown in the following figure, the options for listing all the processes have been selected, as well as all the loaded .dlls. The user has also selected some additional options, such as the local security authority (LSA), recently opened files, Shellback information, the Shim cache, and identifying the security identifiers (SIDs) found within the memory file. This selection will allow the analyst to understand what applications have been running on the system before the analyst collected the data in RAM.

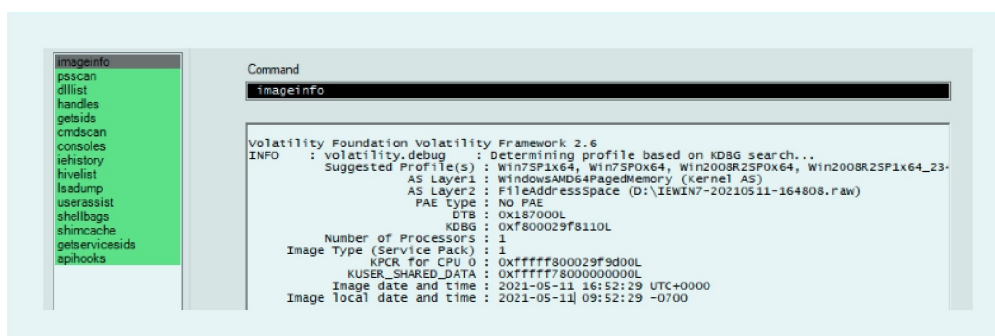
Figure 79: VOLIX II Setup Screen: Options



Source: Created on behalf of IU (2022).

Once Volatility has completed running the plug-ins that have been selected, the analyst can view the results in the “results” tab (as shown in the following figure). The first plug-in selected is `imageinfo`. The GMT date and time of when the image was created is listed, along with information about the operating system, processors, and where the image file was stored.

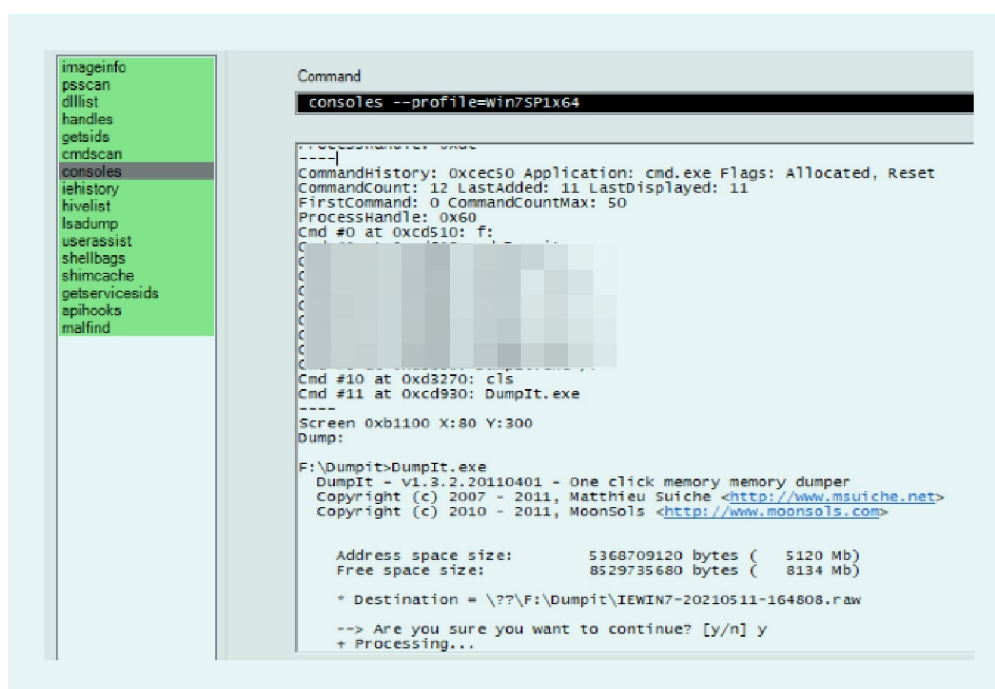
Figure 80: VOLIX II ImageInfo Results Screen



Source: Created on behalf of IU (2022).

“Consoles” is the next plug-in highlighted on the left. The information displayed is the console information that was displayed to the user. Examining the lower portion of the output screen shows the command line of `F:\DumpIt>DumpIt.exe`. The output of the command is shown, which depicts what occurred when the command `DumpIt` was executed on the system, as well as how the user interacted with the console (pressing “Y” to start the memory dump). An analyst can use many plug-ins with the Volatility framework, providing them access to an enormous number of artifacts that they can examine to help prove or disprove their hypothesis.

Figure 81: VOLIX II Consoles Results Screen



Source: Created on behalf of IU (2022).

The collection of RAM is a planned and systematic operation. Every collection will be different due to the wide variety of results that may be encountered. The data found in RAM may also coexist in other non-volatile locations. Artifacts cannot be analyzed in isolation; they must be placed in the proper context so the analyst can investigate what took place.

4.3 Data Exfiltration

Data exfiltration occurs when the confidentiality of data has been violated and the attacker completes an unauthorized data transfer. The attacker can target almost any type of data, but the following are categories of data that the attackers focus on:

- user authentication information. This includes usernames, passwords, and any information that is used for user authentication.
- decision-making information. This includes emails, memos, text messages, and any other information that can show the organization's strategic decision-making process.
- encryption and decryption keys. These are used to open secure containers and violate the confidentiality of the contents.
- financial information. This includes account numbers, credit card numbers, and any information that allows the attacker to gain access to funding resources.
- personal identifying information (PII). This includes user's names, birthdates, government identification numbers, and any information that will allow a third party to impersonate a user.

Why would an attacker want to access and exfiltrate data? The compromise dataset can act as a type of currency. The contents can be sold and resold to any number of bad actors, who can then use it to forward their criminal efforts.

There are many methods that an attacker can use to compromise a data set and exfiltrate it. The insider threat can be very dangerous when it comes to data loss because the insider threat may already have physical access to the workstations or the servers. While inside the organization, they can use their social engineering skills to compromise their coworkers' credentials. The insider may decide to increase their privilege level to gain access to data typically out of reach. With access to workstations or servers, the insider can attach an external storage device and copy the data onto it. There have been many incidents where a trusted insider used their position within the organization to access and exfiltrate data:

- Anthem, Inc. reported in 2017 that an employee of the consulting firm LaunchPoint Ventures had accessed the medical records of over eighteen thousand people and emailed them to a third party (Olenick, 2018).
- The Federal Deposit Insurance Corporation (FDIC) reported that an employee compromised forty-four thousand customer records. Lawrence Gross, the chief information officer of the FDIC, stated that an employee downloaded the customer records onto the personal storage device. The data loss was characterized as without malicious intent. The employee had legitimate access to the data set as part of their duties. The employee ended their employment on the same day they placed the data onto the external device. The breach was discovered three days later when the security protocols identified the data transfer (Davidson, 2016).
- General Electric (GE) was attacked by a trusted insider, Jean Delia, who accessed and stole over eight thousand files with the intent to create a new company and compete with GE. Delia also convinced an employee in the IT section to escalate their privileges in order to access a greater number of documents than they would normally have access to. Delia used email and cloud storage for data exfiltration (Federal Bureau of Investigation, 2020).

There are several vectors that an attacker can use to compromise an organization's network. Social engineering is a non-technical method that can be accomplished by phone, email, or in person. The attacker attempts to manipulate the employee into sharing confidential information. The attacker can call the organization and pretend to be a legitimate member of the organization, a vendor, or a representative of a public utility or government agency. The attacker specifically targets the organizations' leadership to have them reveal information about high-level employees or customers.

Attackers can also target organizations in a phishing attack. This occurs when the attacker sends a large number of emails to members of the organization. The attacker wants the email recipient to click on a link and sign into an illegitimate site, allowing the attacker to record the user's credentials. Another option is to have the user open an email attachment that will install malware on their system.

Once an attacker has compromised the network, the next challenge is copying the data set and moving it outside of the organizational confines. The attacker tries to hide in plain sight by disguising their traffic using the organization's legitimate applications with a "hidden tunnel." An organization can use a hidden tunnel to allow a third-party application to connect to the network. This is common in financial organizations using analytical tools, cloud-based financial applications, or ticker feeds from the financial markets.

The attacker can break the dataset into small pieces to help avoid detection as data are transmitted. This method allows the attacker to avoid common security protocols like a firewall or malware scanning. The attackers can embed their communication channel into "GET" requests (the GET request is used to retrieve specific information from the server) or other network artifacts such as headers or cookies. They then hide the illicit traffic among the legitimate traffic of the organization.

Tunneling is similar to using a virtual private network (VPN) to create an encrypted tunnel that connects a user outside of a network to the organization's internal network. The VPN protects the data by using encryption protocols. The encryption does not allow the data to be read by a third party. After an attacker has compromised the network, an IPsec connection can be used as a port of entry for the attacker. The initial point of compromise is at the endpoint of the organization's VPN. If the organization has multiple sites, multiple VPN connections may connect to these sites. The attacker can then use the site-to-site connections once the organization's network has been compromised. The VPN tunnels provide the attacker with the ability to conceal their traffic from the network administrators. Secure shell (SSH) can also create an encrypted tunnel between the source and the destination hosts. The SSH authentication keys will be a specific target of the attacker when the network is first compromised. The authentication keys will allow the attacker to escalate their privileges to gain access to applications and servers.

Another technique the attacker may use for exfiltration is an external web service. A legitimate web service can provide a method that may conceal the attackers' activity within the legitimate traffic communicating with the service. The organization has already created exemptions for hosts on the network to communicate with the service that is not blocked by firewall rules. The web service may also use secure sockets layer and transport layer security (SSL/TLS) protocols, which the attacker can use as data are being exfiltrated. The potential mitigation control that an organization could deploy is monitoring the amount of data being sent and received between the hosts and service, creating a baseline of the traffic and identifying any hosts that suddenly start using the service.

The attacker may use scheduling for the exfiltration of data from the organization. Limiting the exfiltration to only be active when the network is active will allow the attacker to blend in with legitimate network traffic. The attacker can also schedule the exfiltration to be done at specific intervals, such as ten minutes during regular business hours. Deep packet inspection (DPI) can inspect the active data streams in the organization's network. DPI can create a baseline of the network and application behavior to help identify service issues or the exfiltration of data by an attacker.

Encryption is a tool that legitimate organizations use to protect the confidentiality of the data on their network; the confidentiality of encryption also works in favor of the attacker.

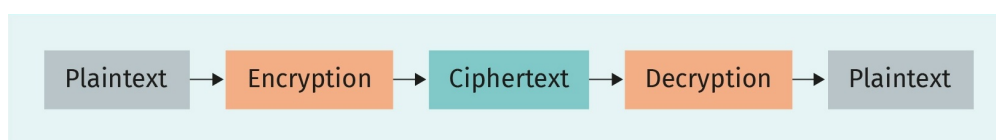
4.4 Attacks Against Computer Forensics

Attacks on digital forensics, also known as anti-forensics, have increased as the importance of digital evidence has been recognized in a judicial or administrative proceeding. The anti-forensics tools and techniques being used by subjects can be an obstacle in determining what occurred during an incident. The digital forensics community must create tools and techniques to combat anti-forensics. Attackers use tools that were not designed as an anti-forensic tool but are maliciously using the tool's functions to hide their activity. An example of this is the 2015 San Bernardino, California, terrorist attack. The government wanted to examine the terrorist's iPhone but could not do so because the terrorist had locked the phone with a four-digit PIN. The government could not brute force the PIN because, after multiple unsuccessful attempts to unlock the phone, it would cause the phone to wipe itself. The phone was also using strong encryption to protect the data. Neither one of these techniques was designed as an anti-forensic tool, but the intended use was to protect the user's privacy (Selyukh, 2016). Next, we will explore some attentional techniques that an investigator may encounter during a digital forensic investigation.

Data Hiding

Subjects can use various methods to hide data from other users while having them still be available for their use. A common privacy tool used as an anti-forensic technique is the use of encryption. Encryption is used to transform readable data into unreadable data. Readable data are known as plaintext, and encrypted data are known as ciphertext. This process of encryption and decryption is the cryptosystem and can be implemented via hardware or software applications. The following figure is an example of the general functions of the cryptosystem.

Figure 82: Cryptosystem



Source: Created on behalf of IU (2022).

The cryptosystem uses an algorithm in the encryption and decryption process. The algorithm (also known as the cipher) is the determining factor of the strength of the cryptosystem. The algorithm is a rule set that determines how the encryption and decryption will be executed. The “key” is the secret value (crypto variable) that is used with the algorithm. The key space determines the strength and complexity of the key.

