Lab Report

Name: Inor Wang
Title: Capstone
Case: 25-T112
Date: 12/12/2025

Table of Contents

Document Revision History

Name	Revision Date	Version	Description
Inor Wang	12/12/2025	0.1	Draft

Executive Summary

In this lab, the examiner conducted memory and static disk forensics on a Windows 10 virtual machine to identify user activity, encryption usage, and potential malicious intent. Volatility 3 was used to confirm the memory capture context (system time and OS major version) and to enumerate running processes, which showed Microsoft Edge, VeraCrypt, and Windows Subsystem for Linux (WSL) with a Kali Linux distribution. Autopsy was then used to analyze the primary virtual disk image to document system details (hostname, OS edition, timezone, and interactive user), review web artifacts showing research into malware generation (Metasploit/msfvenom), and identify encryption-related artifacts. The examiner confirmed VeraCrypt installer execution through Prefetch, located a high-entropy VeraCrypt container in the user's Documents directory, extracted header bytes, and used Hashcat to recover the plaintext password. After mounting the container, the examiner identified invoice.exe and target.txt, calculated SHA1 hashes, and analyzed invoice.exe with VirusTotal. Results indicated the executable was malicious and consistent with Meterpreter stager/Cobalt Strike indicators. The examiner also identified the target email address stored in target.txt, supporting that the suspect used encrypted storage to retain a payload and targeting information.

Key findings:

- **Validated memory capture timestamp (system time)** — The examiner confirmed the system time as 2021-11-04 03:54:49 UTC from win-10-20h2-websrv-Snapshot1.vmem, using Snapshot 1 as the consistent forensic image rather than the suspend-related temporary .vmem.
- **Confirmed OS major version in memory** — The examiner identified NtMajorVersion = 10, confirming the Windows 10 kernel generation.
- **Browser activity in memory** — Process enumeration showed Microsoft Edge (msedge.exe) running, indicating active web usage.
- **Encryption software observed in memory** — The examiner identified VeraCrypt (VeraCrypt.exe) running with PID 6984 and PPID 4292.
- **VeraCrypt launched interactively** — Process tree results showed explorer.exe as the parent process for VeraCrypt.

- **WSL and Kali Linux present** — The examiner identified wsl.exe (PID 2752) and related execution of kali.exe, indicating access to a penetration-testing Linux environment.
- **System identification from disk artifacts** — Autopsy documented hostname DESKTOP-NHQ5GC9, Windows 10 Education, timezone America/Chicago, and user efudd.
- **Encryption tool progression** — Web downloads and installed programs showed prior TrueCrypt activity and later VeraCrypt usage.
- **Malware-generation research** — Web search artifacts showed research into Metasploit/msfvenom, consistent with access via Kali under WSL.
- **Installer execution confirmed via Prefetch** — The examiner determined VERACRYPT SETUP 1.24-UPDATE7.EXE ran on 2021-11-03 22:33:01 CDT.
- **Encrypted container identified** — Autopsy flagged 2021-Mar-Invoice.ods in C:\Users\efudd\Documents\ as likely encrypted/high-entropy.
- **Password recovered and container mounted** — Using dd and Hashcat, the examiner recovered wabbit123! and mounted the container.
- **Container contents and target** — The container contained invoice.exe and target.txt, and target.txt listed bbunny@acmecorp.com.
- **Executable classified as malicious** — VirusTotal showed 24/62 detections and YARA hits consistent with Meterpreter stager and Cobalt Strike indicators, supporting classification of invoice.exe as a malicious Trojan/backdoor stager (hacktool).

In conclusion, the examiner's memory and disk analysis showed the suspect used Edge, VeraCrypt, and WSL (Kali), researched Metasploit/msfvenom, and stored artifacts in an encrypted container. After cracking and mounting the container, the examiner identified a malicious executable (invoice.exe) consistent with Meterpreter/Cobalt Strike indicators and confirmed the intended target email (bbunny@acmecorp.com).

Synopsis

The examiner was tasked with conducting a full end-to-end forensic investigation of a modern Windows virtual machine using Volatility 3 and static disk analysis to reconstruct the suspect's activities. On the memory side, the examiner must use windows.info.Info to determine when the memory image was captured and what OS version it represents, then pivot to windows.pstree to isolate user-level processes and identify which web browser executables were running, what encryption software was in use (including its executable name, program name, PID, and PPID), and how it was launched. The examiner must also document the built-in Windows virtualization technology in use, including the top-level virtualization process and its PID, and recognize a child process named after a well-known Linux distribution, interpreting how that distro is being leveraged (e.g., as a platform for malware development). The static disk portion requires establishing core system context: hostname, OS version/build, timezone, and interactive user account. Then using web history and artifacts to trace earlier encryption software the user researched or downloaded, as well as tools related to malware generation and where those tools were obtained or executed. The examiner must then identify the active encryption program's container file on disk, locate its path, and extract hash material to perform a targeted Hashcat brute-force attack using a mask derived from the "wabbit___!" post-it note. Finally, after unlocking the container, the examiner will inventory all files inside, calculate their SHA1 hashes, submit the key executable and its hash to VirusTotal to assess detections and classification, and determine the intended target of the suspected malicious payload.

Client Instructions:

Memory Forensics

1. What is the estimated date/timestamp this image was taken (systemtime)?
2. What is the operating system major version of this capture (NtMajorVersion)?
3. The suspect had several web browsers open. What was the executables name?
4. What browser was he using?
5. The suspect appears to be using encryption software. What is the name of the executable, the name of the program, the PID, and PPID for the executable?
6. What is the parent executable that ran this software?

7. It appears that the suspect is using virtualization technology built into this Windows version. What is the name of this technology, the name of the executable, and PID. (Top level executable only, no children)

8. The above executable has a PPID to another executable. The name of this executable is a well-known Linux distribution. What is the name of the executable and what is this distro used for?

Static Disk Forensics

9. What is the hostname of the system?

10. What is the operating system version and build for this system?

11. What is the timezone set to?

12. One interactive user is configured on this system, what is the username?

13. According to web history, it appears that the user downloaded and may have installed an encryption program before the one that is installed now. What is the name of this program?

14. According to web history, it appears that the user was researching how to generate malware. What was the program that they researched?

15. The user did not install this program on their Windows system, where did they get it from?

16. When was the setup executable for the more modern encryption program ran (according to prefetch)?

17. The user is using this encryption program to secure their operation. They have a container file on the hard drive. What is the name of the file? Where is it located?

Decryption

18. What is the password in plaintext?

19. What was the exact command you used to crack the password?

20. What was the mask you used to crack the password?

Encrypted Files

21. What is the filename of all the files in the container?

22. What are the SHA1 hashes of each file?

23. Submit the executables hash to VirusTotal. How many hits did the hash have?

24. Submit the actual executable to VirusTotal. Is it malicious? What would you classify the file as?

25. What was the target?

Evidence Analyzed

This section provides details of the digital evidence collected

Evidence ID	**E001**
Name	is4523-capstone.tar.gz
Type	gzip compressed data, max compression, from Unix, original size modulo 2^32 164243456
Size	10822.42 MB
MD5	5379386433189A58D702C7F34906B3F9
SHA1	809858FA01FFFAA4BA74583C65F0DAFCBD947881
SHA256	EF2189EFDF5EFC45C07BED6475B0CAF65311065AC9B07A7905F6CCAD8E53B00B

Tools Used

Workstation

Hostname	Operating System	Build	Physical / Virtual	Built
IS-4523-001-GREYMHATTER	Fedora	2025	Virtual	10/31/2025
IS-4523-001-WINDOWS	Windows 11	2021	Virtual	09/06/2025

Software

Name	Version	Release	Purpose
Volatility 3	2.26.2	Sep 2025	Memory forensics framework used to extract system metadata and enumerate processes/process trees from a .vmem image.
Autopsy	4.22.1	Apr 2025	Digital forensics platform used to ingest a disk image and review OS/user artifacts (hostname, timezone, users), browser history/search/downloads, Prefetch execution, and encryption-suspected/high-entropy results.
VeraCrypt	1.26.24	May 2025	Disk encryption tool used to mount encrypted containers/volumes so files inside the container can be accessed and examined.
VirusTotal	N/A	N/A	GPU-accelerated password recovery tool used to crack the container password using the correct hash mode and a mask/dictionary-style attack workflow.
Hashcat	7.1.2	Aug 2025	Online malware analysis service used to submit hashes/files, review multi-vendor detections, and document community/YARA signature matches for classification.
dd	coreutils 9.9	Nov 2025	Command-line utility used to copy/extract raw bytes into a separate file for downstream cracking or analysis.

Analysis Findings

Overview of Examination Procedures

The examiner conducted a combined memory and static disk forensic examination of the provided Windows 10 virtual machine artifacts to identify user activity, installed tools, and evidence of potential malicious intent. Using Volatility 3 on the Snapshot 1 memory image, the examiner validated the capture context by extracting system time and OS version information, then enumerated running processes to identify the suspect's active applications, including Microsoft Edge and VeraCrypt, and confirmed parent-child process relationships with process tree analysis. The examiner also identified Windows Subsystem for Linux (WSL) activity and verified that a Kali Linux distribution was present and executed, supporting later findings of offensive tool research. For static disk analysis, the examiner imported the primary flat VMDK into Autopsy to document system configuration (hostname, OS build, timezone, and interactive user account) and to review web history artifacts that showed prior encryption software interest (TrueCrypt) and research into malware generation using Metasploit/msfvenom, which aligned with the presence of Kali under WSL. The examiner then validated program execution through Prefetch artifacts to determine when the VeraCrypt installer was run, located a high-entropy encrypted container file in the user's Documents folder, and extracted the necessary header data to enable password cracking. Using Hashcat, the examiner recovered the plaintext password, mounted the VeraCrypt container, and enumerated its contents, which included invoice.exe and target.txt. Finally, the examiner preserved integrity by hashing both files, assessed the executable's reputation and signatures via VirusTotal (including vendor detections and YARA matches), classified the file as a malicious stager consistent with Meterpreter/Cobalt Strike indicators, and identified the intended target email address from target.txt. Additional targeted analysis was performed using:

- **Volatility 3 —** Memory forensics framework used to extract system metadata (system time, OS version) and enumerate processes/process trees (e.g., info.Info, windows.pslist, windows.pstree) from the .vmem image.
- **Autopsy —** Digital forensics platform used to ingest the .vmdk disk image and review OS artifacts (hostname, OS info, timezone, users), web history/search/downloads, Prefetch "Run Programs," and encryption-suspected/high-entropy results.

