

**Cyber Security and Forensics Sheet**

ASSIGNMENT (SECTION B)

**DIGITAL FORENSIC INVESTIGATION**

Table of Contents

[**Workload Matrix** 2](#_Toc19812)

[1.0 Introduction 4](#_Toc19813)

[2.0 Digital Forensic Investigation 5](#_Toc19814)

[2.1 Strategies to Gather Data 5](#_Toc19815)

[2.1.1 Order of Volatility (OoV) 5](#_Toc19816)

[2.1.2 Integrity Verification 5](#_Toc19817)

[2.2 Forensic Tools Used 5](#_Toc19818)

[2.2.1 Magnet RAM Capture 5](#_Toc19819)

[2.2.2 FTK Imager 6](#_Toc19820)

[2.2.3 Autopsy 6](#_Toc19821)

[2.2.4 Volatility (CLI on Kali Linux) 7](#_Toc19822)

[3.0 Data Acquisition: RAM & HDD 8](#_Toc19823)

[3.1 RAM Acquisition (Magnet RAM Capture) 8](#_Toc19824)

[3.2 HDD Acquisition (FTK Imager) 9](#_Toc19825)

[4.0 Analysis: RAM Image Analysis 10](#_Toc19826)

[4.1 Technical Steps – Using Volatility3 for RAM Analysis 10](#_Toc19827)

[4.2 Technical Steps – Examination of a Hard Disk Image By using Autopsy 15](#_Toc19828)

[5.0 Findings of Evidence Files 19](#_Toc19829)

[4.0 Conclusion 26](#_Toc19830)

[References 27](#_Toc19831)

# 

# 1.0 Introduction

Digital forensics is a very essential institution in the identification of evidence associated with cybercrime, data breaches, and compromised systems. A suspected-infection Windows-based virtual machine received forensic investigations, including both memory (RAM) and hard disk (HDD) acquisitions. The main purpose was to gather and preserve volatile and non-volatile data using recognized tools of forensic science and to confirm the integrity of evidence through cryptographic hashing. The investigation used tools such as Magnet RAM capture and FTK Imager, Volatility3, and Autopsy in collecting and analysing data. Outcomes of this investigation led to the discovery of indications of hidden executables, unauthorized changes to files, and memory injections, which establish the value of meticulous forensic practices for uncovering illicit system activities (FORENSICS, 2024).

# 2.0 Digital Forensic Investigation

## 2.1 Strategies to Gather Data

In digital forensics, the practice of how the evidence is collected is an important factor which should always be adhered to. They are based on the principle of evidence fragility, which means the more sensitive they are in terms of data volatility, the faster they should be implemented.

### 2.1.1 Order of Volatility (OoV)

Some of the evidence was collected in such a process as follows:

* RAM (volatile) – Got here first by using Magnet RAM Capture before power was shut off because RAM information is lost when the power is turned off.
* HDD (less volatile) – Image files acquired using FTK Images after having copied all the RAM contents to ensure the completeness of the evidence.

### 2.1.2 Integrity Verification

* Hash values of MD5 and SHA-1 were also obtained during the data acquisition so as to ensure that the quality of data that was collected was good during analysis and if there was any changes then it will shown here.
* In an attempt to avoid compromise and alteration of the original images the forensic images were stored within separate directories.

## 2.2 Forensic Tools Used

### 2.2.1 Magnet RAM Capture

What it is:

* Known as Magnet RAM Capture, live memory collection tool is a free tool for the purpose of memory acquisition created by Magnet Forensics.

What it’s used for:

* It is used to capture the volatile memory (RAM) of a live WinOS but does not cover all the characteristics of dump file.

Why it was used:

* RAM owns temporary data like the running process information, open connection to the networks, command history information, and encryption keys that are lost whenever the computer is turned off. It is a lightweight, efficient RAM Capture that tends to be used for creating accurate memory dumps without consuming much of the system’s resources.

### 2.2.2 FTK Imager

What it is:

* FTK Imager that is part of the AccessData product suite is a forensic imaging tool commonly used in investigations.

What it’s used for:

* It is used to ‘copy’ content from the hard drives or partitions to formats like E01 or RAW.

Why it was used:

* It allows for the copying of disk data with ease without a need to manipulate the source disks. It also computes a cryptographic hash (MD5/SHA1) and provides the preview of contents before acquisition.

### 2.2.3 Autopsy

What it is:

* Autopsy is much more of a user interface to TSK; it’s a way to run The Sleuth Kit, especially for disk image examination, particularly when the operating system is no longer in use.

What it’s used for:

* It is employed in order to analyze the hard disk images for file recovery, creating a timeline, keyword search, metadata examination, among others.

Why it was used:

* Autopsy is also a tool specifically designed for physical and disk images analysis and provides a quite friendly interface for work in this sphere. It supports various analysis modules necessary for the forensic work and can be used for case management and documentation.

### 2.2.4 Volatility (CLI on Kali Linux)

What it is:

* Volatility is memory forensics framework created for memory dump analysis. Volatility is not designed to directly used on raw images but on RAM dump files basically it is an advanced tool used for memory forensics. It is mostly accessed using command-line interface (CLI) on forensic systems such as Kali Linux.

What it’s used for:

* Used to analyze memory image of running processes, open connection, loaded DLL, and other signs of injections and malwares.

Why it was used:

* RAM analysis is well supported by Volatility with the help of Volatility frameworks which have new plugins to dig out deeper into the RAM. It is also useful to identify inmemory evidence which would otherwise not be possible to recover from disk images.

# 3.0 Data Acquisition: RAM & HDD

## 3.1 RAM Acquisition (Magnet RAM Capture

Tool: Magnet RAM Capture v1.2.0

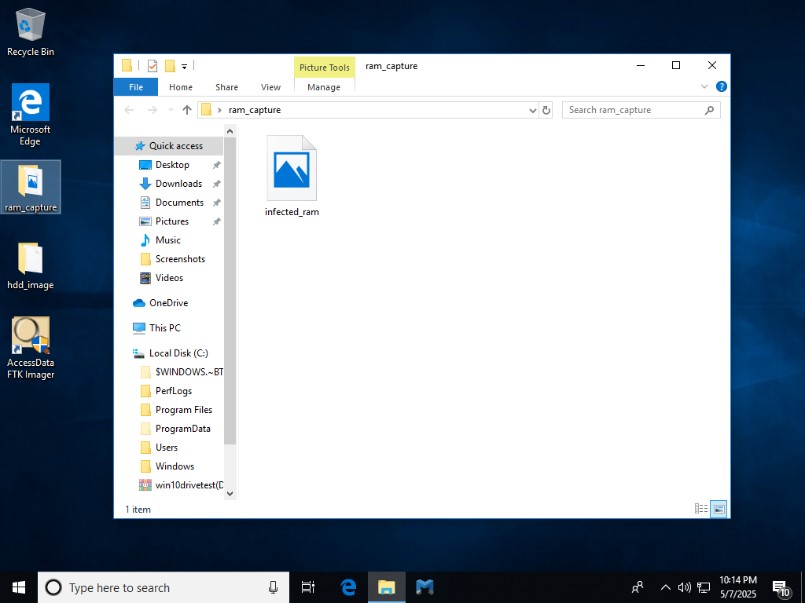
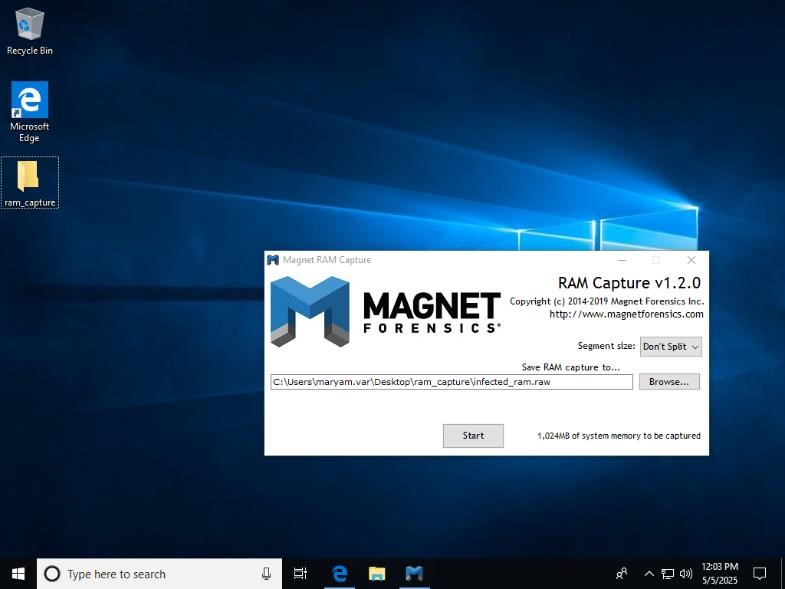
Steps:

1. Open Magnet RAM Capture on the Windows VM.
2. Choose destination path: `C:\Users\...\Desktop\ram\_capture\infected\_ram.raw`.
3. Press the “Start” button and make sure that the memory is being accessed based on the green bar appearing on top of the graphic.