Steganography is hiding data in another file, known as a carrier. This anti-forensic technique conceals the very existence of the hidden dataset. An application is used to place a file(s) or a message into a carrier file. The carrier file is typically a digital image, such as a .gif or .bmp but can also be an audio file. The carrier file can then be uploaded to a cloud destination or emailed to another user. Once the intended recipient can access the

file, the same program is used to extract it. If we were to look at the carrier file before and after the dataset was placed inside, there would be no visible difference. There are two general categories in which a user can implement steganography: insertion and substitution.

Insertion makes use of the unused space in the carrier file to store the data. When using the insertion method, there is no alteration of the file contents. This will cause an increase in the file size, which may be a flag to investigators.

The substitution method uses the least significant bit of the carrier file to embed the data set. The steganography tool identifies the least significant bit of the content and substitute the bits from the content they wish to hide. Changing the least significant bit of the carrier file creates minor variations in the visual depiction. These changes are not substantial enough to be visually different to the average user. Three color values define each pixel of an image file. The color values are red, green, and blue. Each color value has a one-byte value that specifies the specific shades of the color. The colors are blended to create the proper shading in the image's presentation. The changes to the color bytes do not change the visual depiction of the digital image.

Detecting steganography is very difficult. There is no way to detect it using the visual characteristics of a carrier file. Some clues an investigator will look for during the computer forensic exam are steganography tools. When steganography tools are found on the host, the analysis must start looking for additional artifacts to identify potential carrier files. Even if the investigator determines the specific steganography tool that has been used, they must still identify the password used to secure the file. Even if the investigator can extract the hidden file from the carrier, they may have to deal with encryption, as it is not uncommon for a user to encrypt a data set before hiding using steganography.

Artifact Wiping

Users wipe artifacts to remove specific files (and, sometimes, file systems), such as log files, cache files, or any file the user does not want to be analyzed. The user has a choice of several utilities. If a user has been viewing illicit images on their work desktop, they will not want anyone to find the images, so they "wipe" the files. Depending on the operating system, if a user deletes a file, the system has not removed it from the file system or the storage medium. When a user deletes a file, it is simply moved to a recycle bin or trash can. The system then stores the file in this new container until the user decides to "empty the trash." The file is then removed from the file system but is still on the physical storage device. Wiping will overwrite the clusters the file occupies with random characters or with a pattern of characters. A single pass of the characters onto the file's clusters is usually enough to prevent the file from being recovered. The user can also specify that all the sectors of the entire storage device are wiped, which is referred to as disk sanitation. Just like when a user wipes a single file, the data that were on the device can no longer be recovered.

Degassing uses a magnetic field to make the storage device unusable. This technique requires specialized equipment to ensure proper execution. The user can also wipe operating system artifacts to hide their activities. A registry cleaner can identify and remove

registry keys for a system using the Microsoft Windows operating system. There are utilities designed to remove specific types of data. The internet cache and internet history, for example, can contain information about the user's browsing habits. The internet history files may also track which websites the user had visited, keeping the username and passwords and logging whether they have downloaded or uploaded any files.

Operating systems also create a thumbnail cache. The digital images on the system are indexed, and smaller images (thumbnails) are made to enhance the user experience by providing faster previews of the digital images. The thumbnail cache may also keep a thumbnail of a deleted image, proving that the full-size image was available on the system at an earlier date and time. The thumbnail cache is not known or accessed by the casual user.

Obfuscation

The user may also take precautions to hide their true identity or mask their activities on the system. A user may use a proxy server when they remotely connect to a server, hiding their physical location. Using the Tor browser or the Onion network is an example of a user obfuscating their identity and location. The user may also want to obfuscate their activities on the host network. We have already discussed deleting and cleaning registry entries and log files, but the militant user may also change the date and time stamps that are recorded.

The Linux operating system has the "touch" command, which can modify the date and time stamps to a value decided on by the user. The Metasploit Project is a computer security project that developed the Metasploit Framework (Metasploit, n.d.). The framework is a penetration testing tool designed to compromise a host. The framework comprises several anti-forensic tools to include the utility Timestomp. This utility can change files' modified, accessed, and created (MAC) date and time stamps. The altered timestamps can increase the difficulty of determining when a given user accessed files.

Another tool included in the framework is Transmogrify. Transmogrify allows the user to change the file signatures that are contained in the file header. A user with malicious intent could change the file signature of the system's .jpg files to appear as an .mp3 or .doc file, for example.



SUMMARY

To present evidence in administrative or judicial proceedings, the evidence must be collected in a manner that does not contaminate it. As an investigator, several forensic image formats can protect digital evidence. Investigators can use the dd forensic image with most, if not all, forensic tools. The dd image is based on a Unix command and now works on the three major operating systems. The other forensic image is the expert witness or the e01 file. When creating an e01 file, a file header will contain user input information such as the name of the investigator, case

number, and subject name. There will also be a cyclic redundancy check (CRC) value for every 64 sectors, and the system will save CRC value within the forensic image format. Once the forensic image has been created, the investigator can examine the digital evidence.

During the exam, the investigator may find malware within the forensic image; this will allow the investigator to conduct a static analysis. The static analysis of malware can be a time-consuming endeavor. The other option is to conduct a dynamic malware analysis by placing the malware on a testing machine. We want to ensure that the analyst does not connect the testing environment to the production environment. The investigator can continue to examine the forensic image for additional artifacts to determine whether any data have been exfiltrated. Some artifacts that the investigator may look for are the use of encryption or email. Additionally, there may be anti-forensic artifacts, such as the Timestomp tool or steganography tools. If, during the exam, the investigator finds that the log files appear to have been wiped, the user may obfuscate the evidence to hide their actions.

LEKTION 5

NETWORK FORENSICS

STUDY GOALS

On completion of this unit, you will be able to ...

- process information and generate intelligence.
- understand the functions of the F3EAD model.
- explain the goals of incident response.
- understand the attacks on network forensics.
- describe why intelligence is essential.
- emphasize the importance of network security.

5. NETWORK FORENSICS

Introduction

Network forensics is an ambiguous term with a wide variety of definitions. An investigator can analyze the contents of a hard drive to look for file system and operating system artifacts, but what exactly do they analyze when performing network forensics?

Digital forensic and incident response (DFIR)

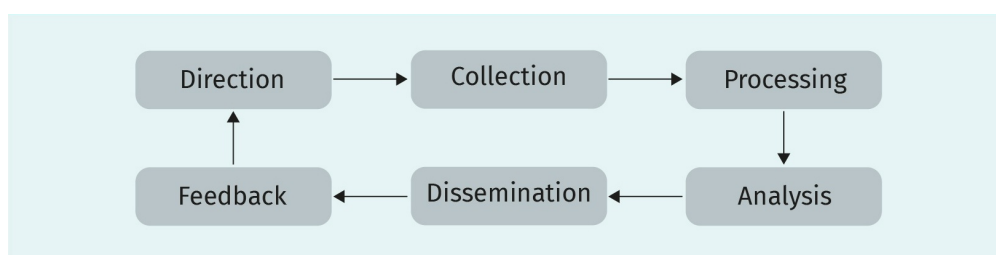
This is a specialized cybersecurity functional sub-field traditionally associated with computer emergency response teams.

Network forensics is a function of **digital forensic and incident response (DFIR)**, whose purpose is to identify, investigate, and remediate attacks against an organization's computer network. An investigator can collect evidence from the network traffic from trace files while an attack is in progress, or collect information from sources such as firewall log files and applications to determine if there has been an attack. Information sharing is critical to an organization's ability to identify, analyze, and defend against a network intrusion. The report *Operation SMN: Axiom Threat Actor Group Report* highlights an example of this information-sharing process (Jones-Powe, 2015). The Axiom Group attacked Fortune 500 companies, non-government organizations (NGOs), journalists, and other organizations for over six years (Jones-Powe, 2015). The attackers were very sophisticated in their approach, and maintained and expanded their access within the victim organizations' networks. As individual organizations identified the malware being used to attack their networks, they collaborated and shared the intelligence they collected from the malware. The organizations recognized the attacks were coordinated and much more complex than usual. As the organizations collected and analyzed the data, they were able to develop a coordinated plan, which resulted in eradicating over forty thousand pieces of malware (Jones-Powe, 2015).

What is the difference between information and intelligence? Roberts and Brown (2017) define intelligence as information that has been augmented to be actionable. Information is standalone; it can be a statistic, fact, or description of something. At the same time, information cannot be properly used without exploring its context. Only when the context is explored can we determine whether the information is important. If intelligence is not shared, there is little point to the entire exercise. Intelligence is produced through the analysis and sharing of information. The analyst can then use the intelligence to answer questions about the incident taking place.

The intelligence cycle is used to create and analyze intelligence. This cycle runs continuously; where one process stops, another process follows, building upon the previously generated intelligence. The six steps of the intelligence cycle are shown in the following figure. The intelligence cycle can be very flexible, but it is important not to omit a critical step, as this can impede the generation of intelligence.

Figure 83: The Intelligence Cycle



Source: Created on behalf of IU (2022).

The first step of the intelligence cycle is “direction.” This step sets the intelligence cycle in motion by providing the question that needs to be answered. This question should be specific and clearly understandable. We must then ask what will drive the rest of the intelligence cycle (United States Joint Chiefs of Staff, 2013).

Once we have determined the direction of the intelligence cycle, the collection process begins. We should collect as many data points (i.e., as much information) as possible. Information must also be gathered from multiple sources, as this will corroborate other data points. The information collected can be tactical, such as information about the network infrastructure, known vulnerabilities, current malware versions, and information about potential adversaries. The collection of information is an ongoing process that is part of the security team’s daily activities. The more knowledge the security team has about ongoing threats, the more effective they can be during a response. As information is collected, it can also generate secondary or tertiary levels of collection. An example of this is identifying the IP address of the attacker. Once this has been ascertained, a WHOIS search can be carried out to determine the vendor or owner of the IP address. A domain name system (DNS) search can also determine if any domain names are associated with this IP address. Once the information has been collected, it must be processed. This information has not yet been placed into context. The information that has been compiled should be put into a standard format so that it can be properly analyzed.

The analysis phase is used to answer the question that started off the intelligence cycle. We may not gather every bit of information to create a definitive answer. There may be some gaps in the data set that cause us to make predictions. As we analyze the information, we may realize there are gaps in the data set that are significant enough to stop the analysis from being completed. This may require circling back to the collection phase to gather more information.

Once the analysis has been completed, it is now time for the intelligence to be disseminated. We are no longer dealing with “information”; we have now created intelligence by analyzing the information and placing it into context. If we do not disseminate the results of the process to the appropriate audience, then the cycle has failed.

To know whether cycle has been carried out properly, there must be feedback to determine if the answers that the process generated effectively answered the question identified in the direction phase. This is a binary answer; either the cycle was successful, or it wasn’t. If the process was successful, it will generate additional questions, which will

cause the cycle to repeat. If the process was unsuccessful, the feedback phase must identify why. Was the direction not specific enough? Was the analysis incorrect? Was the intelligence disseminated to the correct audience? A no to any one of these questions will lead to a failure in the cycle. Let's now look at the different incident response models that can identify and respond to a network attack.

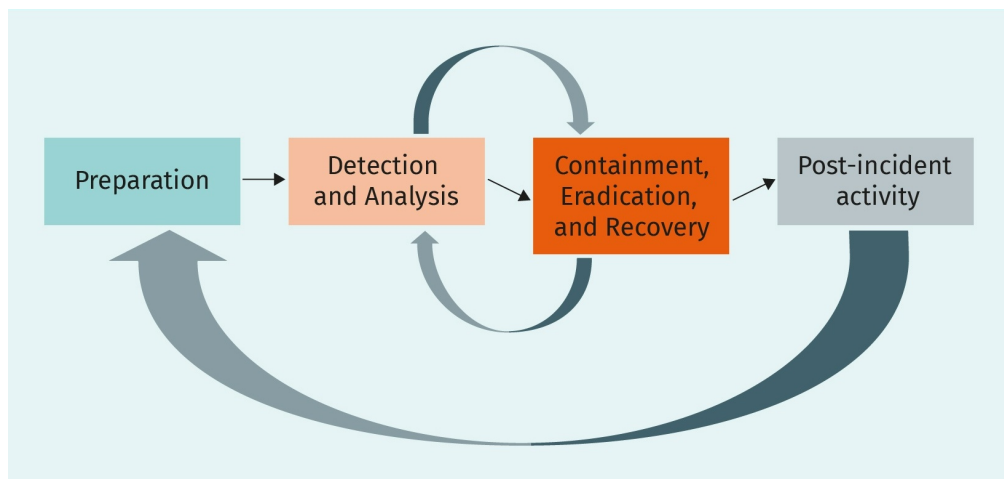
5.1 Indicators of Compromise

Intrusion detection is just one part of the organization's incident response. Incident response and incident detection share some characteristics, but only the incident response handles the identification of the network breach and the development and execution of the response. Many incident response models help take a very complex situation and make it manageable so that the organization can plan an effective response. Like anything else in the world, none of the models provide a perfect plan or response. However, they can provide the structure used as a baseline for the response. Let's look at some models that are in use today.