- **VeraCrypt —** Disk encryption tool used to mount the identified encrypted container file after recovering the password and to access files stored inside the container.
- **Hashcat —** GPU-accelerated password-cracking tool used to crack the VeraCrypt container password using the appropriate hash mode and a masked brute-force attack.
- **VirusTotal —** Online malware analysis service used to assess invoice.exe via hash and file submission, review vendor detection results, and document community/YARA signature matches.
- **dd —** Command-line utility used to extract the required header bytes from the encrypted container into a hash file for cracking workflows.

Throughout the process, all findings were documented, and evidence files were correctly hashed.

Evidence Reviewed

is4523-capstone.tar.gz (E01): Workstation virtual machine clone

Key Findings

Memory Forensics

1. What is the estimated date/timestamp this image was taken (systemtime)?

- **Analysis Performed:**
  - The examiner used Volatility 3 via the command line shown in Figure 1, to extract the image info of "win-10-20h2-websrv-Snapshot1.vmem", which contains the date timestamp (system time) of which this snapshot image was captured.
  - Command: *vol -f win-10-20h2-websrv-Snapshot1.vmem info.Info*
- **Answer:**
  The system time of the "win-10-20h2-websrv-Snapshot1.vmem" image file is **2021-11-04 03:54:49 UTC**, as shown in Figure 1. The reason for why the examiner used "win-10-20h2-websrv-Snapshot1.vmem" instead of "win-10-20h2-websrv-b466fd3e.vmem" is because the other file is a temporary memory file created during a suspend, often tied to a .vmss file. It does not represent a consistent snapshot used for forensic memory analysis. Therefore, the examiner used the Snapshot 1 memory image file.

- **Supporting Evidence:**

```
Evidence/is4523-capstone/win-10-20h2-websrv ❯ vol -f win-10-20h2-websrv-Snapshot1.vmem info.Info
Volatility 3 Framework 2.26.2
Progress:  100.00               PDB scanning finished
Variable        Value

Kernel Base     0xf8050ea00000
DTB     0x1ad000
Symbols file:///home/hatter/.local/lib/python3.13/site-packages/volatility3/symbols/windows/ntkrnlmp.pdb/641F55C592201DCC4F59FACC72EA54DA-1.json.xz
Is64Bit True
IsPAE   False
layer_name      0 WindowsIntel32e
memory_layer    1 VmwareLayer
base_layer      2 FileLayer
meta_layer      2 FileLayer
KdVersionBlock  0xf8050f60f2e0
Major/Minor     15.19041
MachineType     34404
KeNumberProcessors      2
SystemTime      2021-11-04 03:54:49+00:00
NtSystemRoot    C:\Windows
NtProductType   NtProductWinNt
NtMajorVersion  10
NtMinorVersion  0
PE MajorOperatingSystemVersion  10
PE MinorOperatingSystemVersion  0
PE Machine      34404
PE TimeDateStamp        Wed Nov 22 15:44:41 2056
```

Figure 1. Volatility 3 info.Info for "win-10-20h2-websrv-Snapshot1.vmem"

2. What is the operating system major version of this capture (NtMajorVersion)?

- **Analysis Performed:**
  - The examiner used Volatility 3 via the command line shown in Figure 2, to extract the image info of "win-10-20h2-websrv-Snapshot1.vmem", which contains the NtMajorVersion.
  - Command: *vol -f win-10-20h2-websrv-Snapshot1.vmem info.Info*
- **Answer:**
  The NtMajorVersion of the memory image is **10**, as shown in Figure 2. The NtMajorVersion is the Windows kernel's major version number stored in memory (e.g., 10 for Windows 10), which identifies the core operating system generation.

- **Supporting Evidence:**

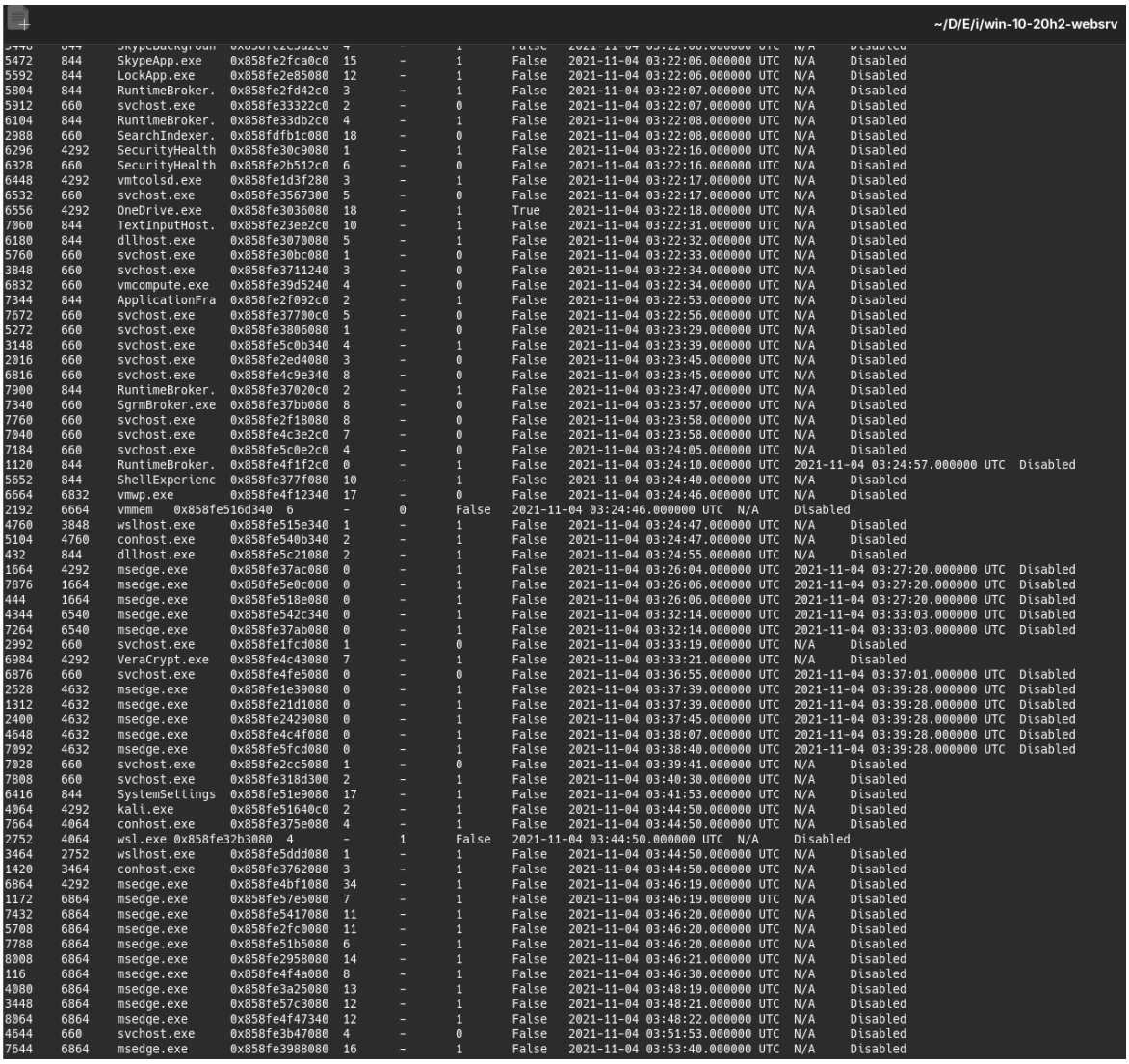
```
Evidence/is4523-capstone/win-10-20h2-websrv ❯ vol -f win-10-20h2-websrv-Snapshot1.vmem info.Info
Volatility 3 Framework 2.26.2
Progress:  100.00               PDB scanning finished
Variable        Value

Kernel Base     0xf8050ea00000
DTB     0x1ad000
Symbols file:///home/hatter/.local/lib/python3.13/site-packages/volatility3/symbols/windows/ntkrnlmp.pdb/641F55C592201DCC4F59FACC72EA54DA-1.json.xz
Is64Bit True
IsPAE   False
layer_name      0 WindowsIntel32e
memory_layer    1 VmwareLayer
base_layer      2 FileLayer
meta_layer      2 FileLayer
KdVersionBlock  0xf8050f60f2e0
Major/Minor     15.19041
MachineType     34404
KeNumberProcessors      2
SystemTime      2021-11-04 03:54:49+00:00
NtSystemRoot    C:\Windows
NtProductType   NtProductWinNt
NtMajorVersion  10
NtMinorVersion  0
PE MajorOperatingSystemVersion  10
PE MinorOperatingSystemVersion  0
PE Machine      34404
PE TimeDateStamp        Wed Nov 22 15:44:41 2056
```

Figure 2. Voltaility 3 info.Info for "win-10-20h2-websrv-Snapshot1.vmem

3. The suspect had several web browsers open. What was the executables name?

- **Analysis Performed:**
  - The examiner used Volatility 3 via the command line shown in Figure 3, to extract the processes (pslist) of the "win-10-20h2-websrv-Snapshot1.vmem" image.
  - Command: *vol -f win-10-20h2-websrv-Snapshot1.vmem windows.pslist*
- **Answer:**
  The suspect had several web browsers open, the executable's name were **msedge.exe**, as shown in Figure 3.