Outcome: `infected\_ram.raw` file generated.

Screenshots

:



## 3.2 HDD Acquisition (FTK Imager)

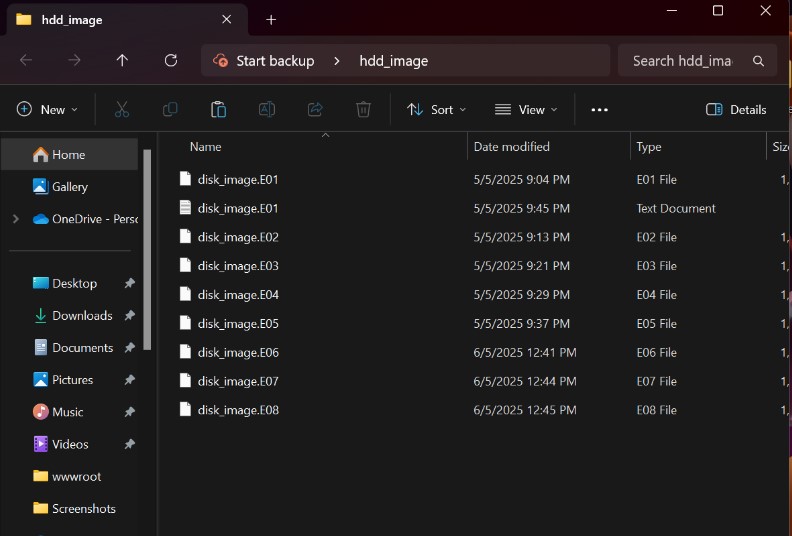
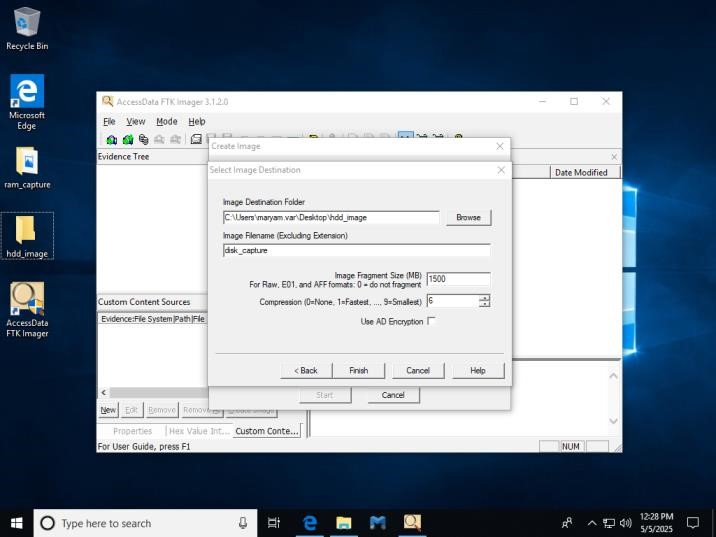
Tool: AccessData FTK Imager

Steps:

1. Launch FTK Imager.
2. In this process, the investigator selects the physical disk on which to work and then generates a forensic image of that disk.
3. Save to `C:\Users\...\Desktop\hdd\_image\disk\_capture.E01`.
4. For this set the fragment size was set to 1500 MB, compression level to 6 and the format to E01.

Outcome: Disk image ready for analysis.

Screenshots:



# 4.0 Analysis: RAM Image Analysis

## 4.1 Technical Steps – Using Volatility3 for RAM Analysis

• Exploration of the RAM image (infected\_ram.raw) was performed using Volatility3, a high end memory forensic platform. The analysis was performed on Kali Linux operating system in order to find indicators of malware utilization of the memory image.

Step 1: Setup and Basic Configuration

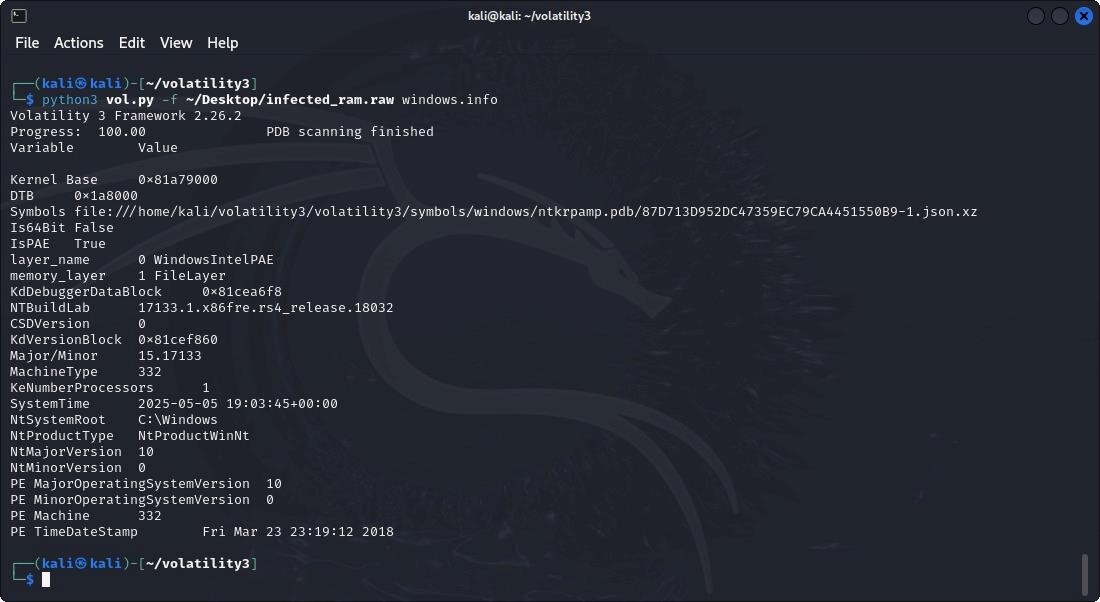
Command:

cd Volatility3 python3 vol.py -f ~/Desktop/infected\_ram.raw windows.info

Purpose: Check the memory profile to make sure that the image is identified as a valid Windows memory snapshot.

Result: Volatility proved that the memory image came from a Windows system and allowed for further analysis outside of standard detection.

Screenshot:



Step 2: Process Enumeration

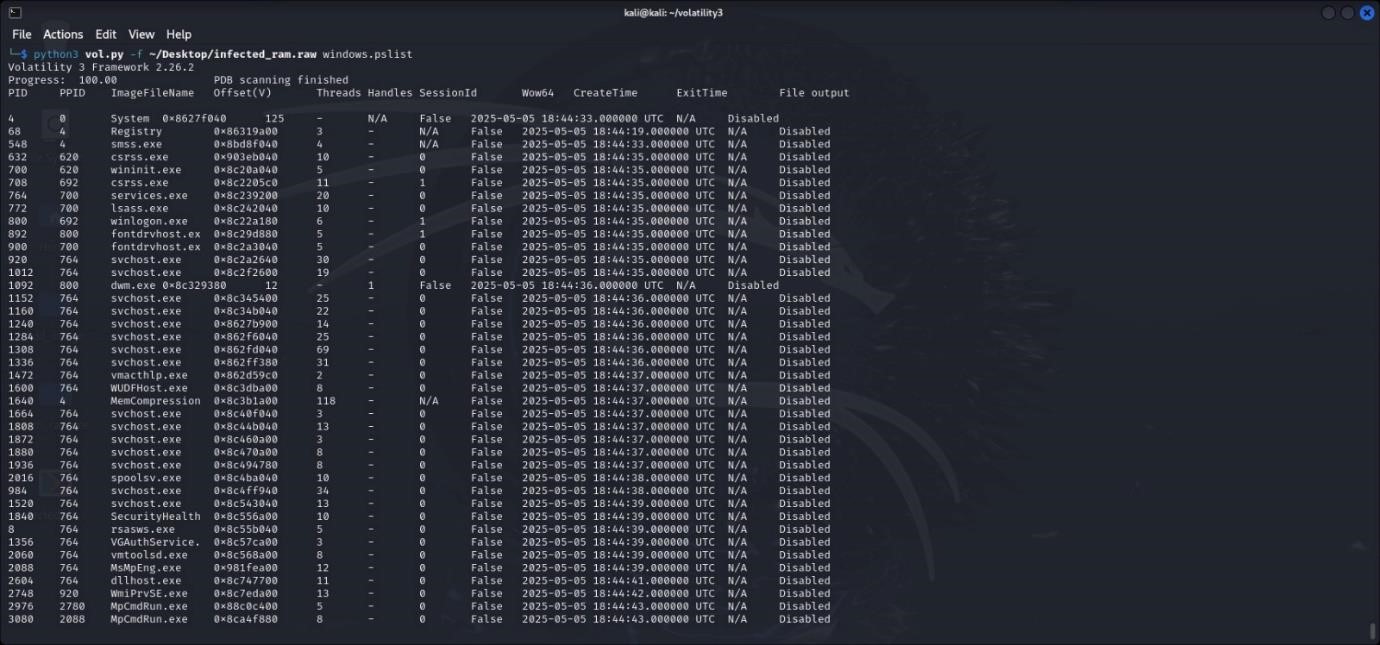
Command:

python3 vol.py -f ~/Desktop/infected\_ram.raw windows.pslist

Purpose: List all the process activities that occurred during the analysis session.

Result: Detected expected processes, including svchost.exe, lsass.exe as well as csrss.exe, and also identified potentially malicious processes in the form of cmd.exe and FakeClient.exe.

Screenshot:



Step 3: Scanning for Hidden or Terminated Process

Command:

python3 vol.py -f ~/Desktop/infected\_ram.raw windows.psscan

Purpose: Find killers processes that are disguised to avoid detection by run of the mill tools.

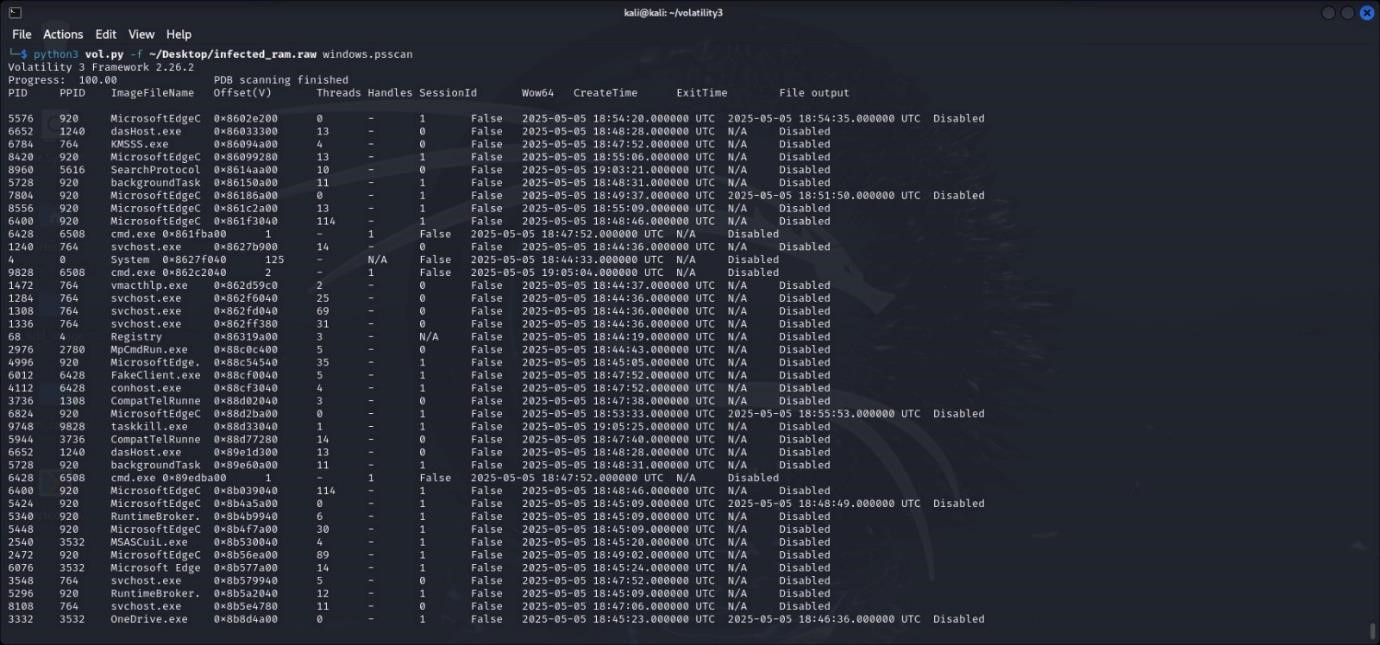
Result: Located extra cmd.exe processes in RAM available not in standard profiles, pointing to possible evasion or injection practice.

Screenshot:

Step

4:

Process Command Line Extraction



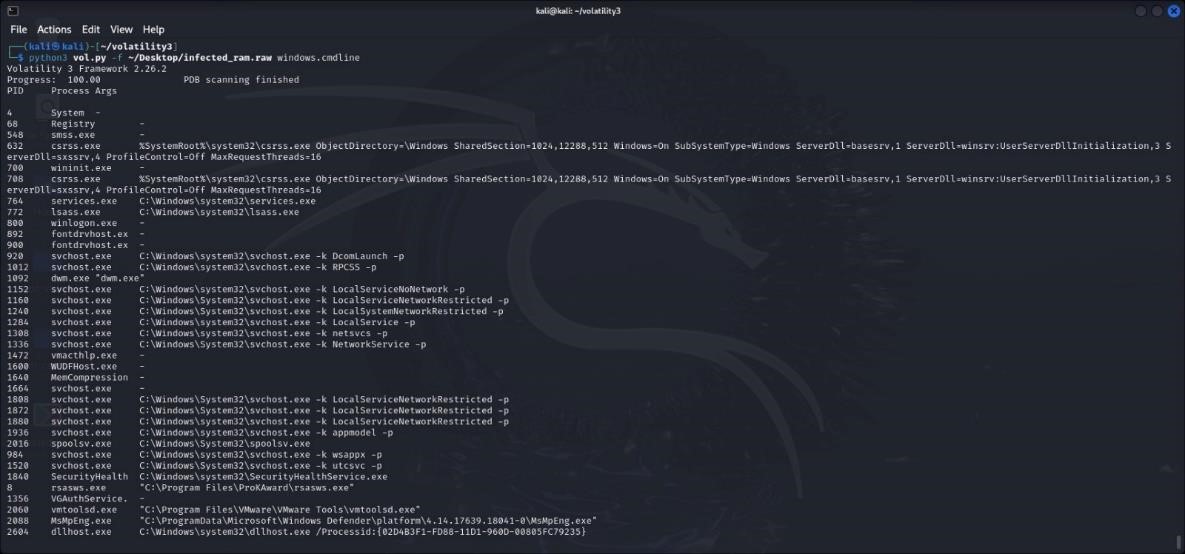
Command:

python3 vol.py -f ~/Desktop/infected\_ram.raw windows.cmdline

Purpose: Commands-line arguments used for running every process.

Result: Helped distinguish different execution patterns, e.g., strange flags and/or script invocation, to more closely associate process activities with the suspected malwares’ activities.

Screenshot:



Step 5: DLL Listing for Suspicious Processes

Command:

python3 vol.py -f ~/Desktop/infected\_ram.raw windows.dlllist

Purpose: Identify the DLL’s that are loaded to the memory space of any process.