Incident Response Life Cycle

The National Institute of Standards and Technology (NIST) released the *Computer Security Incident Handling Guide* and highlighted a basic incident response life cycle, as shown in the following figure (Cichonski et al., 2012).

Figure 84: The Incident Response Life Cycle



Source: Created on behalf of IU (2022).

Generally, the path from preparation to post-incident activity is straightforward. When the preparation phase has been completed, the next phase is detection and analysis, which flow into containment, eradication, and recovery. Activity may move repeatedly between detection and analysis and containment. A security team may also detect additional victim hosts or additional malware while containing previously identified, compromised

hosts or other malware infections. Once the incident has been contained and the security team has eradicated the attacker, post-incident activity takes place. The organization should generate a report that will identify the positive and negative aspects of the response. The cause of the attack, the cost of the response, and the lessons learned by the organization should also be included. The “lessons learned” aspect of the report will have the organization returning to the preparation phase to mitigate any identified deficiencies in the report. Let us now look at the preparation phase in more detail.

Preparation Phase

In the “preparation” phase, the organization must identify its risk tolerance. They are then able to identify and implement the necessary controls to decrease the risk of a successful network attack. It is not possible to remove all risk that an organization will face; the goal of the preparation phase is simply to reduce the risk to an acceptable level. This is the phase where the organization can engage in a number of different controls to minimize the effects of an attack. The organization can implement some of the following controls:

- new definitions for the intrusion detection system,
- a baseline of network traffic,
- new appliances designed to alert about attacks, and
- regular training for the security team.

Communication is an essential part of the preparatory phase; it is used to identify who will be in charge of the response and provide employees with their tasks and assignments. Once the security team has been identified, they must practice their response in a non-production environment (i.e., not practicing on the actual network providing operational services to the organization). Instead, a testing network is created to avoid accidentally bringing down the production network. The security team members must also be familiar with the physical topology of the network (the locations of the servers, routers, and switches). If an attacker can gain physical access to the network, they can more easily launch their attack because they have created a foothold inside the network itself, bypassing many of the security controls. Security should regularly identify potential vulnerabilities, such as weak physical security, problems with security patch deployment, and unnecessary open ports. These actions must also be documented and maintained, as it will be difficult to respond to an incident effectively if the security team is underprepared.

In this sense, training is paramount. C-level personnel must take part in the training to understand, at a high level, what will occur during the response to an attack. The training scenarios should also identify any inadequacies in the plan. What looks good on paper may not be feasible when executed in the physical environment. This will allow the plan to be adjusted to account for any deficiencies that may have been overlooked. After all, the attacker is the one with the greatest advantage; it is very easy to hide in the network and observe what is transpiring around them. Time, as well as the element of surprise, is most often on the side of the attacker.

Detection and Analysis Phase

The detection phase begins when the security team has identified an attacker within the networking environment. This can happen in several ways, for example, when users complain to the helpdesk about IT issues. Common vectors that attackers will use include phishing emails. The user executes the malware via email or enters their user credentials and sends them to the attacker. Another method is abnormal traffic within the network. Traffic spikes may occur when a host has been compromised and the attacker is attempting to spread out into the rest of the network. Another reason for traffic spiking is that the attacker has reached the data set they want to acquire and is now exfiltrating it from the network. In some cases, law enforcement may notify you of criminal activity that has been traced back to the network. Sometimes it is the smaller details that could be indicative of a potential attack, such as unusual characters being used in a file name or finding a configuration change in a log file. No matter which avenue of detection is used, the security team will need to analyze the artifacts to determine the extent of the intrusion. This is the analysis portion of this phase. Once the security team identifies compromised hosts and the vector used in the attack, it is now time to move onto containing the attack to prevent additional hosts from being compromised.

Filtering false positives can be a difficult task. Depending on the organization's size, we could analyze thousands of alerts, and trying to determine which ones are valid seems to be a Herculean task. Even when there are reliable indicators of an intrusion, they may not have been caused by an attacker. A server crash, for example, can cause the modification of critical files, which may generate an alert. A team member must analyze the alert to determine the cause. This could be human error or mechanical failure. We will now look at some actions that an organization can take in the planning phase to help make the analysis phase more efficient. Strong control metrics can be obtained by maintaining a profile of the network. This will require the creation of a "baseline" of the network so it can be categorically noted when usage levels peak and or decline. Identifying the critical files and conducting a file integrity check (using hashing algorithms) can then ensure that essential files have not been modified.

To identify what is "normal" for the network, the security team must have a thorough understanding of the network (the hosts, servers, and applications) and understand the expected behavior of its resources. Depending on the organization's size, it may be difficult for one person to understand the entire system, so breaking it up into regions to be assigned to different members can be useful. Each aspect should have multiple members assigned; if a member cannot respond when an incident occurs, the additional member can stand in their place. Members should make themselves familiar with the logs being generated by reviewing the logs and their security alerts. As members become more familiar with the "normal" contents of the logs and security alerts, they can focus on the abnormal entries.

Many security appliances and servers create log files and generate security alerts, so there needs to be a policy addressing how long the log and security alerts are kept. For example, suppose the security team identifies a potential breach. There is an excellent chance that the prior logs will contain information about the attackers' reconnaissance of the organization. The security team should review different logs and security alerts from various

sources. Each log or alert will contain a small piece of the puzzle. The firewall might have an IP address, while a log file has a username. It is the responsibility of the security team member to take these pieces of information and combine them to gain an accurate understanding of what occurred. This is known as event correlation.

Time analysis is critical when trying to determine whether an attack has taken place, and, if it has, when. All the devices on the network must have synchronized clocks. If the clocks are not synchronized, it will be almost impossible to correlate the events between the different devices on the network. Using network time protocol (NTP) to manage the time systems of the devices on the network is a critical aspect in planning. The security team can also create an organizational knowledge database. Creating this resource will influence the speed with which the secured team understands what has happened. The knowledge base does not have to be very complex; the team can create the knowledge base with text documents, spreadsheets, or small, searchable databases. Anything that might facilitate the exchange of information between the members of the security team can be beneficial. An example of something that should be included in a knowledge base is intrusion detection system (IDS) alerts; if a network generates false positives, this is an excellent location to list it. Important alerts can also be included here, along with explanations of their validity.

We can find the network artifacts we want to analyze in several places, which may make things difficult to correlate. To this end, a packet sniffer can be installed at different points in the network to record what is happening. The packet sniffer logs all the traffic passing through the wire in that network segment. However, capturing these data may result in an information overload, so it is best to filter the traffic to meet certain criteria. Governmental or organizational privacy concerns should also be taken into account here.

Containment, Eradication, and Recovery Phase

The containment phase occurs when the security team interacts with the network and takes specific actions to mitigate the ability of the attacker to continue an attack. There may be multiple strategies being used, such as an immediate strategy to halt any further incursions into the network, followed by a strategy to remove the attacker from the network and prevent the avenue of attack from being used again. Some of the security team's short-term actions can block the attackers' access to network resources, for example, disabling the switch or router connected to the compromised hosts, locking out any compromised user accounts, and disabling the service being exploited by the attacker.

The containment options will vary depending on the type of attack. A distributed denial-of-service (DDoS) attack requires a different response than a USB-based malware attack. Once again, the effort that the organization puts into its planning phase will directly impact the containment phase. Pre-planning potential strategies allows the organization to make a well-informed decision. What information should the organization use to make that well-informed decision? The following are some general criteria that can be used to determine the strategy (Cichonski et al., 2012):

- potential damage to the network
- acceptable risk for data loss

- evidence collection and preservation
- network or organizational downtime
- reaction time to network intrusion
- resources needed to deploy incident response measures, e.g., containment
- effectiveness of the mitigation controls and containment measures
- how long the strategy needs to be deployed

The security team should consult the legal department before executing a strategy. The legal review will help to mitigate any potential liability on behalf of the organization. There is no guarantee that a strategy will be successful. Therefore, will the organization be liable if the strategy cannot contain the attacker?

As the security team is working through this process, there is also the need to ensure evidence is collected and handled appropriately. The regulatory or judicial requirements of the organization's location will be definitive here. As evidence is collected, a log should be maintained highlighting the following information:

- Who? Who found the specific piece of evidence? This information must be recorded. If evidence is transferred to another party, such as a team member responsible for evidence collection and handling, this must also be noted in the log.
- What? What is the evidence? This can be a log file, alert, network trace, or piece of data that can be used in the investigation. A hashing algorithm should be used to create a digital fingerprint of the item. With this, the team can authenticate the piece of evidence during an administrative or judicial proceeding.
- When? When was the log file created, and when was it collected? This is the key reason that all devices on the network must have their clocks synchronized. If the team collected evidence from an off-site location, the time zone in which that device is located should also be included.
- Where? Where was the evidence found? This could be a server, firewall, or intrusion detection system. The location, serial number, hostname, physical address (MAC), and network (IP) address should be recorded.

NIST released the *Guide to Integrating Forensic Techniques into Incident Response*, providing greater detail about incorporating digital forensics into incident response (Kent et al., 2006). After containment, the next step is to begin the eradication process, which includes disabling compromised user accounts, removing malware, and identifying any vulnerabilities that the attacker exploited. The eradication process can be time-consuming because of the need to identify all systems that were compromised.

In the recovery phase, the security team restores the compromised systems and places them back into the production environment. Testing and quality assurance must be carried out to ensure that the systems are back to their pre-attack configurations and then fully patched to protect against further attacks. To do this, the security team may have to restore from backups, replace compromised files, ensure that all security patches have been applied, change any passwords that may have been compromised, and sometimes rebuild some systems from the ground up. Malware-infected systems often spark debate on how to handle this problem. If malware has not been entirely removed, a complete wipe of the system and a fresh install of the operating system will likely be necessary. As

the organization's network is being restored and placed into the production environment, the security team should also increase the level of monitoring to ensure that there are no additional attacks from other hostile actors. Depending on the size of the network and the depth of the attack, the recovery process could take months (or longer) to complete. While the network is being restored, security should be considered at every step.

Post-Incident Activity Phase

In this phase, one of the most important aspects is the "lessons learned." All the participants in the incident response should be part of the lessons learned process. This process is not intended to be a finger-pointing session, but rather an unbiased look at the events to determine what occurred, what strategies were deployed, and which of these strategies were effective. The following questions should be asked and answered during this process:

- What happened? How and when did the incident start?
- How well did the organization respond to the incident? Were the organization's policies and procedures effective? Were all policies and procedures followed?
- Was information shared in a timely manner? If not, why?
- What worked? What actions were effective? What actions were ineffective? What could the organization do differently next time?
- Were there any indicators to show that the attack was imminent? Were these missed or ignored?
- Were there any tools needed for the response? Are newer versions of the existing tools necessary?

The results of the meetings to discuss lessons learned must be documented and retained. The organization can use these results as a teaching tool for new members. Policies and procedures can be updated with the information gathered in these meetings. The data can also be shared with other organizations to help them increase their readiness and recognize any deficiencies they may have within their organization's network. However, before sharing any information, this must be confirmed with the C-level executives, including the owner of the data, legal representatives, and data privacy officials for data security reasons.

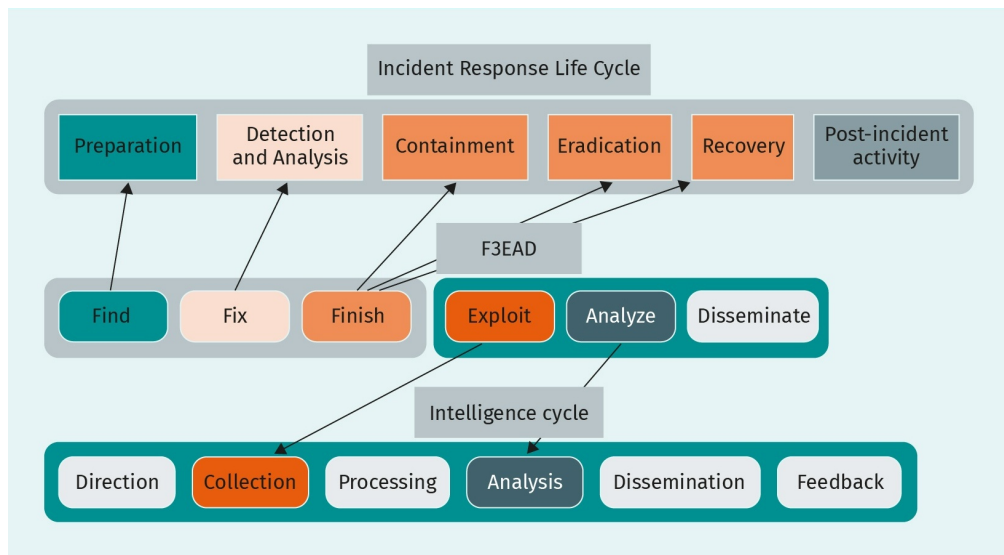
The evidence that was collected during the incident response should be properly maintained. The organization should have a specific policy and procedure detailing how long the evidence should be maintained. This evidence can be used in an administrative or judicial proceeding that may last many years.

