Evidence Acquisition Techniques & Tools

`Evidence acquisition` is a critical phase in digital forensics, involving the collection of digital artifacts and data from various sources to preserve potential evidence for analysis. This process requires specialized tools and techniques to ensure the integrity, authenticity, and admissibility of the collected evidence. Here's an overview of evidence acquisition techniques commonly used in digital forensics:

- Forensic Imaging
- Extracting Host-based Evidence & Rapid Triage
- Extracting Network Evidence

Forensic Imaging

Forensic imaging is a fundamental process in digital forensics that involves creating an exact, bit-by-bit copy of digital storage media, such as hard drives, solid-state drives, USB drives, and memory cards. This process is crucial for preserving the original state of the data, ensuring data integrity, and maintaining the admissibility of evidence in legal proceedings. Forensic imaging plays a critical role in investigations by allowing analysts to examine evidence without altering or compromising the original data.

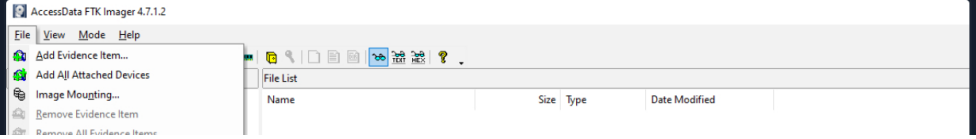
Below are some forensic imaging tools and solutions:

- `FTK Imager`: Developed by AccessData (now acquired by Exterro), FTK Imager is one of the most widely used disk imaging tools in the cybersecurity field. It allows us to create perfect copies (or images) of computer disks for analysis, preserving the integrity of the evidence. It also lets us view and analyze the contents of data storage devices without altering the data.
- `AFF4 Imager`: A free, open-source tool crafted for creating and duplicating forensic disk images. It's user-friendly and compatible with numerous file systems. A benefit of the AFF4 Imager is its capability to extract files based on their creation time, segment volumes, and reduce the time taken for imaging through compression.
- `DD and DCFLDD`: Both are command-line utilities available on Unix-based systems (including Linux and MacOS). DD is a versatile tool included in most Unix-based systems by default, while DCFLDD is an enhanced version of DD with features specifically useful for forensics, such as hashing.
- `Virtualization Tools`: Given the prevalent use of virtualization in modern systems, incident responders will often need to collect evidence from virtual environments. Depending on the specific virtualization solution, evidence can be gathered by temporarily halting the system and transferring the directory that houses it. Another method is to utilize the snapshot capability present in numerous virtualization software tools.

Example 1: Forensic Imaging with FTK Imager

Let's now see a demonstration of utilizing `FTK Imager` to craft a disk image. Be mindful that you'll require an auxiliary storage medium, like an external hard drive or USB flash drive, to save the resultant disk image.

- Select `File -> Create Disk Image`.

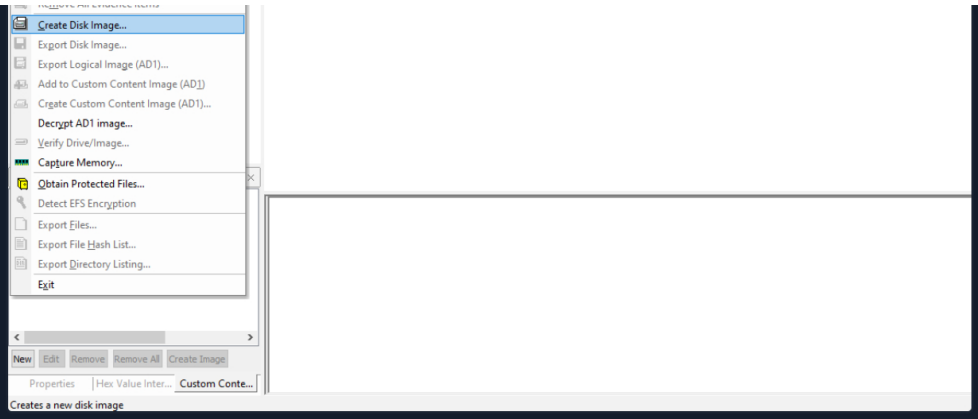


My Workstation

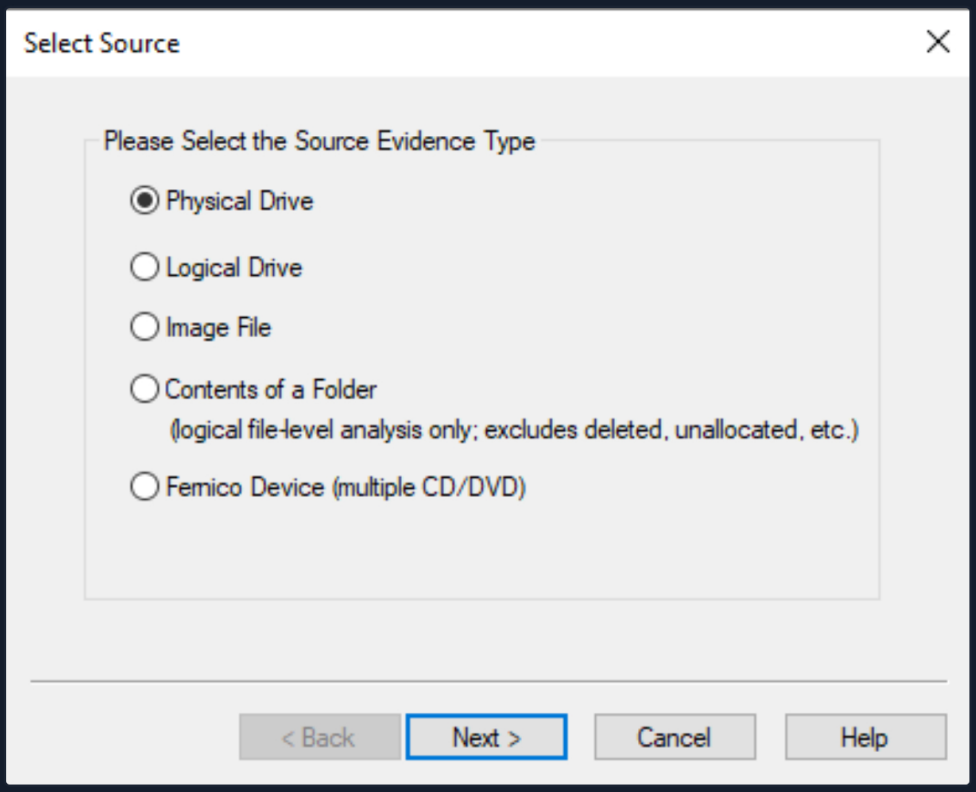
OFFLINE

Start Instance

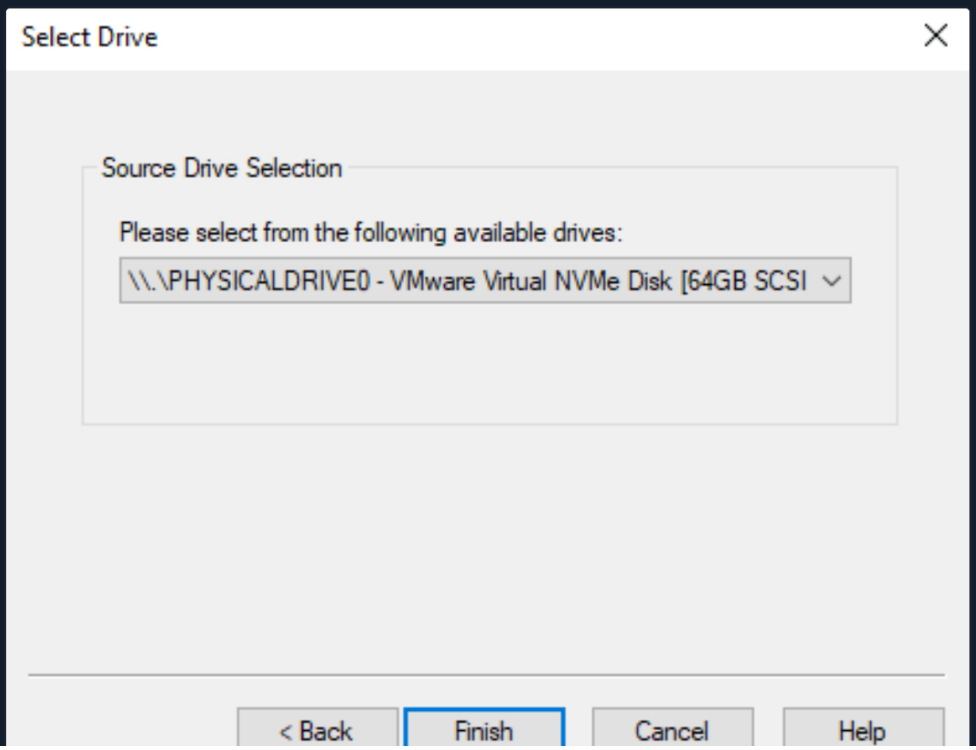
∞ / 1 spawns left



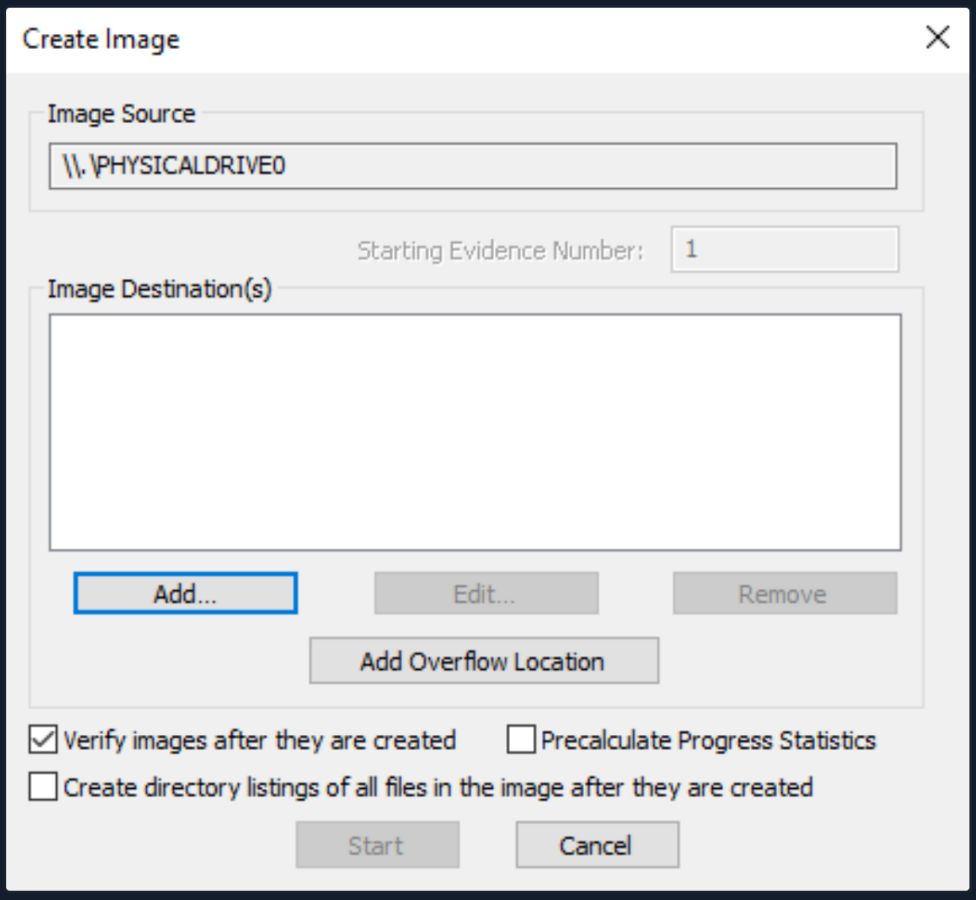
- Next, select the media source. Typically, it's either `Physical Drive` or `Logical Drive`.