- **Supporting Evidence:**



5472	844	SkypeApp.exe	0x858fe2fca0c0	15	-	1	False	2021-11-04 03:22:06.000000 UTC	N/A	Disabled
5592	844	LockApp.exe	0x858fe2e85080	12	-	1	False	2021-11-04 03:22:06.000000 UTC	N/A	Disabled
5804	844	RuntimeBroker.	0x858fe2fd42c0	3	-	1	False	2021-11-04 03:22:07.000000 UTC	N/A	Disabled
5912	660	svchost.exe	0x858fe33322c0	2	-	0	False	2021-11-04 03:22:07.000000 UTC	N/A	Disabled
6104	844	RuntimeBroker.	0x858fe33db2c0	4	-	1	False	2021-11-04 03:22:08.000000 UTC	N/A	Disabled
2988	660	SearchIndexer.	0x858fdfb1c080	18	-	0	False	2021-11-04 03:22:08.000000 UTC	N/A	Disabled
6296	4292	SecurityHealth	0x858fe30c9080	1	-	1	False	2021-11-04 03:22:16.000000 UTC	N/A	Disabled
6328	660	SecurityHealth	0x858fe2b512c0	6	-	0	False	2021-11-04 03:22:16.000000 UTC	N/A	Disabled
6448	4292	vmtoolsd.exe	0x858fe1d3f280	3	-	1	False	2021-11-04 03:22:17.000000 UTC	N/A	Disabled
6532	660	svchost.exe	0x858fe3567300	5	-	0	False	2021-11-04 03:22:17.000000 UTC	N/A	Disabled
6556	4292	OneDrive.exe	0x858fe3036080	18	-	1	True	2021-11-04 03:22:18.000000 UTC	N/A	Disabled
7060	844	TextInputHost.	0x858fe23ee2c0	10	-	1	False	2021-11-04 03:22:31.000000 UTC	N/A	Disabled
6180	844	dllhost.exe	0x858fe3070080	5	-	1	False	2021-11-04 03:22:32.000000 UTC	N/A	Disabled
5760	660	svchost.exe	0x858fe30bc080	1	-	0	False	2021-11-04 03:22:33.000000 UTC	N/A	Disabled
3848	660	svchost.exe	0x858fe3711240	3	-	0	False	2021-11-04 03:22:34.000000 UTC	N/A	Disabled
6832	660	vmcompute.exe	0x858fe39d5240	4	-	0	False	2021-11-04 03:22:34.000000 UTC	N/A	Disabled
7344	844	ApplicationFra	0x858fe2f092c0	2	-	1	False	2021-11-04 03:22:53.000000 UTC	N/A	Disabled
7672	660	svchost.exe	0x858fe37700c0	5	-	0	False	2021-11-04 03:22:56.000000 UTC	N/A	Disabled
5272	660	svchost.exe	0x858fe3806080	1	-	0	False	2021-11-04 03:23:29.000000 UTC	N/A	Disabled
3148	660	svchost.exe	0x858fe5c0b340	4	-	1	False	2021-11-04 03:23:39.000000 UTC	N/A	Disabled
2016	660	svchost.exe	0x858fe2ed4080	3	-	0	False	2021-11-04 03:23:45.000000 UTC	N/A	Disabled
6816	660	svchost.exe	0x858fe4c9e340	8	-	0	False	2021-11-04 03:23:45.000000 UTC	N/A	Disabled
7900	844	RuntimeBroker.	0x858fe37020c0	2	-	1	False	2021-11-04 03:23:47.000000 UTC	N/A	Disabled
7340	660	SgrmBroker.exe	0x858fe37bb080	8	-	0	False	2021-11-04 03:23:57.000000 UTC	N/A	Disabled
7760	660	svchost.exe	0x858fe2f18080	8	-	0	False	2021-11-04 03:23:58.000000 UTC	N/A	Disabled
7040	660	svchost.exe	0x858fe4c3e2c0	7	-	0	False	2021-11-04 03:23:58.000000 UTC	N/A	Disabled
7184	660	svchost.exe	0x858fe5c0e2c0	4	-	0	False	2021-11-04 03:24:05.000000 UTC	N/A	Disabled
1120	844	RuntimeBroker.	0x858fe4f1f2c0	0	-	1	False	2021-11-04 03:24:10.000000 UTC	2021-11-04 03:24:57.000000 UTC	Disabled
5652	844	ShellExperienc	0x858fe377f080	10	-	1	False	2021-11-04 03:24:40.000000 UTC	N/A	Disabled
6664	6832	vmwp.exe	0x858fe4f12340	17	-	0	False	2021-11-04 03:24:46.000000 UTC	N/A	Disabled
2192	6664	vmmem	0x858fe516d340	6	-	0	False	2021-11-04 03:24:46.000000 UTC	N/A	Disabled
4760	3848	wslhost.exe	0x858fe515e340	1	-	1	False	2021-11-04 03:24:47.000000 UTC	N/A	Disabled
5104	4760	conhost.exe	0x858fe540b340	2	-	1	False	2021-11-04 03:24:47.000000 UTC	N/A	Disabled
432	844	dllhost.exe	0x858fe5c21080	2	-	1	False	2021-11-04 03:24:55.000000 UTC	N/A	Disabled
1664	4292	msedge.exe	0x858fe37ac080	0	-	1	False	2021-11-04 03:26:04.000000 UTC	2021-11-04 03:27:20.000000 UTC	Disabled
7876	1664	msedge.exe	0x858fe5e0c080	0	-	1	False	2021-11-04 03:26:06.000000 UTC	2021-11-04 03:27:20.000000 UTC	Disabled
444	1664	msedge.exe	0x858fe518e080	0	-	1	False	2021-11-04 03:26:06.000000 UTC	2021-11-04 03:27:20.000000 UTC	Disabled
4344	6540	msedge.exe	0x858fe542c340	0	-	1	False	2021-11-04 03:32:14.000000 UTC	2021-11-04 03:33:03.000000 UTC	Disabled
7264	6540	msedge.exe	0x858fe37ab080	0	-	1	False	2021-11-04 03:32:14.000000 UTC	2021-11-04 03:33:03.000000 UTC	Disabled
2992	660	svchost.exe	0x858fe1fcd080	1	-	0	False	2021-11-04 03:33:19.000000 UTC	N/A	Disabled
6984	4292	VeraCrypt.exe	0x858fe4c43080	7	-	1	False	2021-11-04 03:33:21.000000 UTC	N/A	Disabled
6876	660	svchost.exe	0x858fe4fe5080	0	-	0	False	2021-11-04 03:36:55.000000 UTC	2021-11-04 03:37:01.000000 UTC	Disabled
2528	4632	msedge.exe	0x858fe1e39080	0	-	1	False	2021-11-04 03:37:39.000000 UTC	2021-11-04 03:39:28.000000 UTC	Disabled
1312	4632	msedge.exe	0x858fe21d1080	0	-	1	False	2021-11-04 03:37:39.000000 UTC	2021-11-04 03:39:28.000000 UTC	Disabled
2400	4632	msedge.exe	0x858fe2429080	0	-	1	False	2021-11-04 03:37:45.000000 UTC	2021-11-04 03:39:28.000000 UTC	Disabled
4648	4632	msedge.exe	0x858fe4c4f080	0	-	1	False	2021-11-04 03:38:07.000000 UTC	2021-11-04 03:39:28.000000 UTC	Disabled
7092	4632	msedge.exe	0x858fe5fcd080	0	-	1	False	2021-11-04 03:38:40.000000 UTC	2021-11-04 03:39:28.000000 UTC	Disabled
7028	660	svchost.exe	0x858fe2cc5080	1	-	0	False	2021-11-04 03:39:41.000000 UTC	N/A	Disabled
7808	660	svchost.exe	0x858fe318d300	2	-	1	False	2021-11-04 03:40:30.000000 UTC	N/A	Disabled
6416	844	SystemSettings	0x858fe51e9080	17	-	1	False	2021-11-04 03:41:53.000000 UTC	N/A	Disabled
4064	4292	kali.exe	0x858fe51640c0	2	-	1	False	2021-11-04 03:44:50.000000 UTC	N/A	Disabled
7664	4064	conhost.exe	0x858fe375e080	4	-	1	False	2021-11-04 03:44:50.000000 UTC	N/A	Disabled
2752	4064	wsl.exe	0x858fe32b3080	4	-	1	False	2021-11-04 03:44:50.000000 UTC	N/A	Disabled
3464	2752	wslhost.exe	0x858fe5ddd080	1	-	1	False	2021-11-04 03:44:50.000000 UTC	N/A	Disabled
1420	3464	conhost.exe	0x858fe3762080	3	-	1	False	2021-11-04 03:44:50.000000 UTC	N/A	Disabled
6864	4292	msedge.exe	0x858fe4bf1080	34	-	1	False	2021-11-04 03:46:19.000000 UTC	N/A	Disabled
1172	6864	msedge.exe	0x858fe57e5080	7	-	1	False	2021-11-04 03:46:19.000000 UTC	N/A	Disabled
7432	6864	msedge.exe	0x858fe5417080	11	-	1	False	2021-11-04 03:46:20.000000 UTC	N/A	Disabled
5708	6864	msedge.exe	0x858fe2fc0080	11	-	1	False	2021-11-04 03:46:20.000000 UTC	N/A	Disabled
7788	6864	msedge.exe	0x858fe51b5080	6	-	1	False	2021-11-04 03:46:20.000000 UTC	N/A	Disabled
8008	6864	msedge.exe	0x858fe2958080	14	-	1	False	2021-11-04 03:46:21.000000 UTC	N/A	Disabled
116	6864	msedge.exe	0x858fe4f4a080	8	-	1	False	2021-11-04 03:46:30.000000 UTC	N/A	Disabled
4080	6864	msedge.exe	0x858fe3a25080	13	-	1	False	2021-11-04 03:48:19.000000 UTC	N/A	Disabled
3448	6864	msedge.exe	0x858fe57c3080	12	-	1	False	2021-11-04 03:48:21.000000 UTC	N/A	Disabled
8064	6864	msedge.exe	0x858fe4f47340	12	-	1	False	2021-11-04 03:48:22.000000 UTC	N/A	Disabled
4644	660	svchost.exe	0x858fe3b47080	4	-	0	False	2021-11-04 03:51:53.000000 UTC	N/A	Disabled
7644	6864	msedge.exe	0x858fe3988080	16	-	1	False	2021-11-04 03:53:40.000000 UTC	N/A	Disabled

Figure 3. Volatility 3 windows.pslist for win-10-20h2-websrv-Snapshot1.vmem

4. What browser was he using?

- **Analysis Performed:**
  - The examiner used Volatility 3 via the command line shown in Figure 4, to extract the processes (pslist) of the "win-10-20h2-websrv-Snapshot1.vmem" image.
  - Command: *vol -f win-10-20h2-websrv-Snapshot1.vmem windows.pslist*
- **Answer:**
  The suspect was using the **Microsoft Edge** (msedge.exe) web browser, as shown in Figure 4. Knowing the suspect's browser helps the examiner target the right artifacts: history files, cache, cookies, logins, and downloads, which are all stored differently by Chrome, Edge, Firefox, etc. It can also reveal how the suspect accessed malicious sites or web applications (for example, if they abused specific Edge features or extensions), helping reconstruct intent, actions, and possible data exfiltration paths.