F3EAD

The find, fix, finish, exploit, analyze and disseminate (F3EAD) cycle, pronounced F3-E-A-D or "feed," is an operation-centric model and intelligence generation framework, designed and used by the US military (Roberts & Brown, 2017). Using this model, the intelligence and operation phases feed into each other. This can be viewed as an incident response operation that leads to an intelligence-gathering process that feeds the next instant

response operation. This creates a continuous cycle, fostering a proactive security team that protects the organization’s network. Let us now discuss the different aspects of this model.

Figure 85: The Intelligence Cycle, F3EAD Model, and Incident Response Life Cycle



Source: Created on behalf of IU (2022).

The “find” phase determines the threats facing the organization and how these threats can be targeted. We can ascertain intelligence about the threats from many sources, such as vendors, open-source locations, or other organizations with whom intelligence is shared. This phase is comparable with the “preparation” phase of the incident response life cycle.

The “fix” phase uses the information that the security team has gathered in the “find” phase of the model. This is not a “repair phase,” but a phase meant to acquire a “fix” (location) on the attacker. Using the information gathered in previous phase, information is being analyzed to create intelligence. The main goal here is identifying the attacker’s location within the network and any support they may be receiving from outside it. This includes compromised hosts, servers, or other resources within your network. Any communication channels that the attacker may have used must be examined. This could be an internet relay chat (IRC) channel, a Slack channel, or a Discord channel. This phase corresponds to the “detection and analysis” phase of the incident response life cycle.

The “finish” phase is where action is taken against the attacker. The activities that would be taken in the “containment, mitigation, and eradication” phase of the incident response life cycle occur here. After this phase in the incident response model, the incident is technically concluded. With the F3EAD model, we are now moving into the intelligence phases, which involve using the information (intelligence) that we have gained in the three “F” phases (Roberts & Brown, 2017).

The “exploit” phase mirrors the actions taken in the “collections” phase of the intelligence cycle. Information that may be useful is collected. Some categories of the information you may look for include malware, vulnerabilities, exploits, prior incident reports, the goals of the attacker, communications from the attacker, information relating to the attack (such as IP addresses), or email addresses. Any information that can be collected dealing with different phases of the attack should be gathered here (Roberts & Brown, 2017).

It is then time to enter the analysis phase, where the information that was collected in the “exploit” phase is analyzed. The information must be placed into context. As the information is analyzed, it may lead to additional data points that also need to be collected. Ultimately, the goal of the “analysis” phase is to gain a comprehensive understanding of what was and is taking place.

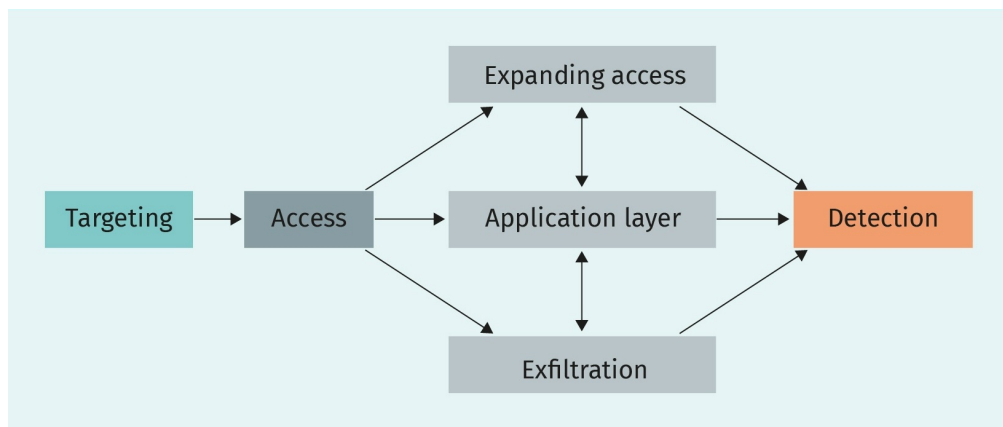
Finally, it is time for the “dissemination” phase, wherein there may be multiple audiences that need to be addressed. The incident response team will want the “tactical” intelligence about the attack. This will allow them to start mitigation measures. The C-level of the organization will want the “strategic” intelligence. The report will inform them which strategies were effective. At this point, we can also identify which mitigation controls were effective. This will allow them to make informed decisions about future funding efforts and other mitigation measures. Finally, intelligence can be passed on to a third party, such as an intelligence-sharing group, law enforcement, or regulatory bodies. The executive level will have to be consulted to determine how much intelligence can be shared with third parties.

F3EAD can be a complicated process to implement. To be successful, there must be a focus on the idea that security operations and incident response feed into the concept of threat intelligence. As the security team members generate reports, the intelligence team should continually analyze the information. This proactive approach will help the organization mitigate attacks with a faster response time.

5.2 Data Enrichment and Pivot Points

How does an attacker determine their target and the method they will use to compromise the network? Just like we just discussed turning information into intelligence, the attacker can use the same models to facilitate their attack. There are five phases to an attack, (Monte, 2015) as shown in the following figure, and we will discuss the generalities of each phase. Depending on your research, you may come across different names for the different phases depending on your reading model.

Figure 86: The Attack Life Cycle



Source: Created on behalf of IU (2022).

Targeting (Reconnaissance)

The first step of an attack is target identification. The attacker may target an organization because of their nationality, location, economic position, or political stance. Once they have decided to attack, they will begin taking steps to compromise the network.

Reconnaissance is a significant part of the actions taken in the targeting phase. The attacker wants to understand the target organization in terms of location, employees, server type, email address format, etc. An intricate understanding of the organization is essential for the attacker. Not many actions can be detected at this phase. The attacker can search social media sites to identify executives, system administrators, and current and former employees. Attackers may also target subcontractors of the target organization. If the organization's vendors or contractors possess the credentials to access the network, then the attacker can also target them for exploitation. The attacker's goal is to use the information gathered to trick the target into revealing their user credentials. It is much easier for an attacker to use a set of compromised credentials to gain access to the network than trying to use an exploit or vulnerability, such as a buffer overflow. Instead, the attacker can perform external network scans against the targeted organization, looking for IP ranges and domain names to help them gain access. The attacker may collect information about the target for use during targeting. Information about the target may be available for purchase from reputable private sources and databases, such as paid subscriptions to technical or threat intelligence data feeds or less-reputable sources, such as the dark web or cyber-crime black markets.

For example, an attacker identifies a commercial organization that uses a point-of-sale (POS) system, connected to their network. Once the attacker identifies the make and model of the POS system, they can perform scans to determine if there are any vulnerabilities and whether a patch has been applied. If a vulnerability exists, it may be possible for the attacker to gain access to the POS system. This occurred in 2008; hackers were able to identify vulnerable POS systems used by the sandwich shop Subway (Zetter, 2011). The attackers were able to identify which POS system was being used and which vendor was being used to deploy it. Subway listed this information in a press release. Once the attack-

ers identified the host on the network, they realized that some of the POS systems came installed with remote desktop software. This compromised over 200 hosts, giving the hackers access to the network and the credit card information of over eighty thousand customers (Zetter, 2011).

Access

The attacker takes this information, analyzes it, and determines the vector of the attack. As they identify the vulnerabilities, the attacker can purchase equipment similar to that used by the organization. This allows them to reverse engineer potential vulnerabilities or create a zero-day exploit. A widespread means of exploitation is the use of social engineering combined with email phishing. Once the attacker has access, several goals arise. The attacker will want to remain in the network, increase their permissions for the credentials they have, and exfiltrate targeted data. Their initial access can range from a restricted user to an administrator account. Ultimately, the attacker does not care. The goal of the access phase is to create a foothold in the network.

The attacker can carry out a phishing attack by sending messages to users to deceive them into exposing their credentials or installing malware. The phishing attack can be targeted at a specific user (this is known as spear phishing). In spear phishing, the attacker will target a particular individual, company, or industry. The emails may contain malicious attachments or links to execute malicious code on victim systems. The attacker may use third-party services, like social media platforms, as a jumping-off point for the phishing attack.

Network intrusion detection systems (IDSs) and email gateways can detect phishing emails that contain malware as an attachment. The IDS can identify the attack by using signature or behavior-based definitions. Unfortunately, some attackers may create the malware and pack it in a manner that avoids detection. Using URL inspectors for links within the email can identify links leading to known malicious sites. Packet inspection can help detect the initial delivery, especially if an secure sockets layer and transport layer security (SSL/TLS) connection is being used. Antiviruses can detect malware when it has been disguised as a legitimate file, and also when a user downloads and executes the malicious files.

The attacker may use the command line or script language to execute commands on the victim host. The command line interface and scripting capabilities are already built into the operating systems. Microsoft Windows uses PowerShell and a command line via the Windows Command Shell. Linux and macOS use a version on the Unix command line shell. An attacker can also use cross-platform programming languages, like Python, to execute commands, scripts, or binaries. Enabling logging will help detect the attacker's command line or scripting activities. Logging the attacker's actions will help identify how the native processes are used, or if the attacker is using a custom toolset.

System administrators should restrict scripting for normal users. The security team should investigate any attempt to change the settings regarding scripts. If scripting is allowed, but scripts are not typically used on the host, the security team should investigate whether

scripts are executing at abnormal times. Some script executions may generate an event log that could lead back to other behaviors, eventually helping to identify the script or command executed by the attacker.

Persistence

After the initial access, the attackers may want to expand their reach by increasing their permissions and making their access repeatable. Installing a remote access program (backdoor) is one of the first actions the attackers will take. The attacker wants to ensure their access persists during the normal operations of the system. Using a backdoor also establishes the command-and-control channel that the attacker can use inside and outside of the network. The attacker will start searching for additional hosts to compromise, this increasing the persistence of their access. These persistence methods can vary depending on the operating system of the host(s) that have been compromised. For example, Microsoft Windows has many AutoRun registry entries that the attacker can use to automatically load the attacker's software. The attacker's goal is to create a persistent connection to avoid having to break through the security controls again.

The attacker will want to manipulate user accounts to maintain access to the victim host. These actions could consist of changing a user's credentials or changing permission groups. Intending to bypass security policies, the attacker could change the security policies for password updates, password duration policies, or plan to preserve the compromised user credentials for as long as possible. Monitoring the following event IDs (Microsoft Windows OS) provides the following alerts:

- 4728A member was added to a security-enabled global group.
- 4738User account was changed.
- 4670Permissions on an object were changed.

These event IDs may identify activities that should be investigated. For example, the date and time stamps may be used when analyzing logs or other artifacts, such as a user's account permissions increasing. The attacker may also use internet browser extensions to maintain access to the victim host. A browser extension is a small program that can add functionality to or customize a user's browser. The user may install the extensions directly or do it through the browser's app store. Once installed and enabled, the extension will have the same access and permissions as the browser. A compromised app store can lead to the installation of malicious extensions that would appear legitimate. The attacker may change an extension's update uniform resource locator (URL), which will cause the updates to come from the attacker. Once the compromised extension has been installed and enabled in the browser, the extension will have the ability to access websites without the user's knowledge and compromise user-inputted information, which could include user credentials. The security team must keep inventory of and monitor browser extension installations, as this will help identify a potential intrusion. When a compromised extension is installed, there may be changes made to the registry.

Expanding Access

Now it is time for the attacker to expand their initial foothold. They will do this by locating a data set they are interested in and solidifying their presence within the network environment. An example of this approach was the Stuxnet worm (Zetter, 2014). After gaining initial access to the network, the worm went after a specific resource on the network, in this case, the centrifuges.

It is common for organizations to use the “defense in depth” concept when trying to protect their network. The **demilitarized zone (DMZ)** is used to host public-facing servers. The DMZ is segregated from the internal network to minimize the ability of an attacker to gain access to critical systems if one of the public-facing servers is compromised. The first breach will not provide any access to confidential information, such as credit card numbers, personal identifying information (PII), payroll data, and internal confidential documents. It may take the attacker several weeks or months to expand their access. During this time, they are sitting silently, monitoring and collecting data. They then analyze the data, generating intelligence that can be used to expand their presence within the compromised network.

Demilitarized zone (DMZ)

This is a network segment that contains and exposes the organization's externally-facing services to the internet, which protects the internal local area network.