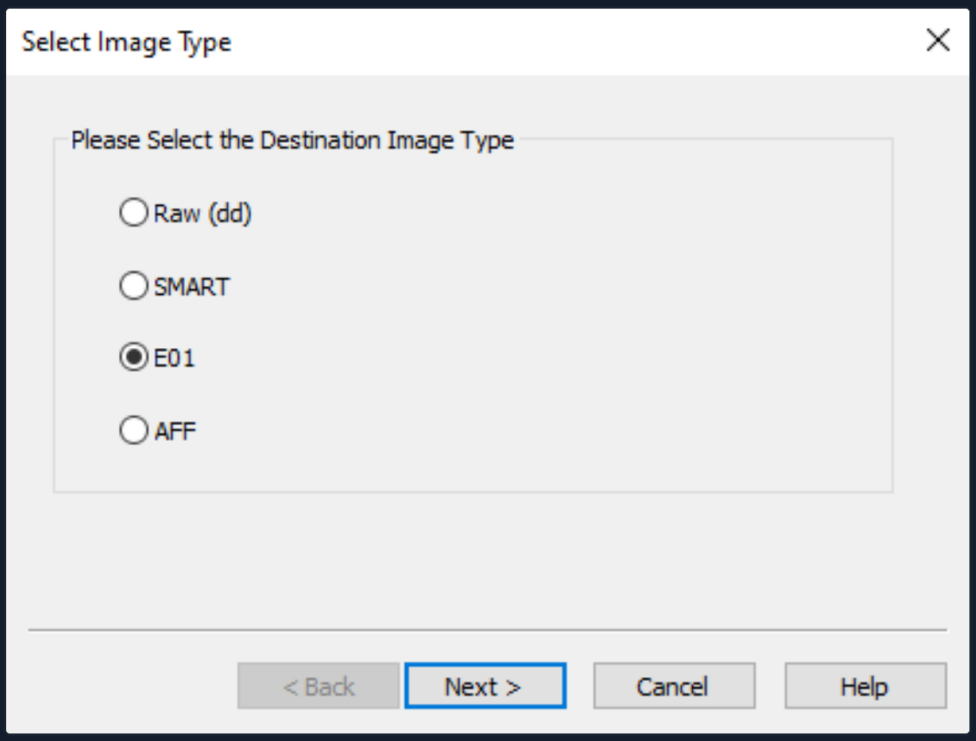
- Choose the drive from which you wish to create an image.



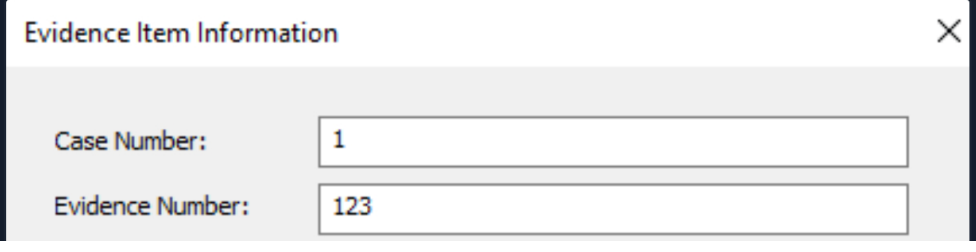
- Specify the destination for the image.

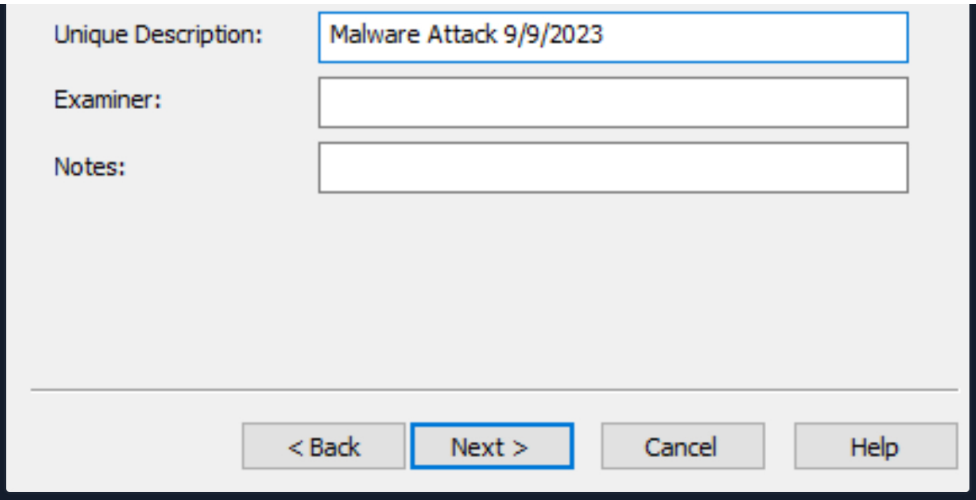


- Select the desired image type.

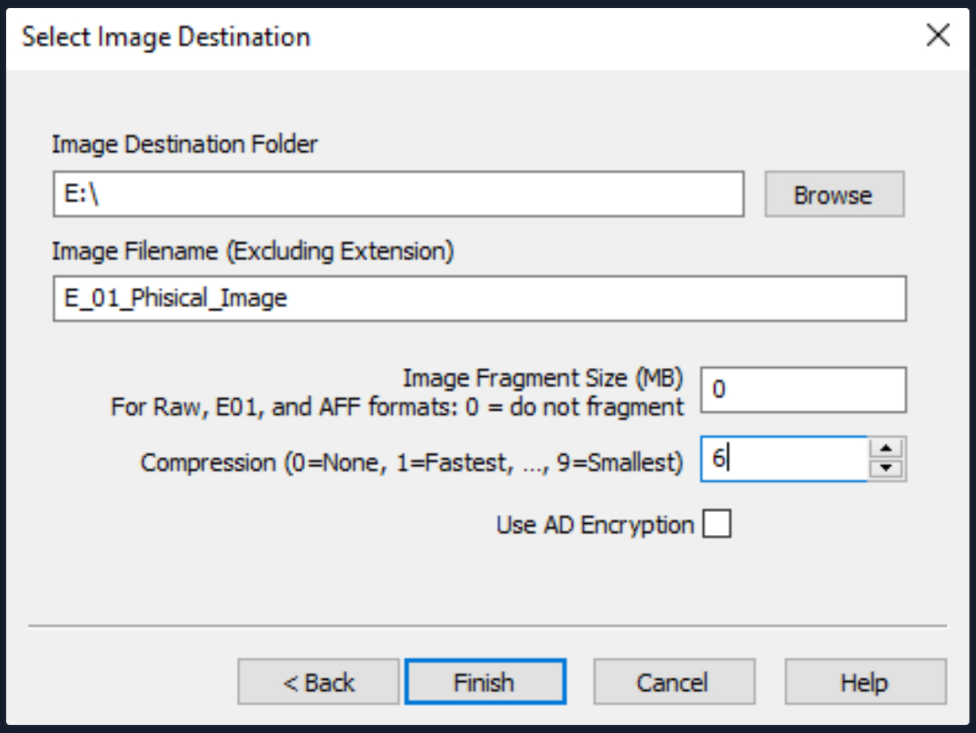


- Input evidence details.

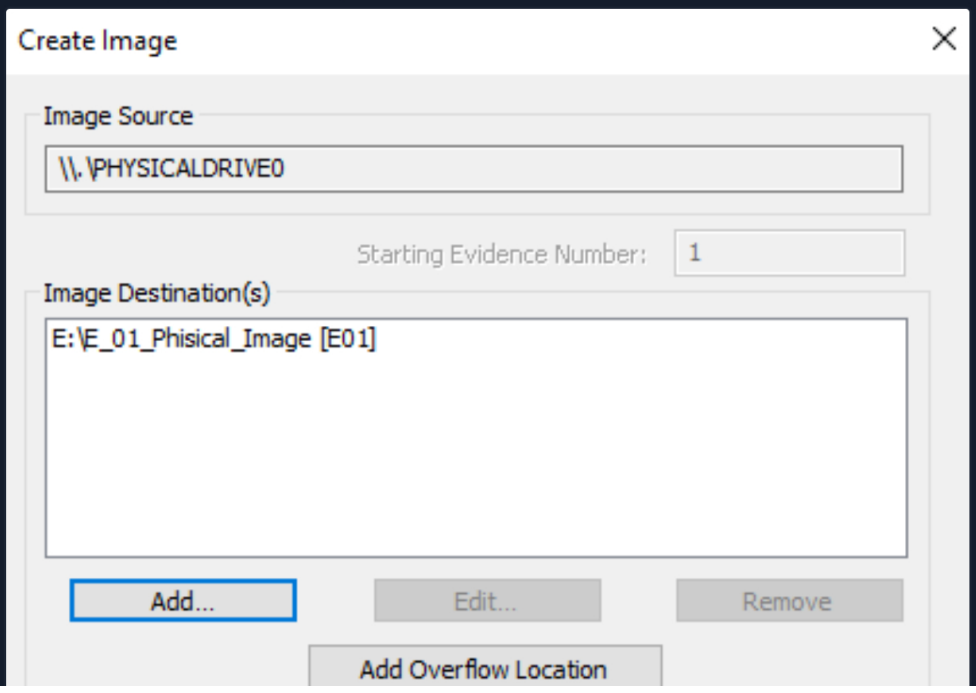


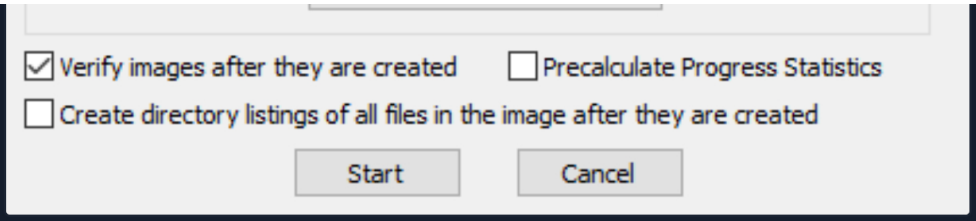


- Choose the destination folder and filename for the image. At this step, you can also adjust settings for image fragmentation and compression.

