Rapid Triage Examination & Analysis Tools

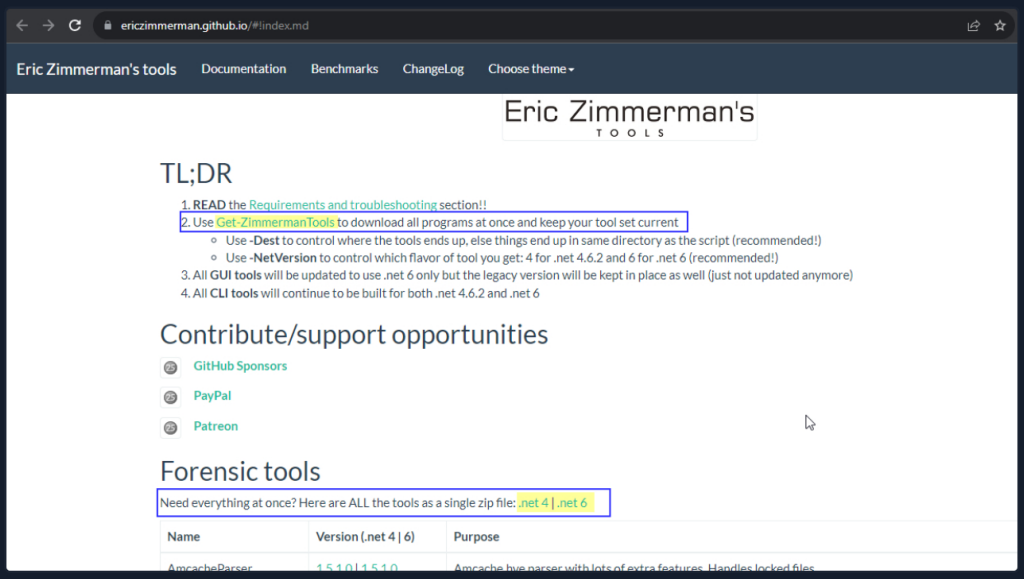
When it comes to Rapid Triage analysis, the right external tools are essential for thorough examination and analysis.

`Eric Zimmerman` has curated a suite of indispensable tools tailored for this very purpose. These tools are meticulously designed to aid forensic analysts in their quest to extract vital information from digital devices and artifacts.

For a comprehensive list of these tools, check out: https://ericzimmerman.github.io/#!index.md

> Let's now navigate to the bottom of this section and click on "Click here to spawn the target system!". Then, let's RDP into the Target IP using the provided credentials. The vast majority of the actions/commands covered from this point up to end of this section can be replicated inside the target, offering a more comprehensive grasp of the topics presented.

To streamline the download process, we can visit the official website and select either the .net 4 or .net 6 link. This action will initiate the download of all the tools in a compressed format.



We can also leverage the provided PowerShell script, as outlined in step 2 of the screenshot above, to download all the tools.

Rapid Triage Examination & Analysis Tools

```
PS C:\Users\johndoe\Desktop\Get-ZimmermanTools> .\Get-ZimmermanTools.ps1

This script will discover and download all available programs
from https://ericzimmerman.github.io and download them to C:\htb\dfir_module\tools

A file will also be created in C:\Users\johndoe\Desktop\Get-ZimmermanTools that tracks the SHA-1 of
so rerunning the script will only download new versions.

To redownload, remove lines from or delete the CSV file created under C:\htb\dfir_module\tools and

Use -NetVersion to control which version of the software you get (4 or 6). Default is 6. Use 0 to g

* Getting available programs...
* Files to download: 27
* Downloaded Get-ZimmermanTools.zip (Size: 10,396)
* C:\htb\dfir_module\tools\net6 does not exist. Creating...
* Downloaded AmcacheParser.zip (Size: 23,60,293) (net 6)
```

? Go to Questions

Table of Contents

Introduction to Digital Forensics
Windows Forensics Overview
Evidence Acquisition Techniques & Tools

Evidence Examination & Analysis

Memory Forensics
Disk Forensics
Rapid Triage Examination & Analysis Tools
Practical Digital Forensics Scenario

Skills Assessment

Skills Assessment

My Workstation

OFFLINE

Start Instance

∞ / 1 spawns left

```
* Downloaded AppCompatCacheParser.zip (Size: 22,62,497) (net 6)
* Downloaded bstrings.zip (Size: 14,73,298) (net 6)
* Downloaded EvtxECmd.zip (Size: 40,36,022) (net 6)
* Downloaded EZViewer.zip (Size: 8,25,80,608) (net 6)
* Downloaded JLECmd.zip (Size: 27,79,229) (net 6)
* Downloaded JumpListExplorer.zip (Size: 8,66,96,361) (net 6)
* Downloaded LECmd.zip (Size: 32,38,911) (net 6)
* Downloaded MFTECmd.zip (Size: 22,26,605) (net 6)
* Downloaded MFTExplorer.zip (Size: 8,27,54,162) (net 6)
* Downloaded PECmd.zip (Size: 20,13,672) (net 6)
* Downloaded RBCmd.zip (Size: 18,19,172) (net 6)
* Downloaded RecentFileCacheParser.zip (Size: 17,22,133) (net 6)
* Downloaded RECmd.zip (Size: 36,89,345) (net 6)
* Downloaded RegistryExplorer.zip (Size: 9,66,96,169) (net 6)
* Downloaded rla.zip (Size: 21,55,515) (net 6)
* Downloaded SDBExplorer.zip (Size: 8,24,54,727) (net 6)
* Downloaded SBECmd.zip (Size: 21,90,158) (net 6)
* Downloaded ShellBagsExplorer.zip (Size: 8,80,06,168) (net 6)
* Downloaded SQLECmd.zip (Size: 52,83,482) (net 6)
* Downloaded SrumECmd.zip (Size: 24,00,622) (net 6)
* Downloaded SumECmd.zip (Size: 20,23,009) (net 6)
* Downloaded TimelineExplorer.zip (Size: 8,77,50,507) (net 6)
* Downloaded VSCMount.zip (Size: 15,46,539) (net 6)
* Downloaded WxTCmd.zip (Size: 36,98,112) (net 6)
* Downloaded iisGeolocate.zip (Size: 3,66,76,319) (net 6)

* Saving downloaded version information to C:\Users\johndoe\Desktop\Get-ZimmermanTools\!!!RemoteFil
```

While we'll be utilizing a subset of these tools to analyze the KAPE output data, it's prudent for us to familiarize ourselves with the entire toolkit. Understanding the full capabilities of each tool can significantly enhance our investigative prowess.

In this section we will be working with certain tools and evidence that reside in the following directories of this section's target.

- **Evidence location**: `C:\Users\johndoe\Desktop\forensic_data`
    - **KAPE's output location**: `C:\Users\johndoe\Desktop\forensic_data\kape_output`
- **Eric Zimmerman's tools location**: `C:\Users\johndoe\Desktop\Get-ZimmermanTools`
- **Active@ Disk Editor's location**: `C:\Program Files\LSoft Technologies\Active@ Disk Editor`
- **EQL's location**: `C:\Users\johndoe\Desktop\eqllib-master`
- **RegRipper's location**: `C:\Users\johndoe\Desktop\RegRipper3.0-master`

MAC(b) Times in NTFS

The term `MAC(b) times` denotes a series of timestamps linked to files or objects. These timestamps are pivotal as they shed light on the chronology of events or actions on a file system. The acronym `MAC(b)` is an abbreviation for `Modified, Accessed, Changed, and (b) Birth` times. The inclusion of `b` signifies the `Birth timestamp`, which isn't universally present across all file systems or easily accessible via standard Windows API functions. Let's delve deeper into the nuances of `MACB` timestamps.

- `Modified Time (M)`: This timestamp captures the last instance when the content within the file underwent modifications. Any alterations to the file's data, such as content edits, trigger an update to this timestamp.
- `Accessed Time (A)`: This timestamp reflects the last occasion when the file was accessed or read, updating whenever the file is opened or otherwise engaged.
- `Changed [Change in MFT Record ] (C)`: This timestamp signifies changes to the MFT record. It captures the moment when the file was initially created. However, it's worth noting that certain file systems, like NTFS, might update this timestamp if the file undergoes movement or copying.
- `Birth Time (b)`: Often referred to as the Birth or Born timestamp, this represents the precise moment when the file or object was instantiated on the file system. Its significance in forensic investigations cannot be overstated, especially when determining a file's original creation time.

General Rules for Timestamps in the Windows NTFS File System

The table below delineates the general rules governing how various file operations influence the timestamps within the Windows NTFS (New Technology File System).

Operation	Modified	Accessed	Birth (Created)
File Create	Yes	Yes	Yes
File Modify	Yes	No	No
File Copy	No (Inherited)	Yes	Yes
File Access	No	No*	No

1. **File Create**:
    - `Modified Timestamp (M)`: The Modified timestamp is updated to reflect the time of file creation.
    - `Accessed Timestamp (A)`: The Accessed timestamp is updated to reflect that the file was accessed at the time of creation.
    - `Birth (Created) Timestamp (b)`: The Birth timestamp is set to the time of file creation.
2. **File Modify**:
    - `Modified Timestamp (M)`: The Modified timestamp is updated to reflect the time when the file's content or attributes were last modified.
    - `Accessed Timestamp (A)`: The Accessed timestamp is not updated when the file is modified.
    - `Birth (Created) Timestamp (b)`: The Birth timestamp is not updated when the file is modified.
3. **File Copy**:
    - `Modified Timestamp (M)`: The Modified timestamp is typically not updated when a file is copied. It usually inherits the timestamp from the source file.
    - `Accessed Timestamp (A)`: The Accessed timestamp is updated to reflect that the file was accessed at the time of copying.
    - `Birth (Created) Timestamp (b)`: The Birth timestamp is updated to the time of copying, indicating when the copy was created.
4. **File Access**:
    - `Modified Timestamp (M)`: The Modified timestamp is not updated when the file is accessed.
    - `Accessed Timestamp (A)`: The Accessed timestamp is updated to reflect the time of access.
    - `Birth (Created) Timestamp (b)`: The Birth timestamp is not updated when the file is accessed.

All these timestamps reside in the `$MFT` file, located at the root of the system drive. While the `$MFT` file will be covered in greater depth later, our current focus remains on understanding these timestamps.

These timestamps are housed within the `$MFT` across two distinct attributes:

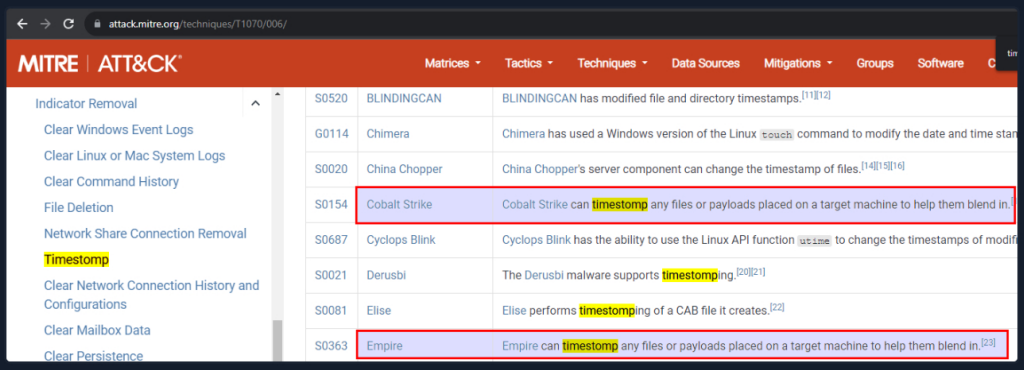
- `$STANDARD_INFORMATION`
- `$FILE_NAME`

The timestamps visible in the Windows file explorer are derived from the `$STANDARD_INFORMATION` attribute.

Timestomping Investigation

Identifying instances of timestamp manipulation, commonly termed as timestomping (T1070.006), presents a formidable challenge in digital forensics. Timestomping entails the alteration of file timestamps to obfuscate the

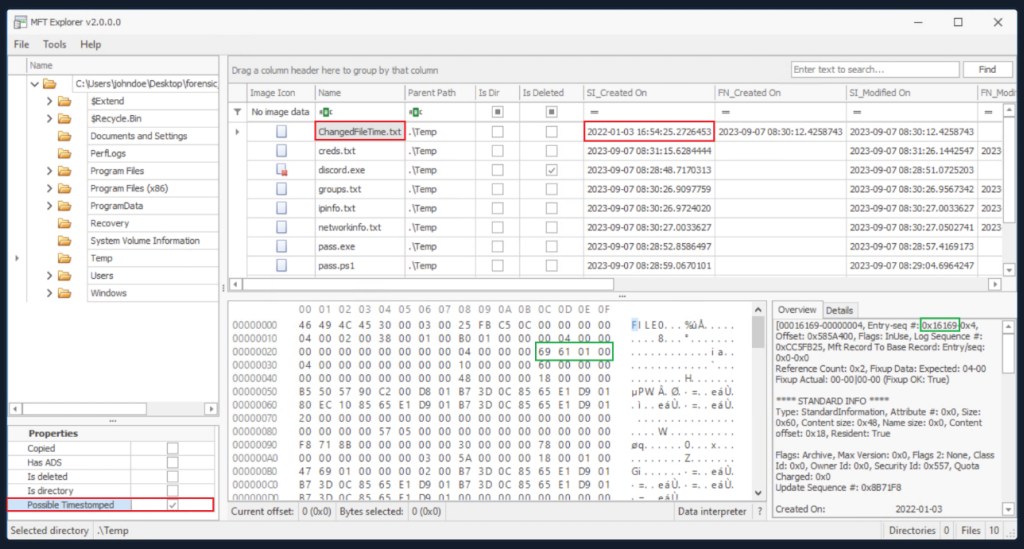
sequence of file activities. This tactic is frequently employed by various tools, as illustrated in the MITRE ATT&CK's timestomp technique.



When adversaries manipulate file creation times or deploy tools for such purposes, the timestamp displayed in the file explorer undergoes modification.

For instance, if we load `C:\Users\johndoe\Desktop\forensic_data\kape_output\D\$MFT` into `MFT Explorer` (available at `C:\Users\johndoe\Desktop\Get-ZimmermanTools\net6\MFTExplorer`) we will notice that the creation time of the file `ChangedFileTime.txt` has been tampered with, displaying `03-01-2022` in the file explorer, which deviates from the actual creation time.

Note: `MFT Explorer` will take 15-25 minutes to load the file.



However, given our knowledge that the timestamps in the file explorer originate from the `$STANDARD_INFORMATION` attribute, we can cross-verify this data with the timestamps from the `$FILE_NAME` attribute through `MFTEcmd` (available at `C:\Users\johndoe\Desktop\Get-ZimmermanTools\net6`) as follows.

Rapid Triage Examination & Analysis Tools

```
PS C:\Users\johndoe\Desktop\Get-ZimmermanTools\net6> .\MFTECmd.exe -f 'C:\Users\johndoe\Desktop\for
MFTECmd version 1.2.2.1

Author: Eric Zimmerman (saericzimmerman@gmail.com)
https://github.com/EricZimmerman/MFTECmd

Command line: -f C:\Users\johndoe\Desktop\forensic_data\kape_output\D\$MFT --de 0x16169

Warning: Administrator privileges not found!

File type: Mft

Processed C:\Users\johndoe\Desktop\forensic_data\kape_output\D\$MFT in 3.4924 seconds

C:\Users\johndoe\Desktop\forensic_data\kape_output\D\$MFT: FILE records found: 93,615 (Free records


Dumping details for file record with key 00016169-00000004
```

```
Entry-seq #: 0x16169-0x4, Offset: 0x585A400, Flags: InUse, Log seq #: 0xCC5FB25, Base Record entry-
Reference count: 0x2, FixUp Data Expected: 04-00, FixUp Data Actual: 00-00 | 00-00 (FixUp OK: True)

**** STANDARD INFO ****
  Attribute #: 0x0, Size: 0x60, Content size: 0x48, Name size: 0x0, ContentOffset 0x18. Resident: T
  Flags: Archive, Max Version: 0x0, Flags 2: None, Class Id: 0x0, Owner Id: 0x0, Security Id: 0x557

  Created On:         2022-01-03 16:54:25.2726453
  Modified On:        2023-09-07 08:30:12.4258743
  Record Modified On: 2023-09-07 08:30:12.4565632
  Last Accessed On:   2023-09-07 08:30:12.4258743

**** FILE NAME ****
  Attribute #: 0x3, Size: 0x78, Content size: 0x5A, Name size: 0x0, ContentOffset 0x18. Resident: T

  File name: CHANGE~1.TXT
  Flags: Archive, Name Type: Dos, Reparse Value: 0x0, Physical Size: 0x0, Logical Size: 0x0
  Parent Entry-seq #: 0x16947-0x2

  Created On:         2023-09-07 08:30:12.4258743
  Modified On:        2023-09-07 08:30:12.4258743
  Record Modified On: 2023-09-07 08:30:12.4258743
  Last Accessed On:   2023-09-07 08:30:12.4258743

**** FILE NAME ****
  Attribute #: 0x2, Size: 0x80, Content size: 0x68, Name size: 0x0, ContentOffset 0x18. Resident: T

  File name: ChangedFileTime.txt
  Flags: Archive, Name Type: Windows, Reparse Value: 0x0, Physical Size: 0x0, Logical Size: 0x0
  Parent Entry-seq #: 0x16947-0x2

  Created On:         2023-09-07 08:30:12.4258743
  Modified On:        2023-09-07 08:30:12.4258743
  Record Modified On: 2023-09-07 08:30:12.4258743
  Last Accessed On:   2023-09-07 08:30:12.4258743

**** DATA ****
  Attribute #: 0x1, Size: 0x18, Content size: 0x0, Name size: 0x0, ContentOffset 0x18. Resident: Tr

  Resident Data

  Data:

    ASCII:
    UNICODE:
```

```
**** STANDARD INFO ****
  Attribute #: 0x0, Size: 0x60, Content size: 0x48, Name size: 0x0, ContentOffset 0x18. Resident: True
  Flags: Archive, Max Version: 0x0, Flags 2: None, Class Id: 0x0, Owner Id: 0x0, Security Id: 0x557, Quota charged: 0x0,

  Created On:         2022-01-03 16:54:25.2726453
  Modified On:        2023-09-07 08:30:12.4258743
  Record Modified On: 2023-09-07 08:30:12.4565632
  Last Accessed On:   2023-09-07 08:30:12.4258743
```

Timestompping in $STANDARD_INFO

```
**** FILE NAME ****
  Attribute #: 0x2, Size: 0x80, Content size: 0x68, Name size: 0x0, ContentOffset 0x18. Resident: True

  File name: ChangedFileTime.txt
  Flags: Archive, Name Type: Windows, Reparse Value: 0x0, Physical Size: 0x0, Logical Size: 0x0
  Parent Entry-seq #: 0x16947-0x2

  Created On:         2023-09-07 08:30:12.4258743
  Modified On:        2023-09-07 08:30:12.4258743
  Record Modified On: 2023-09-07 08:30:12.4258743
  Last Accessed On:   2023-09-07 08:30:12.4258743

**** DATA ****
  Attribute #: 0x1, Size: 0x18, Content size: 0x0, Name size: 0x0, ContentOffset 0x18. Resident: True
```

Original creation time in $FILE_NAME

In standard Windows file systems like NTFS, regular users typically lack the permissions to directly modify the timestamps of filenames in `$FILE_NAME`. Such modifications are exclusively within the purview of the system kernel.

To kickstart our exploration, let's first acquaint ourselves with filesystem-based artifacts. We'll commence with the `$MFT` file, nestled in the root directory of the KAPE output.

MFT File

The `$MFT` file, commonly referred to as the Master File Table, is an integral part of the NTFS (New Technology File System) used by contemporary Windows operating systems. This file is instrumental in organizing and cataloging files and directories on an NTFS volume. Each file and directory on such a volume has a corresponding entry in the Master

File Table. Think of the MFT as a comprehensive database, meticulously documenting metadata and structural details about every file and directory.

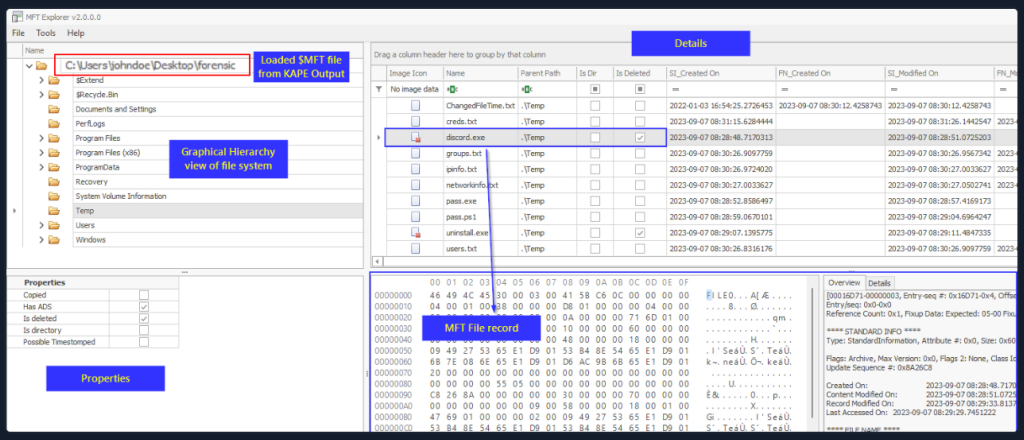
For those in the realm of digital forensics, the `$MFT` is a treasure trove of information. It offers a granular record of file and directory activities on the system, encompassing actions like file creation, modification, deletion, and access. By leveraging the `$MFT`, forensic analysts can piece together a detailed timeline of system events and user interactions.

Note: A standout feature of the MFT is its ability to retain metadata about files and directories, even post their deletion from the filesystem. This trait elevates the MFT's significance in forensic analysis and data recovery.