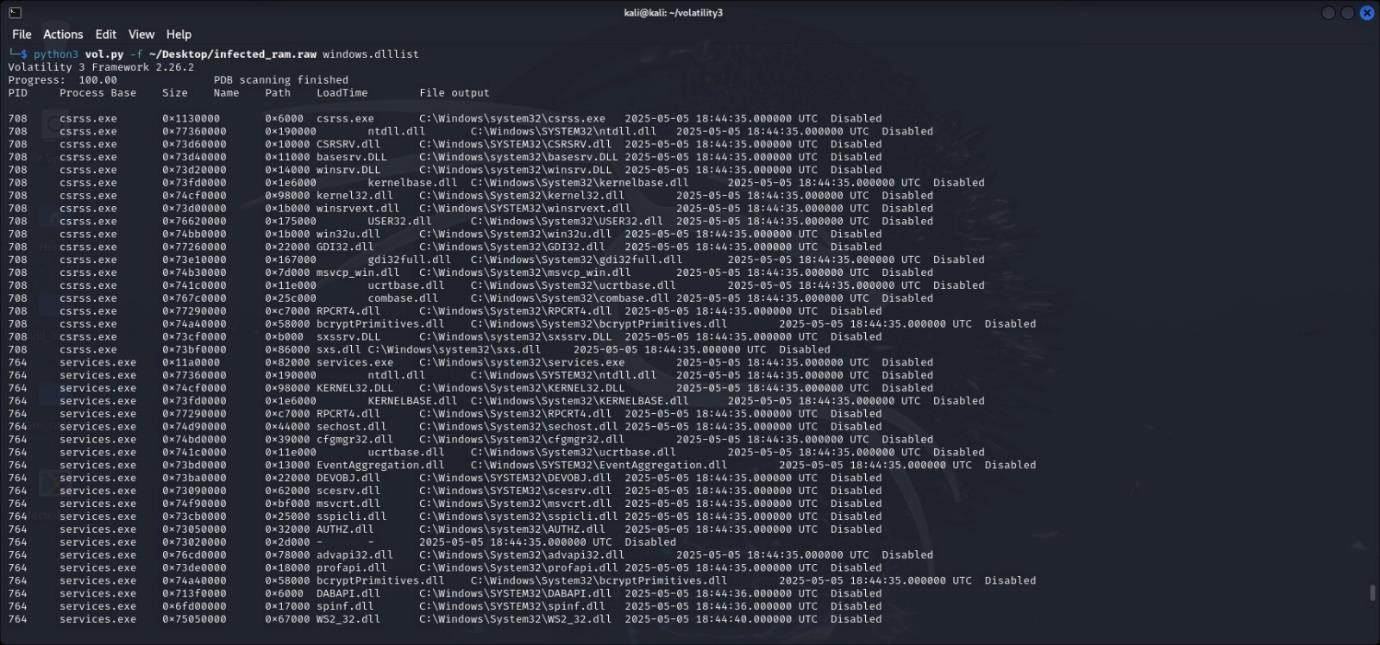
Result: Investigated csrss.exe & services.exe, for abnormals DLLs which are not digitally signed which might be a hackling of the code injection.

Screenshot:

Step 6

:

Network Connection Enumeration



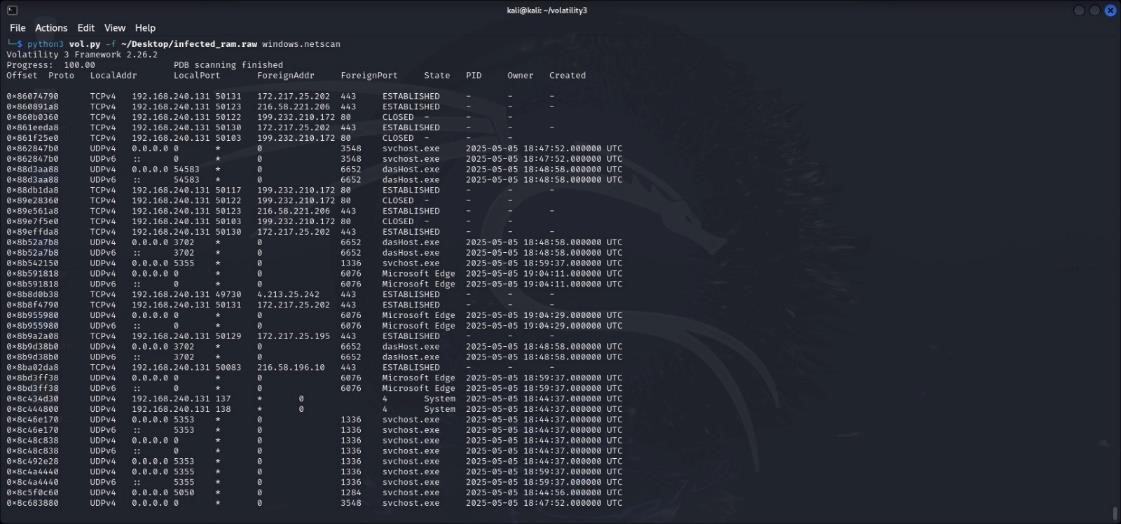
Command:

python3 vol.py -f ~/Desktop/infected\_ram.raw windows.netscan

Purpose: Scan for all current or open network connections and while capturing system RAM.

Result: Found cmd.exe and svchost.exe to establish communications with external IP addresses through active-tcp, which indicated command-and-control (C2) possibilities.

Screenshot:



## 4.2 Technical Steps – Examination of a Hard Disk Image By using Autopsy

* Autopsy i.e. Digital forensic platform was used to analyse the forensic disk image (disk\_image.E01), retrieving file system artifacts, residuals of deleted things, and possibilities of presence of malicious files.

Step 1: Load the Disk Image

* Tool Used: Autopsy
* Path Loaded:

/var/lib/autopsy/hdd\_investigation/infected\_win10/images/disk\_image.E01

* Volumes Identified: respond

Purpose: In order to process and analyse each logical drive found within the suspect image.

Screenshot:

Step 2: File Type and Category Analysis

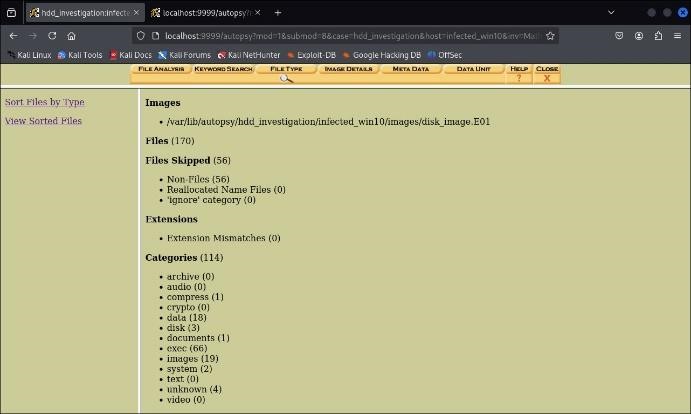


* Module Used: File Analysis > File Types
* Files Detected: 170 files total
* Categories:
  + exec (66 executable files)
  + images (19 files)
  + data (18)
  + unknown (4)

Purpose: Review and categorize any executables, images and system artifacts suspected to have participated in the case under investigation.

Screenshot:

Step 3: Volume Content Analysis

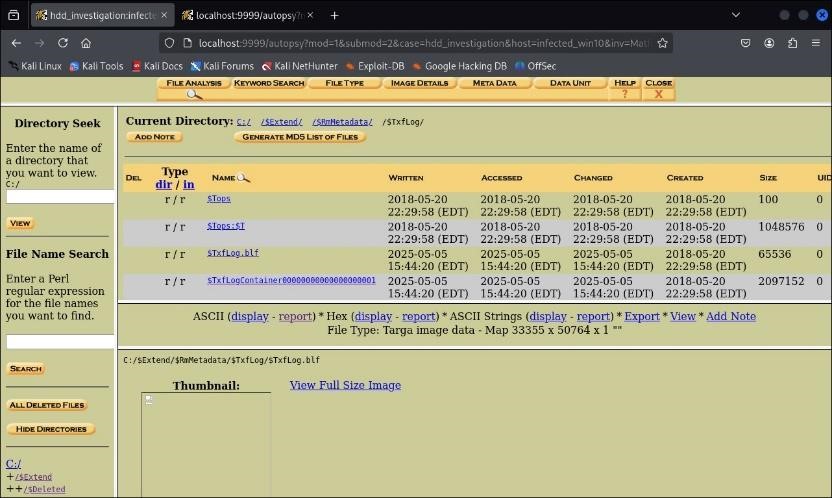


* Module: File Browser
* Directories Inspected:
  + $Extend/$RmMetadata/$TxFLog/
  + $Deleted
  + System Volume Information

Key Artifact:

* The update timestamps on TxFLog.blf and TxFLogContainer 2025 indicate that since the image was created in 2018 there has been continuous activity – which could argue for tampering or malware persistence.