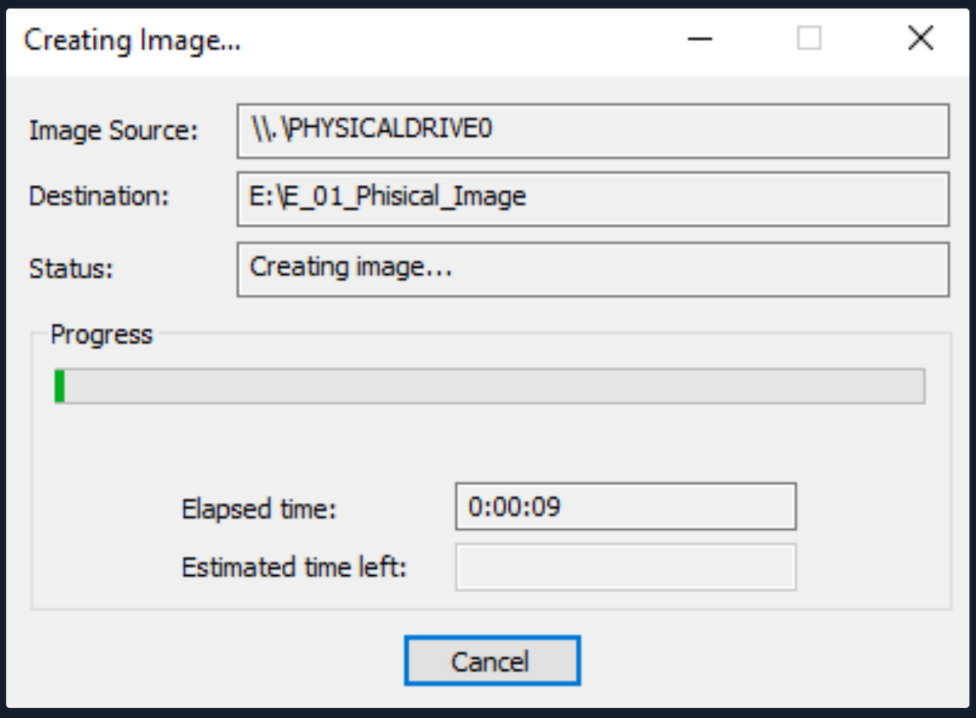


- Once all settings are confirmed, click `Start`.

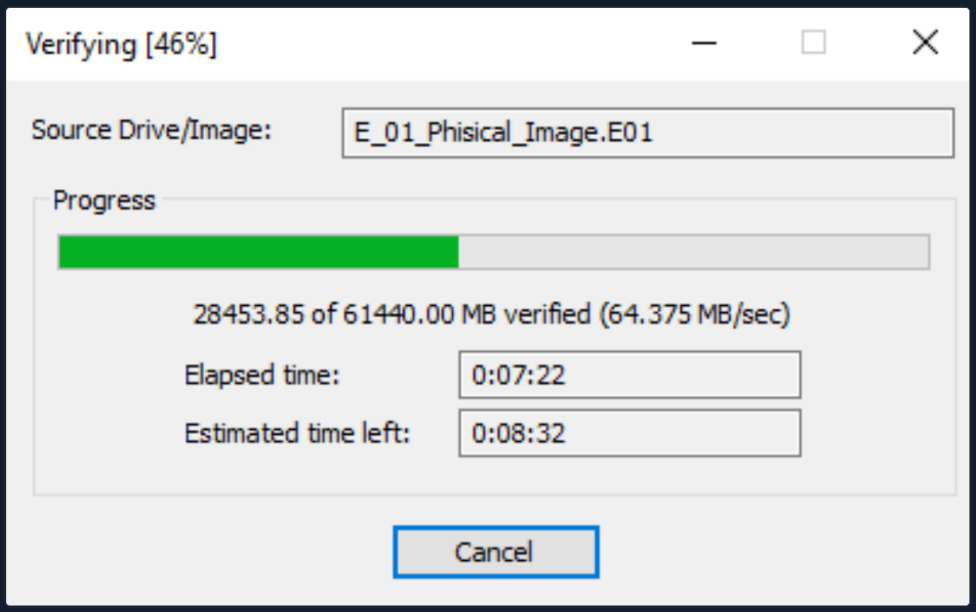




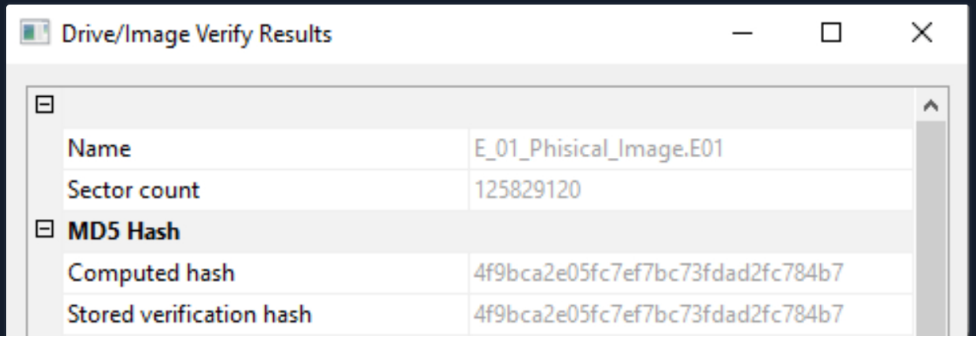
- You'll observe the progress of the imaging.

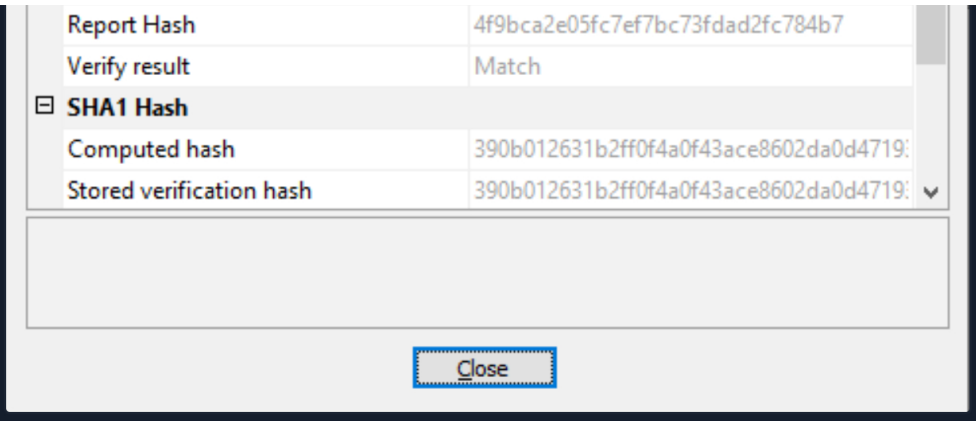


- If you opted to verify the image, you'll also see the verification progress.



- After the image has been verified, you'll receive an imaging summary. Now, you're prepared to analyze this dump.

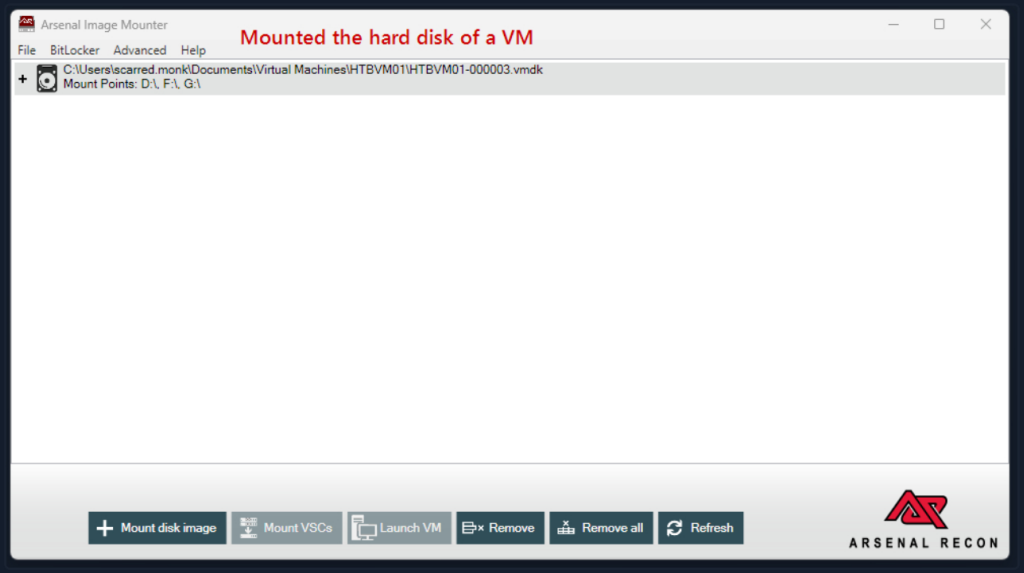




Example 2: Mounting a Disk Image with Arsenal Image Mounter

Let's now see another demonstration of utilizing Arsenal Image Mounter to mount a disk image we have previously created (not the one mentioned above) from a compromised Virtual Machine (VM) running on VMWare. The virtual hard disk of the VM has been stored as `HTBVM01-000003.vmdk`.

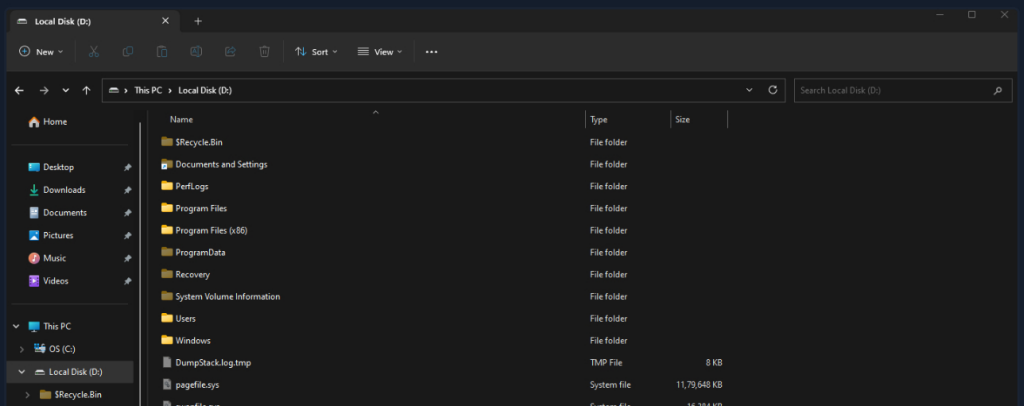
After we've installed Arsenal Image Mounter, let's ensure we launch it with `administrative rights`. In the main window of Arsenal Image Mounter, let's click on the `Mount disk image` button. From there, we'll navigate to the location of our `.VMDK` file and select it.



