**WINDOWS REGISTRY**

Windows forensics is a branch of digital forensics focused on the investigation of Windows-based operating systems to uncover evidence of cybercrimes, security incidents, or system misuse. It involves collecting, analyzing, and preserving digital evidence from Windows devices, ensuring the integrity of the data for legal or investigative purposes.

**WINDOWS FILE SYSTEM**

The Windows file system is a method for storing and organizing files on a storage device, such as a hard drive or solid-state drive (SSD). Windows primarily uses two types of file systems: **FAT (File Allocation Table)** and **NTFS (New Technology File System)**. Each has its features and advantages, with NTFS being the more advanced and widely used in modern versions of Windows.

**Types of File Systems in Windows:**

1. **FAT (File Allocation Table):**
   * **FAT16** and **FAT32** are earlier versions of the FAT file system.
   * **FAT32** is the most commonly used in removable drives (like USB flash drives) due to its simplicity and compatibility with different operating systems.
   * **Limitations**:
     + Maximum file size: 4 GB.
     + Maximum volume size: 8 TB (with special formatting).
   * **Advantages**: Simple, widely compatible with other OSes (Linux, macOS, etc.).
2. **NTFS (New Technology File System):**
   * **NTFS** is the default file system for modern Windows operating systems (from Windows XP onward).
   * **Features**:
     + **Security**: Supports file-level security using Access Control Lists (ACLs) to define who can access or modify files.
     + **Compression**: Can compress files or folders to save space.
     + **Encryption**: Supports encryption through the **Encrypting File System** (EFS).
     + **Journaling**: Keeps a transaction log to help recover from system crashes, ensuring data integrity.
     + **Large File Support**: Can support very large files (up to 16 exabytes) and volumes (up to 256 terabytes).
   * **Advantages**: Robust, secure, and optimized for large files and volumes.
3. **exFAT (Extended File Allocation Table):**
   * This is a variant designed to address the limitations of FAT32, mainly for flash drives and SD cards.
   * **exFAT** allows larger file sizes and is compatible with both Windows and macOS.
   * **Use cases**: Best for external drives and high-capacity storage devices.

**File System Structure (NTFS Example):**

NTFS organizes data in a hierarchical structure using several key components:

1. **Master File Table (MFT)**:
   * The heart of NTFS, which holds metadata about all files and directories. Every file, directory, and system object has a corresponding entry in the MFT.
   * Contains information like file size, attributes, data location, access control lists (ACLs), and timestamps.
2. **File Records**:
   * Each file in NTFS has an MFT entry that points to the actual data and its location on the disk.
3. **Clusters**:
   * The smallest unit of disk space allocated by NTFS for storing files. A file is split across one or more clusters if it’s large enough.
4. **NTFS Metadata Files**:
   * Special files used for managing the file system, like the $MFT, $LogFile, $Volume, and $Bitmap. These are crucial for system integrity and recovery in case of errors or crashes.
5. **Attributes**:
   * Files and directories in NTFS can have several attributes, such as **ReadOnly**, **Hidden**, **System**, and **Archive**, or more advanced attributes like **Compression** or **Encryption**.
6. **Security Descriptors and Access Control Lists (ACLs)**:
   * These define who has access to files and what kind of operations they can perform (e.g., read, write, execute).
7. **Journaling and Log Files**:
   * NTFS uses journaling to keep track of changes to the file system. This helps ensure data integrity and recovery after a system crash or power failure.

**Key NTFS Features Relevant to Forensics:**

1. **Timestamps**:
   * NTFS stores several timestamps for each file, including:
     + **Creation Time**
     + **Last Modified Time**
     + **Last Accessed Time**
   * These timestamps are critical for forensic investigators in building a timeline of events.
2. **File Recovery**:
   * NTFS maintains metadata that allows for recovery of deleted files in some cases, especially if the data blocks haven’t been overwritten yet.
   * Forensics tools can scan the MFT and attempt to recover deleted files or examine unallocated space.
3. **Alternate Data Streams (ADS)**:
   * NTFS supports storing additional metadata within a file via alternate data streams.
   * These streams can sometimes contain hidden or suspicious data, often used for steganography or malware hiding.
4. **Volume Shadow Copies**:
   * NTFS has the ability to create **Volume Shadow Copies**, which are snapshots of the system at a certain point in time. These can be used to recover deleted files or analyze previous states of the system during forensic investigations.
5. **File System Metadata Analysis**:
   * Investigators can analyze MFT records to uncover important evidence about file usage, access patterns, and user activities. The MFT contains details like file name, size, timestamps, permissions, and file system location.

**Common Windows Forensics Tasks Involving the File System:**

* **File Recovery**: Using forensic tools to recover deleted files or partitions from an NTFS file system.
* **Timeline Creation**: Analyzing timestamps to build a timeline of user actions or system events.
* **Artifact Recovery**: Identifying and recovering artifacts such as recent document lists, browser history, or system logs.
* **Metadata Analysis**: Investigating file metadata for hidden clues about usage, manipulation, or tampering.