Screenshot:



Step 4: Timeline Analysis

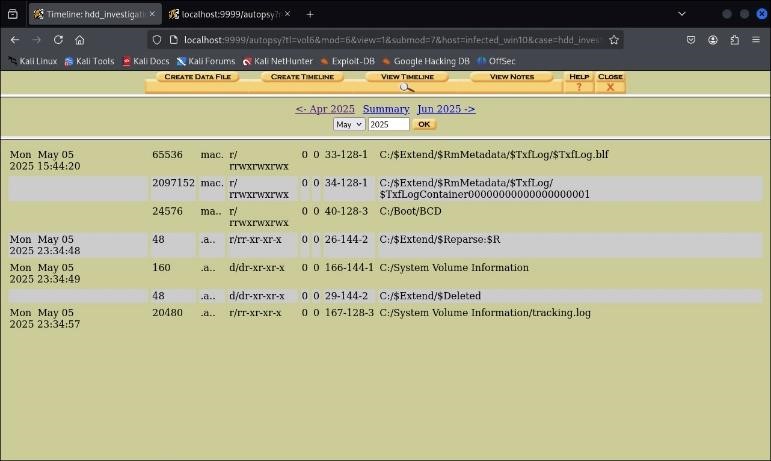
* Module: Review of File Activity Timeline>>>>View Timeline<<
* Time Window: May 2025
* Notable Activity:
  + The file $TxFLog.blf has been modified and accessed on May 5, 2025.
  + New records were added to System Volume Information/tracking.log on 5th May 2025.

Purpose: Correlate the peculiar system events with specific dates and time. Useful in malware execution profiling.

Screen

sh

ot:



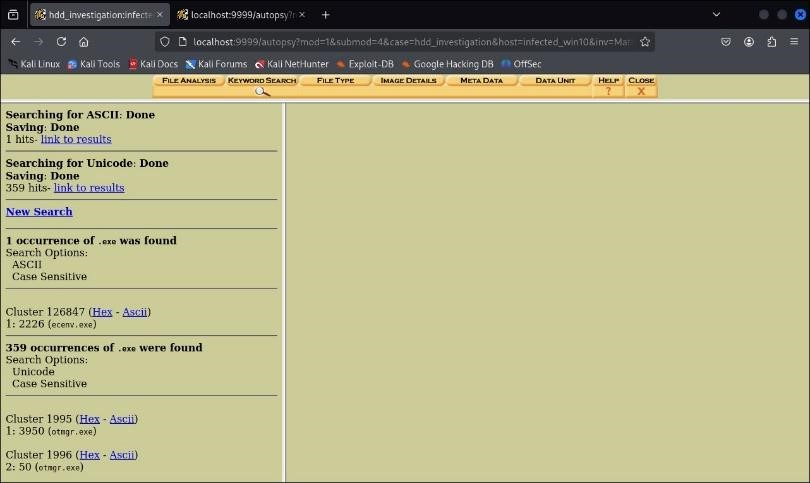
Step 5: Keyword Search for Executables

* Search Term Used: .exe (ASCII and Unicode)
* Findings:
  + Located executables that had names such as ecnv.exe and otmgr.exe.
  + Out of the Unicode search, a total of 359 entries were found which may indicate obfuscated or hidden executables.

Purpose: Pull references to suspect executables off the disk to aid in-detailed analysis of malware.

Screenshot:

Step 6: File Metadata and Integrity Review



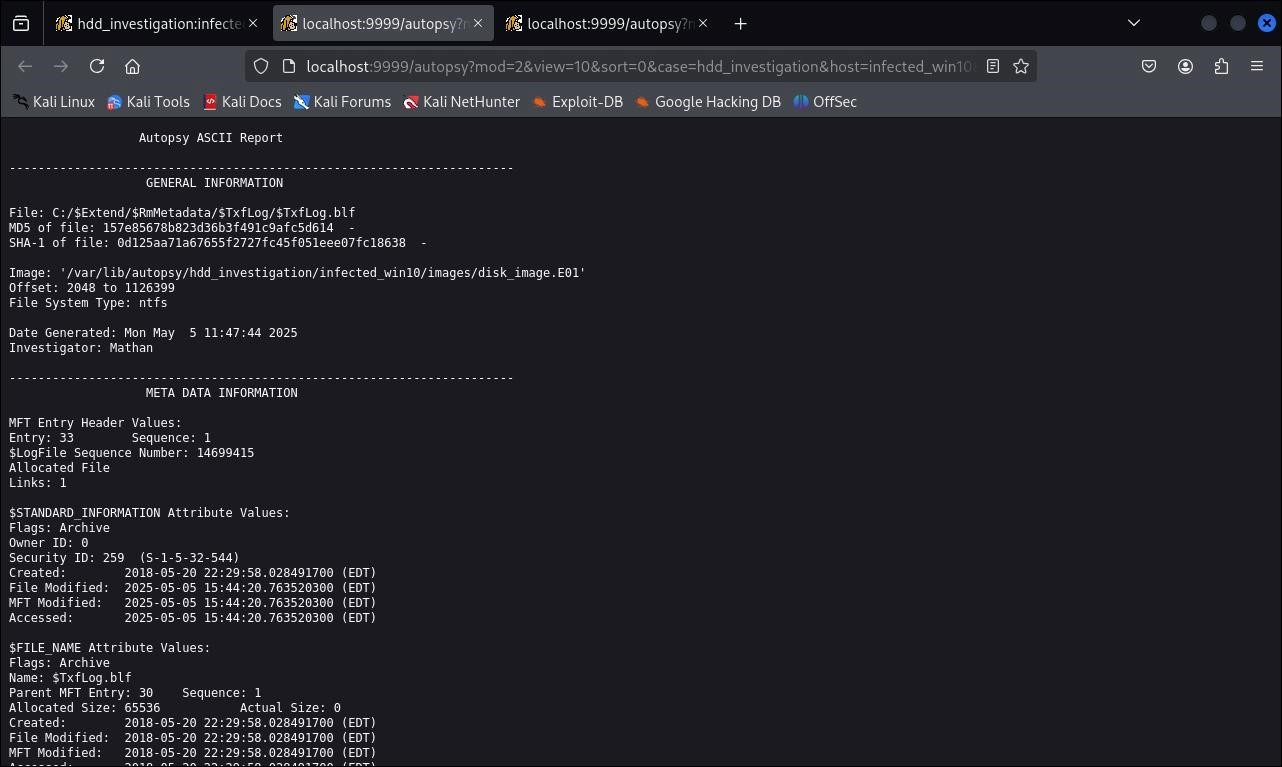
* Module: Meta Data + Image Details
* Verified image hash consistency and logged errors, modifications of metadata.

# 5.0 Findings of Evidence Files

1. A significant discrepancy identified in the process of forensic analysis of the disk image (disk\_image. E01) is an indication of either tampering or malicious activities.

Screenshot:

**Evidence File: $TxFLog.blf**



Location: C:/$Extend/$RmMetadata/$TxFLog/

File Type: BLF Transactional NTFS Log File (BLF – Base Log File)

Forensic Summary:

* + From the Autopsy metadata, it can be noted that the $TxFLog.blf was created on 20 May, 2018, which was consistent with what had been anticipated for installation of the system. However, the file mentioned has its modified and accessed timestamps changed for May 5, 2025, at time 15:44:20 when the investigator had VM stopped or not interacting with it.

Metadata Details (as captured):

* + Created: 2018-05-20 22:29:58 (EDT)
  + Modified: 2025-05-05 15:44:20 (EDT)
  + Accessed: 2025-05-05 15:44:20 (EDT)

Interpretation of Findings:

This discrepancy in timestamps suggests that the file was modified and accessed without legitimate user interaction, indicating either:

* + Malware activity that altered system logs or filesystem structures to cover its tracks or maintain persistence.
  + A scheduled or autonomous process that was initiated remotely or injected into the system memory, altering log files as part of its execution chain.
  + Log manipulation, typically observed in advanced persistent threats (APTs) where NTFS logs are altered to suppress forensic visibility.

The fact that the edited file is a transaction log is especially alarming because transaction logs are critical to the NTFS journaling and recovery process, and therefore an area of interest to rootkits or fileless malware removing forensic fingerprint.