Arsenal Image Mounter will then start its analysis of the VMDK file. We'll also have the choice to decide if we want to mount the disk as `read-only` or `read-write`, based on our specific requirements.

Choosing to mount a disk image as read-only is a foundational step in digital forensics and incident response. This approach is vital for preserving the original state of evidence, ensuring its authenticity and integrity remain intact.

Once mounted, the image will appear as a drive, assigned the letter `D:\`.



Extracting Host-based Evidence & Rapid Triage

Host-based Evidence

Modern operating systems, with Microsoft Windows being a prime example, generate a plethora of evidence artifacts. These can arise from application execution, file modifications, or even the creation of user accounts. Each of these actions leaves behind a trail, providing invaluable insights for incident response analysts.

Evidence on a host system varies in its nature. The term `volatility` refers to the persistence of data on a host system, with volatile data being information that disappears after events such as logoffs or power shutdowns. One crucial type of volatile evidence is the system's active memory. During investigations, especially those concerning malware infections, this live system memory becomes indispensable. Malware often leaves traces within system memory, and losing this evidence can hinder an analyst's investigation. To capture memory, tools like FTK Imager are commonly employed.

Some other memory acquisition solutions are:

- WinPmem: WinPmem has been the default open source memory acquisition driver for windows for a long time. It used to live in the Rekall project, but has recently been separated into its own repository.
- DumpIt: A simplistic utility that generates a physical memory dump of Windows and Linux machines. On Windows, it concatenates 32-bit and 64-bit system physical memory into a single output file, making it extremely easy to use.
- MemDump: MemDump is a free, straightforward command-line utility that enables us to capture the contents of a system's RAM. It's quite beneficial in forensics investigations or when analyzing a system for malicious activity. Its simplicity and ease of use make it a popular choice for memory acquisition.
- Belkasoft RAM Capturer: This is another powerful tool we can use for memory acquisition, provided free of charge by Belkasoft. It can capture the RAM of a running Windows computer, even if there's active anti-debugging or anti-dumping protection. This makes it a highly effective tool for extracting as much data as possible during a live forensics investigation.
- Magnet RAM Capture: Developed by Magnet Forensics, this tool provides a free and simple way to capture the volatile memory of a system.
- LiME (Linux Memory Extractor): LiME is a Loadable Kernel Module (LKM) which allows the acquisition of volatile memory. LiME is unique in that it's designed to be transparent to the target system, evading many common anti-forensic measures.

Example 1: Acquiring Memory with WinPmem

Let's now see a demonstration of utilizing `WinPmem` for memory acquisition.

To generate a memory dump, simply execute the command below with Administrator privileges.

Evidence Acquisition Techniques & Tools

```
C:\Users\X\Downloads> winpmem_mini_x64_rc2.exe memdump.raw
```

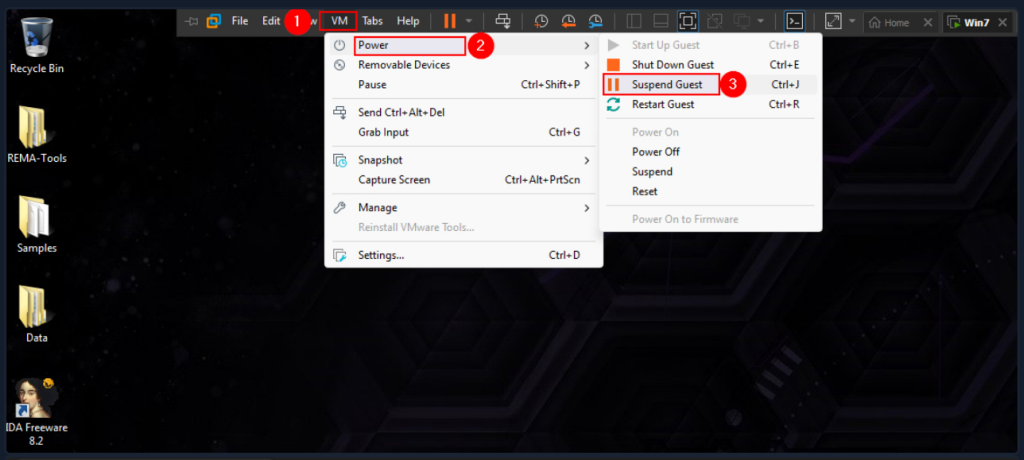
```
C:\Users\Kevin\Downloads+>winpmem_mini_x64_rc2.exe memdump.raw
WinPmem64
Extracting driver to C:\Users\Kevin\AppData\Local\Temp\pme143C.tmp
Driver Unloaded.
Loaded Driver C:\Users\Kevin\AppData\Local\Temp\pme143C.tmp.
Deleting C:\Users\Kevin\AppData\Local\Temp\pme143C.tmp
The system time is: 16:13:11
Will generate a RAW image
 - buffer_size_: 0x1000
CR3: 0x00001AD002
 5 memory ranges:
Start 0x00001000 - Length 0x0009F000
Start 0x00100000 - Length 0x0EEB0000
Start 0x0EFB4000 - Length 0x0000E000
Start 0x0EFC7000 - Length 0x00F1F000
```

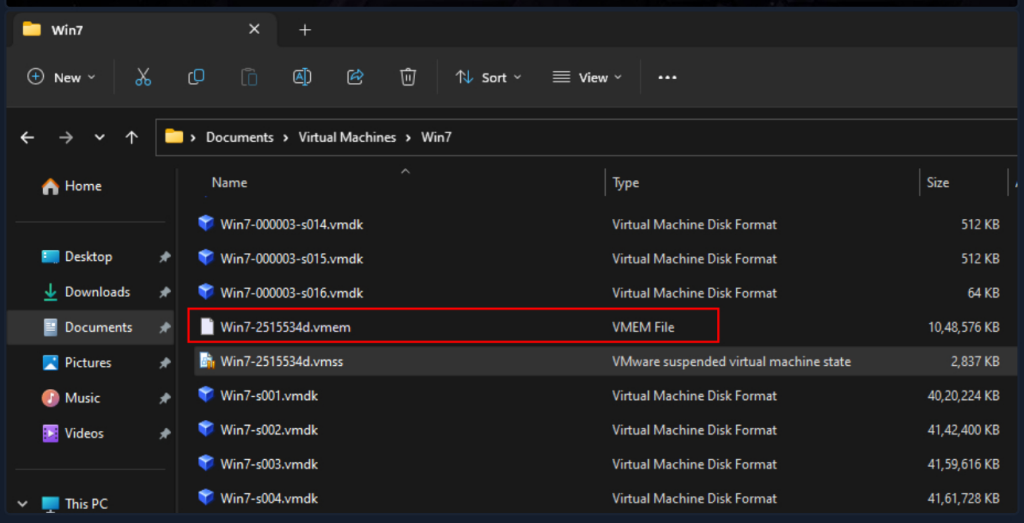
```
Start 0x0FF76000 - Length 0x7008A000
max_physical_memory_ 0x80000000
Acquisition mode PTE Remapping
Padding from 0x00000000 to 0x00001000
pad
 - length: 0x1000

00% 0x00000000 .
copy_memory
 - start: 0x1000
 - end: 0xa0000

00% 0x00001000 .
Padding from 0x000A0000 to 0x00100000
pad
 - length: 0x60000

00% 0x000A0000 .
copy_memory
 - start: 0x100000
 - end: 0xefb0000
```

Example 2: Acquiring VM Memory

Here are the steps to acquire memory from a Virtual Machine (VM).

1. Open the running VM's options
2. Suspend the running VM
3. Locate the `.vmem` file inside the VM's directory.





On the other hand, non-volatile data remains on the hard drive, typically persisting through shutdowns. This category includes artifacts such as:

- Registry
- Windows Event Log
- System-related artifacts (e.g., Prefetch, Amcache)
- Application-specific artifacts (e.g., IIS logs, Browser history)

Rapid Triage

This approach emphasizes collecting data from potentially compromised systems. The goal is to centralize high-value

data, streamlining its indexing and analysis. By centralizing this data, analysts can more effectively deploy tools and techniques, honing in on systems with the most evidentiary value. This targeted approach allows for a deeper dive into digital forensics, offering a clearer picture of the adversary's actions.

One of the best, if not the best, rapid artifact parsing and extraction solutions is KAPE (Kroll Artifact Parser and Extractor). Let's see how we can employ `KAPE` to retrieve valuable forensic data from the image we previously mounted with the help of Arsenal Image Mounter (`D:\`).

`KAPE` is a powerful tool in the realm of digital forensics and incident response. Designed to aid forensic experts and investigators, KAPE facilitates the collection and analysis of digital evidence from Windows-based systems. Developed and maintained by `Kroll` (previously known as `Magnet Forensics`), KAPE is celebrated for its comprehensive collection features, adaptability, and intuitive interface. The diagram below illustrates KAPE's operational flow.