Internal monitoring is going to be essential to identify an attacker. If the attacker can compromise administrative credentials, they may change appliances on the network, such as routers or switches. If the attacker uses a router or switch and creates a new conduit to a secure resource, the security team must recognize that abnormality and investigate it.

The attacker can create or change system-level processes to continually execute malware to maintain their access to the compromised host. As operating systems boot up, system processes are initiated and become able to perform system-level operations in the background. The attacker may install new malware, modify existing processes, or configure processes to execute at startup to maintain or expand access on the host. Some existing processes can be created with administrator privileges, and the attacker could use these processes to create or change system processes to increase their privileges on the host. The organization should have a trusted, established baseline of the system to identify any unauthorized changes in the system-level processes. The security team should not view the artifacts as single events, but as one piece in the totality of circumstances.

The attacker may compromise remote systems by adding malicious content, such as scripts, trojans, or malicious code to a shared storage location (such as a network drive). When the user executes the malicious code, one potential effect is that the attacker can now move laterally to another host on the network. A directory share pivot is a technique used to upload malware into shared network directories as users access the directories. The shortcut for modification of the directory is an `.lnk` file that appears to be an actual directory. The attacker will embed a command in an `.lnk` file to execute a hidden malware file. The real directory will open after the user has executed the malware, so the expected action still occurs. This technique may cause reinfections and access to system-level or higher privileged accounts when used with network shares (Routin, 2017).

The security team should monitor the system to identify processes that write or overwrite files in a network shared directory. When this is discovered, the team should start an investigation. In addition, they should start monitoring the processes that are executed from removable media and monitor the content of the network shares for .lnk files, hidden files, malware, or other file types that may not typically exist in the specific directories.

Exfiltration

Exfiltration occurs when the attacker can copy the desired data set. The goal is to copy the data without being detected. Exfiltration can be challenging to catch because the attacker can hide the process within the regular network traffic. Which data is the attacker looking for? It will depend on their specific motivation. Financial records, customer records, intellectual property, or trade secrets are among the data that can be targeted. The attacker has several options to carry out data theft. If they have physical access to the host, the data can be downloaded to an external device and carried out of the organization.

When operating remotely, the attacker can use a virtual private network (VPN), secure socket layer and transport layer security (SSL/TLS), steganography, encryption, or a covert channel to extract the data. A timing channel will send the data a few packets at a time. Depending on the size of the data set, this could result in a lengthy exfiltration process, albeit one that is much harder to detect. Other alternatives include using emails and instant messages to send the data. To help identify the data exfiltration, the security team should look for a data flow that appears abnormal. This could be a host that usually receives data but now is transmitting them. A communication channel that has been opened using an unusual network port can also indicate a compromised network. If an organization uses a cloud service provider, an attacker can access an account with the same cloud service provider. The attacker can then move the data set within the cloud service provider network and avoid monitoring for unusual data transfers. By using the same cloud service provider, the attacker can hide the movement of the dataset.

5.3 Attacks Against Network Forensics

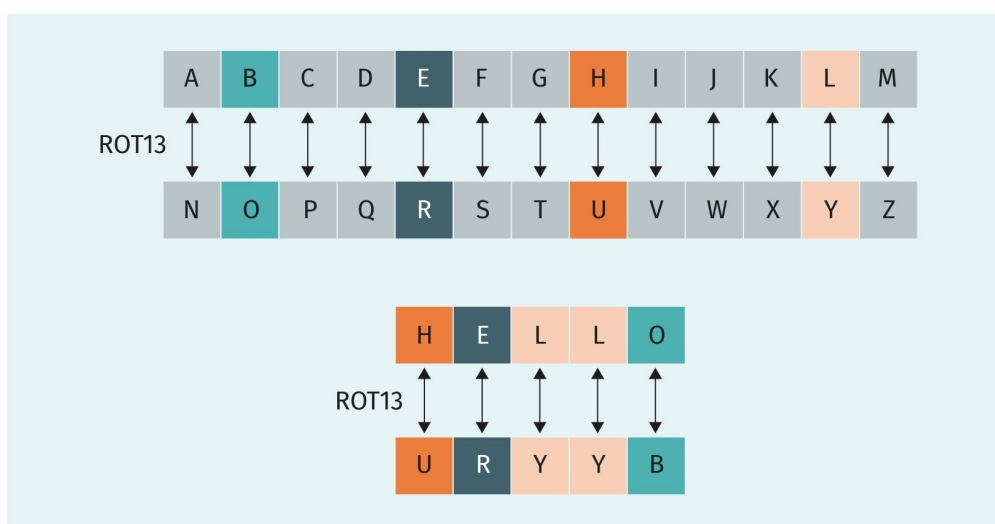
A signature-based antivirus uses the hash value of known malware to identify potential threats to the network. Another aspect of obfuscation is to hide malware by changing the digital fingerprint (i.e., hash value or signature) of the malware. The attacker has access to vendors who sell pre-compiled malware packages. The price of a pre-compiled malware package can range from free (open-source malware) to thousands of dollars (Osborne, 2018). These pre-compiled packages are known to the vendors of the antivirus and intrusion detection systems; the attacker will not attempt to change the malware signature, preferring instead to hide its true nature, i.e., through obfuscation. There are several techniques that an attacker can use:

- Packer—this will use compression to hide the malware. A compressed file will have a different signature than an uncompressed file.

- Code insertion—the attacker will insert inert code into the malware. The added lines of code will change the file signature without changing how the malware operates. The code the attacker inserts does not perform any function other than being placed in the functioning code and changing the file signature.
- Encryption—encrypting the malware (or portions of the it) changes the malware signature.
- Encoding—encoding the malware will change its code, thus changing its signature.

As the vendors notice the attackers' techniques, they will continue to add the signatures to the database. For example, if an attacker was using ROT-13 as their encryption process, the vendor can generate a file signature on the encrypted malware with that process. ROT-13 is a simple substitution cipher. It replaces every character with a character that is 13 places after it. The following figure depicts how this process works.

Figure 87: The ROT-13 Process



Source: Esham (2007). Public domain.

An attacker can also use obfuscation to put off the discovery of their incursion into the network for as long as possible. They want to mask the origins of the attack, disguise the vector they used for the exploitation, and remove any footprints they may leave behind when they traverse the network from host to host. They also want to make it much more difficult for the defender to identify that the network has been compromised. Finally, the attacker wants to confuse the defender by removing the artifacts they will use for the investigation and recovery process. These are all techniques for obfuscation and can be considered the last step in the anatomy of an attack.

The attacker can use several tools and techniques to change the logs, spoof their location, and jump from user account to user account. To bypass the system's file access controls, the attacker needs to have direct access to the volume. A utility such as Ninja Copy operates within PowerShell on a Microsoft Windows operating system and can accomplish this task (PowerShellMafia, 2017). This attack would not be easy to mitigate because the

attacker is exploiting a feature of the utility. To help discover this attack, the security team should monitor the use of PowerShell scripts and the processes accessing the logical drives.

Some operating systems can hide important system files and administrative processes. This prevents the typical user from accessing these aspects of the operating system to prevent unintentional changes. An attacker may also use these techniques to hide their behavior. An attacker can use virtualization to create a region isolated from the security systems. The security team would have to monitor command line files and processes that could indicate that this type of activity has occurred. In 2017, the United States Computer Emergency Readiness Team (US Cert) released a malware analysis report on the remote access tool, Bankshot. The report showed that the malware could change keys within the Windows registry, gather system information to include processes and process IDs, and “delete all artifacts associated with the malware from the infected machine” (National Cybersecurity and Communications Integration Center, 2017, p. 11).

To help identify when the attacker has deleted log files or other artifacts, the security team can focus their mitigation efforts on encrypting and encoding logs and event files, forwarding the logs and event files to an external location, and protecting log and event files with specific security permissions. Using encryption and encoding of the log and event files will help protect the confidentiality of the files. However, if the attacker gains access to the host, this mitigation effort requires additional steps on the attacker’s part.

 **SUMMARY**

Understanding network forensics is a crucial aspect of the role of a digital investigator. To be effective, an analyst must not only gather information, but analyze it, place it into context, and convert it into intelligence. Intelligence allows better preparation for when an attack takes place and increases the efficiency of mitigation measures for potential attacks. We must also share intelligence with other parties who can use the intelligence to mitigate attacks. These parties can be divisions within the same organization or third parties experiencing with similar attacks. Using the intelligence cycle can facilitate an organization’s ability to create effective intelligence and use it as a proactive measure against attacks.

Several intelligent cycles can be deployed, including the generic “intelligence cycle” and the “F3EAD.” The incident response life cycle comprises multiple phases: preparation, detection and analysis, containment, eradication, recovery, and post incident activity. The detection and analysis phase feeds into the containment, eradication, and recovery phase, which can also feed back into the detection and analysis phase as we discover information about the attack. The post-incident activity phase is vital due to its “lessons learned” report. This allows the organization to identify which processes were effective and which were not. The

F3EAD process is a combination of an operation-centric and intelligence generation process. Each phase feeds into the next phase, which generates additional operations and intelligence gathering processes.

LEKTION 6

ATTACKS AS VIEWED FROM THE HOST AND NETWORK

STUDY GOALS

On completion of this unit, you will be able to ...

- explain the importance of the TEACH concept.
- identify different strategies used to protect networks.
- use tactics, techniques, and procedures.
- optimally place an IDPS.
- describe the MITRE framework.
- understand the concept of correlation of alerts.

6. ATTACKS AS VIEWED FROM THE HOST AND NETWORK

Introduction

Organizations have several mitigation strategies that can be used to recognize when an attack is occurring and reduce the effectiveness of an attacker's actions. Each method has a different level of effectiveness and its own pros and cons. Let's first look at these strategies.

Indicators of Compromise

An indicator of compromise (IoC) is an artifact that suggests that an attacker has compromised the network. These artifacts assist the security team in locating suspicious network activity. The cybersecurity company [CrowdStrike \(2021\)](#) has identified some common examples of IoCs:

- unusual network traffic
- network traffic from locations where the organization does not have a presence
- unknown applications installed on the system
- unusual activity from administrator or privileged accounts
- an increase in incorrect logins or access requests
- anomalous activity (such as an increase in database read volume)
- large numbers of requests for the same file
- suspicious registry or system file changes
- unusual domain name server (DNS) requests and registry configurations
- unauthorized settings changes, including mobile device profiles
- large amounts of compressed files in unusual locations

Signature-Based

This strategy is static and reactive. As long as the attacker does not change the malware being used, the security team will receive alerts about files in situations that match a pre-determined set of criteria. However, if the attacker is dynamic and changes their approach, signature-based protection may not be adequate.

Anomaly-Based

This strategy uses algorithms, analysis, and sometimes artificial intelligence to recognize when network behavior matches known attack values. This strategy requires an enormous amount of data collection, as well as a high degree of false positive identification. The alerts are typically not placed into context, meaning that an analyst is required. Too many false positives can lead to an analyst dismissing the alert and potentially missing an attack.

Techniques, Tactics, and Procedures (TTP)

The defender collects and filters data based on the attacker's known tactics and techniques. The cybersecurity firm [Trustnet \(2021\)](#) provides the following definitions:

- techniques. These are non-specific, intermediate methods or tools that an attacker will use.

- tactics. These are general, beginning-to-end strategies that attackers use to gain access to valuable systems and information.
- procedures. These are step-by-step descriptions of how the attacker plans to go about achieving their purpose.

The defender can utilize all of these strategies together, creating a multilayer defense strategy. In the first section of this unit, we will focus on the tactics, techniques, and procedures strategy.