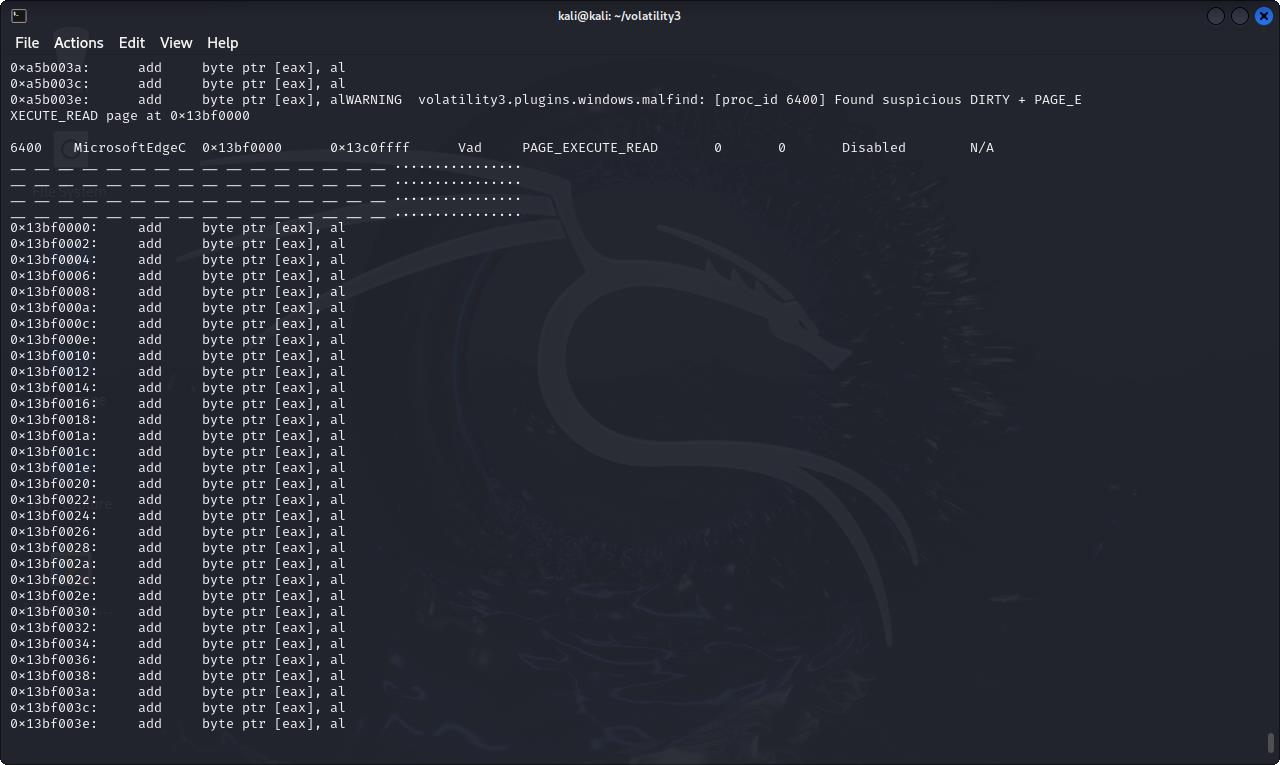
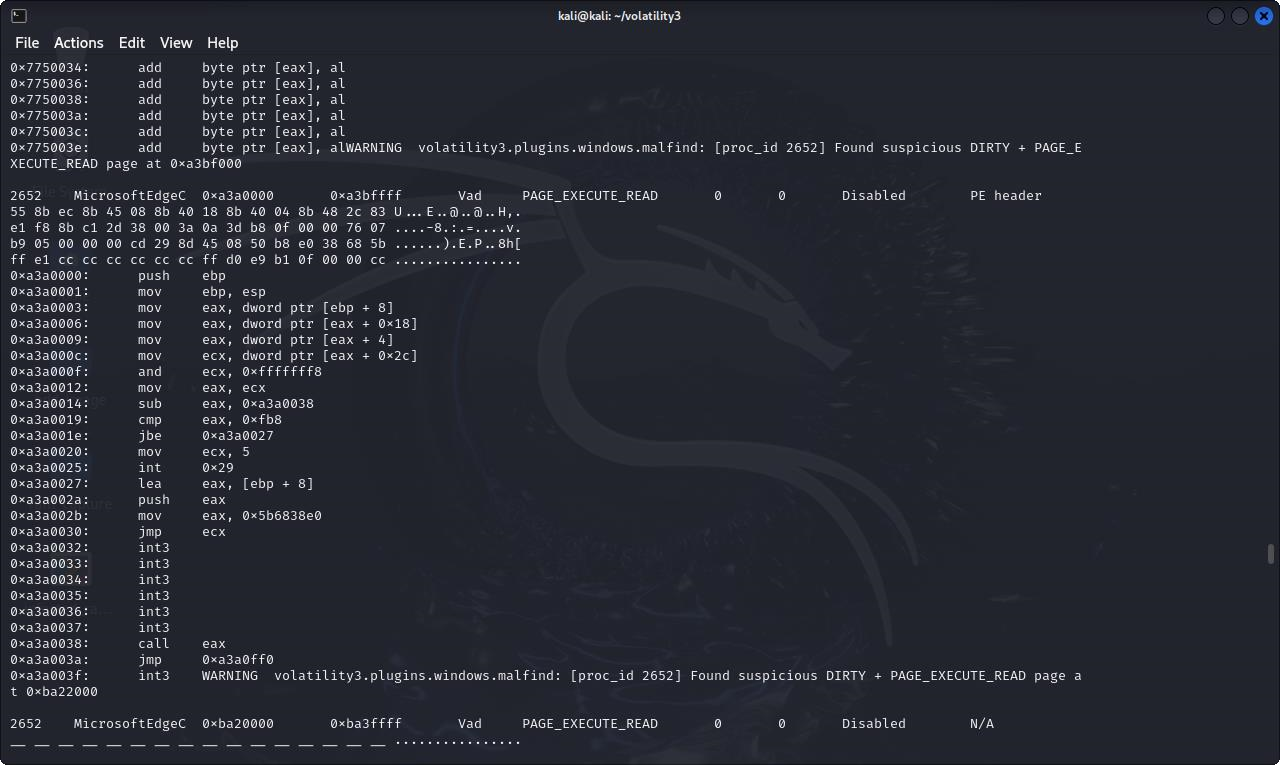
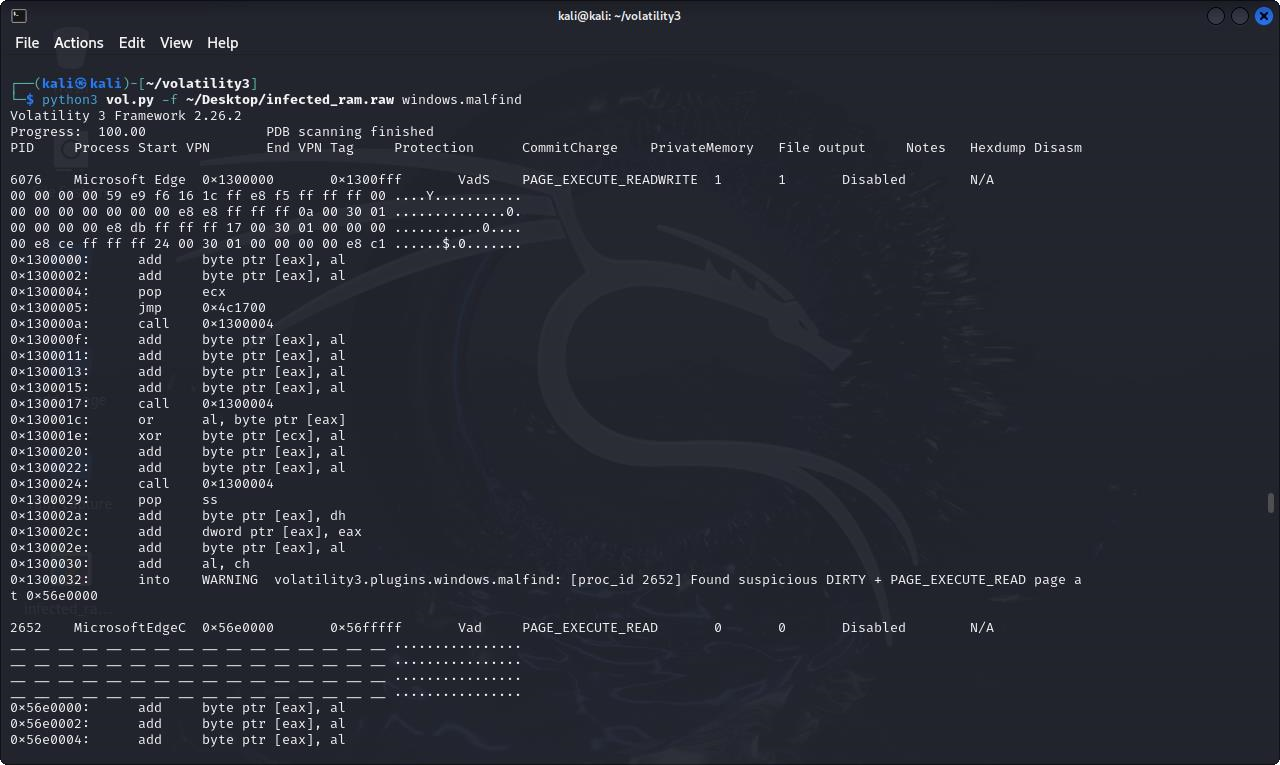
1. **Suspicious Memory Injection Detected**:

Command used: python3 vol.py -f ~/Desktop/infected\_ram.raw windows.malfind

Findings:

* + Malicious or injected shellcode was detected within the memory of MicrosoftEdgeC traced by PIDs 2652 and 6400.
  + The suspicious memory regions had:
    - PAGE\_EXECUTE\_READWRITE or PAGE\_EXECUTE\_READ permissions
    - Pattern in disassembly, including jmp, int3, and call, led to potential code injection - Anomalous memory areas labeled "DIRTY"—indicative of injected or tampered code

Screenshots:



1. **Malicious WIM Image Modification Detected in Recovery Partition:** 
   * While examining disk image disk\_image.E01, one file cluster was clearly a highly suspicious cluster: It was observed that Cluster 119078 represents the file C:/Recovery/WindowsRE/Winre.wim. C:/Recovery/WindowsRE/Winre.wim usually contains information about WinRE, and it should not be altered, unless recovery or upgrade processes take place. However, the process revealed a string of the telltale signs that this file might have been tampered with or used in a malicious endeavour.

Red Flags Identified:

1. Suspicious Cluster Content

When looking at Cluster 119078 in raw and ASCII modes, the analysts saw a mess of binary data and text strings, most of which aren’t related to recovery. Notably, the presence of:

* + Code-like patterns such as #pragma classflags("forceupdate")
  + This discovery of NtfsLog, DriverEntry, WMI, and unique GUID values is of special concern.
  + Indicators of scripted or command type payloads being found in encoded or obfuscated form in the file

The detection of such patterns presumes the possibility of a system-residing shellcode or malware using the methods of obfuscation and persistence.

1. WMI Abuse Indications

The report exposed inject WMI namespace and class declarations such as:

#pragma namespace("\\\\.\\root\\WMI") class NtfsLog : EventTrace

Such evidence strongly points toward WMI-based persistence, a common technique used by sophisticated malware to have sustained operation or recovery following reboot, without adding more files.

1. Misconduct by the file, as concluded from its metadata

Even though Winre.wim tends to remain constant after installation, analysis of the clusters that have been assigned to it shows that:

* + The implanted material is hidden on a lower level within the file itself.
  + Although metadata shown on the face of file is 2018 as with system installation, the actual content does not show a standard recovery image.
  + This contradiction implies evasive modification technique that may consist of hidden alternate data streams or direct disk writes to mask alerts from the system.