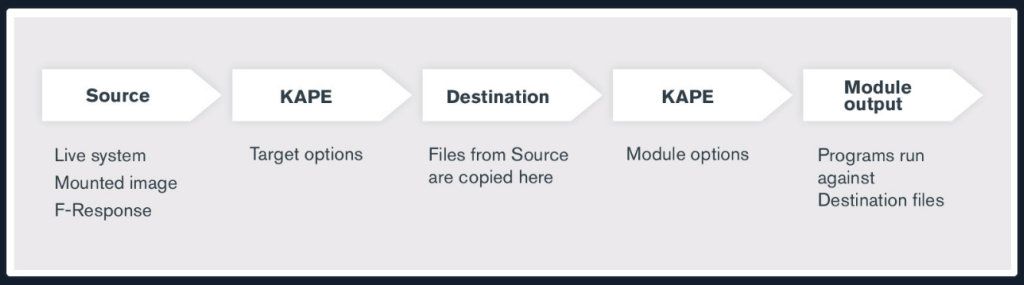
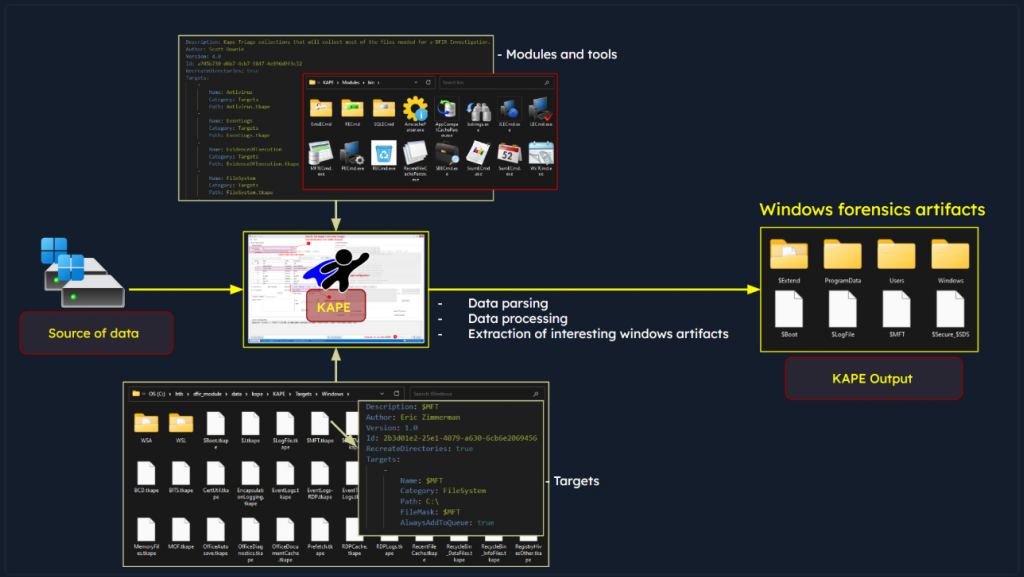
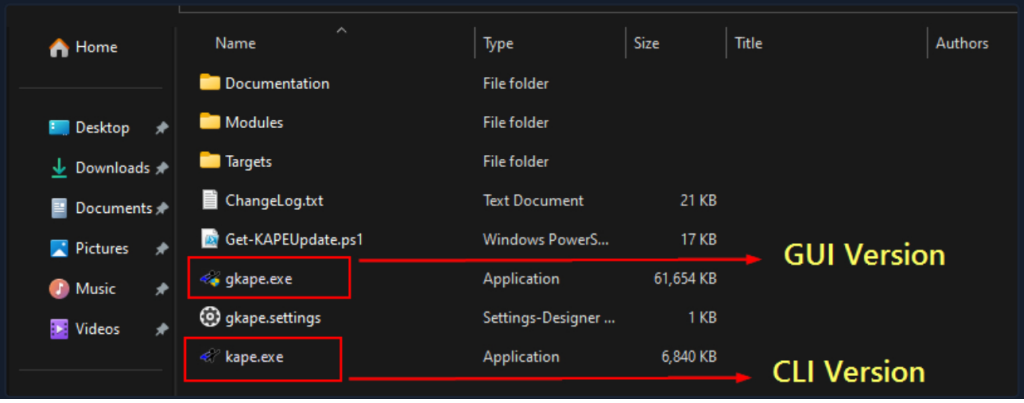


Image reference: https://ericzimmerman.github.io/KapeDocs/#!index.md

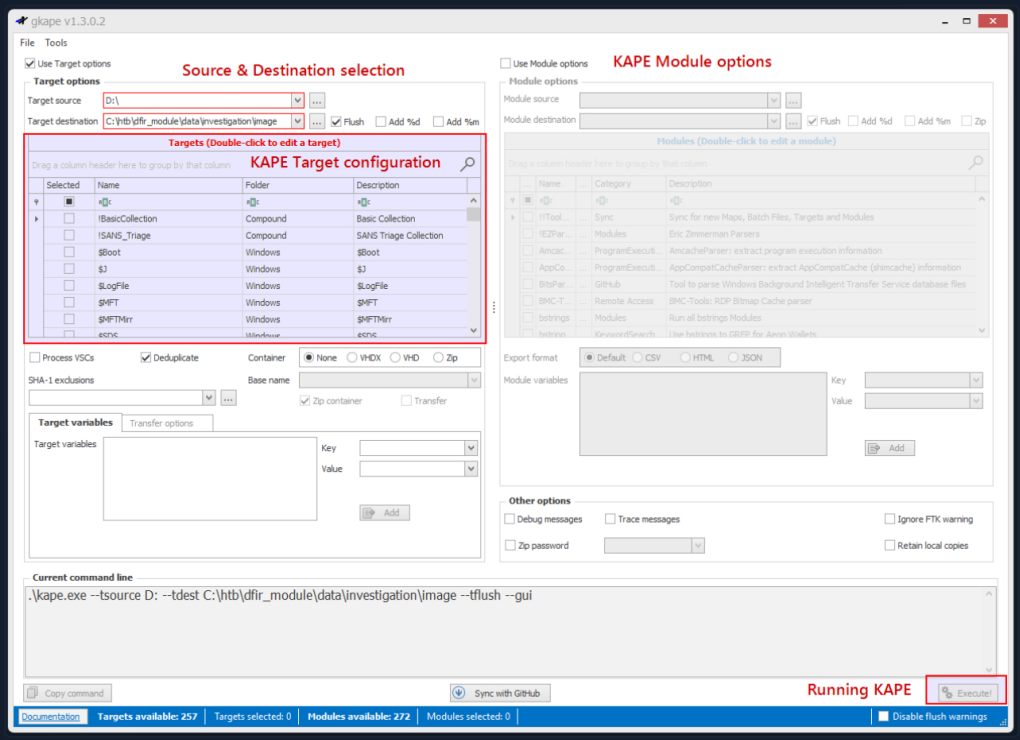
KAPE operates based on the principles of `Targets` and `Modules`. These elements guide the tool in processing data and extracting forensic artifacts. When we feed a source to KAPE, it duplicates specific forensic-related files to a designated output directory, all while maintaining the metadata of each file.



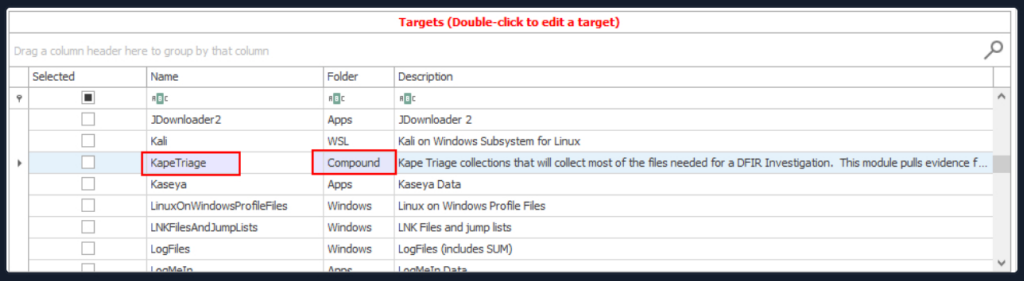
After downloading, let's unzip the file and launch KAPE. Within the KAPE directory, we'll notice two executable files: `gkape.exe` and `kape.exe`. KAPE provides users with two modes: CLI (`kape.exe`) and GUI (`gkape.exe`).



Let's opt for the GUI version to explore the available options more visually.

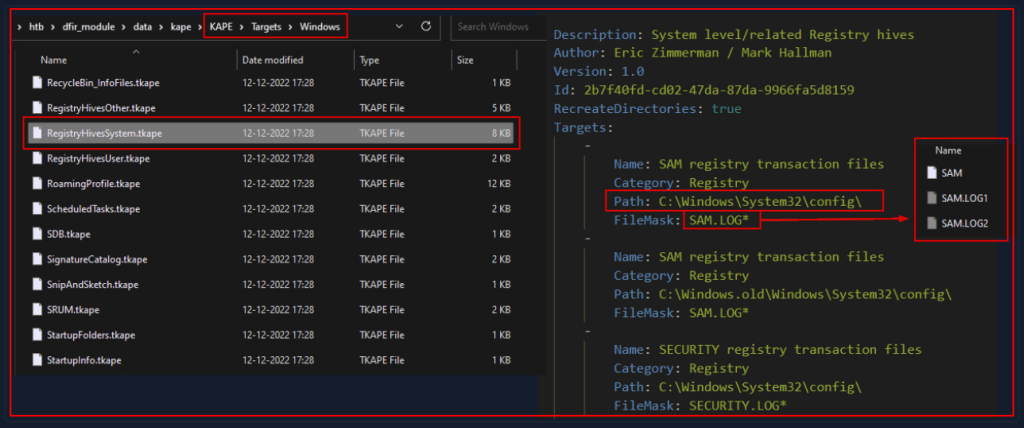


The crux of the process lies in selecting the appropriate target configurations.

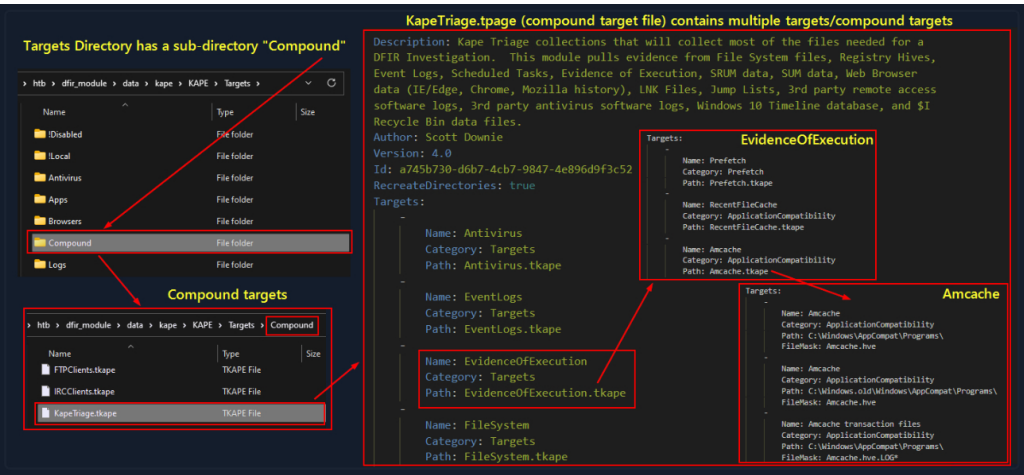


In KAPE's terminology, `Targets` refer to the specific artifacts we aim to extract from an image or system. These are then duplicated to the output directory.

KAPE's target files have a `.tkape` extension and reside in the `<path to kape>\KAPE\Targets` directory. For instance, the target `RegistryHivesSystem.tkape` in the screenshot below specifies the locations and file masks associated with system-related registry hives. In this target configuration, `RegistryHivesSystem.tkape` contains information to collect the files with file mask `SAM.LOG*` from the `C:\Windows\System32\config` directory.