6.1 Techniques, Tactics, and Procedures

Traditionally, organizations have used information to combat attackers. This resulted in the blocking of IP addresses, using hash values to identify known malware, and the application of security patches to the operating system and applications within the network. These tactics had some limited success against the novice or mediocre attacker, but were unsuccessful against the advanced attacker. However, by utilizing the techniques, tactics, and procedures (TTP) strategy, an organization **could now** deploy intelligence-based mitigation measures. The nonprofit organization MITRE has identified the three elements needed to deploy TTP (MITRE, 2014):

1. Threat intelligence analysis
2. Defensive engagement
3. Information sharing and collaboration

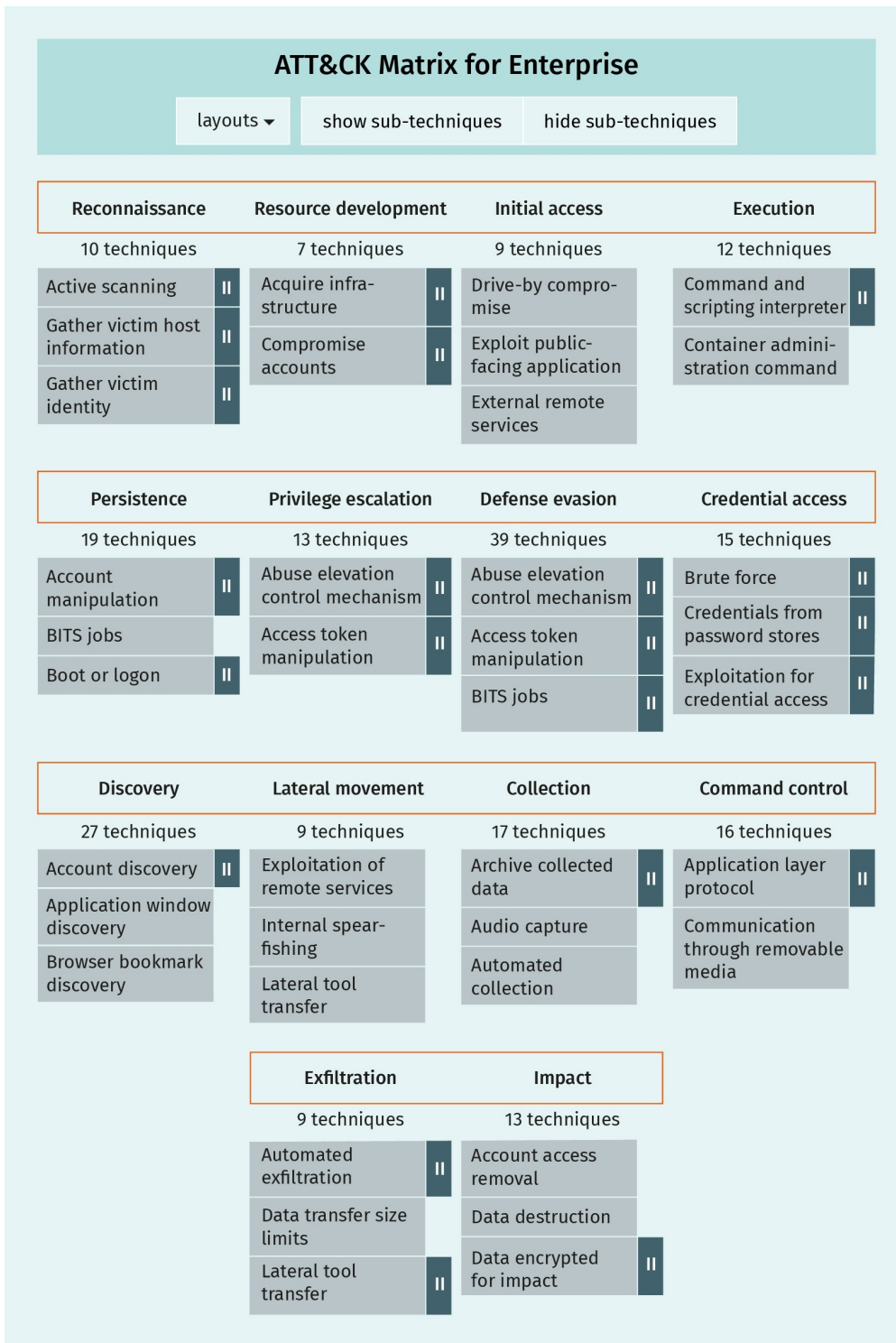
When using threat intelligence analysis, the analyst uses information that has been developed from either their organization or a partner organization to identify potential strategies the attacker might use. Some sources the analyst can use include the following, **as identified by the cybersecurity firm Trustnet (2021)**:

- Open-source intelligence (OSINT) are data found throughout the internet using low-cost, sharable platforms.
- Organization “darknets” are parts of an organization’s network that have no traffic and are not in use. By implementing procedures to monitor these segments the network for sudden changes, an infiltration-in-progress **could be caught**.
- Telemetry is the collective name for all of the data and measurements that flow throughout a network into a receiving device. These are usually gathered by scanning results, uploads, downloads, and traffic flow.
- Scanning for threats involves cataloging threat information from the internet. This strategy is a slow but effective and proactive threat intelligence tool.
- Malware analysis involves testing the most recent iterations of malicious code programs.
- Human intelligence involves using undercover reconnaissance techniques to gain access to closed forums, servers, and communities.

The defense of the organization relies heavily on identifying and preventing potential attacks. Organizations can be proactive in the early steps of the attack before the attacker gains a foothold within the network. Once the attack is in the network, organizations can no longer be proactive and must switch reactive measures. The sharing of information and collaboration with both internal and external organizations can strengthen the defenses of all participants. Information sharing is key to being proactive and recognizing the telltale signs of an incoming attack.

MITRE (MITRE ATT&CK, 2021b) has created a framework to help create an organization's detection and response capabilities. This framework is called ATT&CK, and provides a taxonomy of the different sections within an organization used to build a collaborative plan for organizational identification and response efforts (MITRE ATT&CK, 2021b). This framework can identify specific groups, tactics, and techniques that attackers have used. The framework uses a tab format with labels focused around the TTP approach. The different tactics of the ATT&CK framework are shown in the following figure.

Figure 88: Tactics Identified by MITRE's ATT&CK Framework



Source: MITRE ATT&CK (2021b).

The framework identifies fourteen different main tactics. The tactics have names that correspond to the anatomy of an attack, such as reconnaissance, initial access, persistence, and exfiltration. As an example, let's look at the techniques that have been identified under "reconnaissance." The attacker uses the reconnaissance step to gain information, which helps plan the attack and identify the victim of an upcoming attack. The attacker can use passive and active techniques to collect information. The attacker could, for example, look for information about personnel or internal infrastructure.

The framework breaks the techniques down into ID numbers that comprise a letter followed by four digits. If there are sub-techniques, a ".001" is appended to the ID. The numbering increases by one for each different sub-technique. The first technique identified by the framework, "active scanning," has the ID T1595.

This technique has two sub-techniques: "scanning IP blocks," which has ".001" appended to the ID, and "vulnerability scanning," which has ".002." A brief description is provided for each technique. On the MITRE ATT&CK (2021b) website, we can click on the hyperlink of the ID number or (sub-)technique, which takes us to a page with more detailed information.

Figure 89: Reconnaissance Tactics Identified by MITRE's ATT&CK Framework

ID	Name	Description
T1595	Active Scanning	Adversaries may execute active reconnaissance that do not involve direct i
.001	Scanning IP Blocks	Adversaries may scan victim IP blocks to c
.002	Vulnerability Scanning	Adversaries may scan victims for vulnerab exploit the adversary may seek to use.

Source: Created on behalf of IU, 2022.

When we click on the hyperlink for "vulnerability scanning," it takes us to a page with additional information. Here we find "procedure examples," "mitigations," and "detection" sections. Each example provided under the different headings has its own identification number. Under procedure examples, we see three identified procedures and a brief description of each. Clicking on the hyperlink will take us to a page with detailed information about the procedure. For example, when we click on the hyperlink for advanced persistent threat group 28 (APT28), we find that they have associated this procedure with a group from the Russian Federation, and that APT28 was used to compromise the computers used by Hillary Clinton and the Democratic National Committee in 2016 (MITRE ATT&CK, 2021a).

Figure 90: Procedure Examples as Identified by MITRE’s ATT&CK Framework

ID	Name	Description
G0007	APT28	APT28 has performed large-scale scans in an attempt to find vulnerable servers. ^[2]
G0034	Sandworm Team	Sandworm Team has scanned network infrastructure for vulnerabilities as part of its operational planning. ^[3]
G0123	Volatile Cedar	Volatile Cedar has performed vulnerability scans of the target server. ^{[4][5]}

Source: Created on behalf of IU, 2022.

The framework also provides information about mitigation and detection efforts that the defender can use. Unfortunately, this technique is not easily mitigated because it is conducted outside of the organization’s control. The organization can monitor this activity, but they will receive a sizeable number of false positives, and the defender may see a high occurrence rate from this behavior.

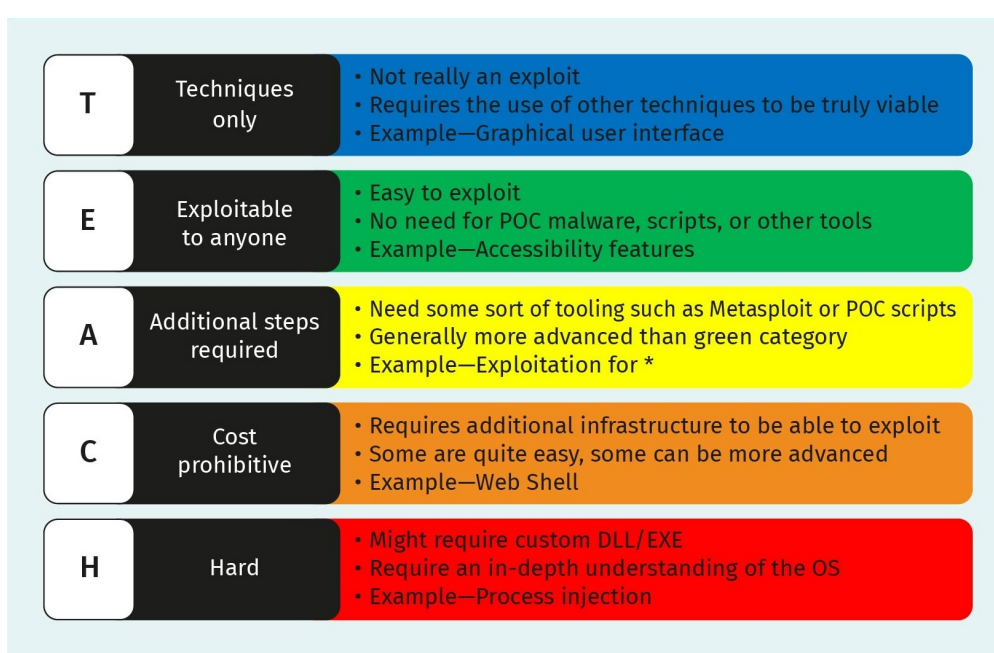
Two sources can be used to collect information to be analyzed with the goal of identifying, and mitigating, potential attacks: network- and host-based information. Network-based information is typically a data set from appliances on the perimeter of the network. These could be firewalls, intrusion detection systems, switches, or any other device that can monitor the perimeter of the organization’s network. The downside to this source is that the organization has a reduced ability to identify anomalous or malicious traffic within the network boundary of the organization. The network perimeter appliances typically face outward, like guard towers hoping to spot the enemy before they can compromise the perimeter wall. If the enemy successfully compromises the perimeter, their network activity is now on the wrong side of the organization’s sensors. Placing the sensors within the network can also be problematic because of the sheer size of the data set that needs to be analyzed and the cost of deploying the additional sensors. As the organization grows, and its network grows with it, the difficulty of this increases exponentially.

Host-based information can provide a more granular data set, especially since operating systems are designed with data collection in mind. Windows 10 has increased event tracing and forwarding and provided additional audit tools, while Linux is being deployed with integrity management architecture. Invincea Labs researched using host-based information and found that analyzing up to 200 MB of log files per day allowed them to identify over eighty percent of the compromised hosts (Berlin et al., 2015). This framework can also be used as a training tool. Travis Smith (2018) created a set of categories that can be applied to the framework in order to identify techniques usable by new (or less advanced) colleagues, as well as other techniques that more experienced colleagues could use. The categories are as follows (Smith, 2018):

- techniques only. These techniques are not considered exploits. They can be used alongside other techniques to accomplish an objective. Using PowerShell or a graphical user interface is an example of this category.
- exploitable to anyone. These are easy-to-use techniques that can be given to newer employees. They do not require proof of concept (POC) malware, scripts, or other tools. Accessibility features and registry run keys are examples of this category.
- additional steps required. Some sort of additional resources, such as Metasploit, is required. These are generally more advanced than the previous category.
- cost prohibitive. These require infrastructure to be used or researched. They may also require a secure environment for testing. Some are quite easy to use, while others are more advanced. Web Shell is an example of this category.
- hard. These may require a custom .dll or executable. An advanced understanding of the operating system or hardware is likely needed. An example of this category would be a rootkit or process injection technique.

The categories are also color-coded, as shown in the following figure.

Figure 91: TEACH Categories for MITRE's ATT&CK Framework



Source: Smith (2018). Used with permission.

We can now apply color coding to the framework navigator through .json files available on GitHub, allowing us to quickly identify the techniques and the difficulty level.