The MFT is strategically positioned at the root of the system drive.

We've already extracted the MFT while showcasing KAPE's capabilities and saved it at `C:\Users\johndoe\Desktop\forensic_data\kape_output\D\$MFT`.

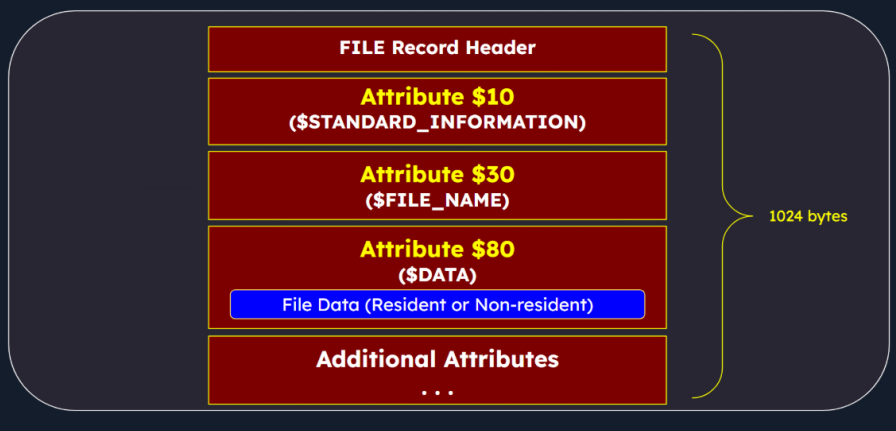
Let's navigate to the `$MFT` file within the KAPE Output directory above and load it in `MFT Explorer`. This tool, one of Eric Zimmerman's masterpieces, empowers us to inspect and analyze the metadata nestled in the MFT. This encompasses a wealth of information about files and directories, from filenames and timestamps (created, modified, accessed) to file sizes, permissions, and attributes. A standout feature of the MFT Explorer is its intuitive interface, presenting file records in a graphical hierarchy reminiscent of the familiar Windows Explorer.



Note: It's worth noting that MFT records, once created, aren't discarded. Instead, as new files and directories emerge, new records are added to the MFT. Records corresponding to deleted files are flagged as "free" and stand ready for reuse.

Structure of MFT File Record

Every file or directory on an NTFS volume is symbolized by a record in the MFT. These records adhere to a structured format, brimming with attributes and details about the associated file or directory. Grasping the MFT's structure is pivotal for tasks like forensic analysis, system management, and data recovery in Windows ecosystems. It equips forensic experts to pinpoint which attributes are brimming with intriguing insights.

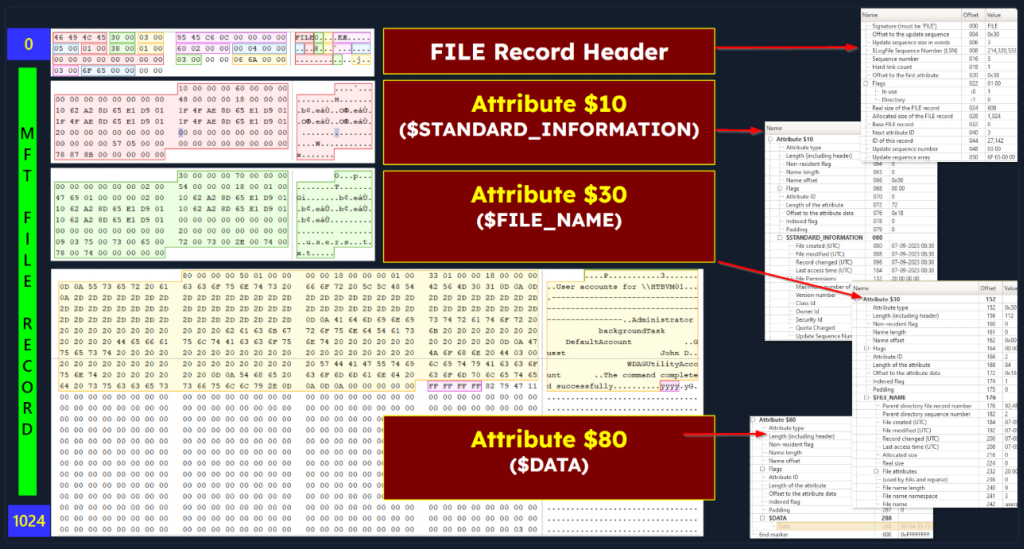


Here's a snapshot of the components:

Here's a snapshot of the components:

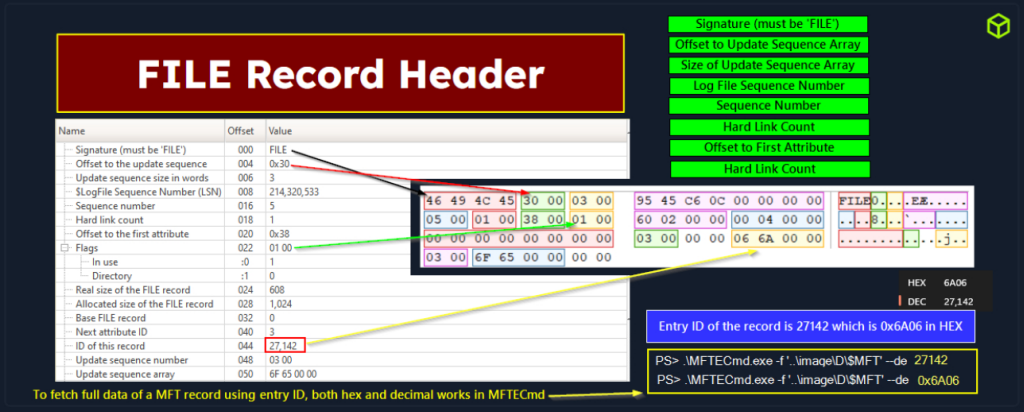
- `File Record Header`: Contains metadata about the file record itself. Includes fields like signature, sequence number, and other administrative data.
- `Standard Information Attribute Header`: Stores standard file metadata such as timestamps, file attributes, and security identifiers.
- `File Name Attribute Header`: Contains information about the filename, including its length, namespace, and Unicode characters.
- `Data Attribute Header`: Describes the file data attribute, which can be either `resident` (stored within the MFT record) or `non-resident` (stored in external clusters).
    - `File Data (File content)`: This section holds the actual file data, which can be the file's content or references to non-resident data clusters. For small files (less than 512 bytes), the data might be stored within the MFT record (`resident`). For larger files, it references `non-resident` data clusters on the disk. We'll see an example of this later on.
- `Additional Attributes (optional)`: NTFS supports various additional attributes, such as security descriptors (SD), object IDs (OID), volume name (VOLNAME), index information, and more.

These attributes can vary depending on the file's characteristics. We can see the common type of information which is stored inside these header and attributes in the image below.



File Record Header

Contains metadata about the file record itself. Includes fields like signature, sequence number, and other administrative data.



The file record begins with a header that contains metadata about the file record itself. This header typically includes the following information:

- `Signature`: A four-byte signature, usually "FILE" or "BAAD", indicating whether the record

- Signature: A four-byte signature, usually "FILE" or "BAAD," indicating whether the record is in use or has been deallocated.
- Offset to Update Sequence Array: An offset to the Update Sequence Array (USA) that helps maintain the integrity of the record during updates.
- Size of Update Sequence Array: The size of the Update Sequence Array in words.
- Log File Sequence Number: A number that identifies the last update to the file record.
- Sequence Number: A number identifying the file record. The MFT records are numbered sequentially, starting from 0.
- Hard Link Count: The number of hard links to the file. This indicates how many directory entries point to this file record.
- Offset to First Attribute: An offset to the first attribute in the file record.

When we sift through the MFT file using `MFTECmd` and extract details about a record, the information from the file record is presented as depicted in the subsequent screenshot.

Rapid Triage Examination & Analysis Tools

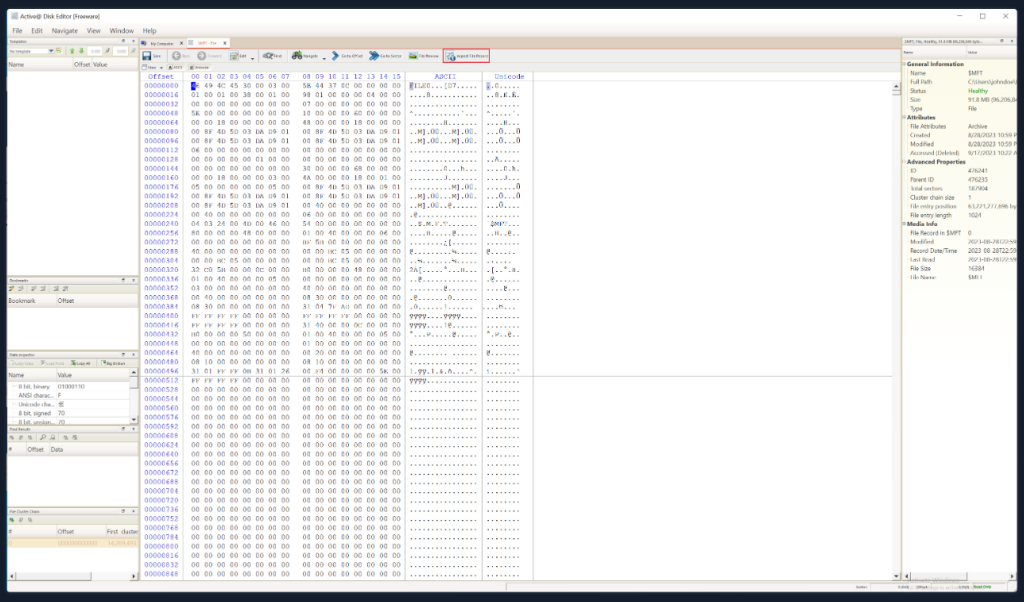
```
PS C:\Users\johndoe\Desktop\Get-ZimmermanTools\net6> .\MFTECmd.exe -f 'C:\Users\johndoe\Desktop\for
MFTECmd version 1.2.2.1

Author: Eric Zimmerman (saericzimmerman@gmail.com)
https://github.com/EricZimmerman/MFTECmd

Command line: -f C:\Users\johndoe\Desktop\forensic_data\kape_output\D\$MFT --de 27142

Warning: Administrator privileges not found!

File type: Mft

Processed C:\Users\johndoe\Desktop\forensic_data\kape_output\D\$MFT in 3.2444 seconds

C:\Users\johndoe\Desktop\forensic_data\kape_output\D\$MFT: FILE records found: 93,615 (Free records


Dumping details for file record with key 00006A06-00000005

Entry-seq #: 0x6A06-0x5, Offset: 0x1A81800, Flags: InUse, Log seq #: 0xCC64595, Base Record entry-s
Reference count: 0x1, FixUp Data Expected: 03-00, FixUp Data Actual: 6F-65 | 00-00 (FixUp OK: True)

**** STANDARD INFO ****
  Attribute #: 0x0, Size: 0x60, Content size: 0x48, Name size: 0x0, ContentOffset 0x18. Resident: T
  Flags: Archive, Max Version: 0x0, Flags 2: None, Class Id: 0x0, Owner Id: 0x0, Security Id: 0x557

  Created On:         2023-09-07 08:30:26.8316176
  Modified On:        2023-09-07 08:30:26.9097759
  Record Modified On: 2023-09-07 08:30:26.9097759
  Last Accessed On:   2023-09-07 08:30:26.9097759

**** FILE NAME ****
  Attribute #: 0x2, Size: 0x70, Content size: 0x54, Name size: 0x0, ContentOffset 0x18. Resident: T

  File name: users.txt
  Flags: Archive, Name Type: DosWindows, Reparse Value: 0x0, Physical Size: 0x0, Logical Size: 0x0
  Parent Entry-seq #: 0x16947-0x2

  Created On:         2023-09-07 08:30:26.8316176
  Modified On:        2023-09-07 08:30:26.8316176
  Record Modified On: 2023-09-07 08:30:26.8316176
  Last Accessed On:   2023-09-07 08:30:26.8316176

**** DATA ****
  Attribute #: 0x1, Size: 0x150, Content size: 0x133, Name size: 0x0, ContentOffset 0x18. Resident:

  Resident Data

  Data: 0D-0A-55-73-65-72-20-61-63-63-6F-75-6E-74-73-20-66-6F-72-20-5C-5C-48-54-42-56-4D-30-31-0D-0

    ASCII:
User accounts for \\HTBVM01

-------------------------------------------------------------------------------
Administrator            backgroundTask           DefaultAccount
Guest                    John Doe                 WDAGUtilityAccount
The command completed successfully.


    UNICODE: ??????????????????????????????????????????????????????????????????+++++????????+++++????
```

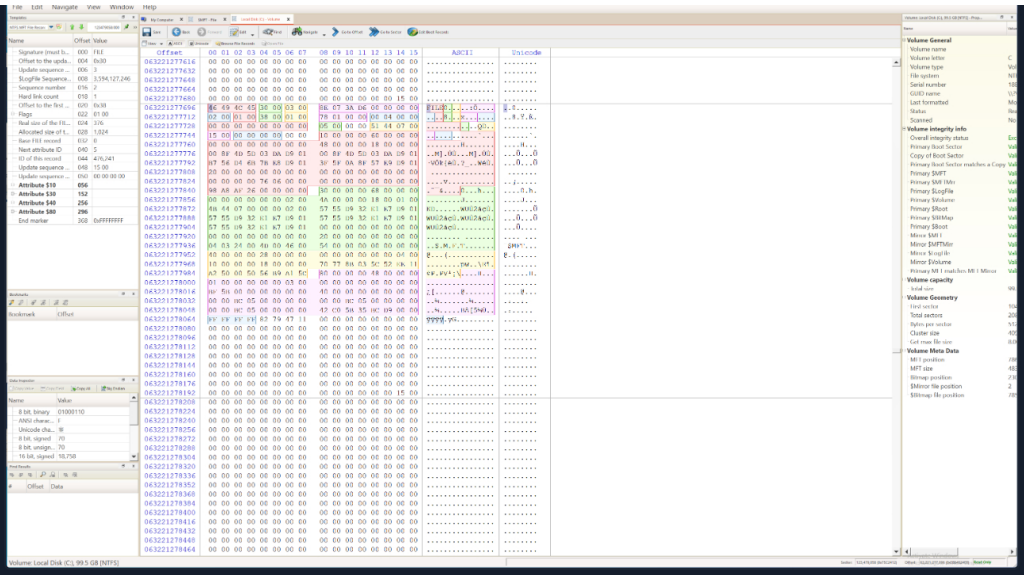
Each attribute signifies some entry information, identified by type.

Type	Attribute	Description
0x10 (16)	$STANDARD_INFORMATION	General information - flags, MAC times, owner, and security id.
0x20 (32)	$ATTRIBUTE_LIST	Pointers to other attributes and a list of nonresident attributes.
0x30 (48)	$FILE_NAME	File name - (Unicode) and outdated MAC times
0x40 (64)	$VOLUME_VERSION	Volume information - NTFS v1.2 only and Windows NT, no longer used
0x40 (64)	$OBJECT_ID	16B unique identifier - for file or directory (NTFS 3.0+; Windows 2000+)
0x50 (80)	$SECURITY_DESCRIPTOR	File's access control list and security properties
0x60 (96)	$VOLUME_NAME	Volume name
0x70 (112)	$VOLUME_INFORMATION	File system version and other information
0x80 (128)	$DATA	File contents
0x90 (144)	$INDEX_ROOT	Root node of an index tree
0xA0 (160)	$INDEX_ALLOCATION	Nodes of an index tree - with a root in $INDEX_ROOT
0xB0 (176)	$BITMAP	Bitmap - for the $MFT file and for indexes (directories)
0xC0 (192)	$SYMBOLIC_LINK	Soft link information - (NTFS v1.2 only and Windows NT)
0xC0 (192)	$REPARSE_POINT	Data about a reparse point - used for a soft link (NTFS 3.0+; Windows 2000+)
0xD0 (208)	$EA_INFORMATION	Used for backward compatibility with OS/2 applications (HPFS)
0xE0 (224)	$EA	Used for backward compatibility with OS/2 applications (HPFS)
0x100 (256)	$LOGGED_UTILITY_STREAM	Keys and other information about encrypted attributes (NTFS 3.0+; Windows 2000+)

To demystify the structure of an NTFS MFT file record, we're harnessing the capabilities of Active@ Disk Editor. This potent, freeware disk editing tool is available at `C:\Program Files\LSoft Technologies\Active@ Disk Editor` and facilitates the viewing and modification of raw disk data, including the Master File Table of an NTFS system. The same insights can be gleaned from other MFT parsing tools, such as `MFT Explorer`.

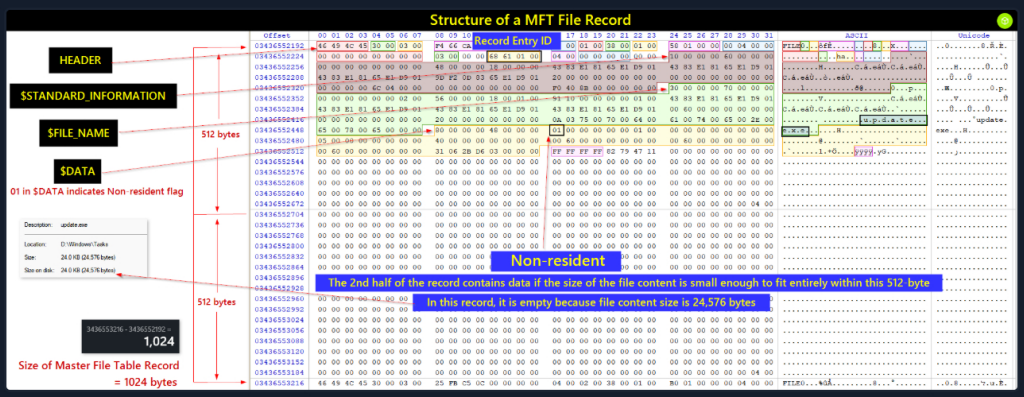
We can have a closer look by opening `C:\Users\johndoe\Desktop\forensic_data\kape_output\D\$MFT` on `Active@ Disk Editor` and then pressing `Inspect File Record`.



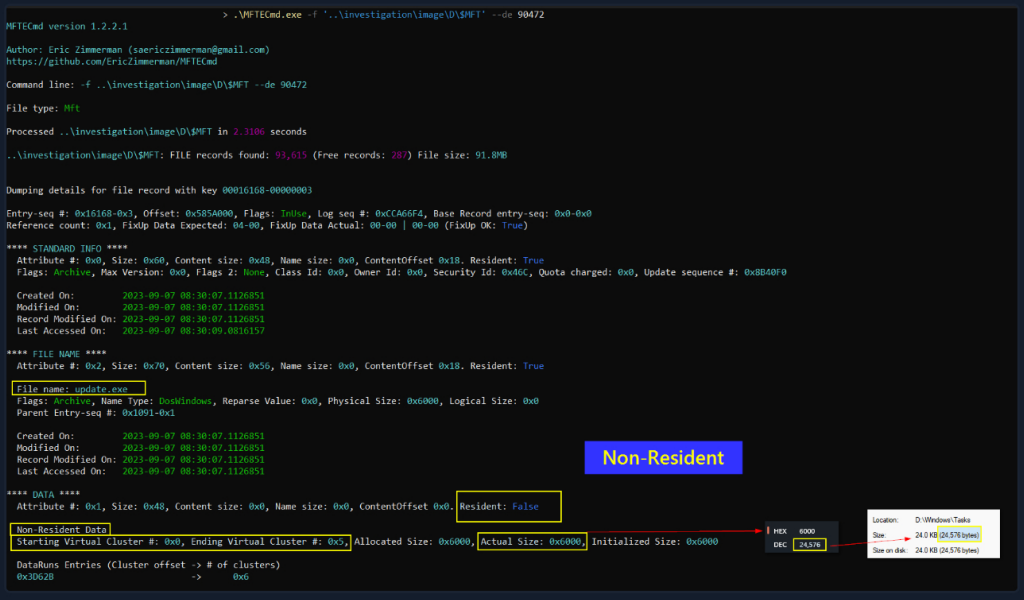


In Disk Editor, we're privy to the raw data of MFT entries. This includes a hexadecimal representation of the MFT record, complete with its header and attributes.

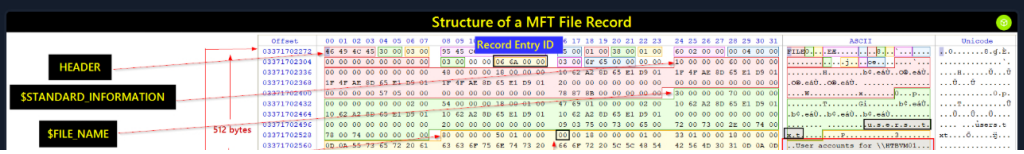
Non-Resident Flag

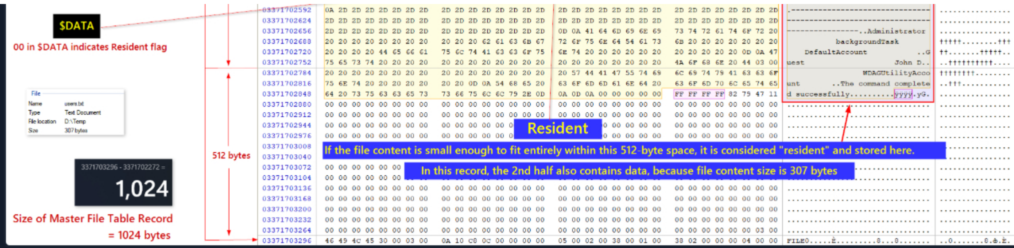


When parsing the entry in MFTECmd, this is how the non-resident data header appears.