**WINDOWS REGISTRY**

The **Windows Registry** is a hierarchical database used by Microsoft Windows to store low-level settings and configurations for the operating system, applications, and hardware devices. It is a critical component of the Windows environment, as it contains detailed information about system configuration, installed software, user preferences, and much more. Because of its importance in system operations and its wealth of evidence, the Windows Registry is often a key focus in digital forensics.

**Structure of the Windows Registry**

The Windows Registry is organized into a set of **hives**, which are the primary containers that hold registry keys and values. Each hive corresponds to a specific part of the system configuration. The main registry hives are:

1. **HKEY\_CLASSES\_ROOT (HKCR)**:
   * Stores file associations and COM (Component Object Model) object registrations. This key controls how files are opened and processed by specific applications (e.g., file extensions like .txt or .exe).
2. **HKEY\_CURRENT\_USER (HKCU)**:
   * Contains user-specific settings and preferences. This includes desktop configurations, application settings, recent documents, and network connections for the currently logged-in user.
3. **HKEY\_LOCAL\_MACHINE (HKLM)**:
   * Contains settings for the local machine that affect all users. This includes hardware configuration, installed software, and system-wide settings.
4. **HKEY\_USERS (HKU)**:
   * Contains information about all user profiles on the machine. Each user profile is stored as a subkey, with settings specific to that user.
5. **HKEY\_CURRENT\_CONFIG (HKCC)**:
   * Contains information about the current hardware profile, including display settings and other device-specific configurations.

**Key Concepts and Components of the Registry**

* **Registry Keys**: These are containers that hold values and other keys (subkeys). Keys are organized in a tree structure.
* **Registry Values**: These are the actual data stored within a key. Values can be different types:
  + **String (REG\_SZ)**: A text string.
  + **Binary (REG\_BINARY)**: Raw binary data.
  + **DWORD (REG\_DWORD)**: A 32-bit integer value.
  + **QWORD (REG\_QWORD)**: A 64-bit integer value.
  + **Expandable String (REG\_EXPAND\_SZ)**: A string that can contain environment variables (e.g., %SystemRoot%).
  + **Multi-String (REG\_MULTI\_SZ)**: A list of strings.
* **Subkeys**: These are keys nested under other keys, creating a hierarchical structure. Subkeys can contain values or other subkeys.

**What Forensics Investigators Look for in the Registry**

1. **User Activity**:
   * **MRU (Most Recently Used)** Lists: The Registry often contains information about recently accessed files, folders, or applications. For example, the key HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\RecentDocs stores recent file activity.
   * **Application Usage**: Keys under HKCU\Software or HKLM\Software can provide evidence of installed applications and their usage.
2. **Login History**:
   * Information about user logins, including the last time a user logged in and the login name, is stored in the Registry. For example, the key HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Authentication\LogonUI can show the last user to log in.
3. **Network Configuration and History**:
   * **Wireless Networks**: In keys like HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\Connections, investigators can find details about Wi-Fi networks previously connected to.
   * **TCP/IP Configuration**: Settings such as IP addresses, DNS configurations, and the last known network interfaces are stored in the Registry.
4. **System Configuration**:
   * **Autostart Items**: Many programs register themselves to start automatically by adding entries in the Registry. For example, HKCU\Software\Microsoft\Windows\CurrentVersion\Run contains entries for programs that run when the user logs in.
   * **Scheduled Tasks**: Keys in the Registry related to the Task Scheduler can reveal scheduled processes and tasks. Look under HKLM\Software\Microsoft\Windows\CurrentVersion\Tasks.
5. **Security and Permissions**:
   * The Registry can contain settings related to system security and user permissions, including Group Policy settings (e.g., HKCU\Software\Microsoft\Windows\CurrentVersion\Policies) and password policies.
6. **System Crashes and Errors**:
   * The **EventLog** key (HKLM\SYSTEM\CurrentControlSet\Services\EventLog) can store logs related to system errors, crashes, and other significant events.
7. **USB Devices**:
   * Information about connected USB devices can be found in the Registry, especially in keys under HKLM\SYSTEM\CurrentControlSet\Enum\USBSTOR. This can be critical for identifying USB device usage, especially for forensic investigators tracing data exfiltration.
8. **Deleted Data and Forensic Evidence**:
   * The Registry may contain remnants of deleted files, programs, or user activity. If a program was uninstalled, there could be lingering traces in the Registry, such as orphaned keys or values that could be vital to an investigation.

**Registry Forensics: Key Techniques and Tools**

1. **Timeline Creation**:
   * Investigators use timestamps in the Registry (such as LastWrite times for keys) to create a timeline of user and system activity.
2. **Recovering Deleted Data**:
   * Even though data in the Registry might seem deleted, it is often possible to recover remnants of registry keys and values that were once active. Tools like **Registry Viewer** can be used to examine these remnants.
3. **Registry Analysis Tools**:
   * **RegRipper**: A popular open-source tool for extracting and parsing information from Windows Registry files. It helps generate a report based on known Registry artifacts related to specific activities.
   * **FTK Imager** and **EnCase**: These tools provide registry analysis features, including the ability to extract and analyze Registry data as part of a larger investigation.
   * **Volatility**: A memory forensics tool that can help in analyzing the Registry from a memory dump.
4. **Forensic Considerations**:
   * **Data Integrity**: When working with the Registry, it's crucial to maintain the integrity of the data by using write-blockers and making forensic copies.
   * **Carving and Searching**: In cases of tampering, investigators may need to search for specific keywords or look for residual data in the Registry files.