Figure 92: TEACH Categories Applied to MITRE's ATT&CK Framework

Privilege escalation	Defense evasion	Credential access	Discovery
Access token manipulation	Access token manipulation	Account manipulation	Account discovery
Accessibility features	Binary padding	Brute force	Application window discovery
AppCert dynamic link libraries (DLLs)	Background intelligence transfer service (BITS) jobs	Credential dumping	Browser bookmark discovery
AppInit DLLs	Bypass user account control	Credentials in files	File and directory discovery
Application shimming	CMSTPExe	Credentials in registry	Network service scanning
Bypass user account control	Code signing	Exploitation for credential access	Network share discovery
DLL search order hijacking	Component firmware	Forced authentication	Password policy discovery
Exploitation for privilege escalation	Component object model hijacking	Hooking	Peripheral device discovery
Extra window memory injection	Control panel items	Input capture	Permission groups discovery
File system permissions weakness	DCShadow	Kerberoasting	Process discovery
Hooking	Deobfuscate / Decode files or information	LLMNR/NBT-NS poisoning	Query registry
Image file execution options injection	Disabling security tools	Network sniffing	Remote system discovery
New service	DLL search order hijacking	Password filter DLL	Security software discovery
Path interception	DLL side-loading	Private keys	System information discovery
Port monitors	Exploitation for defense evasion	Replication through removable media	System network configuration discovery
Process injection	Extra window memory injection	Two-factor authentication interception	System network connections discovery
Scheduled task	File deletion		System owner or user discovery
Service registry permissions weakness	File system logical offsets		System service discovery
SID history injection	Hidden files and directories		System time discovery

Source: Smith (2018). Used with permission.

The previous figure depicts four different categories of the framework with the color-coding of the TEACH categories. All the techniques under “discovery” are highlighted in blue, which shows that the techniques are not an exploit and would need to be combined with another techniques to be viable. To use the framework, the security team would be required to train and research the identified approaches continuously. This can be done by selecting a tactic and a related technique. Once the technique has been identified, the following set of questions should be answered:

- Can the technique be tested or exploited?
- Using the mitigation steps identified in the framework, can the technique still be tested or exploited?
- After the technique has been tested or exploited, which artifacts are left behind?
- Were you able to identify any artifacts or behaviors that were not listed in the framework?

Each technique will have a list of references used in the technique’s page creation and mitigation efforts. These references will provide the resource to understand the technique and how the attacker may use it. As you are reading the documents in the reference, another question must be asked: What is missing? This is a collaborative effort, and an artifact or mitigation effort that is not listed on the page may be identified. Ultimately, the goal is to increase the security team’s knowledge base and take the framework’s information, analyze it, and test it. This will answer the following questions:

- Do the mitigation controls work?
- Can additional mitigation controls be deployed?

A balance must be created so that it does not overrun the security team with alerts. Combining the low-scale TTPs with the mid- and high-level TTPs will create a chain that could identify risk behavior for the organization. No matter which technique is used, indicators of compromise (IoCs) or TTPs, every piece of information that is encountered must be placed into context to be fully understood.

6.2 Intrusion Detection and Prevention

The intrusion detection process comprises either monitoring the network or a specific host for potentially malicious activity. Some events that may cause concern for the security team are the presence of malware on the host, an unauthorized user gaining a foothold within the network, authorized users exceeding their permissions, or an attacker attempting to increase the level of their permissions. An intrusion detection system (IDS) can be a hardware appliance placed on the network to monitor traffic or a software installed on the host connected to the network. An intrusion prevention system (IPS) can also be an appliance connected to the network or installed on a host connected to the network. An IDS is used to detect potential intrusions into the network, while an IPS is designed to prevent unauthorized access to the organization’s network. The two devices provide similar func-

tionality, and, sometimes, an IPS is used by an organization as an IDS with the prevention functions disabled. We can refer to both systems as intrusion detection and prevention systems (IDPS).

When an organization uses an IDPS, the security team will react to alerts generated by the IDPS to determine whether the activity indicates a potential attack. We can also configure the IDPS to recognize violations of the organization's security policies. By creating a rule set, the organization can also monitor the authorized users to ensure they follow the organization's authorized use policies. This can also come into play when dealing with a disgruntled employee (the insider threat). Monitoring the user's ability to transfer data, such as files from the server to a laptop, could generate a flag that the security team could then investigate.

The attacker's ability to conduct reconnaissance against the organization is unhindered by the organization's security policies. Using an IDPS may allow the organization to create an early warning system by monitoring a specific behavior set, such as an attacker scanning for open ports. Suppose the organization is only looking at the internet-facing hosts. In that case, this will create an enormous collection of data to analyze since this is the type of activity that is an everyday staple on the internet. Where this feature would come into play is the internal hosts on the network. If a user, authorized or unauthorized, is conducting port scans on hosts on the internal network, this could indicate an ongoing attack. When an organization deploys an IDPS throughout the network, the following comprise additional benefits other than threat identification or prevention:

- deterrent control. If an authorized user is aware that their behavior may be recorded and monitored, their actions are more likely to conform to the authorized use policy.
- quality control. This can be used to audit current security policies to determine if they are effective, for example, whether the existing firewall rules are blocking the desired traffic.
- threat documentation. As the IDPS identifies and logs potential attacks, the security team can show this documentation to the C-level executives (or decision-making supervisors) to justify additional mitigation efforts or identify changes in the security scheme.

When the organization deploys an IDPS throughout the network environment, they are looking for the following functionalities:

- reports (generating a report about the monitored events or specific incidents)
- incident reporting (creating a local record of events that match the specified criteria—the system may send an event log to an off-site or non-local destination)
- alert notification (sending an alert to the security team when an event or series of events meets the pre-defined criteria)

Some IDPSs can adjust the security profile when circumstances dictate. For example, if the IDPS alerts on malicious activity, it would initiate a more granular collection of data relating to that event. While the technologies used by an IDS and an IPS are similar, an IPS is designed to “prevent” an attack from succeeding. The IPS can do this in several ways, but the approaches will generally fall into the following categories:

- Stop the attack. The connection between the network and the attacker is terminated, blocking the attackers' access to the target host.
- Change the security configuration. When an attack is identified, the IPS can change the configuration of network devices being used by the attacker. This could include a network-based firewall, a router or switch, and host-based firewalls. Some IPS technologies allow for the deployment of security patches to the victim host if the attacker is using a known vulnerability.
- Modify content. This could be as simple as removing malware attached to an email or having the IPS act as a proxy server, which then removes the attacker's information from the header or other parts of the packet.

Even with this advanced technology available to an organization, an IDPS must still be tuned to the organization's environment. The IDPS is not a magical appliance that is always one hundred percent accurate; it will still generate false-positive or false-negative results. A false positive occurs when the IDPS has alerted to activity that is not malicious. A false negative occurs when the IDPS does not create an alert about malicious activity. Using current technology, it is not possible to remove all false positive or false negative alerts. If the organization increases the focus of the IDPS, the occurrence of false positives may also increase. Likewise, if the organization decreases the focus of the IDPS, then the organization may see an increase in false negatives. When comparing a false positive to a false negative, there is a more significant detriment to the organization when a false negative occurs. The organization will need to adjust the settings for the IDPS for the organization's environment, which will create a custom configuration for each IDPS on the network.

The IDPS can use multiple methods to identify and prevent an attack on the organization's network. Using different methodologies allows the IDPS to become more accurate in its workflow. Let us look at some methodologies you may come across: signature-based, anomaly-based, and deep packet inspection. With a signature-based method, the IDPS will hold a database of known "signatures" of malicious behavior. These can include matching file hashes on malware or suspicious behavior. An example of suspicious behavior is an increase in a user's permissions or a user attempting to use an administrative account, such as "root" or "admin," to log into a device. This method requires knowledge of the attack tactics that were used before. If an attacker uses a new vector or changes parts of the known attack, then the signature of the malicious action will not match what is stored in the database.

With an anomaly-based method, the IDPS will require a baseline of the standard behavior of the network. We refer to this as the learning or training period. Any activity that is outside of the norms of the baseline will generate an alert. For example, if the organization's workforce is on a Monday through Friday schedule with 0800 to 1600 work hours, it will find activity during this timeframe normal. It would be normal to have a login spike at 0800, with email activity increasing between 0800 and 0830. The IDPS would therefore flag an increase in network traffic at 2000. By consistently monitoring the day-to-day activity of the organization and the use of the network and its resources, the anomaly-based methodology can be very effective. However, it may take some time before the IDPS is fully aware of the expected behavior of the organization. This also includes behavior that may

be occasional, such as monthly backups or monthly transfers of log files to an off-site location. If the activity does not occur during the learning/training period, then the IDPS will flag that behavior as malicious.

Deep packet inspection (or stateful protocol analysis) occurs when the IDPS monitors and tracks all the network connections. The IDPS will have to understand the application, network, and transport protocols, and recognize when an attacker is misusing them. By analyzing each packet, the IDPS can alert when the attacker issues the same command multiple times or when a command is issued without a requisite qualifying command. The IDPS will come pre-configured with the protocol standards. Using deep packet inspection and protocol analysis allows the device to identify potentially malicious behavior, for example, if a username is required and the max length of the username is **twenty-four** characters, when an attacker attempts to use **one hundred** or **one thousand** characters in the username field, this will create an alert. The vendor of the IDPS will use protocol standards approved by the Internet Engineering Task Force (IETF). Unfortunately, some application and hardware vendors will violate the standard or add proprietary features when using the different protocols. This may create the need to reverse engineer any changes that are being used by the application or hardware device on the organization's network. This can result in a very resource-heavy process. Depending on the size of the network, the number of hosts, the number of outbound or inbound connections, and the number of simultaneous connections, the organization may have to fund additional resources to keep the network experience from being degraded.

Those were the categories where the IDPS will function, but where, within the network topology, can the IDPS be deployed? Possible constellations are as follows:

Network-based (hard-wired) deployment

The IDPS can be attached to the network at specific network segments or specific devices. As traffic is being transmitted across the wire, the IDPS will monitor and identify potential malicious activity. Ideally, the IDPS would be deployed near the network perimeter, near firewalls or on the border of the DMZ. These servers may have access to outside users such as a virtual private network (VPN) server, remote access server, or near the boundary of the wireless network. The outward-facing servers and devices may be the first line that the attacker will identify.

Network-based (wireless) deployment

The IDPS can be attached near the wireless access point or router to the network traffic and identify potential malicious activity.

Behavioral analysis deployment

The IDPS can be attached to the internal network of the organization. It is used to look for unusual data flow, recording what occurs when a worm or virus has been executed on the network. It can also be utilized to identify potential distributed denial of service (DDoS) traffic and identify administrative policy violations.

Host-based deployment

The IDPS is responsible for monitoring the traffic and events occurring on a specific host. It will enable this on critical systems or systems that contain sensitive information. The host-based IDPS can monitor the network traffic that is being sent and received by the host and monitor the event logs, processes, and application activity.

Depending on the vendor of your IDPS, there may be more components present than are covered in this course book. Typically, an IDPS will comprise the following components:

- sensor (or agent). A sensor is used to collect network-based traffic. An agent is used to monitor host-based events.
- management server. This is a centralized server that manages the sensors and agents. The management server can receive information and analyze it to identify malicious events. Since it receives information from various sources, it correlates the information to identify potentially malicious traffic. Suppose sensors A, B, and C report suspicious traffic from the same IP address. The management server analyzes the information and sends an alert to the security team or blocks the traffic. There may be multiple management servers on the network. There may also be a two-tiered system, where a group of management servers reports to an overall management server.
- database server. The database server will be the storage location for the information generated by the sensor, agents or management servers.
- console. This is a program that provides a conduit or interface for the administration of the IDPS system. The console software can be installed on either a desktop or laptop and provide administrative and monitoring capabilities. Some consoles can only be used for administration purposes.

The capabilities of an IDPS can vary depending on the vendor, but, generally speaking, an IDPS is able to collect information about the network. This could be as simple as identifying the operating system being used by the hosts on the network. Logging is a crucial feature of the IDPS. The system alerts on potential malicious events. The analyst must therefore have sufficient information to place that alert into context. Temporal information, such as dates and times, is essential in this setting, especially when correlating events from various sources on the network. The IDPS may flag several events. The organization may label this as a low-level concern and continue up the severity ladder to higher-priority events. If the analyst is looking at a high-priority event, analyzing prior low-level events may provide insight into the situation. Also, if the IDPS initiated any preventive actions, that is information the security team will want to have. If the IDPS is being used for a deep packet inspection and stateful packet analysis, there may be additional logs relating to the types of traffic sent on the network. As the logs are generated, they should be exported to an off-site or central location. If an attacker gains access to the organization's network, one technique they will use is deleting host-based log files to help hide their presence.

A vital feature of an IDPS is its ability to detect malicious traffic and actions. Once an organization has decided to deploy an IDPS, it must be customized or "tuned" for the organization's network. The organization will want to customize the "threshold" levels. The threshold level is used to determine what is and is not malicious behavior. The threshold will set a maximum accepted value for a specific action, such as failed login attempts.