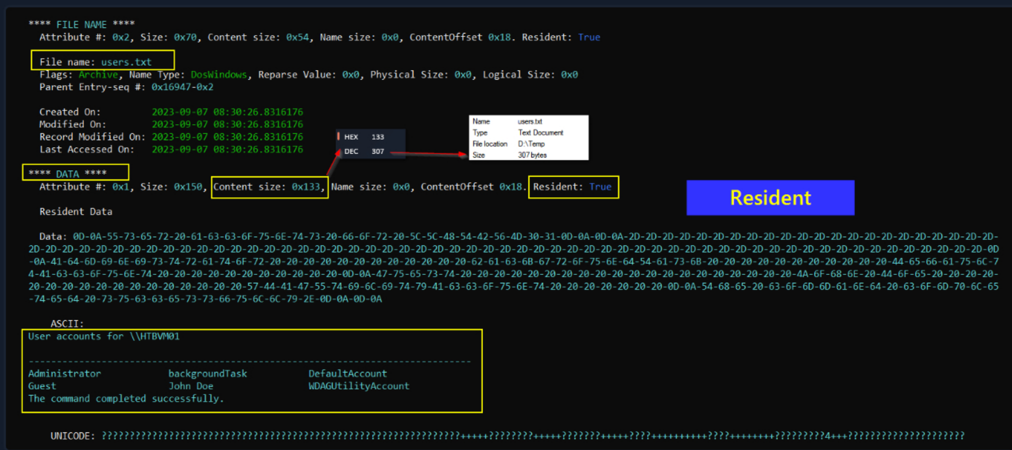


Resident Flag





When parsing the entry in `MFTECmd`, this is how the resident data header appears.



Zone.Identifier data in MFT File Record

The `Zone.Identifier` is a specialized file metadata attribute in the Windows OS, signifying the security zone from which a file was sourced. It's an integral part of the Windows Attachment Execution Service (AES) and is instrumental in determining how Windows processes files procured from the internet or other potentially untrusted origins.

When a file is fetched from the internet, Windows assigns it a Zone Identifier (`ZoneId`). This ZoneId, embedded in the file's metadata, signifies the source or security zone of the file's origin. For instance, internet-sourced files typically bear a `ZoneId` of `3`, denoting the Internet Zone.

For instance, we downloaded various tools inside the `C:\Users\johndoe\Downloads` directory of this section's target. Post-download, a `ZoneID` replete with the Zone.Identifier (i.e., the source URL) has been assigned to them.

Rapid Triage Examination & Analysis Tools

```
PS C:\Users\johndoe\Downloads> Get-Item * -Stream Zone.Identifier -ErrorAction SilentlyContinue


PSPath        : Microsoft.PowerShell.Core\FileSystem::C:\Users\johndoe\Downloads\Autoruns.zip:Zone.
PSParentPath  : Microsoft.PowerShell.Core\FileSystem::C:\Users\johndoe\Downloads
PSChildName   : Autoruns.zip:Zone.Identifier
PSDrive       : C
PSProvider    : Microsoft.PowerShell.Core\FileSystem
PSIsContainer : False
FileName      : C:\Users\johndoe\Downloads\Autoruns.zip
Stream        : Zone.Identifier
Length        : 130

PSPath        : Microsoft.PowerShell.Core\FileSystem::C:\Users\johndoe\Downloads\chainsaw_all_platf
                zip:Zone.Identifier
PSParentPath  : Microsoft.PowerShell.Core\FileSystem::C:\Users\johndoe\Downloads
PSChildName   : chainsaw_all_platforms+rules+examples.zip:Zone.Identifier
PSDrive       : C
PSProvider    : Microsoft.PowerShell.Core\FileSystem
PSIsContainer : False
FileName      : C:\Users\johndoe\Downloads\chainsaw_all_platforms+rules+examples.zip
Stream        : Zone.Identifier
Length        : 679

PSPath        : Microsoft.PowerShell.Core\FileSystem::C:\Users\johndoe\Downloads\disable-defender.p
PSParentPath  : Microsoft.PowerShell.Core\FileSystem::C:\Users\johndoe\Downloads
PSChildName   : disable-defender.ps1:Zone.Identifier
PSDrive       : C
PSProvider    : Microsoft.PowerShell.Core\FileSystem
PSIsContainer : False
FileName      : C:\Users\johndoe\Downloads\disable-defender.ps1
```

```
Stream         : Zone.Identifier
Length         : 55

PSPath         : Microsoft.PowerShell.Core\FileSystem::C:\Users\johndoe\Downloads\USN-Journal-Parser
                 ntifier
PSParentPath   : Microsoft.PowerShell.Core\FileSystem::C:\Users\johndoe\Downloads
PSChildName    : USN-Journal-Parser-master.zip:Zone.Identifier
PSDrive        : C
PSProvider     : Microsoft.PowerShell.Core\FileSystem
PSIsContainer  : False
FileName       : C:\Users\johndoe\Downloads\USN-Journal-Parser-master.zip
Stream         : Zone.Identifier
Length         : 187

PSPath         : Microsoft.PowerShell.Core\FileSystem::C:\Users\johndoe\Downloads\volatility3-develo
                 r
PSParentPath   : Microsoft.PowerShell.Core\FileSystem::C:\Users\johndoe\Downloads
PSChildName    : volatility3-develop.zip:Zone.Identifier
PSDrive        : C
PSProvider     : Microsoft.PowerShell.Core\FileSystem
PSIsContainer  : False
FileName       : C:\Users\johndoe\Downloads\volatility3-develop.zip
Stream         : Zone.Identifier
Length         : 184
```

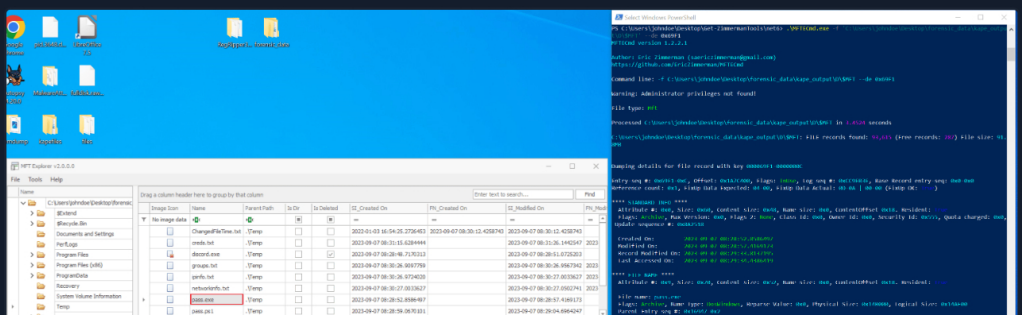
To unveil the content of a `Zone.Identifier` for a file, the following command can be executed in PowerShell.

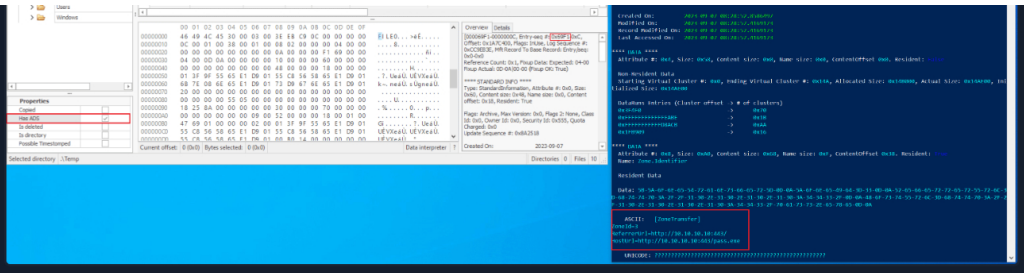
Rapid Triage Examination & Analysis Tools

```
PS C:\Users\johndoe\Downloads> Get-Content * -Stream Zone.Identifier -ErrorAction SilentlyContinue
[ZoneTransfer]
ZoneId=3
ReferrerUrl=https://learn.microsoft.com/
HostUrl=https://download.sysinternals.com/files/Autoruns.zip
[ZoneTransfer]
ZoneId=3
ReferrerUrl=https://github.com/WithSecureLabs/chainsaw/releases
HostUrl=https://objects.githubusercontent.com/github-production-release-asset-2e65be/395658506/222c
[ZoneTransfer]
ZoneId=3
HostUrl=https://github.com/
[ZoneTransfer]
ZoneId=3
ReferrerUrl=https://github.com/PoorBillionaire/USN-Journal-Parser
HostUrl=https://codeload.github.com/PoorBillionaire/USN-Journal-Parser/zip/refs/heads/master
[ZoneTransfer]
ZoneId=3
ReferrerUrl=https://github.com/volatilityfoundation/volatility3
HostUrl=https://codeload.github.com/volatilityfoundation/volatility3/zip/refs/heads/develop
```

One of the security mechanisms, known as the `Mark of the Web` (`MotW`), hinges on the Zone Identifier. Here, the MotW marker differentiates files sourced from the internet or other potentially dubious sources from those originating from trusted or local contexts. It's frequently employed to bolster the security of applications like Microsoft Word. When an app, say Microsoft Word, opens a file bearing a MotW, it can institute specific security measures based on the MotW's presence. For instance, a Word document with a MotW might be launched in `Protected View`, a restricted mode that isolates the document from the broader system, mitigating potential security threats.

While its primary function is to bolster security for files downloaded from the web, forensic analysts can harness it for investigative pursuits. By scrutinizing this attribute, they can ascertain the file's download method. See an example below.





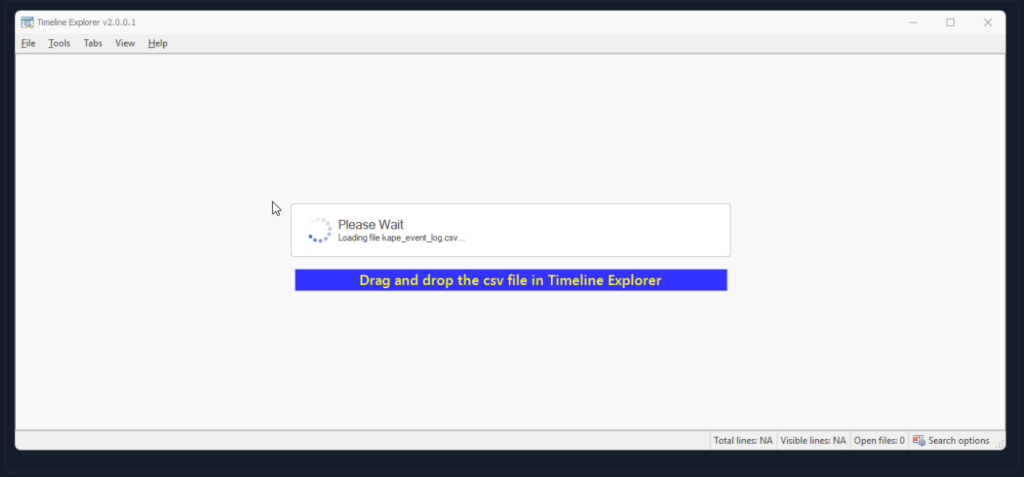
Analyzing with Timeline Explorer

`Timeline Explorer` is another digital forensic tool developed by Eric Zimmerman which is used to assist forensic analysts and investigators in creating and analyzing timeline artifacts from various sources. Timeline artifacts provide a chronological view of system events and activities, making it easier to reconstruct a sequence of events during an investigation. We can filter timeline data based on specific criteria, such as date and time ranges, event types, keywords, and more. This feature helps focus the investigation on relevant information.

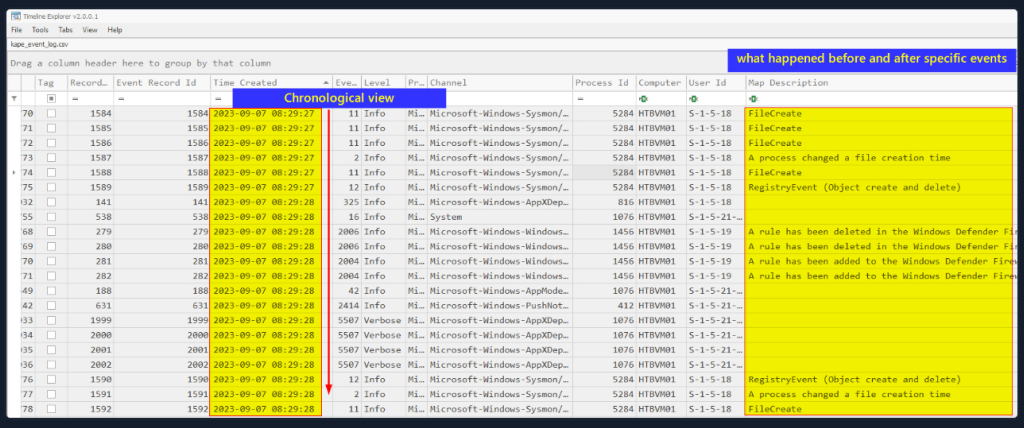
This arrangement of different events following one after another in time is really useful to create a story or timeline about what happened before and after specific events. This sequencing of events helps establish a timeline of activities on a system.

Loading a converted CSV file into Timeline Explorer is a straightforward process. Timeline Explorer is designed to work with timeline data, including CSV files that contain timestamped events or activities. To load the event data csv file into the Timeline Explorer, we can launch Timeline Explorer, and simply drag and drop from its location (e.g., our KAPE analysis directory) onto the Timeline Explorer window.

Once ingested, Timeline Explorer will process and display the data. The duration of this process hinges on the file's size.



We will see the timeline populated with the events from the CSV file in chronological order. With the timeline data now loaded, we can explore and analyze the events using the various features provided by Timeline Explorer. We can zoom in on specific time ranges, filter events, search for keywords, and correlate related activities.



We will provide multiple examples of using Timeline Explorer in this section.

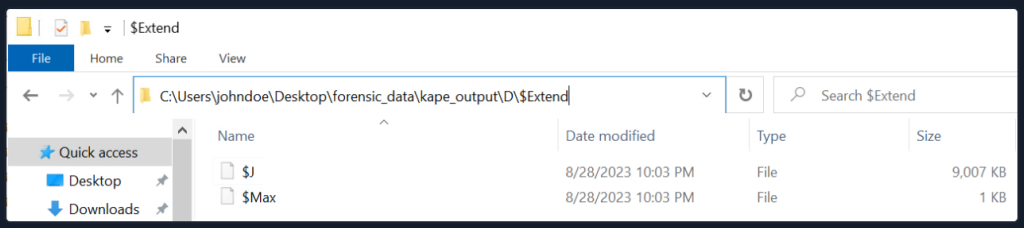
USN Journal

`USN`, or `Update Sequence Number`, is a vital component of the NTFS file system in Windows. The USN Journal is essentially a change journal feature that meticulously logs alterations to files and directories on an NTFS volume.

For those in digital forensics, the USN Journal is a goldmine. It enables us to monitor operations such as File Creation, Rename, Deletion, and Data Overwrite.

In the Windows environment, the USN Journal file is designated as `$J`. The KAPE Output directory houses the collected USN Journal in the following directory: `<KAPE_output_folder>\<Drive>\$Extend`

Here is how it looks in our KAPE's output (`C:\Users\johndoe\Desktop\forensic_data\kape_output\D\$Extend`)



Analyzing the USN Journal Using MFTECmd

We previously utilized `MFTECmd`, one of Eric Zimmerman's tools, to parse the MFT file. While its primary focus is the MFT, MFTECmd can also be instrumental in analyzing the USN Journal. This is because entries in the USN Journal often allude to modifications to files and directories that are documented in the MFT. Hence, we'll employ this tool to dissect the USN Journal.

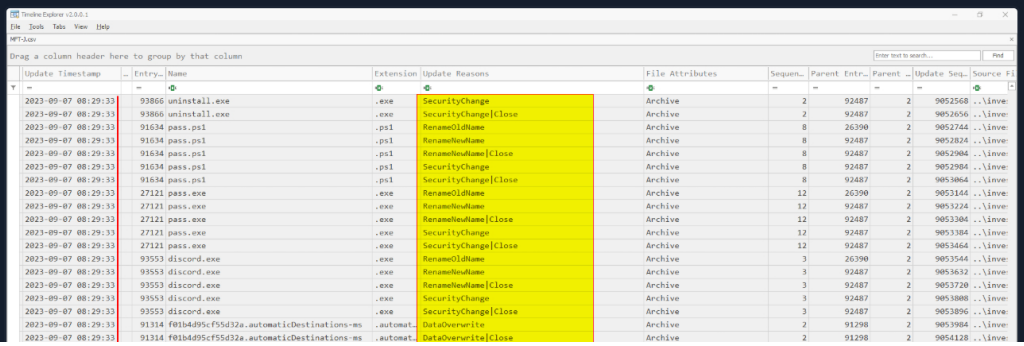
To facilitate the analysis of the USN Journal using `MFTECmd`, execute a command akin to the one below:

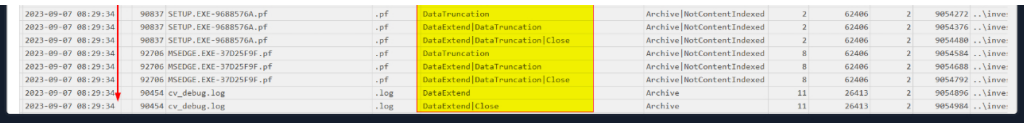
Rapid Triage Examination & Analysis Tools

```
PS C:\Users\johndoe\Desktop\Get-ZimmermanTools\net6> .\MFTECmd.exe -f 'C:\Users\johndoe\Desktop\for
MFTECmd version 1.2.2.1

Author: Eric Zimmerman (saericzimmerman@gmail.com)
https://github.com/EricZimmerman/MFTECmd

Command line: -f C:\Users\johndoe\Desktop\forensic_data\kape_output\D\$Extend\$J --csv C:\Users\joh

Warning: Administrator privileges not found!

File type: UsnJournal


Processed C:\Users\johndoe\Desktop\forensic_data\kape_output\D\$Extend\$J in 0.1675 seconds

Usn entries found in C:\Users\johndoe\Desktop\forensic_data\kape_output\D\$Extend\$J: 89,704
        CSV output will be saved to C:\Users\johndoe\Desktop\forensic_data\mft_analysis\MFT-J.csv
```

The resultant output file is saved as `MFT-J.csv` inside the `C:\Users\johndoe\Desktop\forensic_data\mft_analysis` directory. Let's import it into `Timeline Explorer` (available at `C:\Users\johndoe\Desktop\Get-ZimmermanTools\net6\TimelineExplorer`).

Note: Please remove the filter on the Entry Number to see the whole picture.





2023-09-07 08:29:34	90837	SETUP.EXE-9688576A.pf	.pf	DataTruncation	Archive|NotContentIndexed	2	62406	2	9054272	..\inve
2023-09-07 08:29:34	90837	SETUP.EXE-9688576A.pf	.pf	DataExtend|DataTruncation	Archive|NotContentIndexed	2	62406	2	9054376	..\inve
2023-09-07 08:29:34	90837	SETUP.EXE-9688576A.pf	.pf	DataExtend|DataTruncation|Close	Archive|NotContentIndexed	2	62406	2	9054480	..\inve
2023-09-07 08:29:34	92706	MSEDGE.EXE-37D25F9F.pf	.pf	DataTruncation	Archive|NotContentIndexed	8	62406	2	9054584	..\inve
2023-09-07 08:29:34	92706	MSEDGE.EXE-37D25F9F.pf	.pf	DataExtend|DataTruncation	Archive|NotContentIndexed	8	62406	2	9054688	..\inve
2023-09-07 08:29:34	92706	MSEDGE.EXE-37D25F9F.pf	.pf	DataExtend|DataTruncation|Close	Archive|NotContentIndexed	8	62406	2	9054792	..\inve
2023-09-07 08:29:34	90454	cv_debug.log	.log	DataExtend	Archive	11	26413	2	9054896	..\inve
2023-09-07 08:29:34	90454	cv_debug.log	.log	DataExtend|Close	Archive	11	26413	2	9054984	..\inve

Upon inspection, we can discern a chronologically ordered timeline of events. Notably, the entry for `uninstall.exe` is evident.

By applying a filter on the Entry Number `93866`, which corresponds to the `Entry ID` for `uninstall.exe`, we can glean the nature of modifications executed on this specific file.

Line	Tag	Update Timestamp	…	Entry Number	Name	Extension	Update Reasons
=	▣	=		= 93866			
85130	☐	2023-09-07 08:28:54		93866	49e2ab8a-255c-4ee6-80b2-36d14145934d.tmp	.tmp	FileCreate
85131	☐	2023-09-07 08:28:54		93866	49e2ab8a-255c-4ee6-80b2-36d14145934d.tmp	.tmp	FileCreate|Close
85132	☐	2023-09-07 08:28:54		93866	49e2ab8a-255c-4ee6-80b2-36d14145934d.tmp	.tmp	DataTruncation
85133	☐	2023-09-07 08:28:54		93866	49e2ab8a-255c-4ee6-80b2-36d14145934d.tmp	.tmp	DataTruncation|SecurityChange
85216	☐	2023-09-07 08:29:06		93866	49e2ab8a-255c-4ee6-80b2-36d14145934d.tmp	.tmp	DataTruncation|FileDelete|SecurityChange|Close
85220	☐	2023-09-07 08:29:07		93866	e9378fb6-8a55-4a59-9c4f-53529aa08799.tmp	.tmp	FileCreate
85221	☐	2023-09-07 08:29:07		93866	e9378fb6-8a55-4a59-9c4f-53529aa08799.tmp	.tmp	FileCreate|Close
85222	☐	2023-09-07 08:29:07		93866	e9378fb6-8a55-4a59-9c4f-53529aa08799.tmp	.tmp	DataExtend
85224	☐	2023-09-07 08:29:07		93866	e9378fb6-8a55-4a59-9c4f-53529aa08799.tmp	.tmp	DataExtend|Close
85225	☐	2023-09-07 08:29:07		93866	e9378fb6-8a55-4a59-9c4f-53529aa08799.tmp	.tmp	RenameOldName
85226	☐	2023-09-07 08:29:07		93866	Unconfirmed 407938.crdownload	.crdownload	RenameNewName
85227	☐	2023-09-07 08:29:07		93866	Unconfirmed 407938.crdownload	.crdownload	RenameNewName|Close
85294	☐	2023-09-07 08:29:10		93866	Unconfirmed 407938.crdownload	.crdownload	RenameOldName
85295	☐	2023-09-07 08:29:10		93866	uninstall.exe	.exe	RenameNewName
85296	☐	2023-09-07 08:29:10		93866	uninstall.exe	.exe	RenameNewName|Close
85297	☐	2023-09-07 08:29:11		93866	uninstall.exe	.exe	StreamChange
85298	☐	2023-09-07 08:29:11		93866	uninstall.exe	.exe	NamedDataExtend|StreamChange
85299	☐	2023-09-07 08:29:11		93866	uninstall.exe	.exe	NamedDataExtend|StreamChange|Close
85300	☐	2023-09-07 08:29:11		93866	uninstall.exe	.exe	NamedDataExtend

The file extension, `.crdownload`, is indicative of a partially downloaded file. This type of file is typically generated when downloading content via browsers like Microsoft Edge, Google Chrome, or Chromium. This revelation is intriguing. If the file was downloaded via a browser, it's plausible that the `Zone.Identifier` could unveil the source IP/domain of its origin.