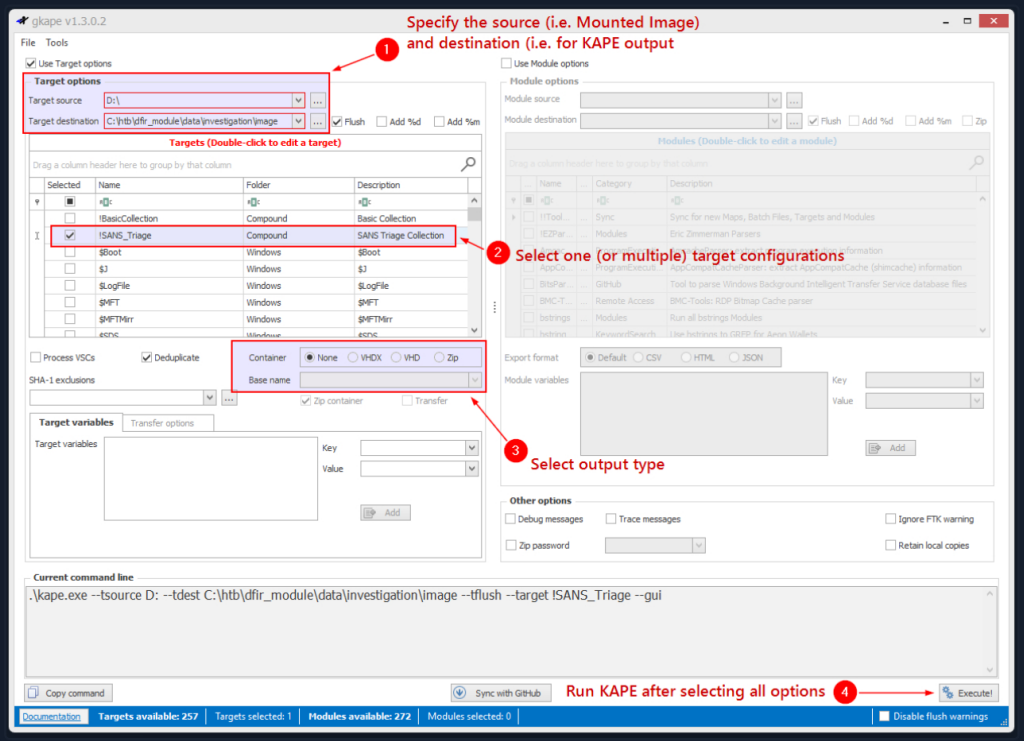


KAPE also offers `Compound Targets`, which are essentially amalgamations of multiple targets. This feature accelerates the collection process by gathering multiple files defined across various targets in a single run. The `Compound` directory's `KapeTriage` file provides an overview of the contents of this compound target.



Let's specify our source (in our scenario, it's `D:\`) and designate a location to store the harvested data. We can also determine an output folder to house the processed data from KAPE.

After configuring our options, let's hit the `Execute` button to initiate the data collection.



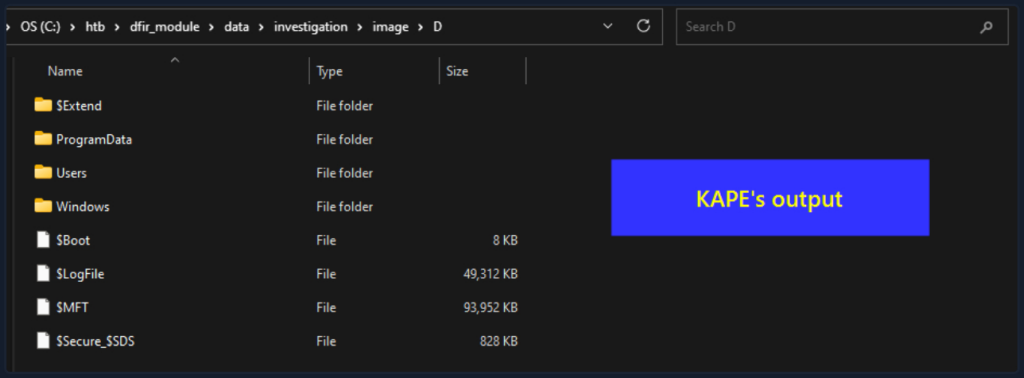
Upon execution, KAPE will commence the collection, storing the results in the predetermined destination.

Evidence Acquisition Techniques & Tools

```
KAPE version 1.3.0.2, Author: Eric Zimmerman, Contact: https://www.kroll.com/kape (kape@kroll.com)

KAPE directory: C:\htb\dfir_module\data\kape\KAPE
Command line:   --tsource D: --tdest C:\htb\dfir_module\data\investigation\image --target !SANS_Tri

System info: Machine name: REDACTED, 64-bit: True, User: REDACTED OS: Windows10 (10.0.22621)

Using Target operations
Found 18 targets. Expanding targets to file list...
Target ApplicationEvents with Id 2da16dbf-ea47-448e-a00f-fc442c3109ba already processed. Skipping!
Target ApplicationEvents with Id 2da16dbf-ea47-448e-a00f-fc442c3109ba already processed. Skipping!
Target ApplicationEvents with Id 2da16dbf-ea47-448e-a00f-fc442c3109ba already processed. Skipping!
Target ApplicationEvents with Id 2da16dbf-ea47-448e-a00f-fc442c3109ba already processed. Skipping!
Target ApplicationEvents with Id 2da16dbf-ea47-448e-a00f-fc442c3109ba already processed. Skipping!
Found 639 files in 4.032 seconds. Beginning copy...
  Deferring D:\Windows\System32\LogFiles\WMI\RtBackup\EtwRTDefenderApiLogger.etl due to Unauthorize
  Deferring D:\Windows\System32\LogFiles\WMI\RtBackup\EtwRTDefenderAuditLogger.etl due to Unauthori
  Deferring D:\Windows\System32\LogFiles\WMI\RtBackup\EtwRTDiagLog.etl due to UnauthorizedAccessExc
  Deferring D:\Windows\System32\LogFiles\WMI\RtBackup\EtwRTDiagtrack-Listener.etl due to Unauthoriz
  Deferring D:\Windows\System32\LogFiles\WMI\RtBackup\EtwRTEventLog-Application.etl due to Unauthor
```

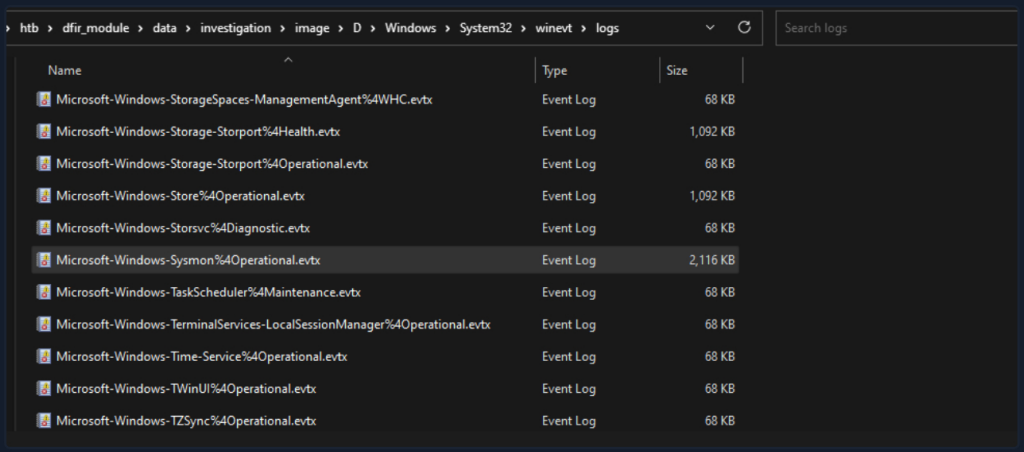
```
Deferring D:\Windows\System32\LogFiles\WMI\RtBackup\EtwRTEventLog-System.etl due to UnauthorizedA
Deferring D:\Windows\System32\LogFiles\WMI\RtBackup\EtwRTSgrmEtwSession.etl due to UnauthorizedAc
Deferring D:\Windows\System32\LogFiles\WMI\RtBackup\EtwRTUBPM.etl due to UnauthorizedAccessExcept
Deferring D:\Windows\System32\LogFiles\WMI\RtBackup\EtwRTWFP-IPsec Diagnostics.etl due to Unautho
Deferring D:\$MFT due to UnauthorizedAccessException...
Deferring D:\$LogFile due to UnauthorizedAccessException...
Deferring D:\$Extend\$UsnJrnl:$J due to NotSupportedException...
Deferring D:\$Extend\$UsnJrnl:$Max due to NotSupportedException...
Deferring D:\$Secure:$SDS due to NotSupportedException...
Deferring D:\$Boot due to UnauthorizedAccessException...
Deferring D:\$Extend\$RmMetadata\$TxfLog\$Tops:$T due to NotSupportedException...
Deferred file count: 17. Copying locked files...
Copied deferred file D:\Windows\System32\LogFiles\WMI\RtBackup\EtwRTDefenderApiLogger.etl to C:\h
Copied deferred file D:\Windows\System32\LogFiles\WMI\RtBackup\EtwRTDiagLog.etl to C:\htb\dfir_mo
Copied deferred file D:\Windows\System32\LogFiles\WMI\RtBackup\EtwRTDiagtrack-Listener.etl to C:\
Copied deferred file D:\Windows\System32\LogFiles\WMI\RtBackup\EtwRTEventLog-Application.etl to C
Copied deferred file D:\Windows\System32\LogFiles\WMI\RtBackup\EtwRTEventLog-System.etl to C:\htb
Copied deferred file D:\$MFT to C:\htb\dfir_module\data\investigation\image\D\$MFT. Hashing sourc
Copied deferred file D:\$LogFile to C:\htb\dfir_module\data\investigation\image\D\$LogFile. Hashi
Copied deferred file D:\$Extend\$UsnJrnl:$J to C:\htb\dfir_module\data\investigation\image\D\$Ext
Copied deferred file D:\$Extend\$UsnJrnl:$Max to C:\htb\dfir_module\data\investigation\image\D\$E
Copied deferred file D:\$Secure:$SDS to C:\htb\dfir_module\data\investigation\image\D\$Secure_$SD
Copied deferred file D:\$Boot to C:\htb\dfir_module\data\investigation\image\D\$Boot. Hashing sou
```

The output directory of KAPE houses the fruits of the artifact collection and processing. The exact contents of this directory can differ based on the artifacts selected and the configurations set. In our demonstration, we opted for the `!SANS_Triage` collection target configuration. Let's navigate to KAPE's output directory to inspect the harvested data.



From the displayed results, it's evident that the `$MFT` file has been collected, along with the `Users` and `Windows` directories.

It's worth noting that KAPE has also harvested the `Windows event logs`, which are nestled within the Windows directory sub-folders.



What if we wanted to perform artifact collection remotely and en masse? This is where EDR solutions and Velociraptor come into play.