If there are **thirty** failed logins within five minutes, they can configure the IDPS to alert the security team. You will typically find the threshold levels used when the IDPS performs deep packet inspection, packet analysis, or anomaly-based detection.

The IDPS can also maintain a white (approved) list and a black (unauthorized) list, which will contain IP addresses, port numbers, uniform resource locators (URLs), applications, usernames, file names, and file types. If the connection comes from a location listed on the unauthorized list, the IDPS sends an alert to the security team and blocks the connection. Using whitelists and blacklists is most common when the IDPS is using signature-based detection methods, and can also be used when performing deep packet inspection or packet analysis.

Securing the IDPS is critical because it is one of the first targets an attacker will seek out. If the attacker can compromise the IDPS, the organization cannot detect current or future malicious actions. The IDPS can also be a repository of critical information such as the configuration information of the network and its hosts. Each user and administrator of the IDPS must have their own credentials for the IDPS. Each user must have the minimum permissions required to accomplish their tasks. If users are sharing credentials, then there is no way to audit user activity. Once the IDPS has been deployed on the network, firewalls and routers will need to be configured to limit access to the hosts that need to access the IDPS. The organization should protect connections to the IDPS. This could be using physical protection, where the user can only connect while on the same physical network, or by restricting connections to virtual networks such as a virtual local area network (VLAN). If connections are to be made outside of the organization's environment, a virtual private network (VPN) or transport layer security (TLS) connection should be considered.

6.3 Correlation of Events

Once the IDPS has been deployed, it will start generating alerts. In a perfect world, the alerts would indicate malicious activity **one hundred** percent of the time. However, there must be an analysis to place these alerts into context and determine whether a false positive has occurred. If the security analyst decides that the alert is documenting malicious activity, the organization initiates the incident response plan. As the security team is attempting to respond to the malicious activity, they will also try to determine when the initial foothold was gained by the attacker and whether there are additional records of the attacker's activity. A security team will start an alert correlation. Multiple IDPSs will have separate log files for each device. Let us look at some devices that can be a source of information for the investigation:

- IDPS. This will be the first stop to collect information. We may find that a sizeable amount of information has been collected, including data fields in the packets traversing the network.

- network devices. This could include information from routers, firewalls, proxy servers, and remote access servers. The information that can be collected here could show trending behavior. The information found here could also help track the attacker. For example, if the IDPS creates an alert for malicious activity, the perimeter firewall may have logs showing where the attacker entered the organization's network.
- packet sniffers. When deploying a sniffer onto a production network, extreme amounts of data will be collected. This could reach millions of packets being recorded. However, most of the packets collected will be related to everyday activity.

Alone, these sources of information may not provide the key to solving the investigation and identifying the attacker. However, when we combine the data sets, we can start creating a complete picture of what had occurred. Naturally, all the devices must have network time protocol (NTP) on the network. If we do not have the time reference, it will be near-impossible to merge the data from all the devices.

Log analysis has been assigned to the system administrators responsible for analyzing logs and identifying events of interest. Log analysis is a challenging aspect of log management. This is due, in part, to the perceived lack of value for the effort. To help prioritize log analyses, the organization can create a training program to ensure all personnel meets the minimum knowledge, skills, and abilities needed for log analysis. The training can also teach employees to use software tools for recognizing the patterns of suspected malicious activity (Souppaya & Kent, 2006). When you combine this with the reactionary nature of log analysis that has been traditionally shown, it is doubtful the organization could be proactive in its approach to prevent malicious activity using log analysis. The following are some practices the organization could implement to increase the efficiency of log management and analysis:

Prioritization

The organization must create a culture where log management and analysis are recognized as essential security protocol processes. Of course, the organization must also ensure that they meet the requirements set by administrative and statutory governing bodies.

Policies and procedures

The organization must create a clear policy regarding its expectations on how log management and analysis are conducted. The organization's policy must also provide employees with explicit instructions for the policy's implementation. Using policies and procedures will help the organization meet their administrative, regulatory, and statutory requirements. In addition, the policies and procedures will need to be audited regularly to ensure compliance and that the policies and procedures are being performed correctly.

Log infrastructure

The organization must ensure that the confidentiality, availability, and integrity of the log files are maintained. Creating infrastructure that dictates how the log files will be handled and preserved will help prevent both intentional and accidental destruction of the log files. In addition, the infrastructure has to handle the creation of log files at both standard and extreme conditions.

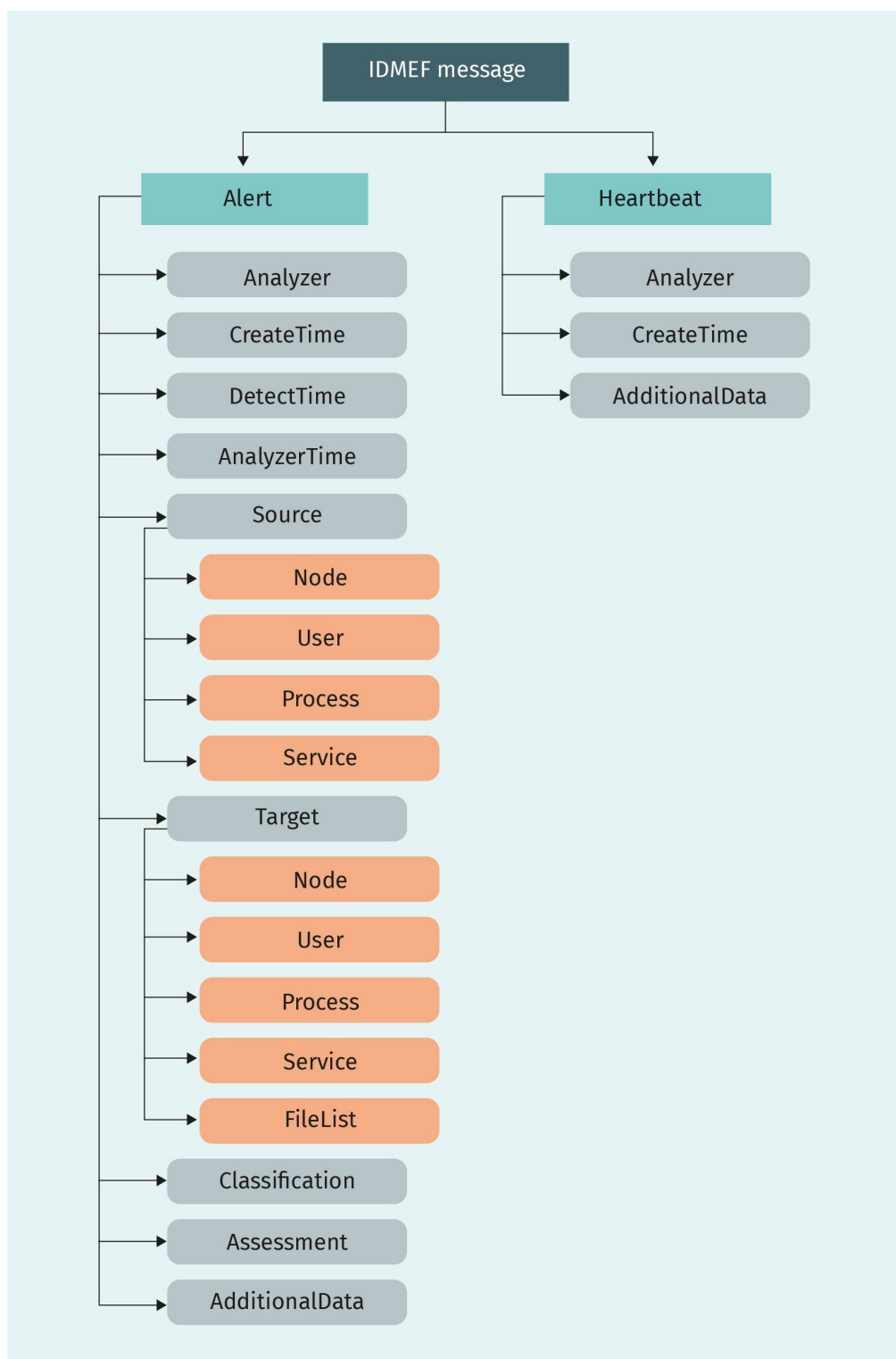
Management responsibility

The organization provides the employees with training and tools for log management and analysis. By creating a culture within the organization that recognizes the importance of log management and analysis, the employees now have a “buy-in” regarding the maintenance and analysis the log files.

The analyst uses alert correlation to take data from a variety of sources. The analysis process creates a report about the security status of the network. The organization’s network may have several components and be targeted by different attacks. This results in a variety of alert logs, whose content comes in various formats. Before the analyst can compile the information, they must convert it into a standard format. We refer to this as normalization. The goal of normalization is to take data and put them into a form that can be easily understood and analyzed. The Internet Engineering Task Force (IETF) created a formatting standard that can exchange information. We know this format as the Intrusion Detection Message Exchange Format (IDMEF), a standard format for use by automated intrusion detection systems to report behavior that is suspicious or of relative interest (Intrusion Detection Exchange Format Working Group, 2007b). IDMEF messages are classified into two different categories: **heartbeat** and an alert message. The following figure shows the makeup of an IDMEF message.

Heartbeat message
A heartbeat message is a signal sent by hardware or software at a pre-set interval to indicate normal operation.

Figure 93: IDMEF Schema



Source: Chloebn78 (2016). CC BY-SA 3.0.

The heartbeat message is sent at a specific interval to indicate that the sender is operational and still monitoring traffic. If a heartbeat message is not received at specific intervals, it may show that the appliance monitoring the network is not functioning and the network traffic is not being monitored. The alert message is sent when the monitoring station has observed suspicious behavior that requires further investigation by a human analyst. The IDMEF has several fields for information. An alert message has significantly more information than a heartbeat message (as shown in the previous figure). An alert message contains the following fields of information:

- analyzer. This is the source of the alert.
- create time. This is the date and time that the alert was created.
- detect time. This is the date and time the suspicious traffic was detected.
- analyzer time. This is the date and time the alert was sent.
- source. This is information about the source of the suspicious traffic (node, service, process, user, etc.).
- target. This is information about the target of the suspicious traffic (node, service, process, user, etc.).
- classification. This is the name of the alert, along with any other additional information such as the common vulnerability and exposure information (CVE).
- assessment. This is a characterization of the damage potential of the suspicious activity.
- additional information. Any other information that may be pertinent is found here.

There are also three alert categories (Intrusion Detection Exchange Format Working Group, 2007a):

1. Correlation alert. This indicates individual, separate instances of suspicious activity that could be related.
2. Overflow alert. This is generated when there is a buffer overflow activity.
3. Tool alert. This identifies suspicious activity based upon the attack tool being used.

Once the data have been normalized, the next stage is pre-processing the alerts to identify and discard false positives. Failure to discard false positives can increase the size of the data set, which also increases the time needed to analyze the information. We can now correlate the data set of the alerts. The correlation process attempts to re-create the attack scenarios based on the information generated in the alerts. We can classify the alerts into several categories, including the following:

- similarity. This can be based on one or more identified fields, such as IP address, destination, or port number of the suspicious traffic.
- statistical. Similar actions will generate similar statistical features, such as repeated patterns. The relationship between alerts is also considered statistical, for example, alert A occurs at 0000 and alert B follows three minutes later.
- knowledge-based. This involves knowing the attack prerequisites and the expected behavior if the attack is successful. This acts as a pre-defined pattern in the attack. For example, if the attacker successfully uses technique Y, technique Z is the next logical step. The MITRE ATT&CK framework is a resource for this category.
- hybrid. This includes any combination of the above-listed categories.

The correlation process is designed to give the analyst a way to take **all the information thrown their way** and transform it into actionable intelligence. This will have to be an everyday activity and responsibility of the security team. Once the process has been implemented and tested, the organization can be more proactive in identifying suspicious behavior before the attacker gains a foothold within the network.



SUMMARY

Network forensics can be a very complicated and intimidating aspect of security, but this intimidation factor can be reduced through education and reputation. There are several ways to identify malicious traffic, from signature-based and anomaly-based strategies to learning about the tactics, techniques, and procedures that attackers will use. When using attackers' tactics, techniques, and policies, the organization will share information with their employees and other organizations. Identifying the techniques that work for the attacker is key to understanding how to mitigate a specific technique. This can also be used as a teaching tool for new or current members of the organization. MITRE has created a framework that organizations can use to develop testing networks to familiarize security analysts with different attacks.

Analyzing the log files from intrusion detection and prevention systems (IDPSs) should be a proactive endeavor by the organization's security team. Alert fatigue is an issue that needs to be addressed, especially since it is unavoidable that a considerable number of alerts will be false positives. IDPS systems can be designed to analyze specific signatures while monitoring traffic, or look for types of behavior that indicate an attack. But before the data from these IDPSs can be used, they must be normalized and placed into the same format.