**Forensic Artifacts in Specific Registry Locations:**

* **Recent Documents**:
  + HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\RecentDocs – A list of recently accessed documents.
* **Installed Software**:
  + HKLM\Software\Microsoft\Windows\CurrentVersion\Uninstall – Information about installed software and its version.
* **USB Devices**:
  + HKLM\SYSTEM\CurrentControlSet\Enum\USBSTOR – Information about USB devices that were connected to the system.
* **User Logins**:
  + HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Authentication\LogonUI – Information about the last logged-in user.

**EVENTS LOGS**

Windows **Event Logs** are essential for tracking the activity and health of a system, providing detailed information about system, security, and application events. These logs help administrators, security professionals, and forensic investigators monitor and troubleshoot issues, as well as detect potentially malicious activity.

There are several different types of event logs in Windows, each designed to capture specific kinds of events. Event logs are recorded in the **Event Viewer**, a built-in tool that allows users to view, analyze, and filter log data.

**Types of Event Logs in Windows**

1. **System Log**:
   * Records events related to the operating system and hardware components, such as drivers, hardware failures, and system services.
   * Examples of events:
     + **Service start/stop** (e.g., a service crashing or failing to start).
     + **Driver issues** (e.g., a device not initializing properly).
     + **System crashes** or critical errors.
2. **Application Log**:
   * Contains events generated by software applications. Applications use this log to store errors, warnings, and other operational information.
   * Examples of events:
     + Software application crashes or errors.
     + Configuration issues within a particular application.
     + Logs from Windows components like IIS or SQL Server.
3. **Security Log**:
   * Tracks security-related events, such as login attempts, privilege use, or access to critical system files.
   * Key for forensic investigators, as it can provide evidence of unauthorized access, failed login attempts, or privilege escalation.
   * Examples of events:
     + **Logon/Logoff events**: Successful or failed login attempts, including user name, timestamp, and source.
     + **Account Lockouts**: A user account locked due to multiple failed login attempts.
     + **Object Access**: Who accessed or attempted to access a protected file or directory.
     + **Privilege Use**: Attempts to access high-level system privileges (e.g., Administrator rights).
4. **Setup Log**:
   * Captures events related to the installation of Windows and updates to system components.
   * Examples of events:
     + **Windows installation** or upgrades.
     + **Driver installations** or failures during installation.
5. **Forwarded Events Log**:
   * Stores events collected from remote machines that are forwarded to a central logging server.
   * Used in enterprise environments where logs from multiple systems are consolidated into a central repository for analysis and monitoring.

**Key Event Log Fields**

Each event in a log is composed of several fields that contain detailed information. Some of the key fields include:

* **Event ID**: A unique identifier for each event. This number can help investigators quickly identify the type of event and its meaning.
* **Source**: The application or system component that generated the event (e.g., "Microsoft-Windows-Security-Auditing").
* **Timestamp**: The date and time when the event occurred.
* **Event Level**: Indicates the severity of the event (e.g., Information, Warning, Error, Critical).
* **Message**: A detailed description of the event, including any associated data (e.g., user names, error codes, or failed action descriptions).
* **User**: The user account associated with the event (if applicable).
* **Computer**: The name of the machine where the event occurred.

**Common Event Log IDs (Security Logs)**

In digital forensics, security logs are among the most important for identifying malicious activities or user behavior on a system. Here are some common **Event IDs** that forensic investigators look for:

1. **4624** (Logon):
   * A successful user logon event. The event contains details about the user who logged in, the time, and the logon type (e.g., local, remote, or through a network).
2. **4625** (Failed Logon):
   * A failed logon attempt. This event helps detect brute-force attacks or unauthorized login attempts.
3. **4634** (Logoff):
   * A successful logoff event. Tracks when a user logs off, which can be useful for determining system usage time.
4. **4720** (Account Created):
   * An event that logs the creation of a new user account on the system. Important for detecting unauthorized user creation.
5. **4723** (Password Change Attempt):
   * Logs an attempt to change a user's password, whether successful or failed.
6. **4740** (Account Locked):
   * Occurs when an account is locked due to too many failed login attempts. This can indicate a brute-force attack or account compromise.
7. **4670** (Permission Change):
   * Logs a change to object permissions, such as file or folder access control. Important for identifying unauthorized modifications to critical system files.
8. **4768** (Kerberos Authentication Ticket Request):
   * Logged when a Kerberos authentication ticket is requested. Useful in tracking authentication events in Windows environments using Kerberos.
9. **5140** (Network Share Accessed):
   * Indicates that a network share has been accessed. Critical for tracking file sharing and potential unauthorized access to shared resources.