- **Supporting Evidence:**

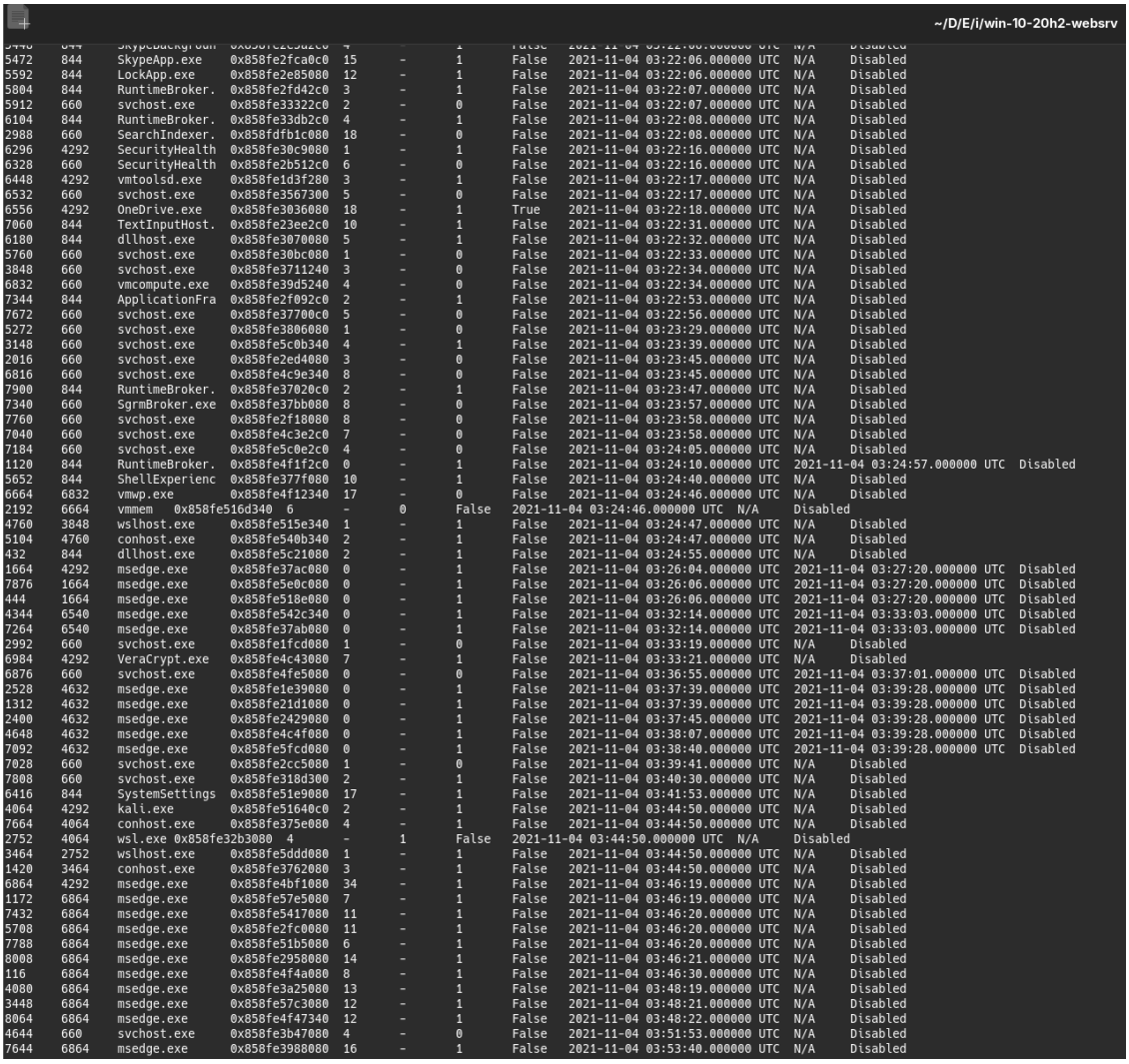
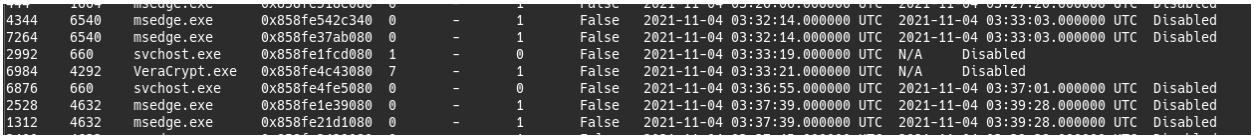


Figure 4. Volatility 3 info.Info for win-10-20h2-websrv-Snapshot1.vmem

5. The suspect appears to be using encryption software. What is the name of the executable, the name of the program, the PID, and PPID for the executable?

- **Analysis Performed:**
  - The examiner used Volatility 3 via the command line shown in Figure 5, to extract the processes (pslist) of the "win-10-20h2-websrv-Snapshot1.vmem" image.
  - Command: *vol -f win-10-20h2-websrv-Snapshot1.vmem windows.pslist*
- **Answer:**
  The name of the encryption software's executable is **VeraCrypt.exe**, the name of the program is **VeraCrypt**, the PID is **6984**, and the PPID is **4292**, as shown in Figure 5.

- **Supporting Evidence:**



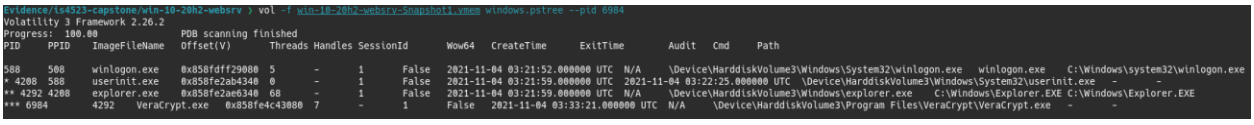
4344	6540	msedge.exe	0x858fe542c340	0	-	1	False	2021-11-04 03:32:14.000000 UTC	2021-11-04 03:33:03.000000 UTC	Disabled
7264	6540	msedge.exe	0x858fe37ab080	0	-	1	False	2021-11-04 03:32:14.000000 UTC	2021-11-04 03:33:03.000000 UTC	Disabled
2992	660	svchost.exe	0x858fe1fcd080	1	-	0	False	2021-11-04 03:33:19.000000 UTC	N/A	Disabled
6984	4292	VeraCrypt.exe	0x858fe4c43080	7	-	1	False	2021-11-04 03:33:21.000000 UTC	N/A	Disabled
6876	660	svchost.exe	0x858fe4fe5080	0	-	0	False	2021-11-04 03:36:55.000000 UTC	2021-11-04 03:37:01.000000 UTC	Disabled
2528	4632	msedge.exe	0x858fe1e39080	0	-	1	False	2021-11-04 03:37:39.000000 UTC	2021-11-04 03:39:28.000000 UTC	Disabled
1312	4632	msedge.exe	0x858fe21d1080	0	-	1	False	2021-11-04 03:37:39.000000 UTC	2021-11-04 03:39:28.000000 UTC	Disabled

Figure 5. Volatility 3 info.Info for win-10-20h2-websrv-Snapshot1.vmem

6. What is the parent executable that ran this software?

- **Analysis Performed:**
  - The examiner used Volatility 3 via the command line shown in Figure 6, to extract the process tree (pstree) of the VeraCrypt software (PID: 6984) within the "win-10-20h2-websrv-Snapshot1.vmem" image.
  - Command: *vol -f win-10-20h2-websrv-Snapshot1.vmem windows.pstree –pid 6984*
- **Answer:**
  The parent executable that ran the VeraCrypt software is **explorer.exe**, as shown in Figure 6.

- **Supporting Evidence:**



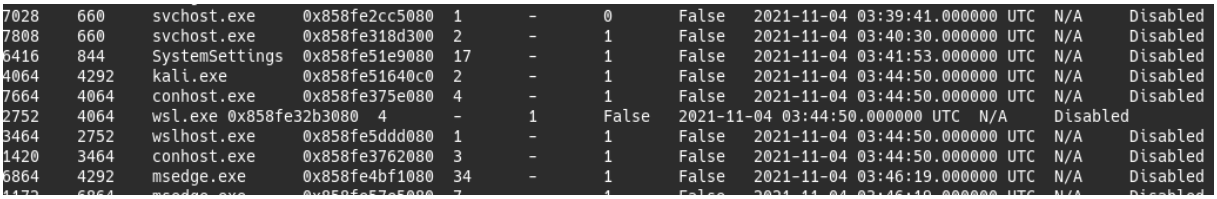
```
Evidence/is4523-capstone/win-10-20h2-websrv ) vol -f win-10-20h2-websrv-Snapshot1.vmem windows.pstree --pid 6984
Volatility 3 Framework 2.26.2
Progress:  100.00               PDB scanning finished
PID     PPID    ImageFileName   Offset(V)       Threads Handles SessionId       Wow64   CreateTime      ExitTime        Audit   Cmd     Path

588     508     winlogon.exe    0x858fdff29080  5       -       1       False   2021-11-04 03:21:52.000000 UTC  N/A     \Device\HarddiskVolume3\Windows\System32\winlogon.exe   winlogon.exe    C:\Windows\system32\winlogon.exe
* 4208  588     userinit.exe    0x858fe2ab4340  0       -       1       False   2021-11-04 03:21:59.000000 UTC  2021-11-04 03:22:25.000000 UTC  \Device\HarddiskVolume3\Windows\System32\userinit.exe   -       -
** 4292 4208    explorer.exe    0x858fe2ae6340  68      -       1       False   2021-11-04 03:21:59.000000 UTC  N/A     \Device\HarddiskVolume3\Windows\explorer.exe    C:\Windows\Explorer.EXE C:\Windows\Explorer.EXE
*** 6984       4292    VeraCrypt.exe   0x858fe4c43080  7       -       1       False   2021-11-04 03:33:21.000000 UTC  N/A     \Device\HarddiskVolume3\Program Files\VeraCrypt\VeraCrypt.exe   -       -
```

Figure 6. Volatility windows.pstree output showing VeraCrypt.exe as a child process of explorer.exe

7. It appears that the suspect is using virtualization technology built into this Windows version. What is the name of this technology, the name of the executable, and PID. (Top level executable only, no children)

- **Analysis Performed:**
  - The examiner used Volatility 3 via the command line shown in Figure 7, to extract the processes (pslist) of the "win-10-20h2-websrv-Snapshot1.vmem" image.
  - Command: *vol -f win-10-20h2-websrv-Snapshot1.vmem windows.pslist*
- **Answer:**
  The suspect is using virtualization technology that is built into Windows 10 (this Windows version), the name of the technology is **Windows Subsystem for Linux (WSL)**, the name of the executable is **wsl.exe**, and the PID is **2752**, as shown in Figure 7.