To investigate this assumption we should:

1. Create a CSV file for `C:\Users\johndoe\Desktop\forensic_data\kape_output\D\$MFT` using `MFTECmd` as we did for `C:\Users\johndoe\Desktop\forensic_data\kape_output\D\$Extend\$J`.
2. Import the $MFT-related CSV into `Timeline Explorer`.
3. Apply a filter on the entry Number `93866`.

Rapid Triage Examination & Analysis Tools

```
PS C:\Users\johndoe\Desktop\Get-ZimmermanTools\net6> .\MFTECmd.exe -f 'C:\Users\johndoe\Desktop\for
MFTECmd version 1.2.2.1

Author: Eric Zimmerman (saericzimmerman@gmail.com)
https://github.com/EricZimmerman/MFTECmd

Command line: -f C:\Users\johndoe\Desktop\forensic_data\kape_output\D\$MFT --csv C:\Users\johndoe\D

Warning: Administrator privileges not found!

File type: Mft

Processed C:\Users\johndoe\Desktop\forensic_data\kape_output\D\$MFT in 3.5882 seconds

C:\Users\johndoe\Desktop\forensic_data\kape_output\D\$MFT: FILE records found: 93,615 (Free records
        CSV output will be saved to C:\Users\johndoe\Desktop\forensic_data\mft_analysis\MFT.csv
```

Timeline Explorer v2.0.0.1

File Tools Tabs View Help

MFT.csv

Drag a column header here to group by that column

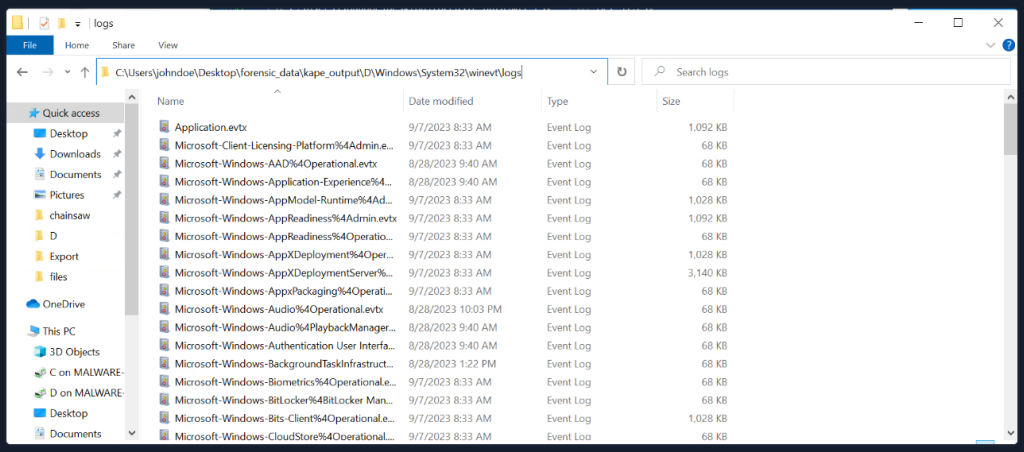
Enter text to search... Find

Entry Number	File Name	Zone Id Contents	Extension	Is Dir
= 93866				
93866	uninstall.exe		.exe	
93866	uninstall.exe:Zone.Identifier	[ZoneTransfer] ZoneId=3 ReferrerUrl=http://10.10.10.10:443/ HostUrl=http://10.10.10.10:443/uninstall.exe	.Identifier	

Windows Event Logs Investigation

Probing into Windows Event Logs is paramount in digital forensics and incident response. These logs are repositories of invaluable data, chronicling system activities, user behaviors, and security incidents on a Windows machine. When KAPE is executed, it duplicates the original event logs, ensuring their pristine state is preserved as evidence. The KAPE Output directory houses these event logs in the following directory:

`<KAPE_output_folder>\Windows\System32\winevt\logs`



This directory is populated with `.evtx` files, encapsulating a myriad of windows event logs, including but not limited to Security, Application, System, and Sysmon (if activated).

Our mission is to sift through these event logs, on the hunt for any anomalies, patterns, or indicators of compromise (IOCs). As forensic sleuths, we should be vigilant, paying heed to event IDs, timestamps, source IPs, usernames, and other pertinent log details. A plethora of forensic utilities and scripts, such as log parsing tools and SIEM systems, can bolster our analysis. It's imperative to identify the tactics, techniques, and procedures (TTPs) evident in any dubious activity. This might entail delving into known attack patterns and malware signatures. Another crucial step is to correlate events from diverse log sources, crafting a comprehensive timeline of events. This holistic view aids in piecing together the sequence of events.

The analysis of Windows Event Logs has been addressed in the modules titled `Windows Event Logs & Finding Evil` and `YARA & Sigma for SOC Analysts`.

Windows Event Logs Parsing Using EvtxECmd (EZ-Tool)

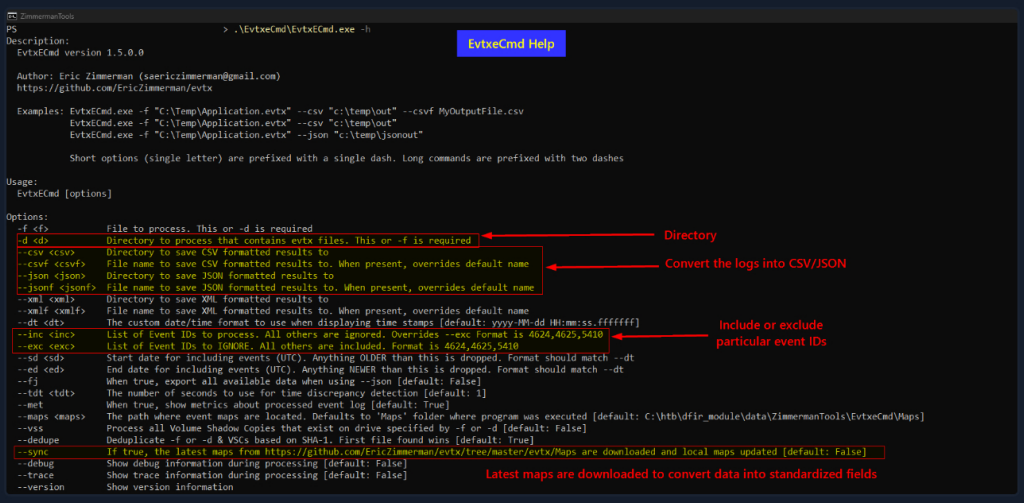
`EvtxECmd` (available at `C:\Users\johndoe\Desktop\Get-ZimmermanTools\net6\EvtxeCmd`) is another brainchild of Eric Zimmerman, tailored for Windows Event Log files (EVTX files). With this tool at our disposal, we can extract specific event logs or a range of events from an EVTX file, converting them into more digestible formats like JSON, XML, or CSV.

Let's initiate the help menu of EvtxECmd to familiarize ourselves with the various options. The command to access the help section is as follows.

Rapid Triage Examination & Analysis Tools

```
PS C:\Users\johndoe\Desktop\Get-ZimmermanTools\net6\EvtxeCmd> .\EvtxECmd.exe -h
Description:
  EvtxECmd version 1.5.0.0

  Author: Eric Zimmerman (saericzimmerman@gmail.com)
  https://github.com/EricZimmerman/evtx

  Examples: EvtxECmd.exe -f "C:\Temp\Application.evtx" --csv "c:\temp\out" --csvf MyOutputFile.csv
            EvtxECmd.exe -f "C:\Temp\Application.evtx" --csv "c:\temp\out"
            EvtxECmd.exe -f "C:\Temp\Application.evtx" --json "c:\temp\jsonout"

            Short options (single letter) are prefixed with a single dash. Long commands are prefix

Usage:
```

```
  EvtxECmd [options]

Options:
  -f <f>           File to process. This or -d is required
  -d <d>           Directory to process that contains evtx files. This or -f is required
  --csv <csv>      Directory to save CSV formatted results to
  --csvf <csvf>    File name to save CSV formatted results to. When present, overrides default name
  --json <json>    Directory to save JSON formatted results to
  --jsonf <jsonf>  File name to save JSON formatted results to. When present, overrides default nam
  --xml <xml>      Directory to save XML formatted results to
  --xmlf <xmlf>    File name to save XML formatted results to. When present, overrides default name
  --dt <dt>        The custom date/time format to use when displaying time stamps [default: yyyy-MM
  --inc <inc>      List of Event IDs to process. All others are ignored. Overrides --exc Format is
  --exc <exc>      List of Event IDs to IGNORE. All others are included. Format is 4624,4625,5410
  --sd <sd>        Start date for including events (UTC). Anything OLDER than this is dropped. Form
  --ed <ed>        End date for including events (UTC). Anything NEWER than this is dropped. Format
  --fj             When true, export all available data when using --json [default: False]
  --tdt <tdt>      The number of seconds to use for time discrepancy detection [default: 1]
  --met            When true, show metrics about processed event log [default: True]
  --maps <maps>    The path where event maps are located. Defaults to 'Maps' folder where program w
                   [default: C:\Users\johndoe\Desktop\Get-ZimmermanTools\net6\EvtxeCmd\Maps]
  --vss            Process all Volume Shadow Copies that exist on drive specified by -f or -d [defa
  --dedupe         Deduplicate -f or -d & VSCs based on SHA-1. First file found wins [default: True
  --sync           If true, the latest maps from https://github.com/EricZimmerman/evtx/tree/master/
                   downloaded and local maps updated [default: False]
  --debug          Show debug information during processing [default: False]
  --trace          Show trace information during processing [default: False]
  --version        Show version information
  -?, -h, --help   Show help and usage information
```



Maps in EvtxECmd

Maps in `EvtxECmd` are pivotal. They metamorphose customized data into standardized fields in the CSV (and JSON) data. This granularity and precision are indispensable in forensic investigations, enabling analysts to interpret and extract salient information from Windows Event Logs with finesse.

Standardized fields in maps:

- `UserName`: Contains information about user and/or domain found in various event logs
- `ExecutableInfo`: Contains information about process command line, scheduled tasks etc.
- `PayloadData1,2,3,4,5,6`: Additional fields to extract and put contextual data from event logs
- `RemoteHost`: Contains information about IP address

`EvtxECmd` plays a significant role in:

- Converting the unique part of an event, known as EventData, into a more standardized and human-readable format.
- Ensuring that the map files are tailored to specific event logs, such as Security, Application, or custom logs, to handle differences in event structures and data.
- Using a unique identifier, the Channel element, to specify which event log a particular map file is designed for, preventing confusion when event IDs are reused across different

logs.

To ensure the most recent maps are in place before converting the EVTX files to CSV/JSON, employ the command below.

Rapid Triage Examination & Analysis Tools

```
PS C:\Users\johndoe\Desktop\Get-ZimmermanTools\net6\EvtxeCmd> .\EvtxECmd.exe --sync
EvtxECmd version 1.5.0.0

Author: Eric Zimmerman (saericzimmerman@gmail.com)
https://github.com/EricZimmerman/evtx

Checking for updated maps at https://github.com/EricZimmerman/evtx/tree/master/evtx/Maps...

Updates found!

New maps
Application_ESENT_216
CiscoSecureEndpoint-Events_CiscoSecureEndpoint_100
CiscoSecureEndpoint-Events_CiscoSecureEndpoint_1300
CiscoSecureEndpoint-Events_CiscoSecureEndpoint_1310
Kaspersky-Security_OnDemandScan_3023
Kaspersky-Security_Real-Time_File_Protection_3023
Microsoft-Windows-Hyper-V-VMMS-Admin_Microsoft-Windows-Hyper-V-VMMS_13002
Microsoft-Windows-Hyper-V-VMMS-Admin_Microsoft-Windows-Hyper-V-VMMS_18304
Microsoft-Windows-Hyper-V-VMMS-Admin_Microsoft-Windows-Hyper-V-Worker_13003
Microsoft-Windows-Hyper-V-Worker-Admin_Microsoft-Windows-Hyper-V-Worker_18303
Microsoft-Windows-Hyper-V-Worker-Admin_Microsoft-Windows-Hyper-V-Worker_18504
Microsoft-Windows-Hyper-V-Worker-Admin_Microsoft-Windows-Hyper-V-Worker_18512
Microsoft-Windows-Windows-Defender-Operational_Microsoft-Windows-Windows-Defender_2050
PowerShellCore-Operational_PowerShellCore_4104
Security_Microsoft-Windows-Security-Auditing_6272
Security_Microsoft-Windows-Security-Auditing_6273

Updated maps
Microsoft-Windows-Hyper-V-Worker-Admin_Microsoft-Windows-Hyper-V-Worker_18500
Microsoft-Windows-Hyper-V-Worker-Admin_Microsoft-Windows-Hyper-V-Worker_18502
Microsoft-Windows-Hyper-V-Worker-Admin_Microsoft-Windows-Hyper-V-Worker_18508
Microsoft-Windows-Hyper-V-Worker-Admin_Microsoft-Windows-Hyper-V-Worker_18514
Microsoft-Windows-SMBServer-Security_Microsoft-Windows-SMBServer_551
Security_Microsoft-Windows-Security-Auditing_4616
```

With the latest maps integrated, we're equipped to infuse contextual information into distinct fields, streamlining the log analysis process. Now, it's time to transmute the logs into a format that's more palatable.

To render the EVTX files more accessible, we can employ `EvtxECmd` to seamlessly convert event log files into user-friendly formats like JSON or CSV.

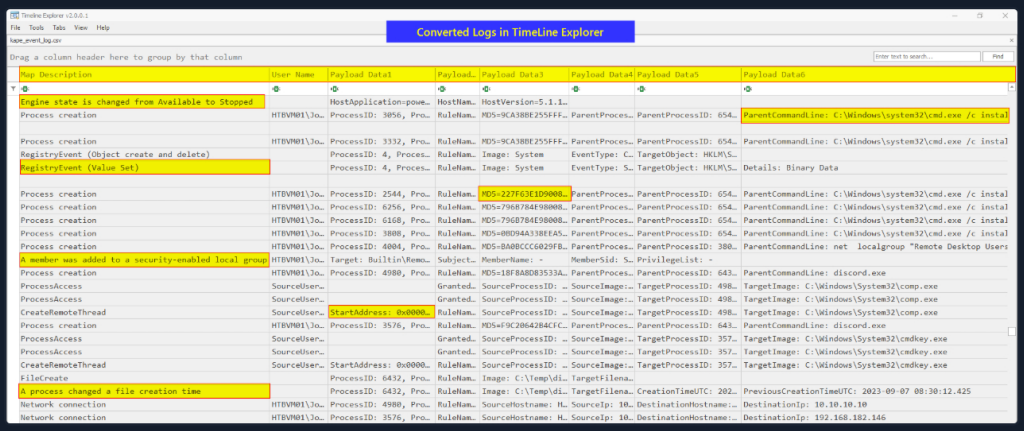
For instance, the command below facilitates the conversion of the `C:\Users\johndoe\Desktop\forensic_data\kape_output\D\Windows\System32\winevt\logs\Microsoft-Windows-Sysmon%4Operational.evtx` file to a CSV file:

Rapid Triage Examination & Analysis Tools

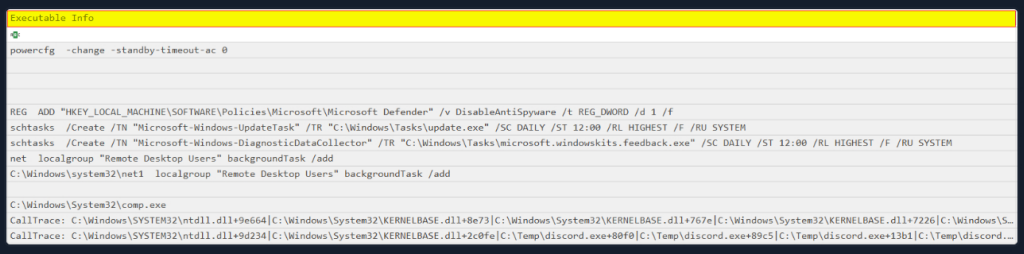
```
PS C:\Users\johndoe\Desktop\Get-ZimmermanTools\net6\EvtxeCmd> .\EvtxECmd.exe -f "C:\Users\johndoe\D
EvtxECmd version 1.5.0.0

Author: Eric Zimmerman (saericzimmerman@gmail.com)
https://github.com/EricZimmerman/evtx

Command line: -f C:\Users\johndoe\Desktop\forensic_data\kape_output\D\Windows\System32\winevt\logs\

Warning: Administrator privileges not found!

CSV output will be saved to C:\Users\johndoe\Desktop\forensic_data\event_logs\csv_timeline\kape_eve

Processing C:\Users\johndoe\Desktop\forensic_data\kape_output\D\Windows\System32\winevt\logs\Micros

Event log details
Flags: None
Chunk count: 28
Stored/Calculated CRC: 3EF9F1C/3EF9F1C
Earliest timestamp: 2023-09-07 08:23:18.4430130
Latest timestamp:   2023-09-07 08:33:00.0069805
Total event log records found: 1,820
```

```
Records included: 1,920 Errors: 0 Events dropped: 0

Metrics (including dropped events)
Event ID        Count
1               95
2               76
3               346
4               1
8               44
10              6
11              321
12              674
13              356
16              1

Processed 1 file in 8.7664 seconds
```

After importing the resultant CSV into `Timeline Explorer`, we should see the below.



Executable Information:



Investigating Windows Event Logs with EQL

Endgame's Event Query Language (EQL) is an indispensable tool for sifting through event logs, pinpointing potential security threats, and uncovering suspicious activities on Windows systems. EQL offers a structured language that facilitates querying and correlating events across multiple log sources, including the Windows Event Logs.

Currently, the EQL module is compatible with Python versions 2.7 and 3.5+. If you have a supported Python version installed, execute the following command.

Rapid Triage Examination & Analysis Tools

```
C:\Users\johndoe>pip install eql
```

Should Python be properly configured and included in your PATH, eql should be accessible. To verify this, execute the command below.

Rapid Triage Examination & Analysis Tools

```
C:\Users\johndoe>eql --version
eql 0.9.18
```

Within EQL's repository (available at `C:\Users\johndoe\Desktop\eqllib-master`), there's a PowerShell module brimming with essential functions tailored for parsing Sysmon events from Windows Event Logs. This module resides in the `utils` directory of `eqllib`, and is named `scrape-events.ps1`.

From the EQL directory, initiate the scrape-events.ps1 module with the following command:

Rapid Triage Examination & Analysis Tools

```
PS C:\Users\johndoe\Desktop\eqllib-master\utils> import-module .\scrape-events.ps1
```

By doing so, we activate the `Get-EventProps` function, which is instrumental in parsing event properties from Sysmon logs. To transform, for example, `C:\Users\johndoe\Desktop\forensic_data\kape_output\D\Windows\System32\winevt\logs\Microsoft-Windows-Sysmon%4Operational.evtx` into a JSON format suitable for EQL queries, execute the command below.

Rapid Triage Examination & Analysis Tools

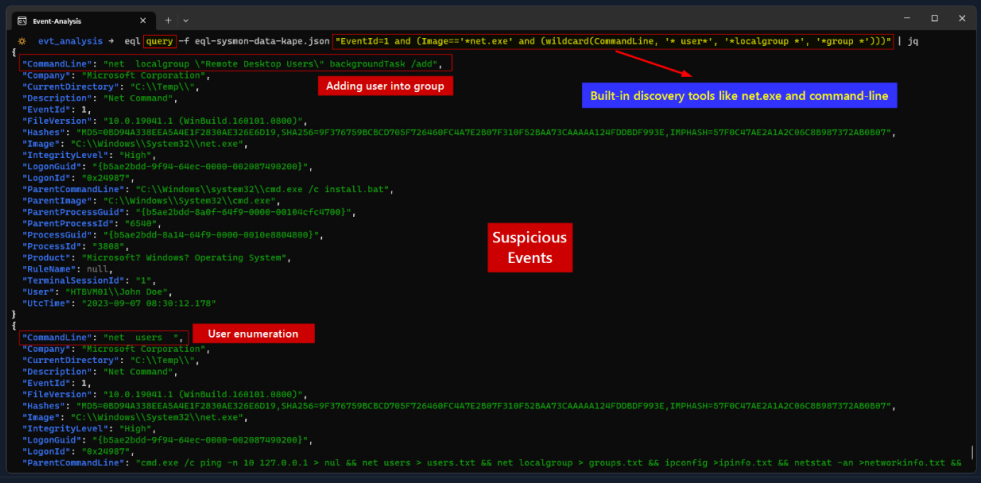
```
PS C:\Users\johndoe\Desktop\eqllib-master\utils> Get-WinEvent -Path C:\Users\johndoe\Desktop\forens
```

This action will yield a JSON file, primed for EQL queries.