**Using Event Logs for Forensics**

For forensic investigators, Windows event logs are a treasure trove of information. Some typical use cases include:

1. **Timeline Construction**:
   * By correlating timestamps from security, system, and application logs, investigators can reconstruct a timeline of system activity, including user logins/logoffs, application events, and system errors.
2. **Tracking User Activity**:
   * Investigators can look at **Event ID 4624** and **Event ID 4625** (successful and failed logon events) to identify which user accounts were used to access the system and whether there were any suspicious login patterns (e.g., logins at odd times or from unfamiliar IP addresses).
3. **Investigating Unauthorized Access**:
   * Reviewing **Event ID 4740** (account locked) or unusual **Event ID 4625** (failed logons) can help identify attempts at unauthorized access, such as brute-force attacks.
4. **Correlating with Other Evidence**:
   * Event logs can be cross-referenced with data from other sources, such as file system analysis or network traffic logs, to confirm the scope of an incident, such as malware infections, unauthorized data access, or privilege escalation.
5. **Intrusion Detection**:
   * **Event IDs** related to account creation (4720), changes in security permissions (4670), and authentication failures can point to malicious activity, like privilege escalation, unauthorized account creation, or password changes.

**Tools for Analyzing Event Logs**

There are several tools that can help forensic investigators analyze event logs:

1. **Event Viewer**:
   * Built into Windows, it allows you to view, filter, and analyze events from all system logs.
2. **PowerShell**:
   * PowerShell provides command-line tools to query and extract event log information, useful for automated analysis and scripting.
   * Example: Get-EventLog -LogName Security -After (Get-Date).AddDays(-1) retrieves security logs from the last 24 hours.
3. **Log Parser**:
   * A versatile tool from Microsoft that can parse event log files and output them into different formats for further analysis.
4. **Splunk**:
   * A comprehensive log analysis platform that can ingest and analyze event logs from multiple systems, and is widely used for large-scale investigations.
5. **ELK Stack (Elasticsearch, Logstash, Kibana)**:
   * A powerful open-source platform for ingesting, storing, and analyzing event log data, commonly used in enterprise environments for continuous log monitoring.
6. **X-Ways Forensics** and **Autopsy**:
   * These forensic tools support the analysis of event logs as part of a broader investigation.

**RECYCLE BIN:**

The **Recycle Bin** in Windows is a special system folder that temporarily holds deleted files and folders before they are permanently removed from the system. It provides users with a safety net to recover files that were accidentally deleted. From a forensics perspective, the Recycle Bin is an important location to examine, as it can provide crucial evidence about user activity, deleted files, and attempts to conceal or erase data.

**How the Recycle Bin Works**

When a file or folder is deleted by the user (either through the "Delete" key or the "Delete" option in the context menu), it’s not immediately removed from the system. Instead, the file is moved to the Recycle Bin, which is a system folder designed to temporarily store deleted items. This allows users to restore files if they change their mind.

The Recycle Bin's behavior is governed by a few factors:

* **File System Location**: Deleted files are stored in a hidden folder inside the Recycle Bin, located on the disk. This folder is typically called $Recycle.Bin on NTFS systems. Each user has their own subfolder within $Recycle.Bin, identified by their unique user SID (Security Identifier).
* **File Renaming**: When files are moved to the Recycle Bin, they are renamed to a random string, with a unique file extension ($R for a regular file and $I for an index file). For example, a deleted file Document.txt might be renamed to $R123abc.txt inside the Recycle Bin folder.
* **Retention**: Files in the Recycle Bin remain there until the user either empties the Recycle Bin or the space exceeds the storage quota set for the Recycle Bin. If the bin is emptied or the storage capacity is exceeded, the files are typically permanently deleted. However, even then, traces of the files can remain on the disk until they are overwritten.

**Location of the Recycle Bin Data**

On an NTFS file system, the Recycle Bin is located in the $Recycle.Bin folder at the root of each volume (partition). The exact location of files within the Recycle Bin depends on the user who deleted the file. Each user has a unique subfolder identified by their SID (Security Identifier).

For example:

* C:\$Recycle.Bin\S-1-5-21-1234567890-1234567890-1234567890-1001\
  + This is where deleted files for a particular user are stored, identified by their SID.

Inside the user’s subfolder, deleted files are stored with random file names and a corresponding $I file that holds metadata about the deleted file.

**Forensic Relevance of the Recycle Bin**

For digital forensics, the Recycle Bin can provide key evidence. While files that have been deleted from the Recycle Bin may be permanently erased, the Recycle Bin often holds remnants of user activity that can help reconstruct events leading up to an incident.