- **Supporting Evidence:**



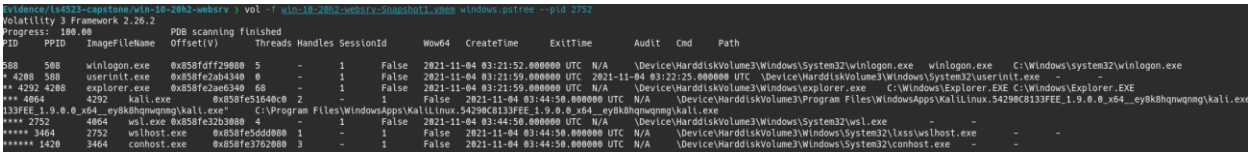
```
7028    660     svchost.exe     0x858fe2cc5080  1       -       0       False   2021-11-04 03:39:41.000000 UTC  N/A     Disabled
7808    660     svchost.exe     0x858fe318d300  2       -       1       False   2021-11-04 03:40:30.000000 UTC  N/A     Disabled
6416    844     SystemSettings  0x858fe51e9080  17      -       1       False   2021-11-04 03:41:53.000000 UTC  N/A     Disabled
4064    4292    kali.exe        0x858fe51640c0  2       -       1       False   2021-11-04 03:44:50.000000 UTC  N/A     Disabled
7664    4064    conhost.exe     0x858fe375e080  4       -       1       False   2021-11-04 03:44:50.000000 UTC  N/A     Disabled
2752    4064    wsl.exe 0x858fe32b3080  4       -       1       False   2021-11-04 03:44:50.000000 UTC  N/A     Disabled
3464    2752    wslhost.exe     0x858fe5ddd080  1       -       1       False   2021-11-04 03:44:50.000000 UTC  N/A     Disabled
1420    3464    conhost.exe     0x858fe3762080  3       -       1       False   2021-11-04 03:44:50.000000 UTC  N/A     Disabled
6864    4292    msedge.exe      0x858fe4bf1080  34      -       1       False   2021-11-04 03:46:19.000000 UTC  N/A     Disabled
```

Figure 7. Volatility windows.pslist output showing wsl.exe

8. The above executable has a PPID to another executable. The name of this executable is a well-known Linux distrubtion. What is the name of the executable and what is this distro used for?

- **Analysis Performed:**
  - The examiner used Volatility 3 via the command line shown in Figure 8, to extract the process tree (pstree) of the WSL software (PID: 2752) within the "win-10-20h2-websrv-Snapshot1.vmem" image.
  - Command: *vol -f win-10-20h2-websrv-Snapshot1.vmem windows.pstree –pid 2752*
- **Answer:**
  The name of the executable is **kali.exe** and this distribution of Linux is mainly used for penetration testing, ethical hacking, and digital forensics investigations, as shown in Figure 8.

- **Supporting Evidence:**



```
Evidence/ls4523-capstone/win-10-20h2-websrv > vol -f win-10-20h2-websrv-Snapshot1.vmem windows.pstree --pid 2752
Volatility 3 Framework 2.26.2
Progress:  100.00               PDB scanning finished
PID     PPID    ImageFileName   Offset(V)       Threads Handles SessionId       Wow64   CreateTime      ExitTime        Audit   Cmd     Path

588     508     winlogon.exe    0x858fdff29080  5       -       1       False   2021-11-04 03:21:52.000000 UTC  N/A     \Device\HarddiskVolume3\Windows\System32\winlogon.exe   winlogon.exe    C:\Windows\system32\winlogon.exe
* 4200  588     userinit.exe    0x858fe2ab4340  0       -       1       False   2021-11-04 03:21:59.000000 UTC  2021-11-04 03:22:25.000000 UTC  \Device\HarddiskVolume3\Windows\System32\userinit.exe    -       -       -
** 4292 4200    explorer.exe    0x858fe2ae6340  68      -       1       False   2021-11-04 03:21:59.000000 UTC  N/A     \Device\HarddiskVolume3\Windows\explorer.exe    C:\Windows\Explorer.EXE C:\Windows\Explorer.EXE
*** 4064       4292    kali.exe        0x858fe51640c0  2       -       1       False   2021-11-04 03:44:50.000000 UTC  N/A     \Device\HarddiskVolume3\Program Files\WindowsApps\KaliLinux.54290C8133FEE_1.9.0.0_x64__ey8k8hqnwqnmg\kali.exe
133FEE_1.9.0.0_x64__ey8k8hqnwqnmg\kali.exe"     C:\Program Files\WindowsApps\KaliLinux.54290C8133FEE_1.9.0.0_x64__ey8k8hqnwqnmg\kali.exe
**** 2752      4064    wsl.exe 0x858fe32b3080  4       -       1       False   2021-11-04 03:44:50.000000 UTC  N/A     \Device\HarddiskVolume3\Windows\System32\wsl.exe        -       -
***** 3464     2752    wslhost.exe     0x858fe5ddd080  1       -       1       False   2021-11-04 03:44:50.000000 UTC  N/A     \Device\HarddiskVolume3\Windows\System32\lxss\wslhost.exe       -       -
****** 1420    3464    conhost.exe     0x858fe3762080  3       -       1       False   2021-11-04 03:44:50.000000 UTC  N/A     \Device\HarddiskVolume3\Windows\System32\conhost.exe    -       -
```

Figure 8. Volatility windows.pstree output showing kali.exe

Static Disk Forensics

9. What is the hostname of the system?

- **Analysis Performed:**
  - The examiner created a new case named, “Captone”, and imported the disk image file (win-10-20h2-websrv-flat.vmdk) into Autopsy.
    - The reason for why the examiner chose, “win-10-20h2-websrv-flat.vmdk”, and not the other disk files is because this disk file is approximately 34 GB, signifying it is the main one.
  - The examiner then went to the “Operating System Information” data artifact within Autopsy which shows information about the operating system, as shown in Figure 9.
- **Answer:**
  The hostname of the system is **DESKTOP-NHQ5GC9**, as shown in Figure 9.

- **Supporting Evidence:**

Type	Value
Name	DESKTOP-NHQ5GC9
Program Name	Windows 10 Education
Processor Architecture	AMD64
Temporary Files Directory	%SystemRoot%\TEMP
Path	C:\Windows
Product ID	00328-00102-10182-AA467
Owner	efudd
Source File Path	/img_win-10-20h2-websrv-flat.vmdk
Artifact ID	-9223372036854775737

Figure 9. The "Operating System Information" data artifact information

10. What is the operating system version and build for this system?

- **Analysis Performed:**
  - The examiner then went to the “Operating System Information” data artifact within Autopsy which shows information about the operating system, as shown in Figure 10.
- **Answer:**
  The operating system of the system is **Windows 10 Education**, as shown in Figure 10.

- **Supporting Evidence:**

Type	Value
Name	DESKTOP-NHQ5GC9
Program Name	Windows 10 Education
Processor Architecture	AMD64
Temporary Files Directory	%SystemRoot%\TEMP
Path	C:\Windows
Product ID	00328-00102-10182-AA467
Owner	efudd
Source File Path	/img_win-10-20h2-websrv-flat.vmdk
Artifact ID	-9223372036854775737

Figure 10. The "Operating System Information" data artifact information

11. What is the timezone set to?

- **Analysis Performed:**
  - The examiner then went to the Data Source section within the tree and clicked on the .vmdk file, which shows what the timezone is set to, as shown in Figure 11.
- **Answer:**
  The timezone is set to **America/Chicago**, as shown in Figure 11.

- **Supporting Evidence:**

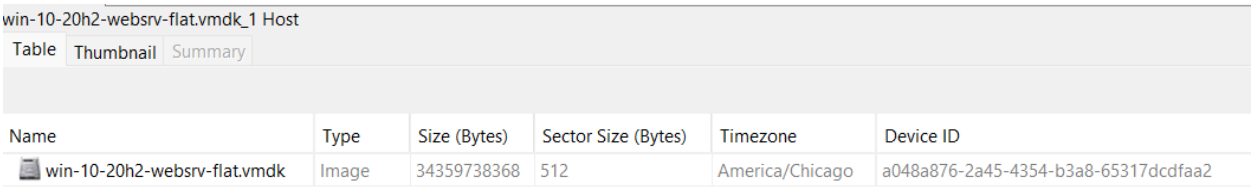


Figure 11. What the timezone is set to for the .vmdk file within Autopsy

12. One interactive user is configured on this system, what is the username?

- **Analysis Performed:**
  - The examiner then went to the file system of the system within the Data Sources tab as shown on the left side of Figure 12. The examiner then went to the Users folder to find the users configured on the system, as shown in Figure 12.
- **Answer:**
  There is one interactive user that is configured on the system, the username is **efudd**, as shown in Figure 12.