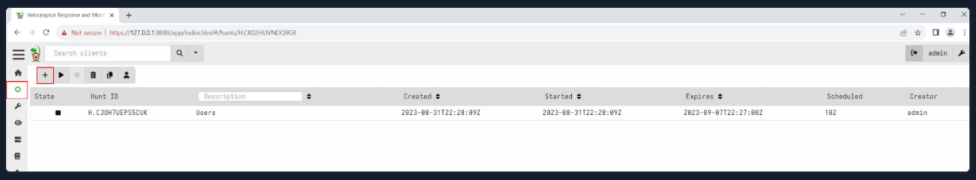
Endpoint Detection and Response (EDR) platforms offer a significant advantage for incident response analysts. They enable remote acquisition and analysis of digital evidence. For instance, EDR platforms can display recently executed

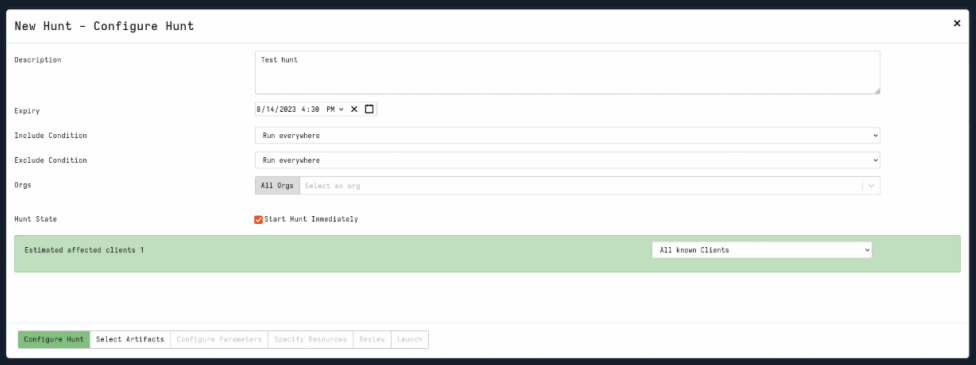
binaries or newly added files. Instead of sifting through individual systems, analysts can search for such indicators across the entire network. Another benefit is the capability to gather evidence, be it specific files or comprehensive forensic packages. This functionality expedites evidence collection and facilitates large-scale searching and collection.

Velociraptor is a potent tool for gathering host-based information using Velociraptor Query Language (VQL) queries. Beyond this, Velociraptor can execute `Hunts` to amass various artifacts. A frequently utilized artifact is the `Windows.KapeFiles.Targets`. While KAPE (Kroll Artifact Parser and Extractor) itself isn't open-source, its file collection logic, encoded in YAML, is accessible via the KapeFiles project. This approach is a staple in Rapid Triage.

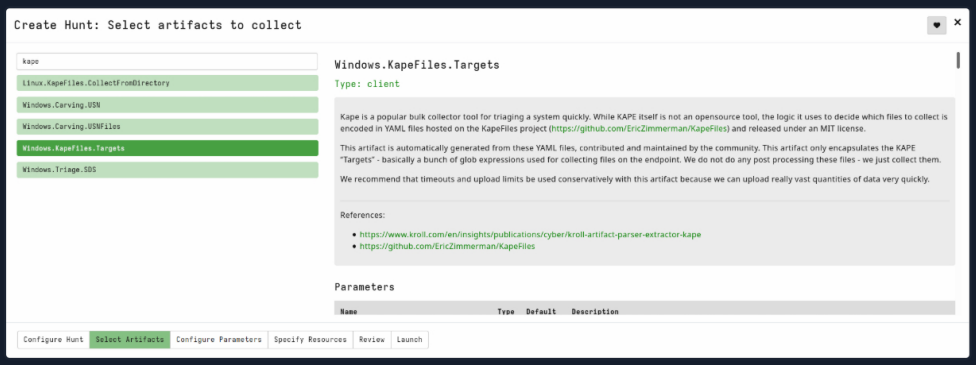
To utilize Velociraptor for KapeFiles artifacts:

- Initiate a new Hunt.

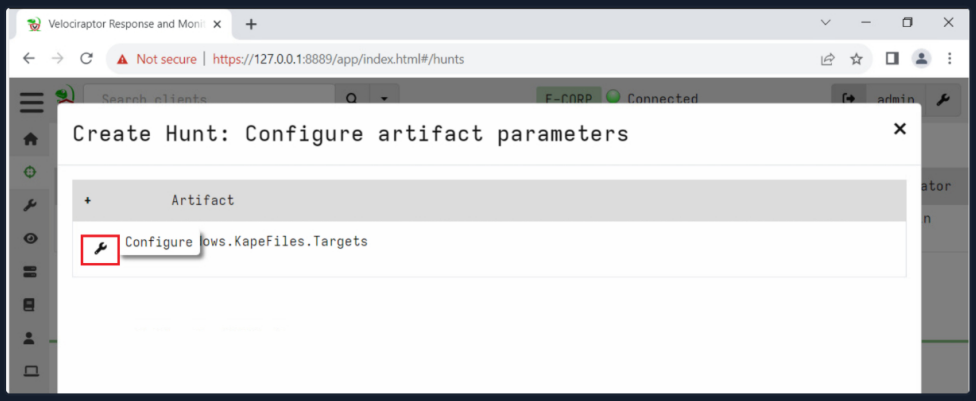


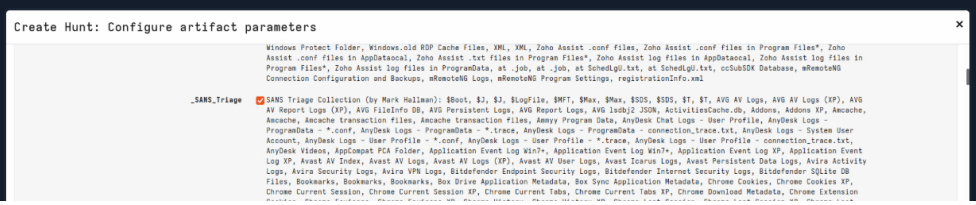


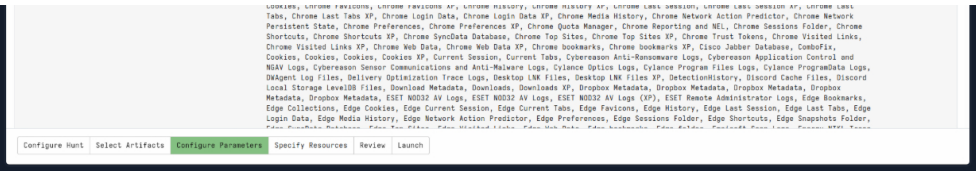
- Choose `Windows.KapeFiles.Targets` as the artifacts for collection.



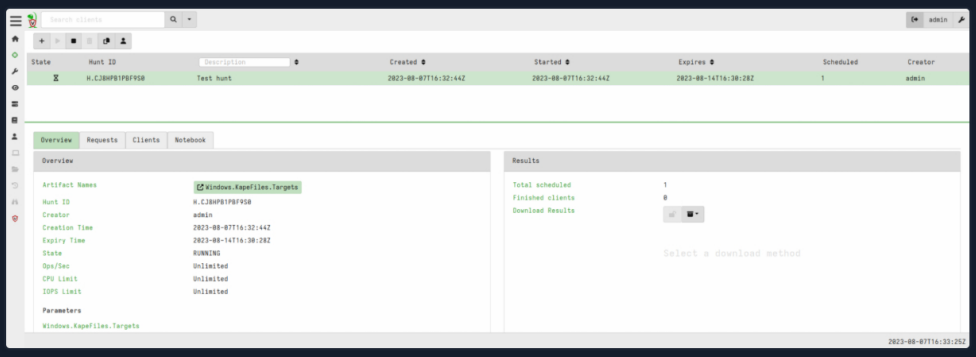
- Specify the collection to use.



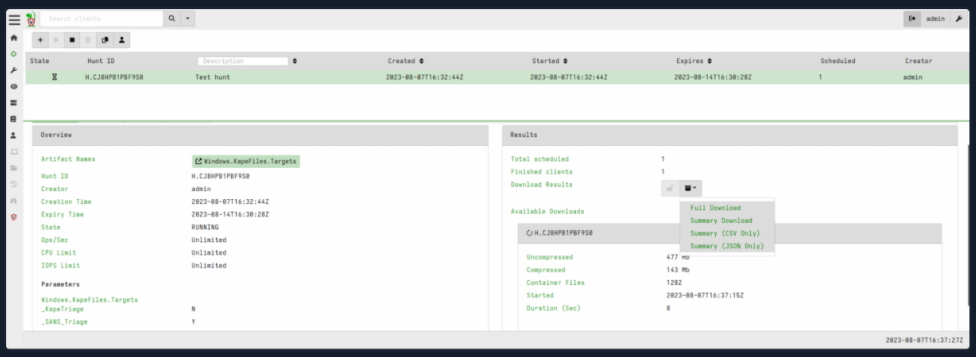




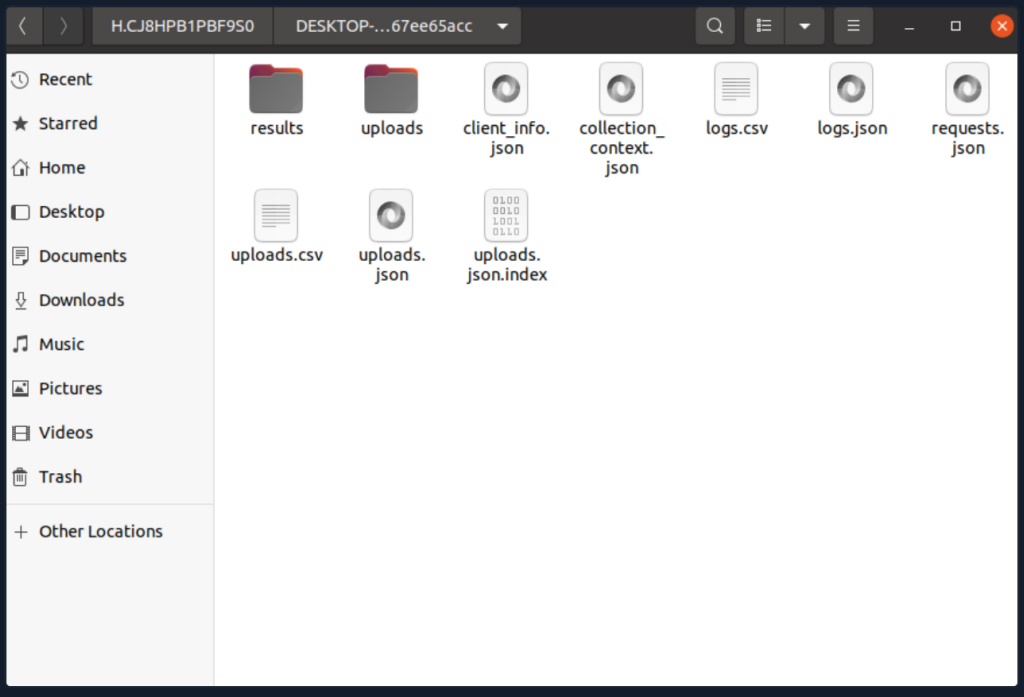
- Click on Launch to start the hunt.