Here are some important forensic aspects related to the Recycle Bin:

1. **Deleted Files Recovery**:
   * Even if a file is deleted from the Recycle Bin (or the user empties the bin), forensic tools may be able to recover fragments of the file or even fully restore it. This is particularly useful in investigations where an individual attempts to conceal their actions by deleting files.
2. **File Metadata**:
   * Files in the Recycle Bin retain some important metadata, such as the original filename, date of deletion, and location (in some cases). This can provide clues about the file’s origin, its relevance, and its last access time.
3. **Recent User Activity**:
   * Analyzing the Recycle Bin can show recent user activity, as it often contains files that were recently deleted. The files may be documents, images, videos, or executables, providing valuable context about what the user was working on or which files they were interacting with.
4. **Forensic Tools for Recovery**:
   * Forensic investigators use various tools to analyze and recover files from the Recycle Bin. Common tools include:
     + **FTK Imager** and **EnCase** for creating disk images and performing detailed analysis.
     + **Recuva** for recovering files that have been deleted but not overwritten.
     + **X-Ways Forensics** and **Autopsy** for in-depth examination of file systems, including the Recycle Bin.
5. **File Evidence**:
   * In a forensic context, files in the Recycle Bin might include important evidence such as:
     + Deleted documents related to a crime or investigation.
     + Evidence of an attempt to delete incriminating files.
     + Files that were deliberately hidden by the user before being deleted.
6. **Time Stamps**:
   * The time a file is deleted (or moved to the Recycle Bin) is typically recorded in the file's metadata. This can be helpful for building a timeline of events in an investigation.

**What Forensic Investigators Look for in the Recycle Bin**

* **Deleted Files**: Investigators will look for deleted files that may have been intentionally erased to conceal evidence. These could be documents, images, or software executables.
* **Timestamps**: File creation, modification, and deletion timestamps can help establish a timeline of user activity or identify suspicious behavior leading up to an event.
* **Recovered Files**: Deleted files that have been recovered from the Recycle Bin could potentially reveal critical information related to the case.
* **File Types**: The types of files in the Recycle Bin (e.g., documents, spreadsheets, emails, images) can help piece together the user’s actions and provide context about their intentions.
* **Cross-referencing with Other Artifacts**: Forensic experts often cross-reference deleted files found in the Recycle Bin with other system artifacts, such as browser history, file system metadata, or event logs, to paint a more comprehensive picture of user actions.

**Tools for Analyzing the Recycle Bin**

There are several forensic tools that allow investigators to examine the Recycle Bin in depth:

1. **FTK Imager**: A popular tool for creating forensic images of hard drives, FTK Imager also allows investigators to explore the contents of the Recycle Bin and recover deleted files.
2. **EnCase**: Another widely used forensic tool, EnCase enables investigators to examine the Recycle Bin, recover deleted files, and extract metadata for analysis.
3. **Recuva**: This is a free tool that can be used to recover files that have been deleted, including those emptied from the Recycle Bin (if the space hasn't been overwritten yet).
4. **X-Ways Forensics**: A powerful tool for file system analysis, X-Ways Forensics includes support for examining the Recycle Bin and recovering deleted files.
5. **Autopsy**: An open-source digital forensics platform that can be used to analyze and recover deleted files from the Recycle Bin, providing detailed reports.
6. **Bulk Extractor**: A tool that helps to extract data such as deleted files, URLs, and email addresses, useful in investigating files removed from the Recycle Bin.

**Best Practices for Investigating the Recycle Bin**

1. **Create a Forensic Image**: Always create a bit-for-bit copy (forensic image) of the storage device before analyzing the Recycle Bin. This ensures that no data is altered during the investigation.
2. **Check for Hidden Files**: The Recycle Bin folder and deleted files are often hidden. Ensure that the system is configured to show hidden files, or use forensic tools to locate them.
3. **Review Metadata**: Pay close attention to file metadata, especially timestamps (e.g., when the file was deleted), to help establish a timeline of events.
4. **Use Specialized Forensic Tools**: Use dedicated forensics tools to ensure the integrity of recovered files and to maximize the chances of successfully recovering deleted or overwritten files.

**PREFETCH FILE:**

**Windows Prefetch Files**

In Windows operating systems, **Prefetch files** are used to speed up the startup of applications by preloading data into memory based on user activity. These files store information about the programs that have been executed on the system, which helps Windows optimize the loading times of frequently used applications.

From a forensic perspective, Prefetch files can provide important clues about user activity, program usage, and system performance. They are an invaluable source of evidence in digital forensics, as they contain timestamps and paths for applications that were run, even after the user has closed them.

**How Prefetch Files Work**

Prefetch files are part of Windows' **SuperFetch** (introduced in Windows Vista) and **Prefetch** mechanisms, which work together to optimize the performance of the system. The purpose of Prefetch files is to store metadata about programs executed by the user.

When a program is run for the first time, Windows creates a **.pf file** for it in the C:\Windows\Prefetch folder (on the system partition). This file stores information such as:

* **Application name and path**
* **File access patterns** (which files and components are loaded when the application runs)
* **Execution timestamps**
* **File loading sequence** (the order in which the files were loaded during program execution)

The next time the application is launched, Windows can use this information to pre-load the necessary files into memory, reducing startup time.

**Location of Prefetch Files**

* **Path**: C:\Windows\Prefetch
* Prefetch files have a **.pf** file extension (e.g., EXPLORER.EXE-12345ABC.pf).

These files are created for each executable that is launched. The naming convention for these files typically includes the program's name and a unique identifier for the session (e.g., the hash of the executable file), followed by a .pf extension.