Let's now see how we could have identified user/group enumeration through an EQL query against the JSON file we created.

Rapid Triage Examination & Analysis Tools

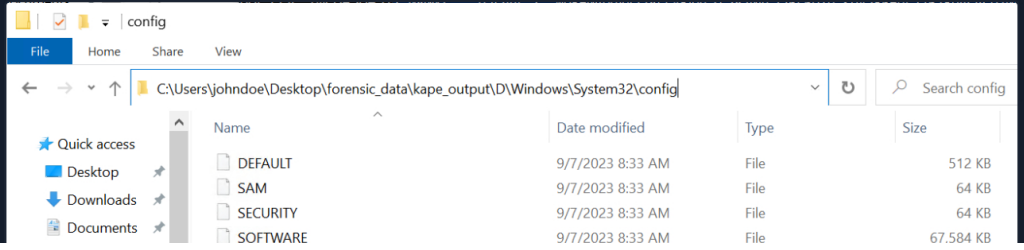
```
C:\Users\johndoe>eql query -f C:\Users\johndoe\Desktop\forensic_data\event_logs\eql_format_json\eql
{"CommandLine": "net  localgroup \"Remote Desktop Users\" backgroundTask /add", "Company": "Microso
{"CommandLine": "net  users  ", "Company": "Microsoft Corporation", "CurrentDirectory": "C:\\Temp\\
{"CommandLine": "net  localgroup  ", "Company": "Microsoft Corporation", "CurrentDirectory": "C:\\T
```



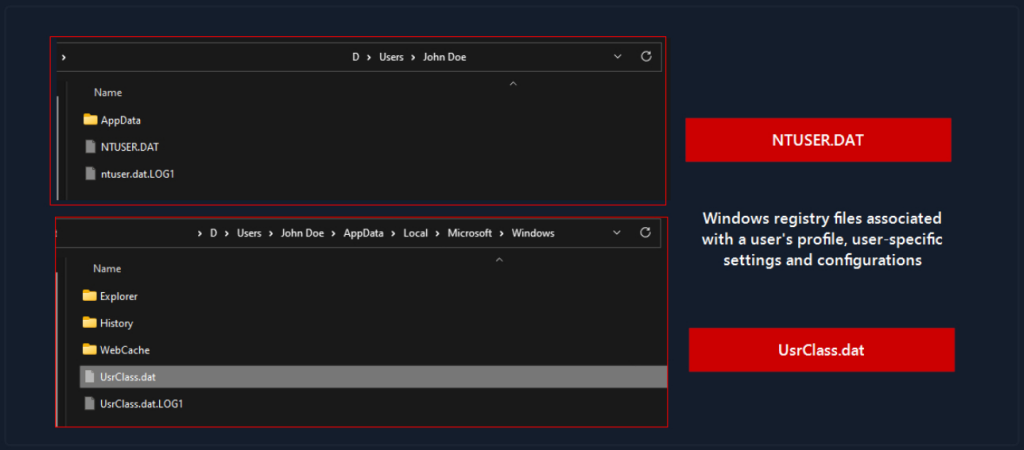
Windows Registry

A deep dive into the registry hives can furnish us with invaluable insights, such as the computer's name, Windows version, owner's name, and network configuration.

Registry-related files harvested from KAPE are typically housed in `<KAPE_output_folder>\Windows\System32\config`

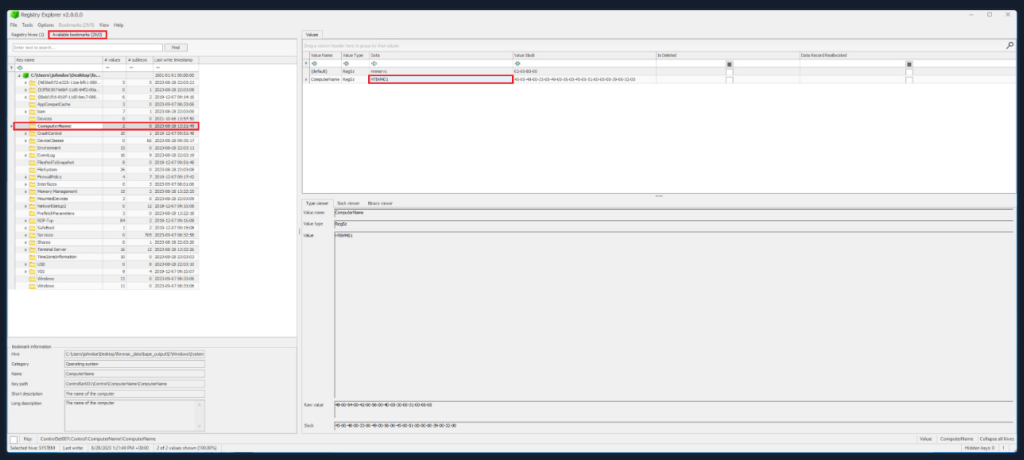


Additionally, there are user-specific registry hives located within individual user directories, as exemplified in the following screenshot.



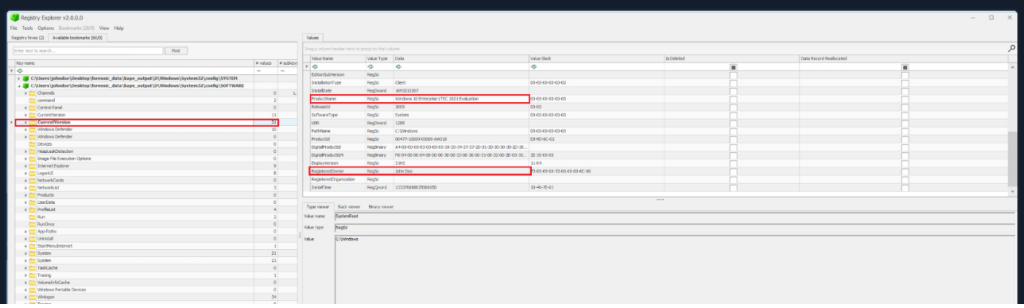
For a comprehensive analysis, we can employ `Registry Explorer` (available at `C:\Users\johndoe\Desktop\Get-ZimmermanTools\net6\RegistryExplorer`) a GUI-based tool masterminded by Eric Zimmerman. This tool offers a streamlined interface to navigate and dissect the contents of Windows Registry hives. By simply dragging and dropping these files into Registry Explorer, the tool processes the data, presenting it within its GUI. The left panel displays the registry hives, while the right panel reveals their corresponding values.

In the screenshot below we have loaded the `SYSTEM` hive, that can be found inside the `C:\Users\johndoe\Desktop\forensic_data\kape_output\D\Windows\System32\config` directory of this section's target.



Registry Explorer boasts a suite of features, including hive analysis, search capabilities, filtering options, timestamp viewing, and bookmarking. The bookmarking utility is particularly handy, allowing users to earmark pivotal locations or keys for subsequent reference.

In the screenshot below we have loaded the `SOFTWARE` hive, that can be found inside the `C:\Users\johndoe\Desktop\forensic_data\kape_output\D\Windows\System32\config` directory of this section's target. Notice the available bookmarks within Registry Explorer.



RegRipper

Another potent tool in our arsenal is `RegRipper` (available at `C:\Users\johndoe\Desktop\RegRipper3.0-master`), a command-line utility adept at swiftly extracting information from the Registry.

To acquaint ourselves with RegRipper's functionalities, let's invoke the help section by executing `rip.exe` accompanied by the `-h` parameter.

Rapid Triage Examination & Analysis Tools

```
PS C:\Users\johndoe\Desktop\RegRipper3.0-master> .\rip.exe -h
Rip v.3.0 - CLI RegRipper tool
Rip [-r Reg hive file] [-f profile] [-p plugin] [options]
Parse Windows Registry files, using either a single module, or a profile.

NOTE: This tool does NOT automatically process Registry transaction logs! The tool
does check to see if the hive is dirty, but does not automatically process the
transaction logs.  If you need to incorporate transaction logs, please consider
using yarp + registryFlush.py, or rla.exe from Eric Zimmerman.

  -r [hive] .........Registry hive file to parse
  -d ................Check to see if the hive is dirty
  -g ................Guess the hive file type
  -a ................Automatically run hive-specific plugins
  -aT ...............Automatically run hive-specific TLN plugins
  -f [profile].......use the profile
  -p [plugin]........use the plugin
  -l ................list all plugins
  -c ................Output plugin list in CSV format (use with -l)
  -s systemname......system name (TLN support)
  -u username........User name (TLN support)
  -uP ...............Update default profiles
  -h.................Help (print this information)

Ex: C:\>rip -r c:\case\system -f system
    C:\>rip -r c:\case\ntuser.dat -p userassist
    C:\>rip -r c:\case\ntuser.dat -a
    C:\>rip -l -c

All output goes to STDOUT; use redirection (ie, > or >>) to output to a file.

copyright 2020 Quantum Analytics Research, LLC
```

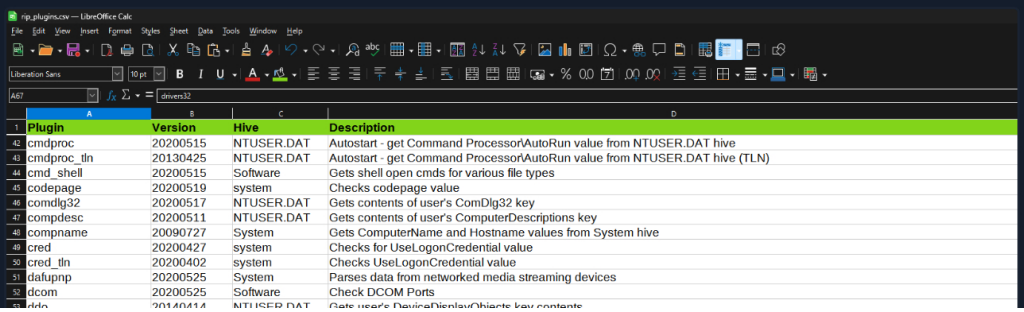
For a seamless experience with RegRipper, it's essential to familiarize ourselves with its plugins. To enumerate all available plugins and catalog them in a CSV file (e.g., `rip_plugins.csv`), use the command below.

Rapid Triage Examination & Analysis Tools

```
PS C:\Users\johndoe\Desktop\RegRipper3.0-master> .\rip.exe -l -c > rip_plugins.csv
```

This action compiles a comprehensive list of plugins, detailing the associated hives, and saves it as a CSV file.

The screenshot below elucidates the contents of this file, highlighting the plugin name, its corresponding registry hive, and a brief description.



	Plugin	Version	Hive	Description
42	cmdproc	20200515	NTUSER.DAT	Autostart - get Command Processor\AutoRun value from NTUSER.DAT hive
43	cmdproc_tln	20130425	NTUSER.DAT	Autostart - get Command Processor\AutoRun value from NTUSER.DAT hive (TLN)
44	cmd_shell	20200515	Software	Gets shell open cmds for various file types
45	codepage	20200519	system	Checks codepage value
46	comdlg32	20200517	NTUSER.DAT	Gets contents of user's ComDlg32 key
47	compdesc	20200511	NTUSER.DAT	Gets contents of user's ComputerDescriptions key
48	compname	20090727	System	Gets ComputerName and Hostname values from System hive
49	cred	20200427	system	Checks for UseLogonCredential value
50	cred_tln	20200402	system	Checks UseLogonCredential value
51	dafupnp	20200525	System	Parses data from networked media streaming devices
52	dcom	20200525	Software	Check DCOM Ports
53	ddo	20140414	NTUSER.DAT	Gets user's DeviceDisplayObjects key contents

54 ddpe	20221128	Software	Get the Machine ID (MCID) and Shield ID (DCID) needed to decrypt files using Dell Encryption. Also reports the Dell Server URI.
55 defender	20200427	Software	Get Windows Defender settings

To kick things off, let's execute the `compname` command on the SYSTEM hive (located at `C:\Users\johndoe\Desktop\forensic_data\kape_output\D\Windows\System32\config`), which retrieves the computer's name.

Rapid Triage Examination & Analysis Tools

```
PS C:\Users\johndoe\Desktop\RegRipper3.0-master> .\rip.exe -r "C:\Users\johndoe\Desktop\forensic_da
Launching compname v.20090727
compname v.20090727
(System) Gets ComputerName and Hostname values from System hive

ComputerName    = HTBVM01
TCP/IP Hostname = HTBVM01
```

Let's see some more examples against different hives.

Timezone

Rapid Triage Examination & Analysis Tools

```
PS C:\Users\johndoe\Desktop\RegRipper3.0-master> .\rip.exe -r "C:\Users\johndoe\Desktop\forensic_da
Launching timezone v.20200518
timezone v.20200518
(System) Get TimeZoneInformation key contents

TimeZoneInformation key
ControlSet001\Control\TimeZoneInformation
LastWrite Time 2023-08-28 23:03:03Z
  DaylightName   -> @tzres.dll,-211
  StandardName   -> @tzres.dll,-212
  Bias           -> 480 (8 hours)
  ActiveTimeBias -> 420 (7 hours)
  TimeZoneKeyName-> Pacific Standard Time
```

Network Information

Rapid Triage Examination & Analysis Tools

```
PS C:\Users\johndoe\Desktop\RegRipper3.0-master> .\rip.exe -r "C:\Users\johndoe\Desktop\forensic_da
Launching nic2 v.20200525
nic2 v.20200525
(System) Gets NIC info from System hive

Adapter: {50c7b4ab-b059-43f4-8b0f-919502abc934}
LastWrite Time: 2023-09-07 08:01:06Z
  EnableDHCP                    0
  Domain
  NameServer                    10.10.10.100
  DhcpServer                    255.255.255.255
  Lease                         1800
  LeaseObtainedTime             2023-09-07 07:58:03Z
  T1                            2023-09-07 08:13:03Z
  T2                            2023-09-07 08:24:18Z
  LeaseTerminatesTime           2023-09-07 08:28:03Z
  AddressType                   0
  IsServerNapAware              0
  DhcpConnForceBroadcastFlag    0
  DhcpInterfaceOptions          ├Ä☻  w              ├Ä☻  /              ├Ä☻  .
  DhcpGatewayHardware           ├ç┬┐┬╢☻♠    PV├Ñ┬ó┬¥
  DhcpGatewayHardwareCount      1
  RegistrationEnabled           1
  RegisterAdapterName           0
  IPAddress                     10.10.10.11
  SubnetMask                    255.0.0.0
  DefaultGateway                10.10.10.100
  DefaultGatewayMetric          0

ControlSet001\Services\Tcpip\Parameters\Interfaces has no subkeys.
Adapter: {6c733e3b-de84-487a-a0bd-48b9d9ec7616}
LastWrite Time: 2023-09-07 08:01:06Z
  EnableDHCP                    1
  Domain
```

```
NameServer
RegistrationEnabled          1
RegisterAdapterName          0
```

The same information can be extracted using the `ips` plugin.

Installer Execution

Rapid Triage Examination & Analysis Tools

```
Microsoft\Windows\CurrentVersion\Installer\UserData not found.
PS C:\Users\johndoe\Desktop\RegRipper3.0-master> .\rip.exe -r "C:\Users\johndoe\Desktop\forensic_da
Launching installer v.20200517
Launching installer v.20200517
(Software) Determines product install information

Installer
Microsoft\Windows\CurrentVersion\Installer\UserData

User SID: S-1-5-18
Key      : 01DCD275E2FC1D341815B89DCA09680D
LastWrite: 2023-08-28 09:39:56Z
20230828 - Microsoft Visual C++ 2019 X86 Additional Runtime - 14.28.29913 14.28.29913 (Microsoft Co

Key      : 3367A02690A78A24580870A644384C0B
LastWrite: 2023-08-28 09:39:59Z
20230828 - Microsoft Visual C++ 2019 X64 Additional Runtime - 14.28.29913 14.28.29913 (Microsoft Co

Key      : 426D5FF15155343438A75EC40151376E
LastWrite: 2023-08-28 09:40:29Z
20230828 - VMware Tools 11.3.5.18557794 (VMware, Inc.)

Key      : 731DDCEEAD31DE64DA0ADB7F8FEB568B
LastWrite: 2023-08-28 09:39:58Z
20230828 - Microsoft Visual C++ 2019 X64 Minimum Runtime - 14.28.29913 14.28.29913 (Microsoft Corpo

Key      : DBBE6326F05F3B048B91D80B6C8003C8
LastWrite: 2023-08-28 09:39:55Z
20230828 - Microsoft Visual C++ 2019 X86 Minimum Runtime - 14.28.29913 14.28.29913 (Microsoft Corpo
```

Recently Accessed Folders/Docs

Rapid Triage Examination & Analysis Tools

```
PS C:\Users\johndoe\Desktop\RegRipper3.0-master> .\rip.exe -r "C:\Users\johndoe\Desktop\forensic_da
Launching recentdocs v.20200427
recentdocs v.20200427
(NTUSER.DAT) Gets contents of user's RecentDocs key

RecentDocs
**All values printed in MRUList\MRUListEx order.
Software\Microsoft\Windows\CurrentVersion\Explorer\RecentDocs
LastWrite Time: 2023-09-07 08:28:20Z
  2 = The Internet
  7 = threat/
  0 = system32
  6 = This PC
  5 = C:\
  4 = Local Disk (C:)
  3 = Temp
  1 = redirect

Software\Microsoft\Windows\CurrentVersion\Explorer\RecentDocs\Folder
LastWrite Time 2023-09-07 08:28:20Z
MRUListEx = 1,0,3,2
  1 = The Internet
  0 = system32
  3 = This PC
  2 = Local Disk (C:)
```

Autostart - Run Key Entries

Rapid Triage Examination & Analysis Tools

```
PS C:\Users\johndoe\Desktop\RegRipper3.0-master> .\rip.exe -r "C:\Users\johndoe\Desktop\forensic_da
```

```
Launching run v.20200511
run v.20200511
(Software, NTUSER.DAT) [Autostart] Get autostart key contents from Software hive

Software\Microsoft\Windows\CurrentVersion\Run
LastWrite Time 2023-09-07 08:30:07Z
  MicrosoftEdgeAutoLaunch_0562217A6A32A7E92C68940F512715D9 - "C:\Program Files (x86)\Microsoft\Edge
  DiscordUpdate - C:\Windows\Tasks\update.exe

Software\Microsoft\Windows\CurrentVersion\Run has no subkeys.

Software\Wow6432Node\Microsoft\Windows\CurrentVersion\Run not found.

Software\Microsoft\Windows\CurrentVersion\RunOnce not found.

Software\Microsoft\Windows\CurrentVersion\RunServices not found.

Software\Microsoft\Windows\CurrentVersion\RunServicesOnce not found.

Software\Microsoft\Windows NT\CurrentVersion\Terminal Server\Install\Software\Microsoft\Windows\Cur

Software\Microsoft\Windows NT\CurrentVersion\Terminal Server\Install\Software\Microsoft\Windows\Cur

Software\Microsoft\Windows\CurrentVersion\Policies\Explorer\Run not found.

Software\Wow6432Node\Microsoft\Windows\CurrentVersion\Policies\Explorer\Run not found.

Software\Microsoft\Windows\CurrentVersion\StartupApproved\Run not found.

Software\Microsoft\Windows\CurrentVersion\StartupApproved\Run32 not found.

Software\Microsoft\Windows\CurrentVersion\StartupApproved\StartupFolder not found.
```

Program Execution Artifacts

When we talk about `execution artifacts` in digital forensics, we're referring to the traces and evidence left behind on a computer system or device when a program runs. These little bits of information can clue us in on the activities and behaviors of software, users, and even those with malicious intent. If we want to piece together what went down on a computer, diving into these execution artifacts is a must.

You might stumble upon some well-known execution artifacts in these Windows components:

- `Prefetch`
- `ShimCache`
- `Amcache`
- `BAM (Background Activity Moderator)`

Let's dive deeper into each of these to get a better grasp on the kind of program execution details they capture.

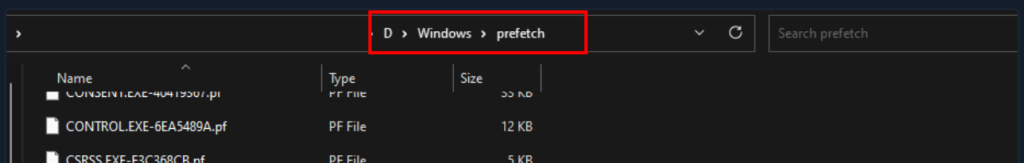
Investigation of Prefetch

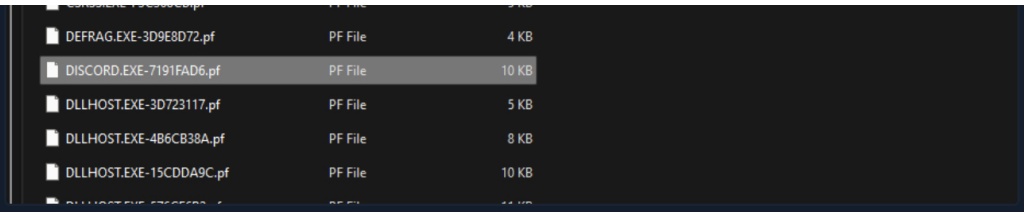
`Prefetch` is a Windows operating system feature that helps optimize the loading of applications by preloading certain components and data. Prefetch files are created for every program that is executed on a Windows system, and this includes both installed applications and standalone executables. The naming convention of Prefetch files is indeed based on the original name of the executable file, followed by a hexadecimal value of the path where the executable file resides, and it ends with the `.pf` file extension.

In digital forensics, the Prefetch folder and associated files can provide valuable insights into the applications that have been executed on a Windows system. Forensic analysts can examine Prefetch files to determine which applications have been run, how often they were executed, and when they were last run.