- **Supporting Evidence:**

Figure 12. Autopsy view of the win-10-20h2-websrv-flat.vmdk image showing volumes and the Users directory.

13. According to web history, it appears that the user downloaded and may have installed an encryption program before the one that is installed now. What is the name of this program?

- **Analysis Performed:**
  - The examiner then went to the "Web Downloads" data artifact within Autopsy which shows information about what was downloaded on the system, as shown in Figure 13.
  - The examiner then went to the Program Files directory in the file system to find the encryption program that is installed now, as shown in Figure 14.
- **Answer:**
  The user downloaded and may have installed an encryption program before the one that is installed now. The encryption program that is installed now is VeraCrypt, as shown in Figure 14. However, the name of the program that was installed before VeraCrypt is **TrueCrypt**, as shown in Figure 13.

- **Supporting Evidence:**

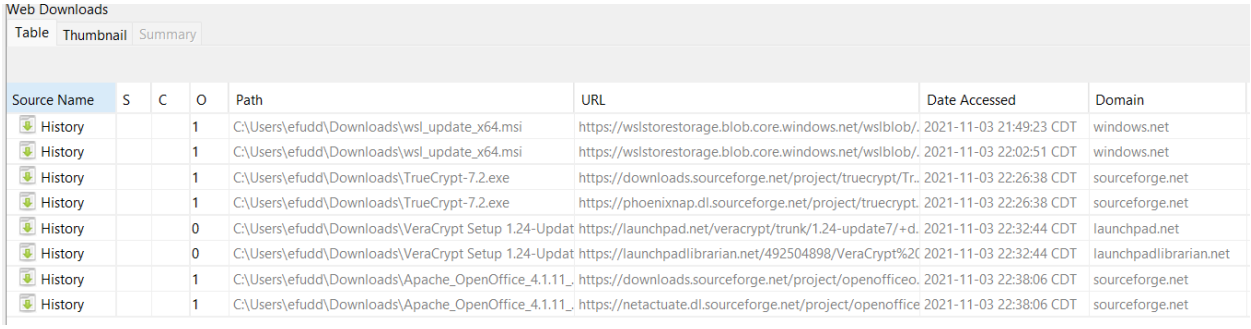

Web Downloads

Table | Thumbnail | Summary

Source Name	S	C	O	Path	URL	Date Accessed	Domain
History			1	C:\Users\efudd\Downloads\wsl_update_x64.msi	https://wslstorestorage.blob.core.windows.net/wslblob/.	2021-11-03 21:49:23 CDT	windows.net
History			1	C:\Users\efudd\Downloads\wsl_update_x64.msi	https://wslstorestorage.blob.core.windows.net/wslblob/.	2021-11-03 22:02:51 CDT	windows.net
History			1	C:\Users\efudd\Downloads\TrueCrypt-7.2.exe	https://downloads.sourceforge.net/project/truecrypt/Tr..	2021-11-03 22:26:38 CDT	sourceforge.net
History			1	C:\Users\efudd\Downloads\TrueCrypt-7.2.exe	https://phoenixnap.dl.sourceforge.net/project/truecrypt.	2021-11-03 22:26:38 CDT	sourceforge.net
History			0	C:\Users\efudd\Downloads\VeraCrypt Setup 1.24-Updat	https://launchpad.net/veracrypt/trunk/1.24-update7/+d.	2021-11-03 22:32:44 CDT	launchpad.net
History			0	C:\Users\efudd\Downloads\VeraCrypt Setup 1.24-Updat	https://launchpadlibrarian.net/492504898/VeraCrypt%2(	2021-11-03 22:32:44 CDT	launchpadlibrarian.net
History			1	C:\Users\efudd\Downloads\Apache_OpenOffice_4.1.11_.	https://downloads.sourceforge.net/project/openofficeo.	2021-11-03 22:38:06 CDT	sourceforge.net
History			1	C:\Users\efudd\Downloads\Apache_OpenOffice_4.1.11_.	https://netactuate.dl.sourceforge.net/project/openoffice	2021-11-03 22:38:06 CDT	sourceforge.net

Figure 13. Autopsy Web Downloads showing efudd's downloads

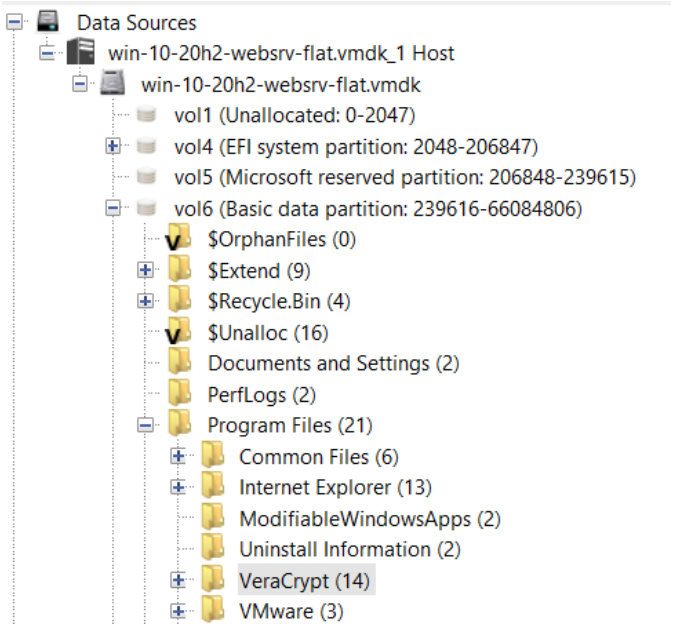
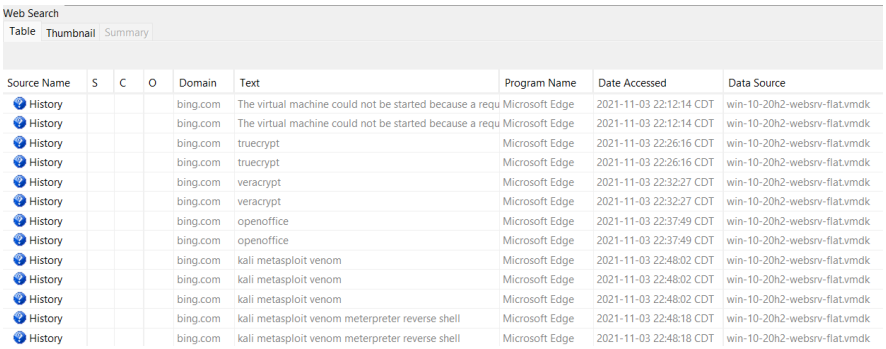


Figure 14. Autopsy view of Program Files showing installed VeraCrypt

14. According to web history, it appears that the user was researching how to generate malware. What was the program that they researched?

- **Analysis Performed:**
  - The examiner then went to the "Web Search" data artifact within Autopsy which shows information about what the user searched, as shown in Figure 15.
- **Answer:**
  It appears that the user was researching how to generate malware. The examiner determined that the suspect was researching **Metasploit's msfvenom payload generator**, based on the web searches, as shown in Figure 15.

- **Supporting Evidence:**



Web Search

Table | Thumbnail | Summary

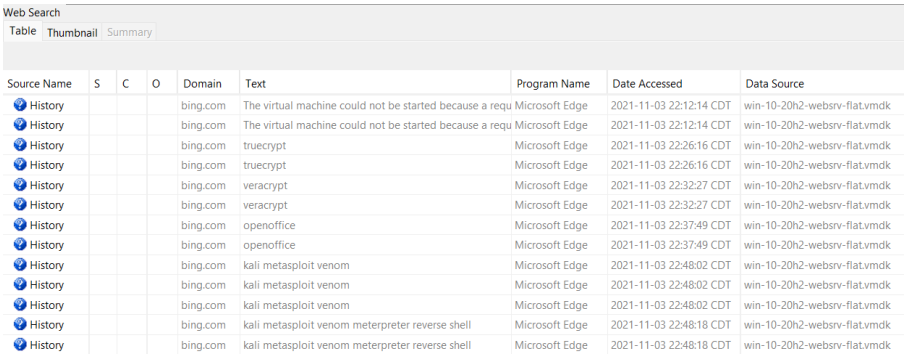
Source Name	S	C	O	Domain	Text	Program Name	Date Accessed	Data Source
History				bing.com	The virtual machine could not be started because a requ	Microsoft Edge	2021-11-03 22:12:14 CDT	win-10-20h2-websrv-flat.vmdk
History				bing.com	The virtual machine could not be started because a requ	Microsoft Edge	2021-11-03 22:12:14 CDT	win-10-20h2-websrv-flat.vmdk
History				bing.com	truecrypt	Microsoft Edge	2021-11-03 22:26:16 CDT	win-10-20h2-websrv-flat.vmdk
History				bing.com	truecrypt	Microsoft Edge	2021-11-03 22:26:16 CDT	win-10-20h2-websrv-flat.vmdk
History				bing.com	veracrypt	Microsoft Edge	2021-11-03 22:32:27 CDT	win-10-20h2-websrv-flat.vmdk
History				bing.com	veracrypt	Microsoft Edge	2021-11-03 22:32:27 CDT	win-10-20h2-websrv-flat.vmdk
History				bing.com	openoffice	Microsoft Edge	2021-11-03 22:37:49 CDT	win-10-20h2-websrv-flat.vmdk
History				bing.com	openoffice	Microsoft Edge	2021-11-03 22:37:49 CDT	win-10-20h2-websrv-flat.vmdk
History				bing.com	kali metasploit venom	Microsoft Edge	2021-11-03 22:48:02 CDT	win-10-20h2-websrv-flat.vmdk
History				bing.com	kali metasploit venom	Microsoft Edge	2021-11-03 22:48:02 CDT	win-10-20h2-websrv-flat.vmdk
History				bing.com	kali metasploit venom	Microsoft Edge	2021-11-03 22:48:02 CDT	win-10-20h2-websrv-flat.vmdk
History				bing.com	kali metasploit venom meterpreter reverse shell	Microsoft Edge	2021-11-03 22:48:18 CDT	win-10-20h2-websrv-flat.vmdk
History				bing.com	kali metasploit venom meterpreter reverse shell	Microsoft Edge	2021-11-03 22:48:18 CDT	win-10-20h2-websrv-flat.vmdk

Figure 15. Autopsy Web Search

15. The user did not install this program on their Windows system, where did they get it from?

- **Analysis Performed:**
  - The examiner then went to the "Web Search" data artifact within Autopsy which shows information about what the user searched, as shown in Figure 16.
- **Answer:**
  The user did not install the msfvenom program on the Windows system. The suspect obtained it from Kali Linux (as previously discovered) running under Windows Subsystem for Linux (WSL). Metasploit and msfvenom come bundled with Kali Linux which the suspect is aware of, as shown in Figure 16.