**Key Characteristics of Prefetch Files**

1. **Timestamp**:
   * Prefetch files store a timestamp of when the application was executed. This can be useful for investigators in determining when specific programs were run and constructing a timeline of system activity.
2. **Execution Count**:
   * The Prefetch file stores information on how many times the application has been executed. This helps Windows determine which applications are frequently used and prioritize them for optimization.
3. **Path and File Names**:
   * The path to the executable file, as well as paths to other files and libraries that were loaded with the program, are recorded. This can provide forensic evidence of what files were accessed during an application’s execution.
4. **File Sequence**:
   * The order in which files and components were loaded during the execution of the application is also recorded. This can be useful for understanding how an application operates or identifying additional files that may be involved.

**Forensic Relevance of Prefetch Files**

For digital forensics investigators, Prefetch files can provide valuable evidence of application usage, even after the program has been closed. Some of the forensic applications include:

1. **Evidence of Program Execution**:
   * Prefetch files can confirm that a specific program was executed, including the exact time and the number of times it was run. This is especially useful when trying to establish a timeline of events or activity on a system.
2. **Timestamps**:
   * The timestamps stored in Prefetch files can be used to reconstruct a timeline of user actions. Even if a user attempts to delete or conceal their actions (e.g., by clearing logs or removing shortcuts), the Prefetch files can still provide evidence of when a program was last run.
3. **Identification of Hidden Programs**:
   * Prefetch files can reveal programs that a user may have tried to conceal by deleting shortcuts or removing entries from the Start Menu. If an investigator finds that a program was run multiple times, it may indicate that the program was used regularly, even if the user took steps to hide it.
4. **File Access and Dependencies**:
   * The Prefetch file also records which files were accessed during the execution of the application. Investigators can analyze the associated files to identify other potentially relevant files or dependencies that were used during the execution of the program.
5. **Malware Analysis**:
   * In the case of malware investigations, Prefetch files can provide critical evidence of the execution of malicious programs. Malware often creates Prefetch files to speed up its execution, and these files may help investigators identify the malware's behavior, the files it interacted with, and its execution timeline.
6. **Deleted or Hidden Files**:
   * In cases where files are deleted but not overwritten (such as malware or illicit files), Prefetch files can provide a path to those files. Even if the files themselves are deleted from the system, Prefetch files may still point to their previous location.

**How to Analyze Prefetch Files in Forensics**

1. **Manual Inspection**:
   * **Event Logs** and **File Explorer** can be used to manually inspect the Prefetch folder (C:\Windows\Prefetch). However, this approach may not give a comprehensive understanding of the data without additional context.
2. **Forensic Tools**:
   * Specialized forensic tools can help parse Prefetch files and extract detailed information. Some of these tools include:
     + **RegRipper**: A widely used tool that extracts and analyzes Windows Prefetch files.
     + **X-Ways Forensics**: A powerful forensic tool that can extract and analyze Prefetch files as part of a full disk analysis.
     + **Autopsy**: An open-source digital forensics platform that includes modules to extract and interpret Prefetch data.
     + **FTK Imager**: A tool for imaging and analyzing disk files, including Prefetch files.
     + **Volatility**: A memory forensics tool that can help correlate Prefetch data with memory dumps to investigate system activity.
3. **Interpreting the Data**:
   * Prefetch files contain hexadecimal data and timestamps, which need to be decoded. Tools like **Prefetch Parser** or **Log2Timeline** are used to automate the process of extracting meaningful data from Prefetch files.
4. **Reconstructing Timelines**:
   * By correlating Prefetch data with other system artifacts (like event logs or file system metadata), investigators can construct a more accurate timeline of system activity and determine when specific programs were run.

**Limitations of Prefetch Files**

1. **Limited Data Retention**:
   * Windows does not store Prefetch data indefinitely. The files are typically overwritten after a certain number of uses or after a system cleanup. Additionally, if a user manually clears the Prefetch folder (though it requires administrative privileges), this data may be lost.
2. **Not All Programs Create Prefetch Files**:
   * Some programs or processes may not generate Prefetch files, especially if they are not commonly run or if they are portable (i.e., do not rely on the Windows registry to register execution).
3. **File Corruption**:
   * In some cases, Prefetch files may become corrupted or fail to record data correctly. This can lead to incomplete or inaccurate information during forensic analysis.
4. **Persistence**:
   * While Prefetch files provide evidence of execution, they do not provide persistent evidence once the files are cleared or overwritten, especially in systems that have undergone repeated usage or clean-up. **Shortcut Files in Windows (LNK Files)**

**SHORTCUT FILES:**

In Windows, **shortcut files** (with the .lnk extension) are used to provide quick access to applications, files, folders, or system resources without having to navigate to their actual location. These shortcuts are often placed on the desktop, taskbar, or Start Menu to make accessing programs or files more convenient.