In general, prefetch files are stored in the `C:\Windows\Prefetch\` directory.

Prefetch-related files harvested from KAPE are typically housed in `<KAPE_output_folder>\Windows\prefetch`.



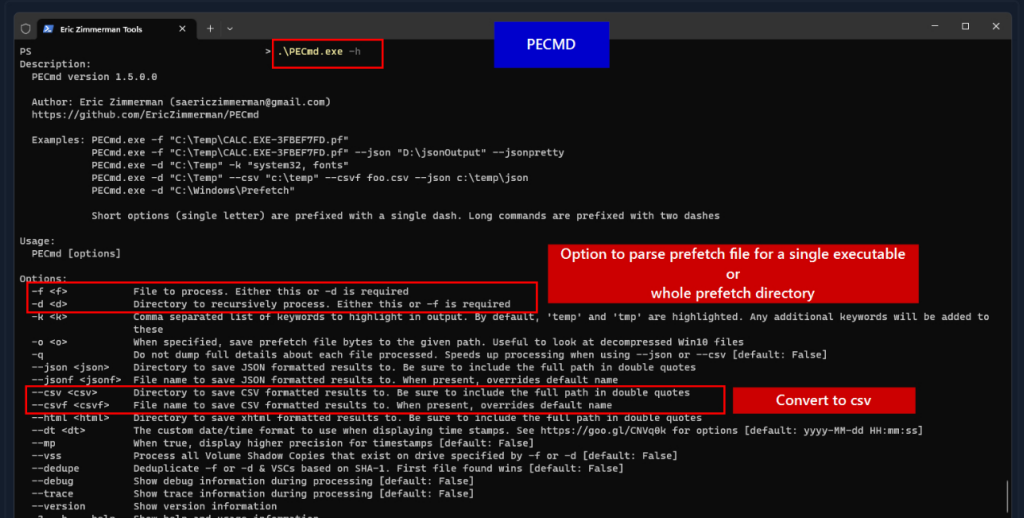


Eric Zimmerman provides a tool for prefetch files: `PECmd` (available at `C:\Users\johndoe\Desktop\Get-ZimmermanTools\net6`).

Here's an example of how to launch PECmd's help menu from the EricZimmerman tools directory.

Rapid Triage Examination & Analysis Tools

```
PS C:\Users\johndoe\Desktop\Get-ZimmermanTools\net6> .\PECmd.exe -h
Description:
  PECmd version 1.5.0.0

  Author: Eric Zimmerman (saericzimmerman@gmail.com)
  https://github.com/EricZimmerman/PECmd

  Examples: PECmd.exe -f "C:\Temp\CALC.EXE-3FBEF7FD.pf"
            PECmd.exe -f "C:\Temp\CALC.EXE-3FBEF7FD.pf" --json "D:\jsonOutput" --jsonpretty
            PECmd.exe -d "C:\Temp" -k "system32, fonts"
            PECmd.exe -d "C:\Temp" --csv "c:\temp" --csvf foo.csv --json c:\temp\json
            PECmd.exe -d "C:\Windows\Prefetch"

            Short options (single letter) are prefixed with a single dash. Long commands are prefix

Usage:
  PECmd [options]

Options:
  -f <f>           File to process. Either this or -d is required
  -d <d>           Directory to recursively process. Either this or -f is required
  -k <k>           Comma separated list of keywords to highlight in output. By default, 'temp' and
                   highlighted. Any additional keywords will be added to these
  -o <o>           When specified, save prefetch file bytes to the given path. Useful to look at de
                   files
  -q               Do not dump full details about each file processed. Speeds up processing when us
                   [default: False]
  --json <json>    Directory to save JSON formatted results to. Be sure to include the full path in
  --jsonf <jsonf>  File name to save JSON formatted results to. When present, overrides default nam
  --csv <csv>      Directory to save CSV formatted results to. Be sure to include the full path in
  --csvf <csvf>    File name to save CSV formatted results to. When present, overrides default name
  --html <html>    Directory to save xhtml formatted results to. Be sure to include the full path i
  --dt <dt>        The custom date/time format to use when displaying time stamps. See https://goo.
                   options [default: yyyy-MM-dd HH:mm:ss]
  --mp             When true, display higher precision for timestamps [default: False]
  --vss            Process all Volume Shadow Copies that exist on drive specified by -f or -d [defa
  --dedupe         Deduplicate -f or -d & VSCs based on SHA-1. First file found wins [default: Fals
  --debug          Show debug information during processing [default: False]
  --trace          Show trace information during processing [default: False]
  --version        Show version information
  -?, -h, --help   Show help and usage information
```



PECmd will analyze the prefetch file (`.pf`) and display various information about the application execution. This generally includes details such as:

- First and last execution timestamps.
- Number of times the application has been executed.
- Volume and directory information.
- Application name and path.
- File information, such as file size and hash values.

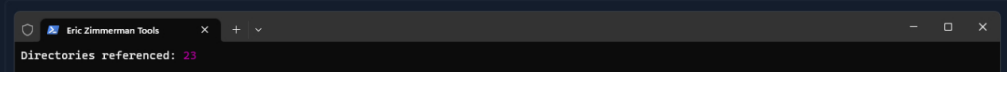
Let's see by providing a path to a single prefetch file, for example the prefetch file related to `discord.exe` (i.e. DISCORD.EXE-7191FAD6.pf located at `C:\Users\johndoe\Desktop\forensic_data\kape_output\D\Windows\prefetch`).

Rapid Triage Examination & Analysis Tools

```
PS C:\Users\johndoe\Desktop\Get-ZimmermanTools\net6> .\PECmd.exe -f C:\Users\johndoe\Desktop\forens
PECmd version 1.5.0.0

Author: Eric Zimmerman (saericzimmerman@gmail.com)
https://github.com/EricZimmerman/PECmd

Command line: -f C:\Users\johndoe\Desktop\forensic_data\kape_output\D\Windows\prefetch\DISCORD.EXE-

Warning: Administrator privileges not found!

Keywords: temp, tmp

Processing C:\Users\johndoe\Desktop\forensic_data\kape_output\D\Windows\prefetch\DISCORD.EXE-7191FA

Created on: 2023-09-07 08:30:16
Modified on: 2023-09-07 08:30:16
Last accessed on: 2023-09-17 15:55:01

Executable name: DISCORD.EXE
Hash: 7191FAD6
File size (bytes): 51,104
Version: Windows 10 or Windows 11

Run count: 1
Last run: 2023-09-07 08:30:06

Volume information:

#0: Name: \VOLUME{01d9da035d4d8f00-285d5e74} Serial: 285D5E74 Created: 2023-08-28 22:59:56 Director

Directories referenced: 23

00: \VOLUME{01d9da035d4d8f00-285d5e74}\$EXTEND
01: \VOLUME{01d9da035d4d8f00-285d5e74}\TEMP (Keyword True)
02: \VOLUME{01d9da035d4d8f00-285d5e74}\USERS
03: \VOLUME{01d9da035d4d8f00-285d5e74}\USERS\JOHN DOE
04: \VOLUME{01d9da035d4d8f00-285d5e74}\USERS\JOHN DOE\APPDATA
05: \VOLUME{01d9da035d4d8f00-285d5e74}\USERS\JOHN DOE\APPDATA\LOCAL
06: \VOLUME{01d9da035d4d8f00-285d5e74}\USERS\JOHN DOE\APPDATA\LOCAL\MICROSOFT
07: \VOLUME{01d9da035d4d8f00-285d5e74}\USERS\JOHN DOE\APPDATA\LOCAL\MICROSOFT\WINDOWS
08: \VOLUME{01d9da035d4d8f00-285d5e74}\USERS\JOHN DOE\APPDATA\LOCAL\MICROSOFT\WINDOWS\CACHES
09: \VOLUME{01d9da035d4d8f00-285d5e74}\USERS\JOHN DOE\APPDATA\LOCAL\MICROSOFT\WINDOWS\INETCACHE
10: \VOLUME{01d9da035d4d8f00-285d5e74}\USERS\JOHN DOE\APPDATA\LOCAL\MICROSOFT\WINDOWS\INETCACHE\IE
11: \VOLUME{01d9da035d4d8f00-285d5e74}\USERS\JOHN DOE\APPDATA\LOCAL\MICROSOFT\WINDOWS\INETCACHE\IE\
12: \VOLUME{01d9da035d4d8f00-285d5e74}\USERS\JOHN DOE\APPDATA\LOCAL\TEMP (Keyword True)
13: \VOLUME{01d9da035d4d8f00-285d5e74}\USERS\JOHN DOE\DOWNLOADS
14: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS
15: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\APPPATCH
16: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\GLOBALIZATION
17: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\GLOBALIZATION\SORTING
18: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\REGISTRATION
19: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32
20: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\DRIVERS
21: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\EN-US
22: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\TASKS

Files referenced: 76

00: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\NTDLL.DLL
01: \VOLUME{01d9da035d4d8f00-285d5e74}\TEMP\DISCORD.EXE (Executable: True)
02: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\KERNEL32.DLL
```

```
03: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\KERNELBASE.DLL
04: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\LOCALE.NLS
05: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\APPHELP.DLL
06: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\APPPATCH\SYSMAIN.SDB
07: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\ADVAPI32.DLL
08: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\MSVCRT.DLL
09: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\SECHOST.DLL
10: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\RPCRT4.DLL
11: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\SHELL32.DLL
12: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\MSVCP_WIN.DLL
13: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\UCRTBASE.DLL
14: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\USER32.DLL
15: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\NETAPI32.DLL
16: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\WIN32U.DLL
17: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\GDI32.DLL
18: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\GDI32FULL.DLL
19: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\WS2_32.DLL
20: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\WININET.DLL
21: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\NETUTILS.DLL
22: \VOLUME{01d9da035d4d8f00-285d5e74}\$MFT
23: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\SAMCLI.DLL
24: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\IMM32.DLL
25: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\DRIVERS\CONDRV.SYS
26: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\NTMARTA.DLL
27: \VOLUME{01d9da035d4d8f00-285d5e74}\TEMP\UNINSTALL.EXE (Keyword: True)
28: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\TASKS\MICROSOFT.WINDOWSKITS.FEEDBACK.EXE
29: \VOLUME{01d9da035d4d8f00-285d5e74}\USERS\JOHN DOE\DOWNLOADS\UNINSTALL.EXE:ZONE.IDENTIFIER
30: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\TASKS\MICROSOFT.WINDOWSKITS.FEEDBACK.EXE:ZONE.IDENTI
31: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\IERTUTIL.DLL
32: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\COMBASE.DLL
33: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\SHCORE.DLL
34: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\GLOBALIZATION\SORTING\SORTDEFAULT.NLS
35: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\SSPICLI.DLL
36: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\WINDOWS.STORAGE.DLL
37: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\WLDP.DLL
38: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\SHLWAPI.DLL
39: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\PROFAPI.DLL
40: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\ONDEMANDCONNROUTEHELPER.DLL
41: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\WINHTTP.DLL
42: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\KERNEL.APPCORE.DLL
43: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\MSWSOCK.DLL
44: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\IPHLPAPI.DLL
45: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\WINNSI.DLL
46: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\NSI.DLL
47: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\URLMON.DLL
48: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\SRVCLI.DLL
49: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\OLEAUT32.DLL
50: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\OLE32.DLL
51: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\DNSAPI.DLL
52: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\RASADHLP.DLL
53: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\FWPUCLNT.DLL
54: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\BCRYPT.DLL
55: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\EN-US\MSWSOCK.DLL.MUI
56: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\WSHQOS.DLL
57: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\EN-US\WSHQOS.DLL.MUI
58: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\C_20127.NLS
59: \VOLUME{01d9da035d4d8f00-285d5e74}\USERS\JOHN DOE\APPDATA\LOCAL\MICROSOFT\WINDOWS\INETCACHE\IE\
60: \VOLUME{01d9da035d4d8f00-285d5e74}\USERS\JOHN DOE\APPDATA\LOCAL\TEMP\DISCORDSETUP.EXE (Keyword:
61: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\TASKS\UPDATE.EXE
62: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\BCRYPTPRIMITIVES.DLL
63: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\RPCSS.DLL
64: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\UXTHEME.DLL
65: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\PROPSYS.DLL
66: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\CFGMGR32.DLL
67: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\CLBCATQ.DLL
68: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\REGISTRATION\R000000000006.CLB
69: \VOLUME{01d9da035d4d8f00-285d5e74}\USERS\JOHN DOE\APPDATA\LOCAL\MICROSOFT\WINDOWS\CACHES\CVERSI
70: \VOLUME{01d9da035d4d8f00-285d5e74}\USERS\JOHN DOE\APPDATA\LOCAL\MICROSOFT\WINDOWS\CACHES\{AFBF9
71: \VOLUME{01d9da035d4d8f00-285d5e74}\USERS\DESKTOP.INI
72: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\SAMLIB.DLL
73: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\CRYPTBASE.DLL
74: \VOLUME{01d9da035d4d8f00-285d5e74}\TEMP\INSTALL.BAT (Keyword: True)
75: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\CMD.EXE


---------- Processed C:\Users\johndoe\Desktop\forensic_data\kape_output\D\Windows\prefetch\DISCORD.
```

Upon scrolling down the output, we can see the directories referenced by this executable.



```
00: \VOLUME{01d9da035d4d8f00-285d5e74}\$EXTEND
01: \VOLUME{01d9da035d4d8f00-285d5e74}\TEMP (Keyword True)
02: \VOLUME{01d9da035d4d8f00-285d5e74}\USERS
03: \VOLUME{01d9da035d4d8f00-285d5e74}\USERS\JOHN DOE
04: \VOLUME{01d9da035d4d8f00-285d5e74}\USERS\JOHN DOE\APPDATA
05: \VOLUME{01d9da035d4d8f00-285d5e74}\USERS\JOHN DOE\APPDATA\LOCAL
06: \VOLUME{01d9da035d4d8f00-285d5e74}\USERS\JOHN DOE\APPDATA\LOCAL\MICROSOFT
07: \VOLUME{01d9da035d4d8f00-285d5e74}\USERS\JOHN DOE\APPDATA\LOCAL\MICROSOFT\WINDOWS
08: \VOLUME{01d9da035d4d8f00-285d5e74}\USERS\JOHN DOE\APPDATA\LOCAL\MICROSOFT\WINDOWS\CACHES
09: \VOLUME{01d9da035d4d8f00-285d5e74}\USERS\JOHN DOE\APPDATA\LOCAL\MICROSOFT\WINDOWS\INETCACHE
10: \VOLUME{01d9da035d4d8f00-285d5e74}\USERS\JOHN DOE\APPDATA\LOCAL\MICROSOFT\WINDOWS\INETCACHE\IE
11: \VOLUME{01d9da035d4d8f00-285d5e74}\USERS\JOHN DOE\APPDATA\LOCAL\MICROSOFT\WINDOWS\INETCACHE\IE\8O7R2XTQ
12: \VOLUME{01d9da035d4d8f00-285d5e74}\USERS\JOHN DOE\APPDATA\LOCAL\TEMP (Keyword True)
13: \VOLUME{01d9da035d4d8f00-285d5e74}\USERS\JOHN DOE\DOWNLOADS
14: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS
15: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\APPPATCH
16: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\GLOBALIZATION
17: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\GLOBALIZATION\SORTING
18: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\REGISTRATION
19: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32
20: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\DRIVERS
21: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\EN-US
22: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\TASKS

Files referenced: 76
```

Further scrolling down the output reveals the files referenced by this executable.

```
Eric Zimmerman Tools
Files referenced: 76

00: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\NTDLL.DLL
01: \VOLUME{01d9da035d4d8f00-285d5e74}\TEMP\DISCORD.EXE (Executable: True)
02: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\KERNEL32.DLL
03: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\KERNELBASE.DLL
04: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\LOCALE.NLS
05: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\APPHELP.DLL
06: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\APPPATCH\SYSMAIN.SDB
07: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\ADVAPI32.DLL
08: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\MSVCRT.DLL
09: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\SECHOST.DLL
10: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\RPCRT4.DLL
11: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\SHELL32.DLL
12: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\MSVCP_WIN.DLL
13: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\UCRTBASE.DLL
14: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\USER32.DLL
15: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\NETAPI32.DLL
16: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\WIN32U.DLL
17: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\GDI32.DLL
18: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\GDI32FULL.DLL
19: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\WS2_32.DLL
20: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\WININET.DLL
21: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\NETUTILS.DLL
22: \VOLUME{01d9da035d4d8f00-285d5e74}\$MFT
23: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\SAMCLI.DLL
24: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\IMM32.DLL
25: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\DRIVERS\CONDRV.SYS
```

Suspicious Activity in Referenced Files

We should also consider the directory where the application was executed from. If it was run from an unusual or unexpected location, it may be suspicious. For example the below screenshot shows some suspicious locations and files.

```
Eric Zimmerman Tools
54: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\BCRYPT.DLL
55: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\EN-US\MSWSOCK.DLL.MUI
56: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\WSHQOS.DLL
57: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\EN-US\WSHQOS.DLL.MUI
58: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\C_20127.NLS
59: \VOLUME{01d9da035d4d8f00-285d5e74}\USERS\JOHN DOE\APPDATA\LOCAL\MICROSOFT\WINDOWS\INETCACHE\IE\8O7R2XTQ\DISCORDSETUP[1].EXE
60: \VOLUME{01d9da035d4d8f00-285d5e74}\USERS\JOHN DOE\APPDATA\LOCAL\TEMP\DISCORDSETUP.EXE (Keyword: True)
61: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\TASKS\UPDATE.EXE
62: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\BCRYPTPRIMITIVES.DLL
63: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\RPCSS.DLL
64: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\UXTHEME.DLL
65: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\PROPSYS.DLL
66: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\CFGMGR32.DLL
67: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\CLBCATQ.DLL
68: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\REGISTRATION\R000000000006.CLB
69: \VOLUME{01d9da035d4d8f00-285d5e74}\USERS\JOHN DOE\APPDATA\LOCAL\MICROSOFT\WINDOWS\CACHES\CVERSIONS.1.DB
70: \VOLUME{01d9da035d4d8f00-285d5e74}\USERS\JOHN DOE\APPDATA\LOCAL\MICROSOFT\WINDOWS\CACHES\{AFBF9F1A-8EE8-4C77-AF34-C647E37CA8D9}.1.VER0X0000000000000003.DB
71: \VOLUME{01d9da035d4d8f00-285d5e74}\USERS\DESKTOP.INI
72: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\SAMLIB.DLL
73: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\CRYPTBASE.DLL
74: \VOLUME{01d9da035d4d8f00-285d5e74}\TEMP\INSTALL.BAT (Keyword: True)
75: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\CMD.EXE
```

Convert Prefetch Files to CSV

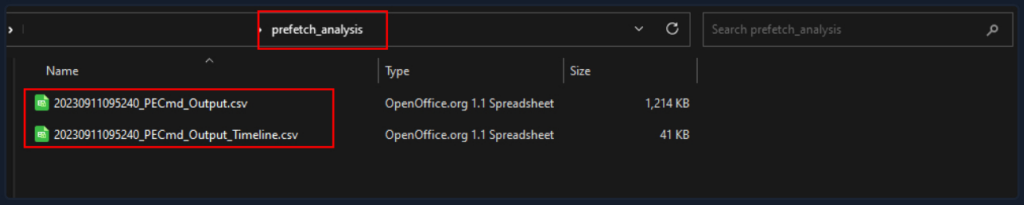
For easier analysis, we can convert the prefetch data into CSV as follows.

```
Rapid Triage Examination & Analysis Tools

PS C:\Users\johndoe\Desktop\Get-ZimmermanTools\net6> .\PECmd.exe -d C:\Users\johndoe\Desktop\forens
PECmd version 1.5.0.0

Author: Eric Zimmerman (saericzimmerman@gmail.com)
https://github.com/EricZimmerman/PECmd

Command line: -d C:\Users\johndoe\Desktop\forensic_data\kape_output\D\Windows\prefetch --csv C:\Use

Warning: Administrator privileges not found!

Keywords: temp, tmp

Looking for prefetch files in C:\Users\johndoe\Desktop\forensic_data\kape_output\D\Windows\prefetch
```