- **Supporting Evidence:**



Web Search

Table | Thumbnail | Summary

Source Name	S	C	O	Domain	Text	Program Name	Date Accessed	Data Source
History				bing.com	The virtual machine could not be started because a requ	Microsoft Edge	2021-11-03 22:12:14 CDT	win-10-20h2-websrv-flat.vmdk
History				bing.com	The virtual machine could not be started because a requ	Microsoft Edge	2021-11-03 22:12:14 CDT	win-10-20h2-websrv-flat.vmdk
History				bing.com	truecrypt	Microsoft Edge	2021-11-03 22:26:16 CDT	win-10-20h2-websrv-flat.vmdk
History				bing.com	truecrypt	Microsoft Edge	2021-11-03 22:26:16 CDT	win-10-20h2-websrv-flat.vmdk
History				bing.com	veracrypt	Microsoft Edge	2021-11-03 22:32:27 CDT	win-10-20h2-websrv-flat.vmdk
History				bing.com	veracrypt	Microsoft Edge	2021-11-03 22:32:27 CDT	win-10-20h2-websrv-flat.vmdk
History				bing.com	openoffice	Microsoft Edge	2021-11-03 22:37:49 CDT	win-10-20h2-websrv-flat.vmdk
History				bing.com	openoffice	Microsoft Edge	2021-11-03 22:37:49 CDT	win-10-20h2-websrv-flat.vmdk
History				bing.com	kali metasploit venom	Microsoft Edge	2021-11-03 22:48:02 CDT	win-10-20h2-websrv-flat.vmdk
History				bing.com	kali metasploit venom	Microsoft Edge	2021-11-03 22:48:02 CDT	win-10-20h2-websrv-flat.vmdk
History				bing.com	kali metasploit venom	Microsoft Edge	2021-11-03 22:48:02 CDT	win-10-20h2-websrv-flat.vmdk
History				bing.com	kali metasploit venom meterpreter reverse shell	Microsoft Edge	2021-11-03 22:48:18 CDT	win-10-20h2-websrv-flat.vmdk
History				bing.com	kali metasploit venom meterpreter reverse shell	Microsoft Edge	2021-11-03 22:48:18 CDT	win-10-20h2-websrv-flat.vmdk

Figure 16. Autopsy Web Search

16. When was the setup executable for the more modern encryption program ran (according to prefetch)?

- **Analysis Performed:**
  - The examiner then went to the "Run Programs" data artifact within Autopsy which shows information about the programs that were ran on the system, as shown in Figure 17.
- **Answer:**
  The setup executable for the more modern encryption program (VeraCrypt) was ran on **2021-11-03 22:33:01 CDT**, as shown in Figure 17.

- **Supporting Evidence:**

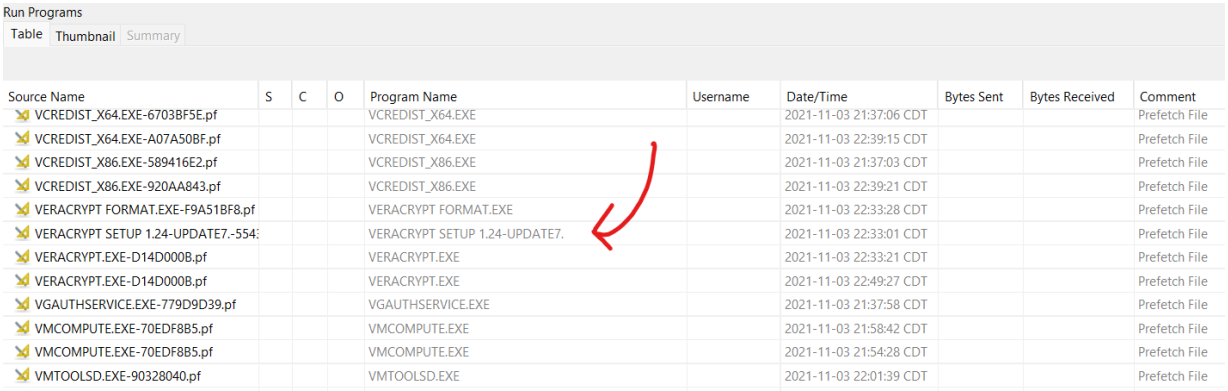

Run Programs

Table | Thumbnail | Summary

Source Name	S	C	O	Program Name	Username	Date/Time	Bytes Sent	Bytes Received	Comment
VCREDIST_X64.EXE-6703BF5E.pf				VCREDIST_X64.EXE		2021-11-03 21:37:06 CDT			Prefetch File
VCREDIST_X64.EXE-A07A50BF.pf				VCREDIST_X64.EXE		2021-11-03 22:39:15 CDT			Prefetch File
VCREDIST_X86.EXE-589416E2.pf				VCREDIST_X86.EXE		2021-11-03 21:37:03 CDT			Prefetch File
VCREDIST_X86.EXE-920AA843.pf				VCREDIST_X86.EXE		2021-11-03 22:39:21 CDT			Prefetch File
VERACRYPT FORMAT.EXE-F9A51BF8.pf				VERACRYPT FORMAT.EXE		2021-11-03 22:33:28 CDT			Prefetch File
VERACRYPT SETUP 1.24-UPDATE7.-554:				VERACRYPT SETUP 1.24-UPDATE7.		2021-11-03 22:33:01 CDT			Prefetch File
VERACRYPT.EXE-D14D000B.pf				VERACRYPT.EXE		2021-11-03 22:33:21 CDT			Prefetch File
VERACRYPT.EXE-D14D000B.pf				VERACRYPT.EXE		2021-11-03 22:49:27 CDT			Prefetch File
VGAUTHSERVICE.EXE-779D9D39.pf				VGAUTHSERVICE.EXE		2021-11-03 21:37:58 CDT			Prefetch File
VMCOMPUTE.EXE-70EDF8B5.pf				VMCOMPUTE.EXE		2021-11-03 21:58:42 CDT			Prefetch File
VMCOMPUTE.EXE-70EDF8B5.pf				VMCOMPUTE.EXE		2021-11-03 21:54:28 CDT			Prefetch File
VMTOOLSD.EXE-90328040.pf				VMTOOLSD.EXE		2021-11-03 22:01:39 CDT			Prefetch File

Figure 17. Autopsy Run Programs artifact showing the prefetch entry for VERACRYPT SETUP 1.24-UPDATE7.EXE and its execution time.

17. The user is using this encryption program to secure their operation. They have a container file on the hard drive. What is the name of the file? Where is it located?

- **Analysis Performed:**
  - The examiner then went to the "Encryption Suspected" analysis result artifact within Autopsy which shows information about the operating system, as shown in Figure 18.
- **Answer:**
  The user is using VeraCrypt to secure their operation. There is a container file on the hard drive and the name of the file is **2021-Mar-Invoice.ods** located at within the **Documents directory of user, efudd**, as shown in Figure 18.

- **Supporting Evidence:**


Encryption Suspected

Table | Thumbnail | Summary

Source Name	S	C	O	Source Type	Score	Conclusion	File Path	Justification
mpcache-740D2ABA8E09D339AF1411CF1EEED2A1			0	File	Likely Notable		/img_win-10-20h2-websrv-flat.vmdk/vol_vol6/ProgramData/Microsoft/Windows Defender/Scans/	Suspected encryption due to high entropy (7.631038).
mpenginedb.db			0	File	Likely Notable		/img_win-10-20h2-websrv-flat.vmdk/vol_vol6/ProgramData/Microsoft/Windows Defender/Scans/	Suspected encryption due to high entropy (7.983179).
2021-Mar-Invoice.ods			0	File	Likely Notable		/img_win-10-20h2-websrv-flat.vmdk/vol_vol6/Users/efudd/Documents/2021-Mar-Invoice.ods	Suspected encryption due to high entropy (8.000000).

Figure 18. Autopsy "Encryption Suspected" results flagging three high-entropy files, including 2021-Mar-Invoice.ods, as likely encrypted

Decryption

18. What is the password in plaintext?

- **Analysis Performed:**
  - The examiner then went to the Linux system (Greymhatter machine) to execute the following command, as shown in Figure 19:
    - *dd if=2021-Mar-Invoice.ods of=hash.txt bs=1 count=512*
    - This command extracts the hash data from the encrypted file.
  - The examiner then proceeded to hashcat in order to find the correct command in order to crack the password. The examiner selected mode, 13721, and understood the charset instructions, as shown in Figures 20 and 21.
  - The examiner then ran the hashcat command to crack the password, as shown in Figure 22.
    - *.\hashcat.exe -m 13721 -a 3 -o crackeddd.txt ..\hash.txt 'wabbit?h?h?h!'*
  - Then, the examiner read the contents of the "crackeddd.txt" file and it shows the password in plaintext, as shown in Figure 23.
- **Answer:**
  The password in plaintext is "**wabbit123!**", as shown in Figure 23.