* + **Key Characteristics:**
  + **File Extension**: .lnk
  + **Location**: Shortcuts are typically stored on the desktop, Start Menu, or within the user's personal folders.
  + **Metadata**: Shortcut files contain useful metadata, including:
  + **Target path**: The original location of the file or program.
  + **Creation/Modification Timestamps**: These can be used to trace when the shortcut was created or last accessed.
  + **Icon**: The graphical representation of the file or program the shortcut points to.
  + **Working directory**: The folder where the program was executed from.
  + **Arguments**: Any command-line arguments used to launch the target file.
  + **Forensic Relevance:**
  + **Evidence of Access**: Shortcut files provide evidence that a specific file, program, or folder was accessed by the user. Even if the target file was deleted, the shortcut file may remain.
  + **Timeline Construction**: The creation and access times of shortcut files can help investigators construct a timeline of user activity.
  + **Malware Investigation**: Malware often creates shortcuts to ensure its execution or disguise itself as legitimate software. Forensic analysis of shortcut files can help uncover malicious activity.

**WINDOWS EXECUTABLES**

In Windows, executable files are programs designed to be run by the system. These files usually have the .exe, .dll, .bat, or .msi extensions, but any file can potentially be an executable if it contains machine code that the system can interpret.

**Key Concepts:**

* **Executable File**: A file that contains instructions for the computer's processor to execute, such as .exe (for programs) or .dll (dynamic link libraries).
* **PE Format (Portable Executable)**: Most Windows executables are in the PE format, which contains headers, sections, and metadata like file timestamps, entry points, and imported functions.
* **Malware Executables**: Malicious executables often disguise themselves as legitimate programs or are delivered through phishing attacks, software vulnerabilities, or drive-by downloads.

**Forensic Focus:**

* **MD5/SHA Hashing**: To ensure integrity and verify whether an executable has been tampered with or if it matches known malware signatures.
* **Strings Analysis**: Extracting readable text from executables can reveal URLs, file paths, or commands, which are crucial for identifying the behavior of the executable.

**VOLATILE AND NON-VOLATILE INFORMATION**

In the context of digital forensics, data can be categorized as **volatile** or **non-volatile** based on how it is stored and its persistence over time.

**Volatile Information:**

* **Definition**: Data that exists temporarily in memory (RAM) or other volatile storage mediums.
* **Examples**:
  + **RAM (Random Access Memory)**: Stores active processes, network connections, passwords, encryption keys, etc.
  + **CPU Registers**: Contain temporary data used by running programs.
  + **Cache Data**: Temporary files or data stored by applications to speed up operations.
  + **Network Connections**: Active network connections can provide evidence of communication to external systems.
* **Forensic Relevance**: Volatile data is highly time-sensitive and may disappear when the system is turned off. Memory dumps, process lists, and network traffic can be crucial for live incident response and malware analysis.

**Non-Volatile Information:**

* **Definition**: Data that persists even when the system is powered off.
* **Examples**:
  + **Hard Drives**: Contain the file system, OS installation, and user data.
  + **File System Metadata**: File names, timestamps, and access control lists (ACLs).
  + **Registry Keys**: Store system configuration and user preferences.
* **Forensic Relevance**: Non-volatile data is more persistent and allows for post-incident forensic investigation, even after a system shutdown.

**WINDOWS MEMORY ANALYSIS**

Memory analysis involves examining the volatile memory (RAM) of a Windows system to uncover evidence of running processes, active network connections, or hidden malware.

**Key Concepts:**

* **Memory Dump**: A snapshot of the contents of a system's memory, which can be captured during an incident or investigation.
* **Tools**:
  + **Volatility Framework**: A popular open-source tool for analyzing memory dumps and extracting information like running processes, open network sockets, and loaded drivers.
  + **Redline**: A tool from FireEye that is often used for memory forensics.

**Forensic Focus:**

* **Running Processes**: Identify malicious processes running in memory, including rootkits or other hidden malware.
* **Network Connections**: Look for connections to suspicious IPs, malware C2 (command and control) servers, or untrusted remote systems.
* **Credentials**: Active credentials (like plaintext passwords or session tokens) may be found in memory.
* **Persistence Mechanisms**: Investigate how malware persists in memory, such as through DLL injection or process hollowing.

**EXECUTABLE FILE ANALYSIS**

Executable file analysis involves inspecting a file's content to identify its functionality and determine whether it is benign or malicious.

**Key Concepts:**

* **Static Analysis**: Involves examining the executable without running it. This includes:
  + **PE Header Inspection**: Analyze the headers for details about sections, imports, and exports.
  + **Strings Analysis**: Extract readable strings from the binary to identify command line arguments, URLs, or suspicious behavior.
  + **Hashing**: Use hashing techniques to verify whether the file matches known malware samples.
* **Dynamic Analysis**: Involves running the executable in a controlled environment (sandbox) to monitor its behavior, such as file creation, registry modifications, and network traffic.

**Tools:**

* **IDA Pro**: A disassembler and debugger used to perform static analysis.
* **Cuckoo Sandbox**: A popular tool for dynamic analysis, providing detailed reports on malware behavior.

**METADATA**

Metadata is data that describes other data, providing context for understanding files, documents, or system events.

**Key Concepts:**

* **File Metadata**: For example, file creation, modification, and access timestamps, author information, and file size.
* **Windows Metadata**: Includes information about file properties, system settings, and file system structures.
* **Forensic Relevance**: Metadata provides evidence of file origin, user activity, and file interactions, making it essential for timeline construction and identifying deleted or hidden files.