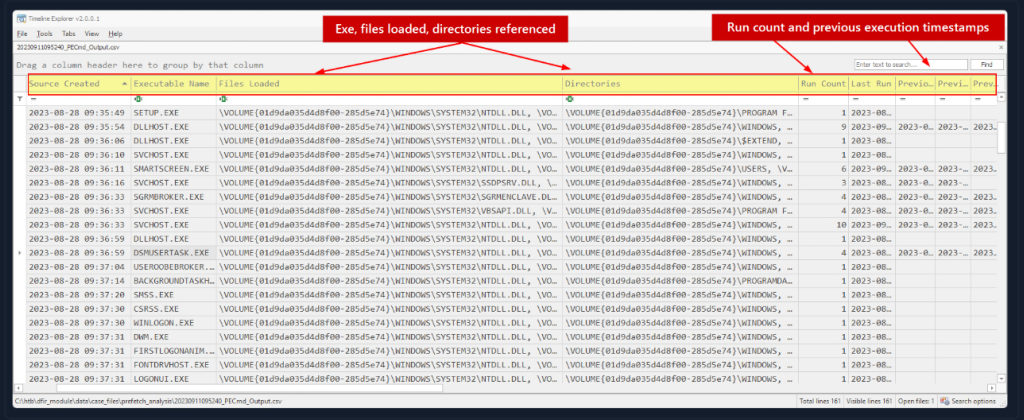
```
Found 161 Prefetch files

Processing C:\Users\johndoe\Desktop\forensic_data\kape_output\D\Windows\prefetch\APPLICATIONFRAMEHO

Created on: 2023-08-28 09:39:12
Modified on: 2023-09-07 08:28:29
Last accessed on: 2023-09-17 16:02:51

Executable name: APPLICATIONFRAMEHOST.EXE
Hash: 8CE9A1EE
File size (bytes): 61,616
Version: Windows 10 or Windows 11

Run count: 2
Last run: 2023-09-07 08:28:18
Other run times: 2023-08-28 09:39:02

Volume information:

#0: Name: \VOLUME{01d9da035d4d8f00-285d5e74} Serial: 285D5E74 Created: 2023-08-28 22:59:56 Director

Directories referenced: 8

00: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS
01: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\FONTS
02: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\GLOBALIZATION
03: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\GLOBALIZATION\SORTING
04: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32
05: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\EN-US
06: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEMAPPS
07: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEMAPPS\MICROSOFT.WINDOWS.SECHEALTHUI_CW5N1H2TXYE

Files referenced: 84

00: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\NTDLL.DLL
01: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\APPLICATIONFRAMEHOST.EXE (Executable: True)
02: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\KERNEL32.DLL
03: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\KERNELBASE.DLL
04: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\LOCALE.NLS
05: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\MSVCRT.DLL
06: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\COMBASE.DLL
07: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\UCRTBASE.DLL
08: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\RPCRT4.DLL
09: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\DXGI.DLL
10: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\WIN32U.DLL
11: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\GDI32.DLL
12: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\GDI32FULL.DLL
13: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\MSVCP_WIN.DLL
14: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\USER32.DLL
15: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\IMM32.DLL
16: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\RPCSS.DLL
17: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\KERNEL.APPCORE.DLL
18: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\BCRYPTPRIMITIVES.DLL
19: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\CLBCATQ.DLL
20: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\REGISTRATION\R000000000006.CLB
21: \VOLUME{01d9da035d4d8f00-285d5e74}\$MFT
22: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\APPLICATIONFRAME.DLL
23: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\SHCORE.DLL
24: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\SHLWAPI.DLL
25: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\OLEAUT32.DLL
26: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\TWINAPI.APPCORE.DLL
27: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\UXTHEME.DLL
28: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\PROPSYS.DLL
29: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\DEVOBJ.DLL
30: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\CFGMGR32.DLL
31: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\TWINAPI.DLL
32: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\SECHOST.DLL
33: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\BCP47MRM.DLL
34: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\D3D11.DLL
35: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\DWMAPI.DLL
36: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\D2D1.DLL
37: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\OLE32.DLL
38: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\ONECOREUAPCOMMONPROXYSTUB.DLL
39: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\WIN32KBASE.SYS
40: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\MSCTF.DLL
41: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\D3D10WARP.DLL
42: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\ADVAPI32.DLL
43: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\RESOURCEPOLICYCLIENT.DLL
44: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\DRIVERS\DXGMMS2.SYS
45: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\DRIVERS\DXGKRNL.SYS
46: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\DXCORE.DLL
47: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\DCOMP.DLL
```

```
48: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\COREMESSAGING.DLL
49: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\WS2_32.DLL
50: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\WIN32KFULL.SYS
51: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\EN-US\APPLICATIONFRAME.DLL.MUI
52: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\UIAUTOMATIONCORE.DLL
53: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\GLOBALIZATION\SORTING\SORTDEFAULT.NLS
54: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\SHELL32.DLL
55: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\WINDOWS.STORAGE.DLL
56: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\WLDP.DLL
57: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\PROFAPI.DLL
58: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\WINDOWS.STATEREPOSITORYCORE.DLL
59: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\WINDOWS.STATEREPOSITORYPS.DLL
60: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\WINDOWSCODECS.DLL
61: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\BCRYPT.DLL
62: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEMAPPS\MICROSOFT.WINDOWS.SECHE
---SNIP---
60: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\UMPDC.DLL
61: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SERVICING\CBSAPI.DLL
62: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SOFTWAREDISTRIBUTION\DOWNLOAD\A766D9CA8E03365B463454
63: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\WINDOWS.STORAGE.DLL
64: \VOLUME{01d9da035d4d8f00-285d5e74}\WINDOWS\SYSTEM32\WLDP.DLL


---------- Processed C:\Users\johndoe\Desktop\forensic_data\kape_output\D\Windows\prefetch\WUAUCLT.
Processed 161 out of 161 files in 14.3289 seconds

CSV output will be saved to C:\Users\johndoe\Desktop\forensic_data\prefetch_analysis\20230917160113
CSV time line output will be saved to C:\Users\johndoe\Desktop\forensic_data\prefetch_analysis\2023
```

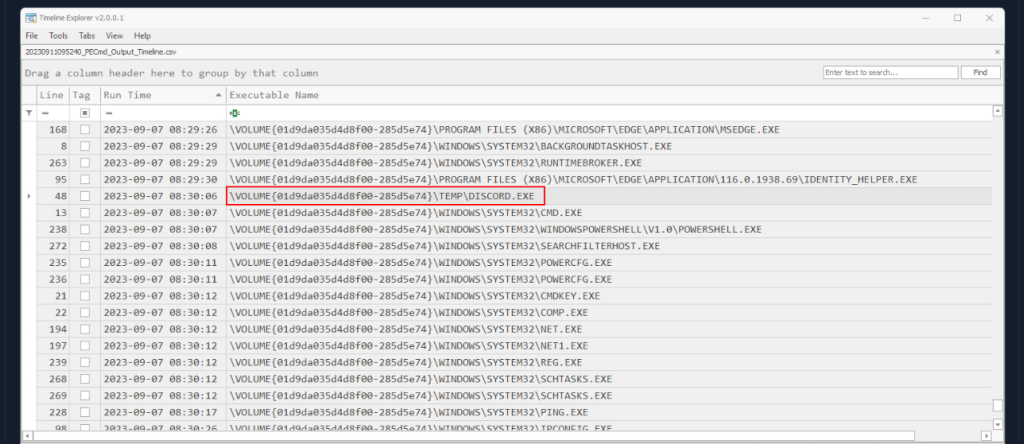
The destination directory contains the parsed output in CSV format.



Now we can easily analyse the output in Timeline Explorer. Let's load both files.



The second output file is the timeline file, which shows the executable details sorted by the run time.



Investigation of ShimCache (Application Compatibility Cache)

`ShimCache` (also known as AppCompatCache) is a Windows mechanism used by the Windows operating systems in order to identify application compatibility issues. This database records information about executed applications, and is stored in the Windows Registry. This information can be used by developers to track compatibility issues with executed programs.

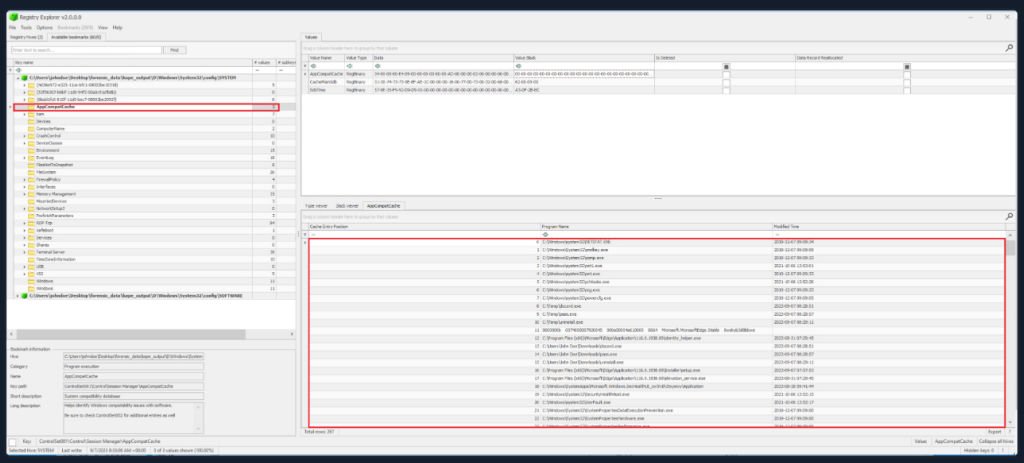
In the `AppCompatCache` cache entries, we can see the information such as:

- `Full file paths`
- `Timestamps`
    - `Last modified time ($Standard_Information)`
    - `Last updated time (Shimcache)`
- `Process execution flag`
- `Cache entry position`

Forensic investigators can use this information to detect the execution of potentially malicious files.

The `AppCompatCache` key is located at the `HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\ControlSet001\Control\Session Manager\AppCompatCache` registry location.

Let's load the `SYSTEM` registry hive (available at `C:\Users\johndoe\Desktop\forensic_data\kape_output\D\Windows\System32\config`) in `Registry Explorer` and see what kind of information it contains. We can do that by opening Registry Explorer and dropping the registry hive files into it. Then we will need to go to bookmarks and select `AppCompatCache`. In the bottom right side, we should see the evidence of application execution as shown in the screenshot.



Investigation of Amcache

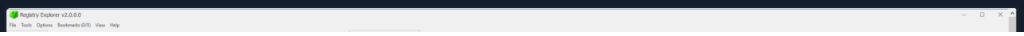
`AmCache` refers to a Windows registry file which is used to store evidence related to program execution. It serves as a valuable resource for digital forensics and security investigations, helping analysts understand the history of application execution and detect signs of any suspicious execution.

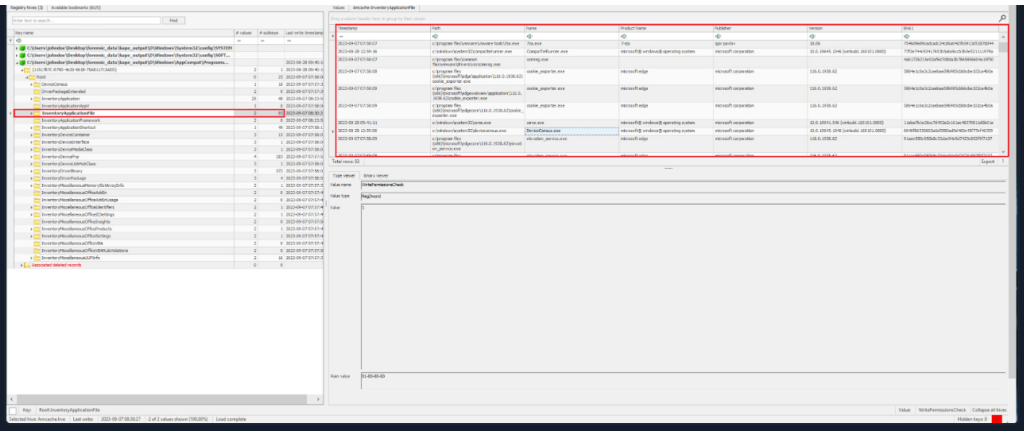
The information that it contains include the execution path, first executed time, deleted time, and first installation. It also provides the file hash for the executables.

On Windows OS the AmCache hive is located at `C:\Windows\AppCompat\Programs\AmCache.hve`

AmCache-related files harvested from KAPE are typically housed in `<KAPE_output_folder>\Windows\AppCompat\Programs`.

Let's load `C:\Users\johndoe\Desktop\forensic_data\kape_output\D\Windows\AppCompat\Programs\AmCache.hve` in Registry Explorer to see what kind of information it contains.





Using Eric Zimmerman's AmcacheParser, we can parse and convert this file into a CSV, and analyze it in detail inside Timeline Explorer.

Rapid Triage Examination & Analysis Tools

```
PS C:\Users\johndoe\Desktop\Get-ZimmermanTools\net6> .\AmcacheParser.exe -f "C:\Users\johndoe\Deskt
AmcacheParser version 1.5.1.0

Author: Eric Zimmerman (saericzimmerman@gmail.com)
https://github.com/EricZimmerman/AmcacheParser

Command line: -f C:\Users\johndoe\Desktop\forensic_data\kape_output\D\Windows\AppCompat\Programs\Am

Warning: Administrator privileges not found!


C:\Users\johndoe\Desktop\forensic_data\kape_output\D\Windows\AppCompat\Programs\AmCache.hve is in n

Total file entries found: 93
Total shortcuts found: 49
Total device containers found: 15
Total device PnPs found: 183
Total drive binaries found: 372
Total driver packages found: 4

Found 36 unassociated file entry

Results saved to: C:\Users\johndoe\Desktop\forensic_data\amcache-analysis

Total parsing time: 0.539 seconds
```

Investigation of Windows BAM (Background Activity Moderator)

The `Background Activity Moderator` (BAM) is a component in the Windows operating system that tracks and logs the execution of certain types of background or scheduled tasks. BAM is actually a kernel device driver as shown in the below screenshot.

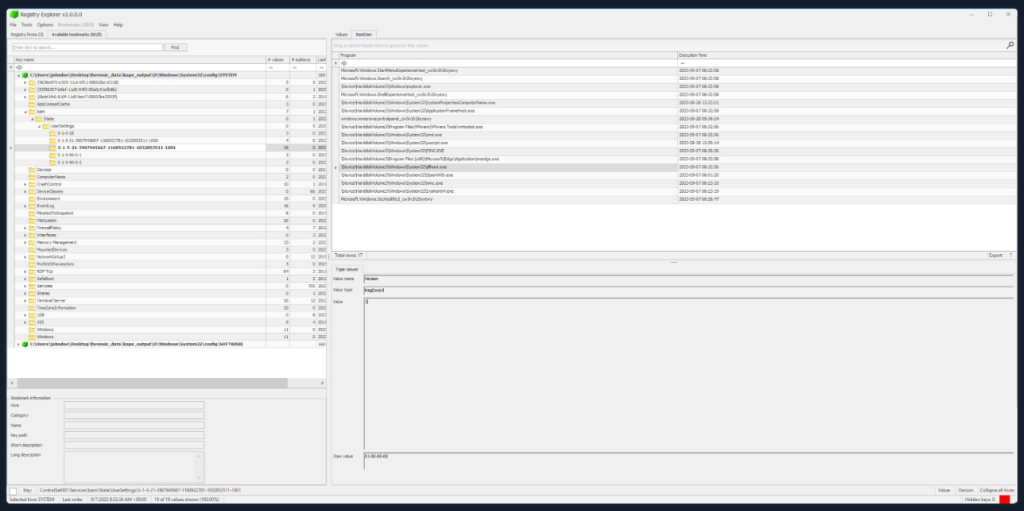
Powershell

```
PS C:\Windows\System32> .\sc.exe qc bam
[SC] QueryServiceConfig SUCCESS

SERVICE_NAME: bam
        TYPE               : 1  KERNEL_DRIVER
        START_TYPE         : 1   SYSTEM_START
        ERROR_CONTROL      : 1   NORMAL
        BINARY_PATH_NAME   : system32\drivers\bam.sys
        LOAD_ORDER_GROUP   :
        TAG                : 0
        DISPLAY_NAME       : Background Activity Moderator Driver
        DEPENDENCIES       :
        SERVICE_START_NAME :
PS C:\Windows\System32>
```

It is primarily responsible for controlling the activity of background applications but it can help us in providing the evidence of program execution which it lists under the bam registry hive. The BAM key is located at the below registry location. `HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\bam\State\UserSettings\{USER-SID}`

Using Registry Explorer, we can browse this inside the SYSTEM hive to see the executable names. Registry explorer

already has a bookmark for `bam`.

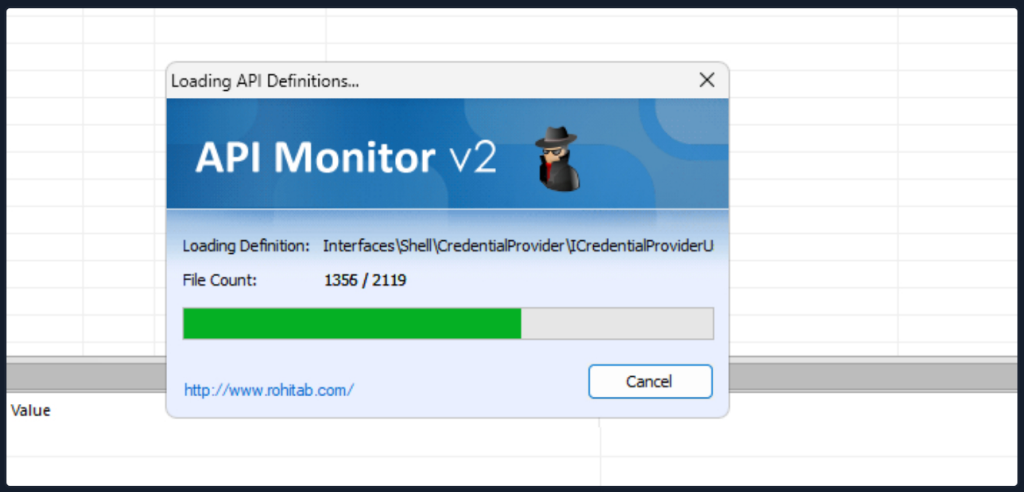


We can also use `RegRipper` to get similar information through its `bam` plugin.

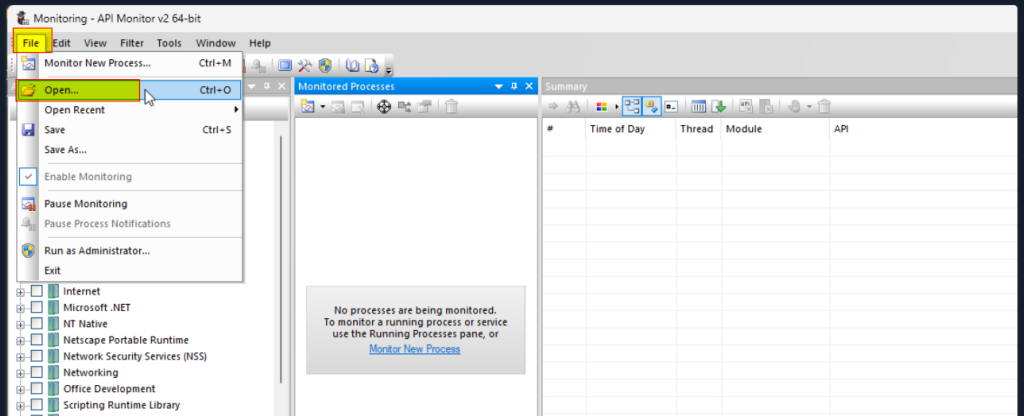
Analyzing Captured API Call Data (`.apmx64`)

`.apmx64` files are generated by API Monitor, which records API call data. These files can be opened and analyzed within the tool itself. API Monitor is a software that captures and displays API calls initiated by applications and services. While its primary function is debugging and monitoring, its capability to capture API call data makes it handy for uncovering forensic artifacts. Let's proceed by loading `C:\Users\johndoe\Desktop\forensic_data\APMX64\discord.apmx64` into API Monitor (available at `C:\Program Files\rohitab.com\API Monitor`) and examining its contents for valuable information.

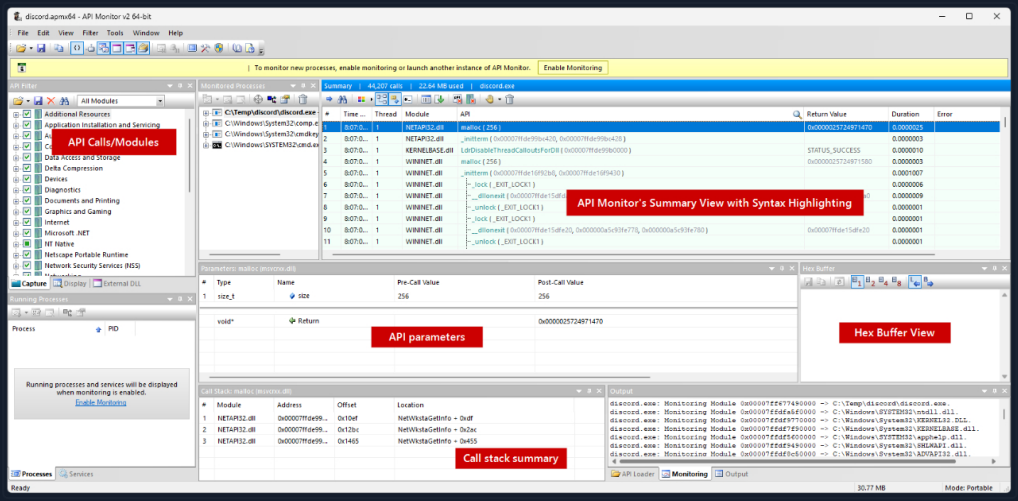
Launching the API Monitor will initiate certain necessary files.



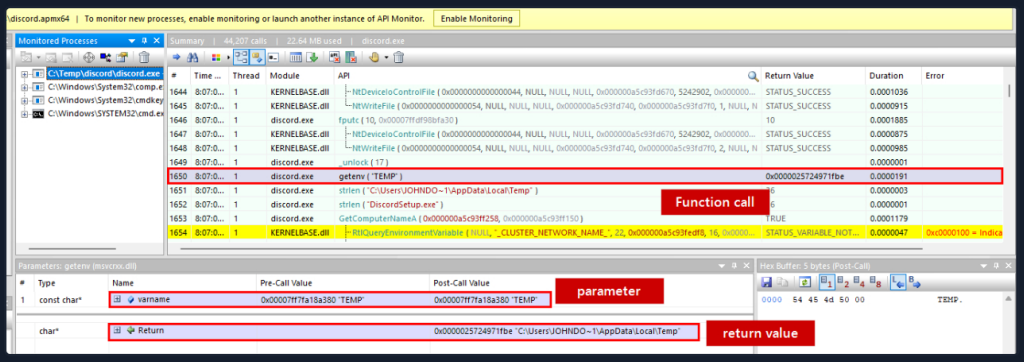
Upon opening the API Monitor application, let's head over to the `File` menu and choose `Open` From there, let's navigate to the location of the `.apmx64` file and select it.



After opening the file, a list of recorded API calls made by the monitored application will be displayed. Typically, this list contains details such as the API function name, parameters, return values, and timestamps. The screenshot below offers a comprehensive view of the API Monitor user interface and its various sections.



Clicking on the monitored processes to the left will display the recorded API call data for the chosen process in the summary view to the right. For illustration, consider selecting the `discord.exe` process. In the summary view, we will observe the API calls initiated by `discord.exe`.