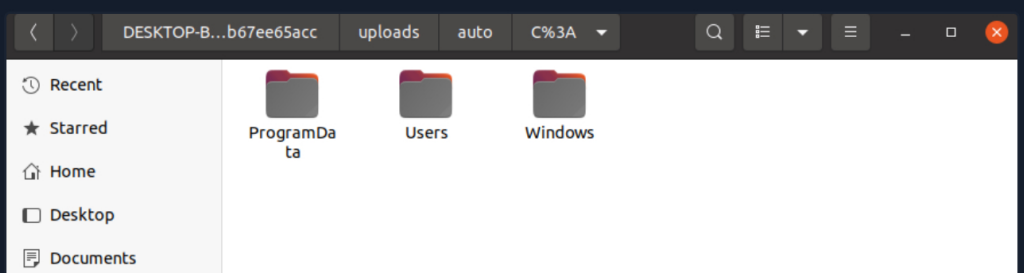


- Once completed, download the results.



Extracting the archive will reveal files related to the collected artifacts and all gathered files.

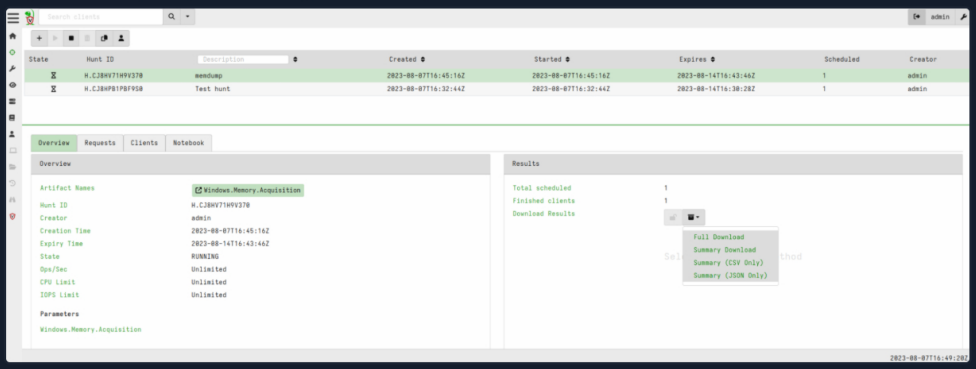




For remote memory dump collection using Velociraptor:

- Start a new Hunt, but this time, select the `Windows.Memory.Acquisition` artifact.

- After the Hunt concludes, download the resulting archive. Within, you'll find a file named `PhysicalMemory.raw`, containing the memory dump.



Extracting Network Evidence

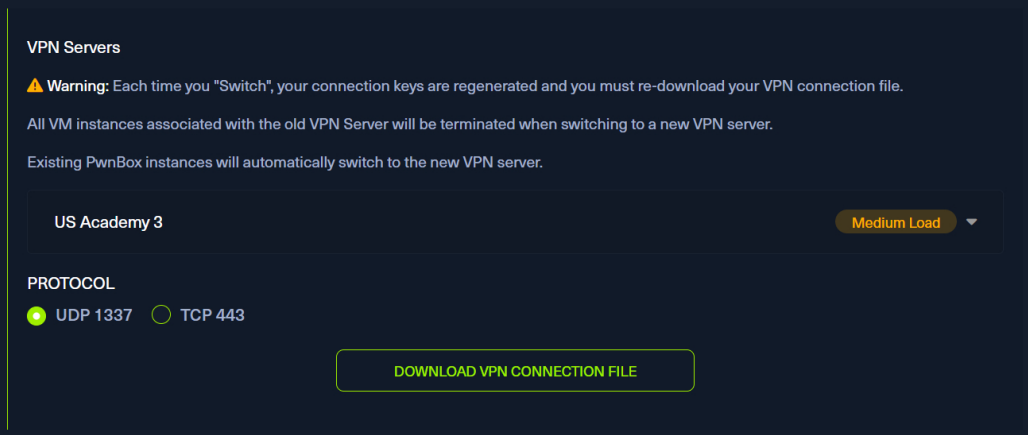
Throughout our exploration of the modules in the `SOC Analyst` path, we've delved extensively into the realm of network evidence, a fundamental aspect for any SOC analyst.

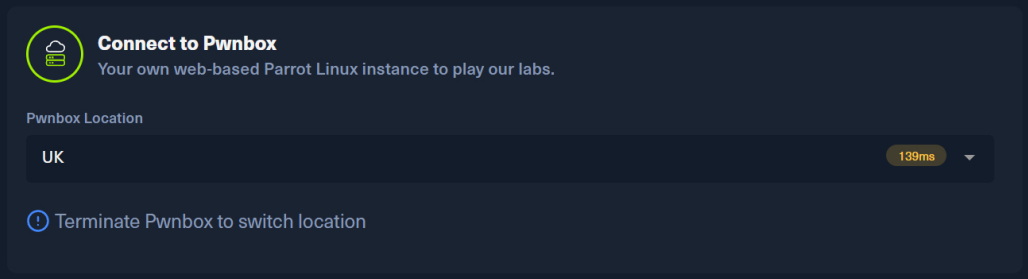
- First up, our `Intro to Network Traffic Analysis` and `Intermediate Network Traffic Analysis` modules covered `traffic capture analysis`. Think of traffic capture as a snapshot of all the digital conversations happening in our network. Tools like `Wireshark` or `tcpdump` allow us to capture and dissect these packets, giving us a granular view of data in transit.

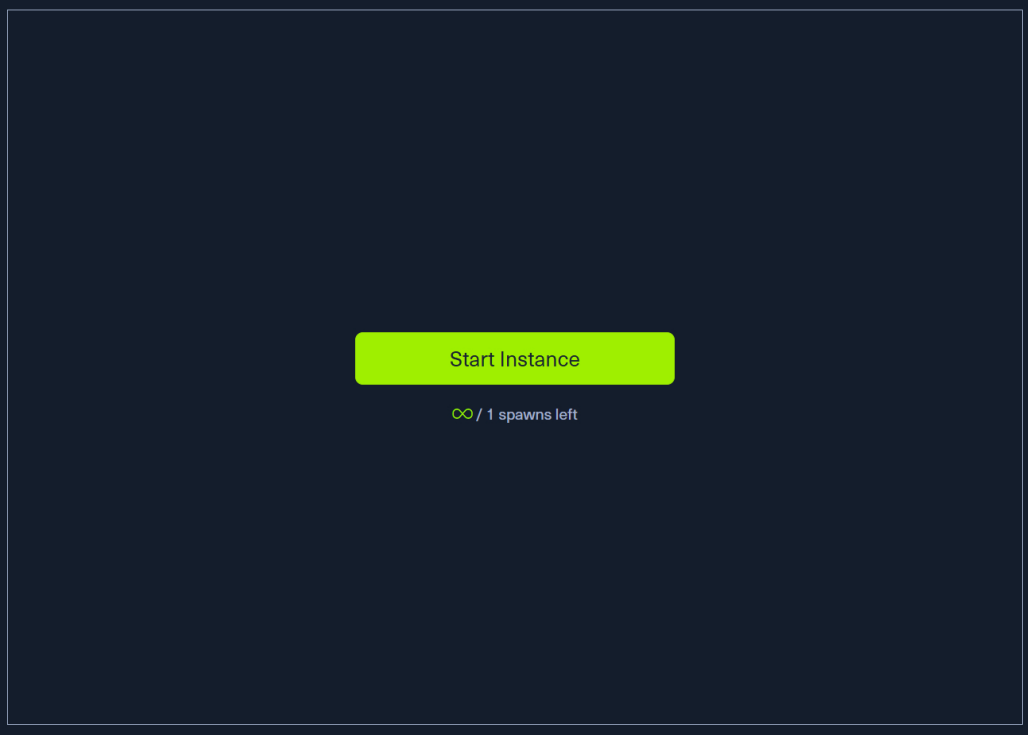
- Then, our `Working with IDS/IPS` and `Detecting Windows Attacks with Splunk` modules covered the usage of IDS/IPS-derived data. `Intrusion Detection Systems (IDS)` are our watchful sentinels, constantly monitoring network traffic for signs of malicious activity. When they spot something amiss, they alert us. On the other hand, `Intrusion Prevention Systems (IPS)` take it a step further. Not only do they detect, but they also take pre-defined actions to block or prevent those malicious activities.

- `Traffic flow` data, often sourced from tools like `NetFlow` or `sFlow`, provides us with a broader view of our network's behavior. While it might not give us the nitty-gritty details of each packet, it offers a high-level overview of traffic patterns.

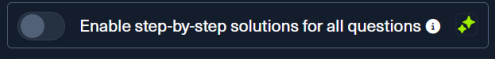
- Lastly, our trusty `firewalls`. These are not just barriers that block or allow traffic based on predefined rules. Modern firewalls are intelligent beasts. They can identify applications, users, and even detect and block threats. By analyzing firewall logs, we can uncover attempts to exploit vulnerabilities, unauthorized access attempts, and other malicious activities.

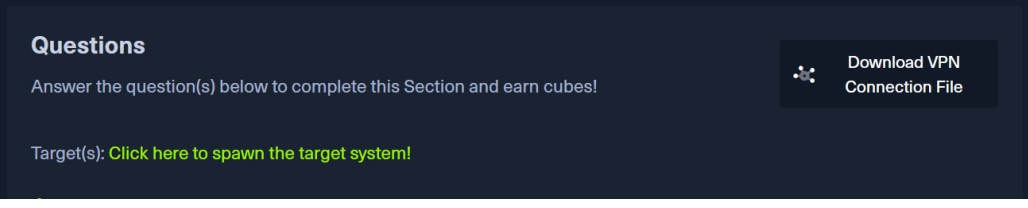






Waiting to start...





RDP to with user "Administrator" and password "password"

+ 1 Visit the URL "https://127.0.0.1:8889/app/index.html#/search/all" and log in using the credentials: admin/password. After logging in, click on the circular symbol adjacent to "Client ID". Subsequently, select the displayed "Client ID" and click on "Collected". Initiate a new collection and gather artifacts labeled as "Windows.KapeFiles.Targets" using the _SANS_Triage configuration. Lastly, examine the collected artifacts and enter the name of the scheduled task that begins with 'A' and concludes with 'g' as your answer.

AutoRunsToWinEventLog

Submit

Previous Next

Mark Complete & Next

Powered by HACKTHEBOX