**Tools:**

* **Examine the File System**: Tools like **FTK Imager** or **X-Ways Forensics** can help extract and analyze file metadata from a system.
* **ExifTool**: A powerful tool for analyzing metadata in images, documents, and multimedia files.

**IIS LOGS (INTERNET INFORMATION SERVICES)**

IIS logs track the activities of web servers running Microsoft's Internet Information Services (IIS). These logs contain detailed information about web server requests, responses, and errors.

**Key Concepts:**

* **Log Data**: IIS logs record each request made to the server, including:
  + **IP Address**: The client’s IP address.
  + **Timestamp**: When the request was made.
  + **Request Type**: GET, POST, etc.
  + **Status Code**: HTTP status code (200 for success, 404 for file not found, etc.).
  + **User-Agent**: Information about the client’s browser and operating system.
  + **URL Requested**: The specific resource being accessed.

**Forensic Relevance:**

* **Identify Unauthorized Access**: Track failed login attempts or requests to unauthorized resources.
* **Malicious Activity**: Detect patterns of attack like brute force login attempts or SQL injection.
* **Incident Response**: Analyze logs to trace the source of a cyberattack or data breach.

**Tools:**

* **LogParser**: A command-line tool from Microsoft that helps to parse IIS log files.
* **ELK Stack**: Elasticsearch, Logstash, and Kibana are often used to parse and analyze IIS logs for larger-scale monitoring.

**PARSING DHCP SERVER LOGS**

The **DHCP (Dynamic Host Configuration Protocol)** server logs contain information about IP address allocation to devices on the network.

**Key Concepts:**

* **DHCP Logs**: Record when an IP address is assigned, the MAC address of the device, and the lease time.
* **Log Entries**:
  + **IP Address Assignment**: When an IP address is granted to a device.
  + **Lease Time**: Duration of the IP address allocation.
  + **Device Information**: The MAC address and host name of the device requesting the IP.

**Forensic Relevance:**

* **Track Device Activity**: Identify which devices were connected to the network at specific times.
* **Network Forensics**: Correlate DHCP logs with other network traffic logs to trace the source of malicious activity.

**Tools:**

* **Wireshark**: Can capture DHCP traffic in real-time for analysis.
* **Splunk**: For parsing and correlating DHCP log data with other network logs.

**PARSING WINDOWS FIREWALL LOGS**

Windows Firewall logs track incoming and outgoing network traffic that is blocked or allowed by the firewall.

**Key Concepts:**

* **Log Entries**: Includes information like:
  + **Source and Destination IP Address**: The originating and receiving IP addresses.
  + **Protocol**: The type of traffic (TCP, UDP, etc.).
  + **Action**: Whether the traffic was allowed or blocked.
  + **Port Numbers**: Information on the ports used by the communication.

**Forensic Relevance:**

* **Detect Suspicious Traffic**: Identify unauthorized access attempts, including potential intrusions or malware communication.
* **Track Malicious Behavior**: Correlate blocked traffic patterns with known attack signatures or unauthorized activity.

**Tools:**

* **Event Viewer**: A native Windows tool to view firewall log events.
* **Sysinternals Suite**: Tools like TCPView for monitoring active connections and network traffic.

**EVALUATING ACCOUNT MANAGEMENT EVENTS**

Account management events track changes to user accounts and group memberships in a Windows environment.

**Key Concepts:**

* **Event IDs**: Key event IDs to watch for include:
  + **4720**: Account Created.
  + **4722**: Account Enabled.
  + **4723**: Password Change Attempt.
  + **4725**: Account Disabled.

**Forensic Relevance:**

* **Track Unauthorized Access**: Detect when unauthorized accounts are created or when existing accounts are modified.
* **Privilege Escalation**: Identify when user accounts are granted elevated privileges or moved to high-level security groups.

**EXAMINING AUDIT POLICY CHANGE EVENTS**

Audit policies help track changes in the system’s auditing configuration, which defines what types of events should be logged.

**Key Concepts:**

* **Audit Policy Changes**: Key event IDs include:
  + **4719**: Audit policy change.
  + **4902**: Audit policy settings changed.

**Forensic Relevance:**

* **Investigate Potential Tampering**: Changes to audit policies could indicate that an attacker has disabled auditing to cover their tracks.

**SYSTEM LOG ENTRIES**

The **System log** in Windows records events related to the operating system and its components.

**Key Concepts:**

* **Event Types**: Information, Warnings, Errors, Critical.
* **Event IDs**: Logs critical system events like system crashes, driver failures, and service starts/stops.

**Forensic Relevance:**

* **Troubleshoot Failures**: Identify when and why critical system components failed.
* **Track System Compromise**: Identify anomalies or errors related to system compromise or hardware failures.

**APPLICATION LOG ENTRIES**

The **Application log** records events related to software applications and services running on the system.

**Key Concepts:**

* **Event Types**: Similar to System logs but focused on application behavior.
* **Application Errors**: Include application crashes, missing dependencies, and software configuration issues.

**Forensic Relevance:**

* **Correlate with User Activity**: Helps investigate what applications were in use and when issues occurred.