A notable observation from the screenshot is the call to the getenv function. Here's the syntax of this function.

Rapid Triage Examination & Analysis Tools

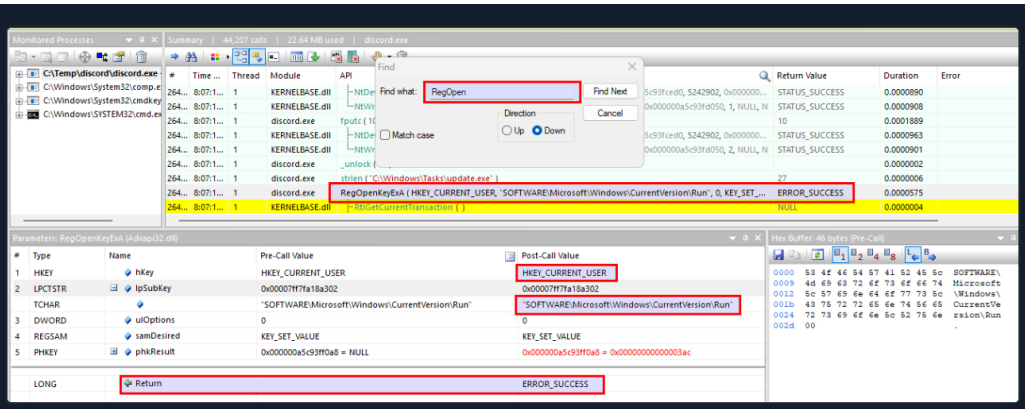
```
char *getenv(
   const char *varname
);
```

This function retrieves the value of a specified environment variable. It requires a `varname` parameter, representing a valid environment variable name, and returns a pointer pointing to the table entry containing the respective environment variable's value.

API Monitor boasts a plethora of filtering and search capabilities. This allows us to hone in on specific API calls based on functions or time frames. By browsing through the summary or utilizing the filter and search functionalities, we can unearth intriguing details, such as API calls concerning file creation, process creation, registry alterations, and more.

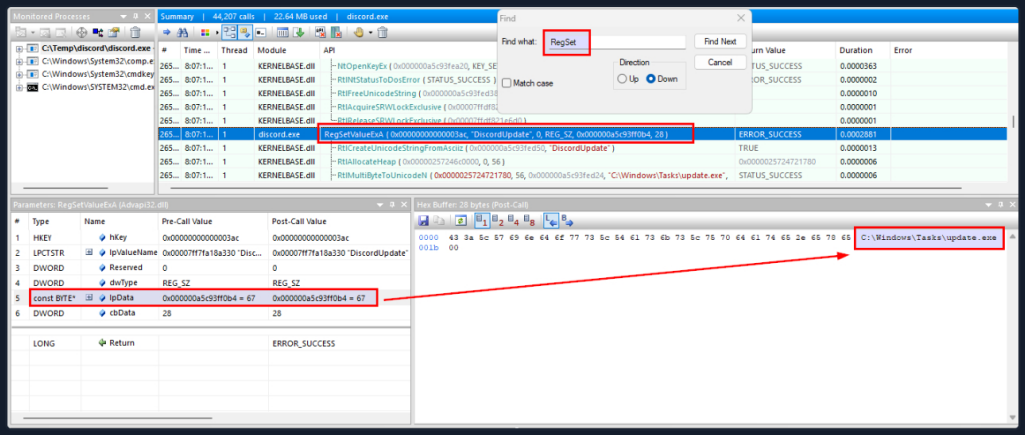
Registry Persistence via Run Keys

An oft-employed strategy by adversaries to maintain unauthorized access to a compromised system is inserting an entry into the `run keys` within the Windows Registry. Let's investigate if there's any reference to the `RegOpenKeyExA` function, which accesses the designated registry key. To perform this search, simply type `RegOpenKey` into the search box, usually situated atop the API Monitor window, and press `Enter`.



From the displayed results, it's evident that the registry key `SOFTWARE\Microsoft\Windows\CurrentVersion\Run` corresponds to the Run registry key, which triggers the designated program upon every user login. Malicious entities often exploit this key to embed entries pointing to their backdoor, a task achievable via the registry API function `RegSetValueExA`.

To explore further, let's seek any mention of the `RegSetValueExA` function, which defines data and type for a specified value within a registry key. Engage the search box, type `RegSet`, and hit `Enter`.

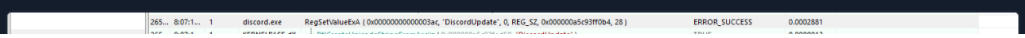


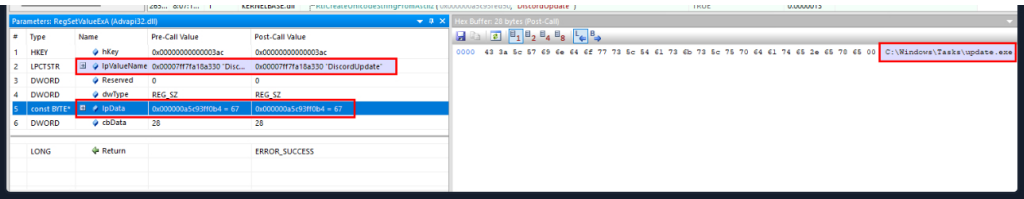
A notable observation is the `RegSetValueExA` invocation. Before diving deeper, let's familiarize ourselves with this function's documentation.

Rapid Triage Examination & Analysis Tools

```
LSTATUS RegSetValueExA(
  [in]           HKEY       hKey,
  [in, optional] LPCSTR     lpValueName,
                 DWORD      Reserved,
  [in]           DWORD      dwType,
  [in]           const BYTE *lpData,
  [in]           DWORD      cbData
);
```

- `hKey1` is a handle to the registry key where you want to set a registry value.
- `lpValueName` is a pointer to a null-terminated string that specifies the name of the registry value you want to set. In this case, it is named as `DiscordUpdate`.
- The `Reserved` parameter is reserved and must be zero.
- `dwType` specifies the data type of the registry value. It's likely an integer constant that represents the data type (e.g., `REG_SZ` for a string value).
- `(BYTE*)lpData` is a type cast that converts the `_lpData_` variable to a pointer to a byte (`BYTE*`). This is done to ensure that the data pointed to by `_lpData_` is treated as a byte array, which is the expected format for binary data in the Windows Registry. In our case, this is shown in the buffer view as `C:\Windows\Tasks\update.exe`.
- `cbData` is an integer that specifies the size, in bytes, of the data pointed to by `_lpData_`.

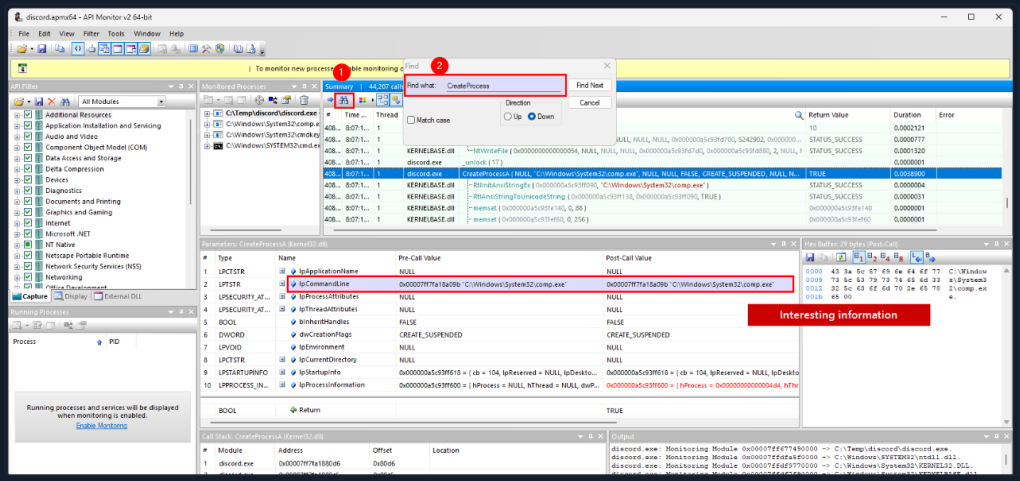




A critical takeaway from this API call is the `lpData` parameter, which reveals the backdoor's location, `C:\Windows\Tasks\update.exe`.

Process Injection

To scrutinize process creation, let's search for the `CreateProcessA` function. Let's key in `CreateProcess` in the search box and press `Enter`.



Presented below is the syntax of the Windows API function, `CreateProcessA`.

Rapid Triage Examination & Analysis Tools

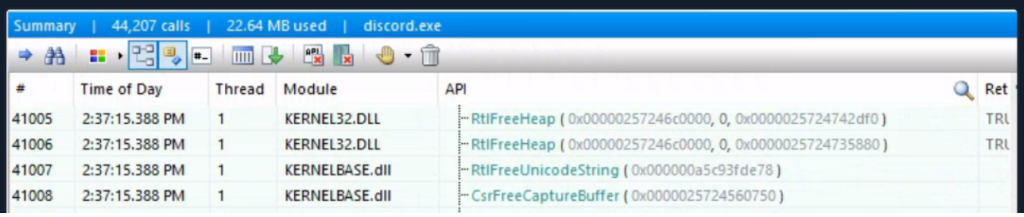
```
BOOL CreateProcessA(
  [in, optional]      LPCSTR                lpApplicationName,
  [in, out, optional] LPSTR                 lpCommandLine,
  [in, optional]      LPSECURITY_ATTRIBUTES lpProcessAttributes,
  [in, optional]      LPSECURITY_ATTRIBUTES lpThreadAttributes,
  [in]                BOOL                  bInheritHandles,
  [in]                DWORD                 dwCreationFlags,
  [in, optional]      LPVOID                lpEnvironment,
  [in, optional]      LPCSTR                lpCurrentDirectory,
  [in]                LPSTARTUPINFOA        lpStartupInfo,
  [out]               LPPROCESS_INFORMATION lpProcessInformation
);
```

An intriguing element within this API is the `lpCommandLine` parameter. It discloses the executed command line, which, in this context, is `C:\Windows\System32\comp.exe`. Notably, the `lpCommandLine` can be specified without delineating the complete executable path in the `lpApplicationName` value.

Another pivotal parameter worth noting is `dwCreationFlags`, set to `CREATE_SUSPENDED`. This indicates that the new process's primary thread starts in a suspended state and remains inactive until the `ResumeThread` function gets invoked.

The `lpCommandLine` parameter of this API call sheds light on the child process that was initiated, namely, `C:\Windows\System32\comp.exe`.

Further down we also notice process injection-related functions being utilized by `discord.exe`.



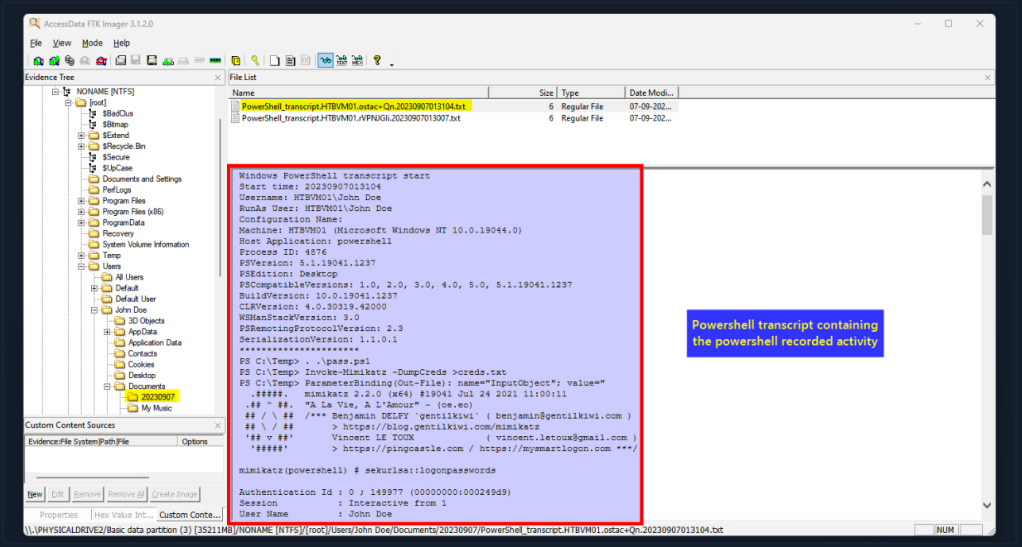
41009	2:37:15.388 PM	1	KERNELBASE.dll	RtlFreeUnicodeString (0x000000a5c93ff138)	
41010	2:37:15.388 PM	1	KERNELBASE.dll	RtlFreeUnicodeString (0x000000a5c93ff148)	
41011	2:37:15.388 PM	1	KERNELBASE.dll	RtlFreeUnicodeString (0x000000a5c93ff158)	
41012	2:37:15.388 PM	1	KERNELBASE.dll	RtlFreeHeap (0x00000257246c0000, 0, NULL)	TRU
41013	2:37:15.388 PM	1	KERNELBASE.dll	RtlFreeHeap (0x00000257246c0000, 0, NULL)	TRU
41014	2:37:15.388 PM	1	KERNELBASE.dll	RtlFreeHeap (0x00000257246c0000, 0, NULL)	TRU
41015	2:37:15.388 PM	1	discord.exe	OpenProcess (PROCESS_ALL_ACCESS, FALSE, 3276)	0x0
41016	2:37:15.388 PM	1	KERNELBASE.dll	NtOpenProcess (0x000000a5c93ff308, PROCESS_ALL_ACCESS, 0x00000...	STA
41017	2:37:15.388 PM	1	discord.exe	VirtualAllocEx (0x00000000000004e0, NULL, 511, MEM_COMMIT | MEM_RE...	0x0
41018	2:37:15.388 PM	1	KERNELBASE.dll	NtAllocateVirtualMemory (0x00000000000004e0, 0x000000a5c93ff2b8, 0,.	STA
41019	2:37:15.388 PM	1	discord.exe	WriteProcessMemory (0x00000000000004e0, 0x00000256275d0000, 0x0000...	TRU
41020	2:37:15.388 PM	1	KERNELBASE.dll	NtQueryVirtualMemory (0x00000000000004e0, 0x00000256275d0000, 8, 0	STA
41021	2:37:15.388 PM	1	KERNELBASE.dll	NtWriteVirtualMemory (0x00000000000004e0, 0x00000256275d0000, 0...	STA
41022	2:37:15.388 PM	1	discord.exe	CreateRemoteThread (0x00000000000004e0, NULL, 0, 0x00000256275d0000,	0x0
41023	2:37:15.388 PM	1	KERNELBASE.dll	NtDuplicateObject (GetCurrentProcess(), 0x00000000000004e0, GetCur...	STA

All the above are strong indicators of process injection.

PowerShell Activity

PowerShell transcripts meticulously log both the commands issued and their respective outputs during a PowerShell session. Occasionally, within a user's documents directory, we might stumble upon PowerShell transcript files. These files grant us a window into the recorded PowerShell activities on the system.

The subsequent screenshot, showcases the PowerShell transcript files nestled within the user's documents directory on a mounted forensic image.

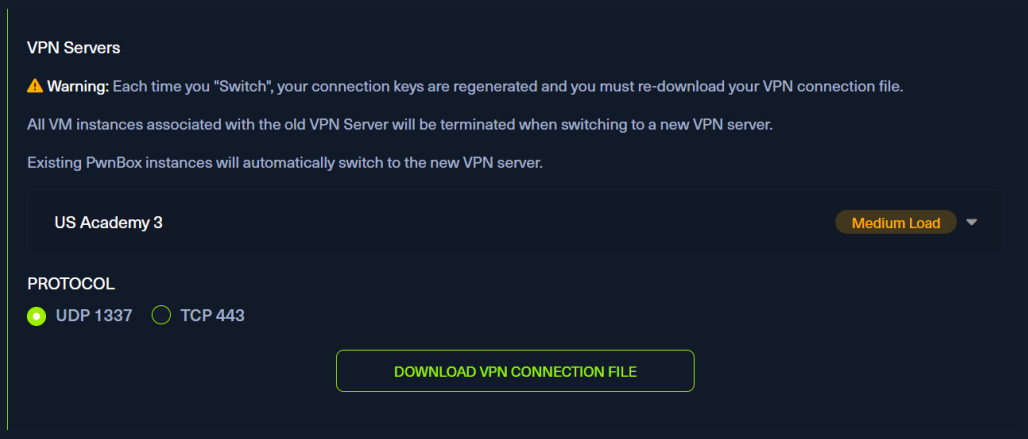


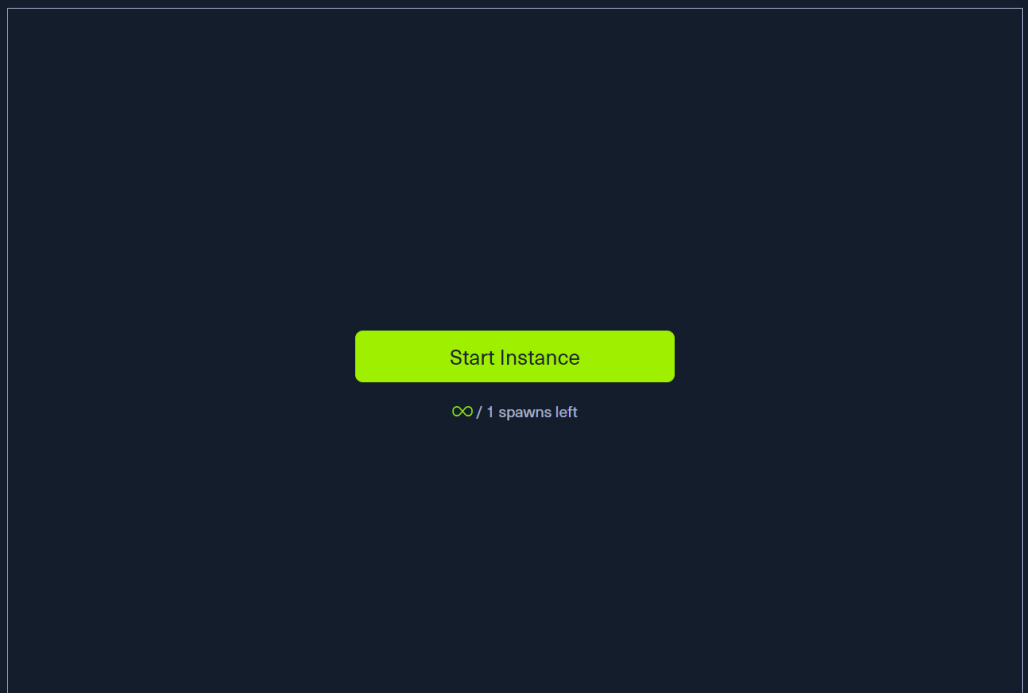
Reviewing PowerShell-related activity in detail can be instrumental during investigations.

Here are some recommended guidelines when handling PowerShell data.

- **Unusual Commands**: Look for PowerShell commands that are not typical in your environment or are commonly associated with malicious activities. For example, commands to download files from the internet (Invoke-WebRequest or wget), commands that manipulate the registry, or those that involve creating scheduled tasks.
- **Script Execution**: Check for the execution of PowerShell scripts, especially if they are not signed or come from untrusted sources. Scripts can be used to automate malicious actions.
- Encoded Commands: Malicious actors often use encoded or obfuscated PowerShell commands to evade detection. Look for signs of encoded commands in transcripts.
- **Privilege Escalation**: Commands that attempt to escalate privileges, change user permissions, or perform actions typically restricted to administrators can be suspicious.
- **File Operations**: Check for PowerShell commands that involve creating, moving, or deleting files, especially in sensitive system locations.
- **Network Activity**: Look for commands related to network activity, such as making HTTP requests or initiating network connections. These may be indicative of command and control (C2) communications.

- Registry Manipulation: Check for commands that involve modifying the Windows Registry, as this can be a common tactic for malware persistence.
- Use of Uncommon Modules: If a PowerShell script or command uses uncommon or non-standard modules, it could be a sign of suspicious activity.
- User Account Activity: Look for changes to user accounts, including creation, modification, or deletion. Malicious actors may attempt to create or manipulate user accounts for persistence.
- Scheduled Tasks: Investigate the creation or modification of scheduled tasks through PowerShell. This can be a common method for persistence.
- Repeated or Unusual Patterns: Analyze the patterns of PowerShell commands. Repeated, identical commands or unusual sequences of commands may indicate automation or malicious behavior.
- Execution of Unsigned Scripts: The execution of unsigned scripts can be a sign of suspicious activity, especially if script execution policies are set to restrict this.





Waiting to start...

Enable step-by-step solutions for all questions

Questions

Answer the question(s) below to complete this Section and earn cubes!

Download VPN Connection File

Target(s): Click here to spawn the target system!

RDP to with user "johndoe" and password "password"

+ 1 During our examination of the USN Journal within Timeline Explorer, we observed "uninstall.exe". The attacker subsequently renamed this file. Use Zone.Identifier information to determine its new name and enter it as your answer.

MICROSOFT.WINDOWSKITS.FEEDBACK.EXE

Submit

RDP to with user "johndoe" and password "password"

+ 1 Review the file at "C:\Users\johndoe\Desktop\forensic_data\kape_output\D\Windows\System32\winevt\logs\Microsoft-Windows-Sysmon%4Operational.evtx" using Timeline Explorer. It documents the creation of two scheduled tasks. Enter the name of the scheduled task that begins with "M" and concludes with "r" as your answer.

Microsoft-Windows-DiagnosticDataCollector

Submit

RDP to with user "johndoe" and password "password"

+ 1 Examine the contents of the file located at "C:\Users\johndoe\Desktop\forensic_data\APMX64\discord.apmx64" using API Monitor. "discord.exe" performed process injection against another process as well. Identify its name and enter it as your answer.

cmdkey.exe

Submit

Previous Next

Mark Complete & Next