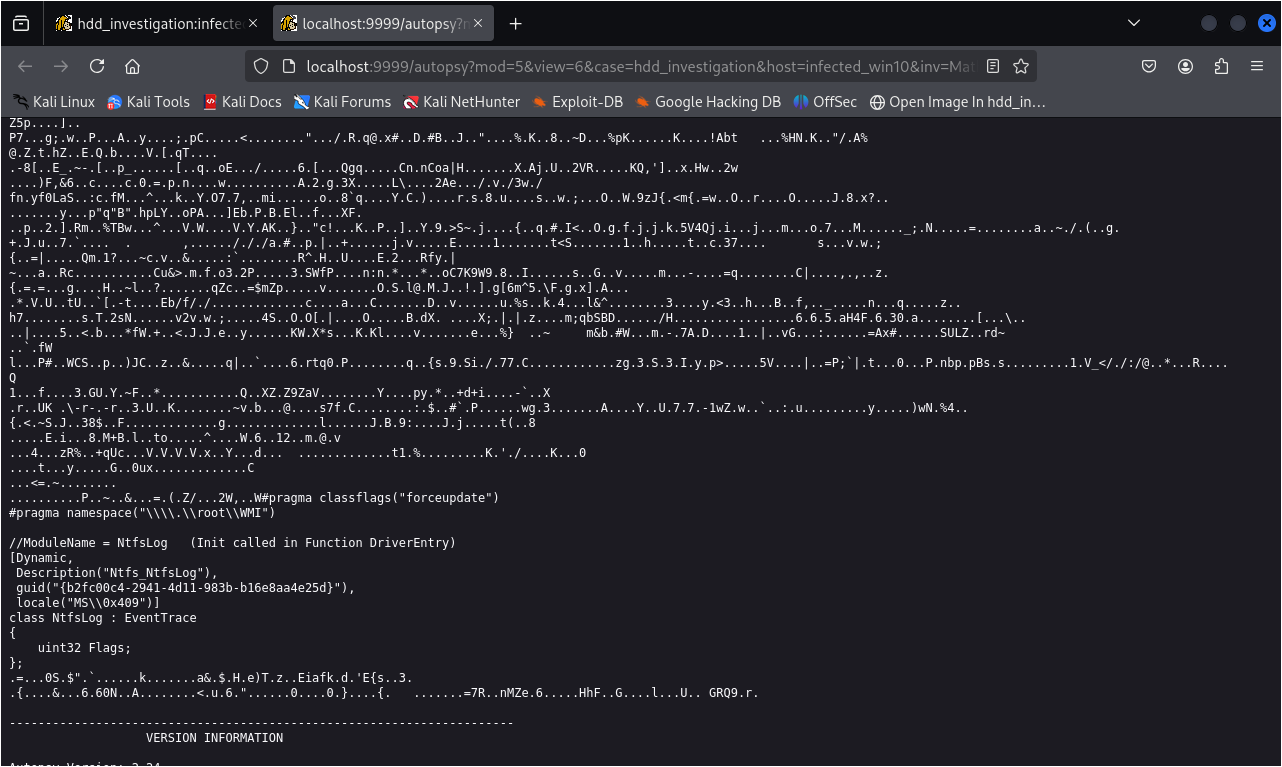
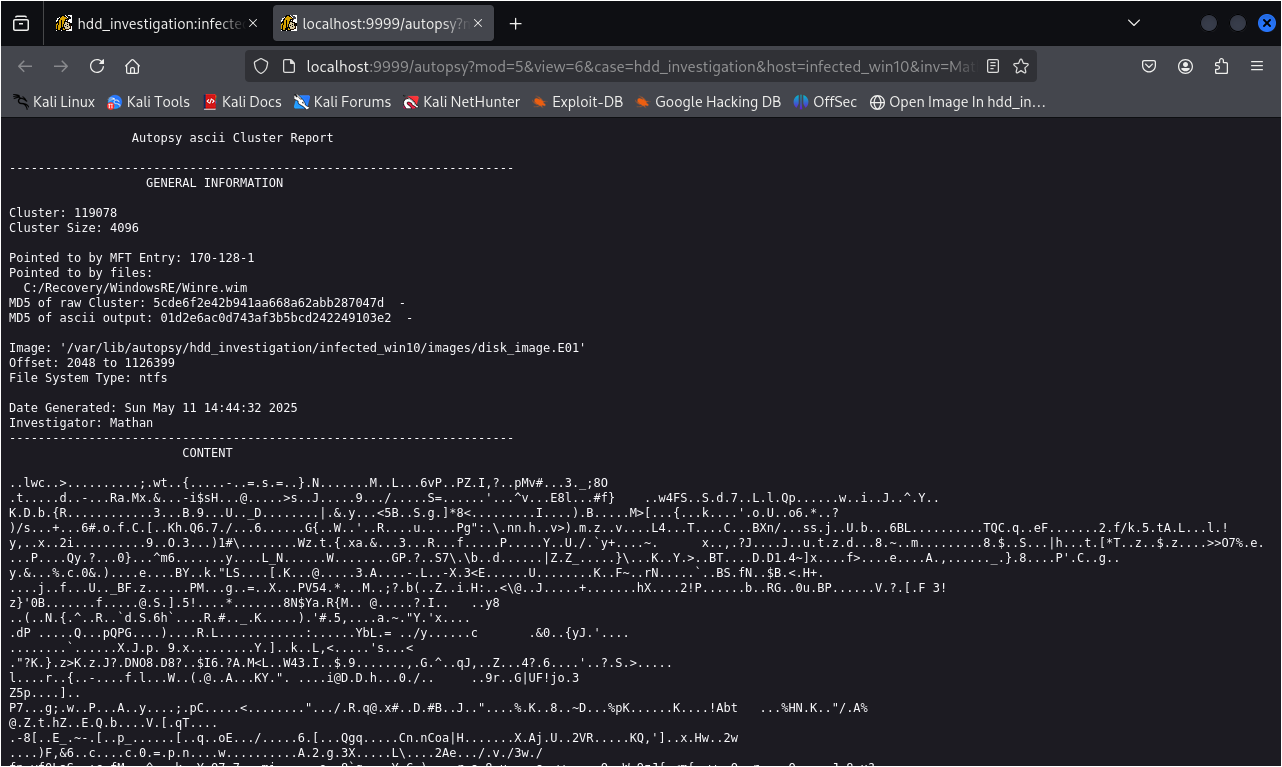
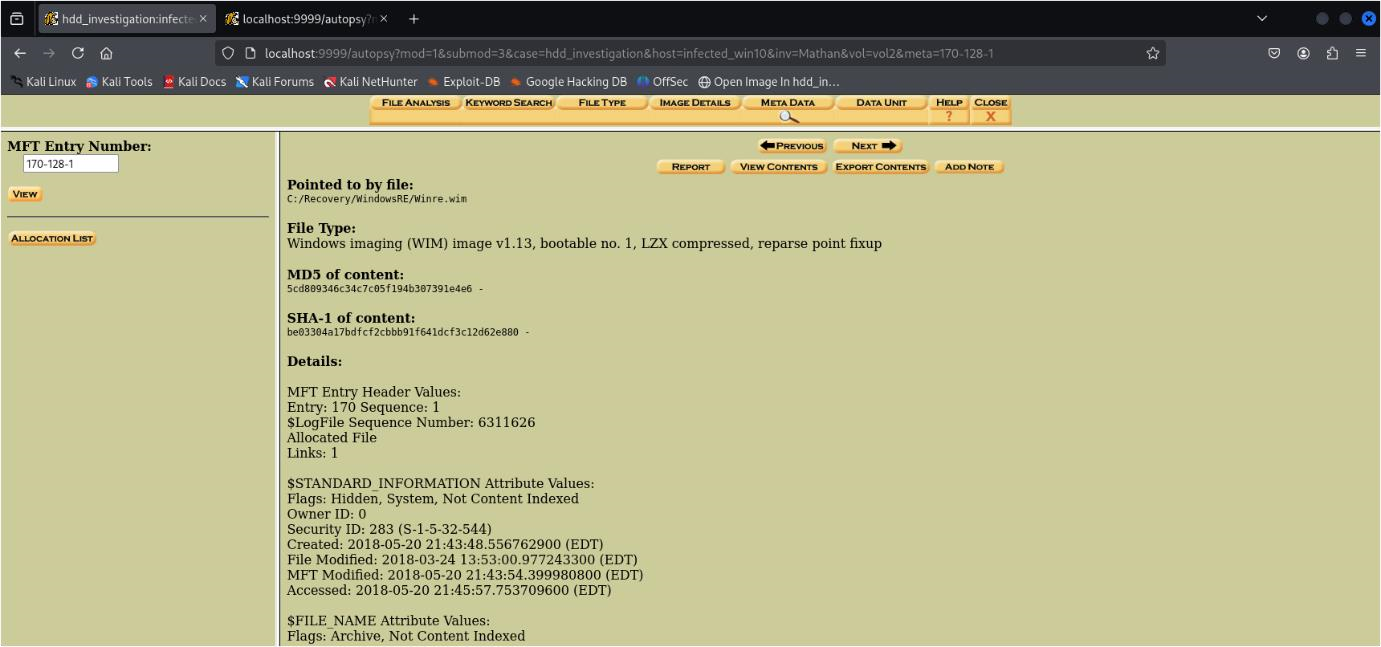
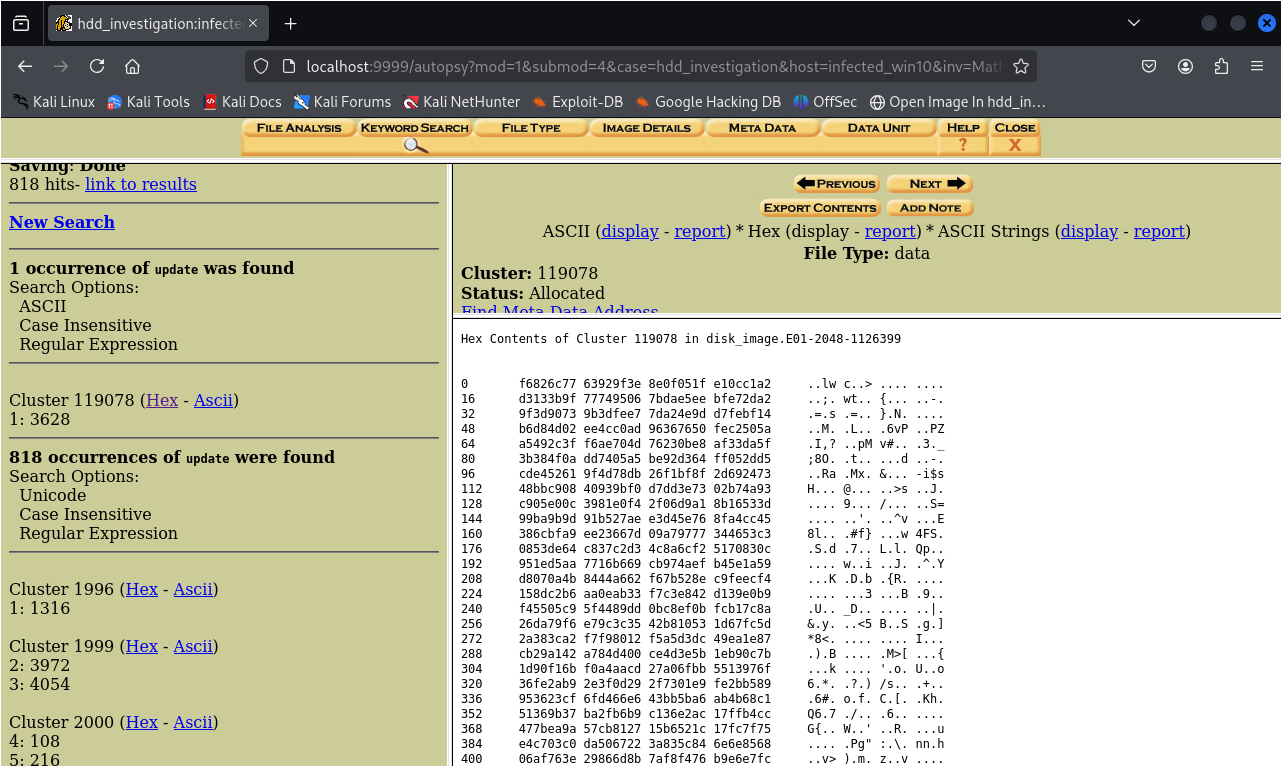
1. Presence of Update Strings

The autopsy’s keyword search identified over 800 occurrences of the word “update”, in Unicode form, with only a single ASCII indication found solely in Cluster 119078. It is very non-standard to see these terms in a recovery file, especially as they never show up in official patching files or installer packages. Such placement implies that the malicious content could very likely have been falsely attributed possibly as part of attempt to obscure the nature of content by pretending to be a system update.

1. File Type and Compression Abuse

The file is a WIM archive compressed with LZX as the compression method is often used for Windows recovery or installation. The compression that is used may be exploited by attackers to inject bad payloads into compressed files, aware that such compressed files are sometimes overlooked by security tools.

Screenshots:



# 

# 4.0 Conclusion

The examination showed various markers of compromise in the memory and storage components of the system. In RAM, investigators used Volatility3 and found embedded shellcode, unrelated process behaviour, and indications of command-and-control communications. Further analysis of the HDD by Autopsy presented hidden or manipulated files such as $TxFLog.blf, with changed access dates that were inconsistent with normal activity. The results highlight the complexity of modern threats, including rootkits and fileless malware, and their use of sophisticated stealth and evasion methods to stay below the radar. Using standard forensic procedures, such as Order of Volatility, ensuring data integrity, and cross-validating tools, was important to determine whether digital evidence was admissible in legal or incident response arenas (Matthew Braid, 2001).

# References

FORENSICS, M. (2024). *Magnet RAM Capture*. Retrieved from https://www.magnetforensics.com/product/magnet-ram-capture/

Matthew Braid, A. (2 August, 2001). *Collecting Electronic Evidence After a System Compromise*.

Retrieved from AUSCERT: https://auscert.org.au/publications/historical-articles/201709-11-collecting-electronic-evidence-after-sy