- **Supporting Evidence:**

```
Evidence/is4523-capstone/win-10-20h2-websrv ❯ dd if=2021-Mar-Invoice.ods of=hash.txt bs=1 count=512
512+0 records in
512+0 records out
512 bytes copied, 0.0181537 s, 28.2 kB/s
```

Figure 19. dd command extracting the first 512 bytes of 2021-Mar-Invoice.ods into hash.txt.

```
 29461 | VeraCrypt SHA256 + XTS 512 bit + boot-mode                 | Full-Disk Encryption (FDE)
 29462 | VeraCrypt SHA256 + XTS 1024 bit + boot-mode                | Full-Disk Encryption (FDE)
 29463 | VeraCrypt SHA256 + XTS 1536 bit + boot-mode                | Full-Disk Encryption (FDE)
 13721 | VeraCrypt SHA512 + XTS 512 bit (legacy)                    | Full-Disk Encryption (FDE)
 13722 | VeraCrypt SHA512 + XTS 1024 bit (legacy)                   | Full-Disk Encryption (FDE)
 13723 | VeraCrypt SHA512 + XTS 1536 bit (legacy)                   | Full-Disk Encryption (FDE)
 29421 | VeraCrypt SHA512 + XTS 512 bit                             | Full-Disk Encryption (FDE)
 29422 | VeraCrypt SHA512 + XTS 1024 bit                            | Full-Disk Encryption (FDE)
```

Figure 20. Hashcat hash-mode list highlighting the VeraCrypt SHA512 + XTS (legacy) mode.

```
- [ Built-in Charsets ] -

  ? | Charset
 ===+=========
  l | abcdefghijklmnopqrstuvwxyz [a-z]
  u | ABCDEFGHIJKLMNOPQRSTUVWXYZ [A-Z]
  d | 0123456789                 [0-9]
  h | 0123456789abcdef           [0-9a-f]
  H | 0123456789ABCDEF           [0-9A-F]
  s |  !"#$%&'()*+,-./:;<=>?@[\]^_`{|}~
  a | ?l?u?d?s
  b | 0x00 - 0xff
```

Figure 21. Hashcat built-in charset table showing mask symbols like ?l, ?u, and ?d.

```
PS D:\UTSA\Fall 2025\Digital Forensics II\Labs\Capstone\hashcat-7.1.2> .\hashcat.exe -m 13721 -a 3 -o crackeddd.txt ..\h
ash.txt 'wabbit?h?h?h!'
hashcat (v7.1.2) starting

CUDA API (CUDA 13.0)
====================
* Device #01: NVIDIA GeForce RTX 3080, 9071/10239 MB, 68MCU

OpenCL API (OpenCL 3.0 CUDA 13.0.97) - Platform #1 [NVIDIA Corporation]
=======================================================================
* Device #02: NVIDIA GeForce RTX 3080, skipped

Minimum password length supported by kernel: 0
Maximum password length supported by kernel: 128
Minimum salt length supported by kernel: 0
Maximum salt length supported by kernel: 256

Hashes: 1 digests; 1 unique digests, 1 unique salts
Bitmaps: 16 bits, 65536 entries, 0x0000ffff mask, 262144 bytes, 5/13 rotates
```

Figure 22. Hashcat brute-force run against hash.txt using the wabbit???! password mask.

```
PS D:\UTSA\Fall 2025\Digital Forensics II\Labs\Capstone\hashcat-7.1.2> cat .\crackeddd.txt
..\hash.txt:wabbit123!
```

Figure 23. Contents of crackeddd.txt showing the recovered password wabbit123!

19. What was the exact command you used to crack the password?

- **Analysis Performed:**
  - The examiner conducted research on the correct command to use to crack the password, as previously mentioned.
  - The examiner then ran the hashcat command to crack the password, as shown in Figure 24.
    - *.\hashcat.exe -m 13721 -a 3 -o crackeddd.txt ..\hash.txt 'wabbit?h?h?h!'*
- **Answer:**
  The exact command the examiner used to crack the password was, **“.\hashcat.exe -m 13721 -a 3 -o crackeddd.txt ..\hash.txt ‘wabbit?h?h?h!’**”, as shown in Figure 24. The examiner came to this command from previous analysis discussed in the previous question.

- **Supporting Evidence:**

```
PS D:\UTSA\Fall 2025\Digital Forensics II\Labs\Capstone\hashcat-7.1.2> .\hashcat.exe -m 13721 -a 3 -o crackeddd.txt ..\h
ash.txt 'wabbit?h?h?h!'
hashcat (v7.1.2) starting

CUDA API (CUDA 13.0)
====================
* Device #01: NVIDIA GeForce RTX 3080, 9071/10239 MB, 68MCU

OpenCL API (OpenCL 3.0 CUDA 13.0.97) - Platform #1 [NVIDIA Corporation]
=======================================================================
* Device #02: NVIDIA GeForce RTX 3080, skipped

Minimum password length supported by kernel: 0
Maximum password length supported by kernel: 128
Minimum salt length supported by kernel: 0
Maximum salt length supported by kernel: 256

Hashes: 1 digests; 1 unique digests, 1 unique salts
Bitmaps: 16 bits, 65536 entries, 0x0000ffff mask, 262144 bytes, 5/13 rotates
```

Figure 24. Hashcat brute-force run against hash.txt using the wabbit???! password mask.

20. What was the mask you used to crack the password?

- **Analysis Performed:**
  - The examiner used the -hh command to find out the built-in charsets in order to create an extensive mask to crack the password, as shown in Figure 26.
  - The examiner then ran the hashcat command to crack the password, as shown in Figure 25.
    - *.\hashcat.exe -m 13721 -a 3 -o crackeddd.txt ..\hash.txt 'wabbit?h?h?h!'*
- **Answer:**
  The mask that the examiner used to crack the password was, "**wabbit?h?h?h!**", as shown in Figure 25. The reason for why the examiner went with the ?h?h?h charset was because it included all of the numbers and also abcdef, as shown in Figure 26.

- **Supporting Evidence:**

```
PS D:\UTSA\Fall 2025\Digital Forensics II\Labs\Capstone\hashcat-7.1.2> .\hashcat.exe -m 13721 -a 3 -o crackeddd.txt ..\h
ash.txt 'wabbit?h?h?h!'
hashcat (v7.1.2) starting

CUDA API (CUDA 13.0)
====================
* Device #01: NVIDIA GeForce RTX 3080, 9071/10239 MB, 68MCU

OpenCL API (OpenCL 3.0 CUDA 13.0.97) - Platform #1 [NVIDIA Corporation]
=======================================================================
* Device #02: NVIDIA GeForce RTX 3080, skipped

Minimum password length supported by kernel: 0
Maximum password length supported by kernel: 128
Minimum salt length supported by kernel: 0
Maximum salt length supported by kernel: 256

Hashes: 1 digests; 1 unique digests, 1 unique salts
Bitmaps: 16 bits, 65536 entries, 0x0000ffff mask, 262144 bytes, 5/13 rotates
```

Figure 25. Hashcat brute-force run against hash.txt using the wabbit???! password mask

```
- [ Built-in Charsets ] -

  ? | Charset
 ===+=========
  l | abcdefghijklmnopqrstuvwxyz [a-z]
  u | ABCDEFGHIJKLMNOPQRSTUVWXYZ [A-Z]
  d | 0123456789                 [0-9]
  h | 0123456789abcdef           [0-9a-f]
  H | 0123456789ABCDEF           [0-9A-F]
  s |  !"#$%&'()*+,-./:;<=>?@[\]^_`{|}~
  a | ?l?u?d?s
  b | 0x00 - 0xff
```

Figure 26. Hashcat built-in charset table showing mask symbols like ?l, ?u, and ?d.

Encrypted Files

21. What is the filename of all the files in the container?

- **Analysis Performed:**
  - The examiner then downloaded and installed VeraCrypt, from there, the examiner mounted the encrypted file container and inputted the password found as found previously.
  - The examiner accessed the mounted encrypted file container, as shown in Figure 27.
- **Answer:**
  The filename of all of the files in the container are: **invoice.exe** and **target.txt**, as shown in Figure 27

- **Supporting Evidence:**

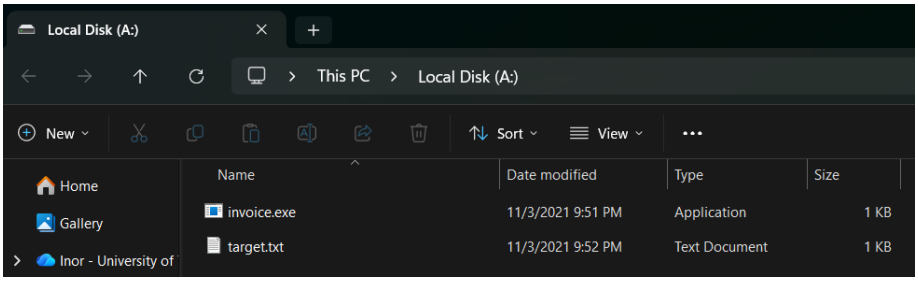


Figure 27. The files within the encrypted file container

22. What are the SHA1 hashes of each file?

- **Analysis Performed:**
  - The examiner then proceeded to the terminal within the folder to get the SHA1 hash of each file, as shown in Figure 28.
- **Answer:**
  The SHA1of invoice.exe is: **FF11C4CA855391031E2F5D38F8EBBF90E9E7D1B5**, as shown in Figure 28.
  The SHA1 of target.txt is **4BABAE25A000D6E3A6C95ED3FF9A1F4A24AC956E**, as shown in Figure 28.

- **Supporting Evidence:**

```
PS A:\> Get-FileHash -a sha1 '.\invoice.exe'

Algorithm       Hash                                                                   Path
---------       ----                                                                   ----
SHA1            FF11C4CA855391031E2F5D38F8EBBF90E9E7D1B5                               A:\invoice.exe

PS A:\> Get-FileHash -a sha1 '.\target.txt'

Algorithm       Hash                                                                   Path
---------       ----                                                                   ----
SHA1            4BABAE25A000D6E3A6C95ED3FF9A1F4A24AC956E                               A:\target.txt
```

Figure 28. PowerShell Get-FileHash output showing SHA1 hashes for A:\invoice.exe and A:\target.txt

23. Submit the executable's hash to VirusTotal. How many hits did the hash have?

- **Analysis Performed:**
  - The examiner then copy and pasted the file hash of invoice.exe to VirusTotal, as shown in Figure 29.
- **Answer:**
  After the examiner submitted invoice.exe's SHA1 file has to VirusTotal, the hash had 24 hits, as shown in Figure 29.

- **Supporting Evidence:**

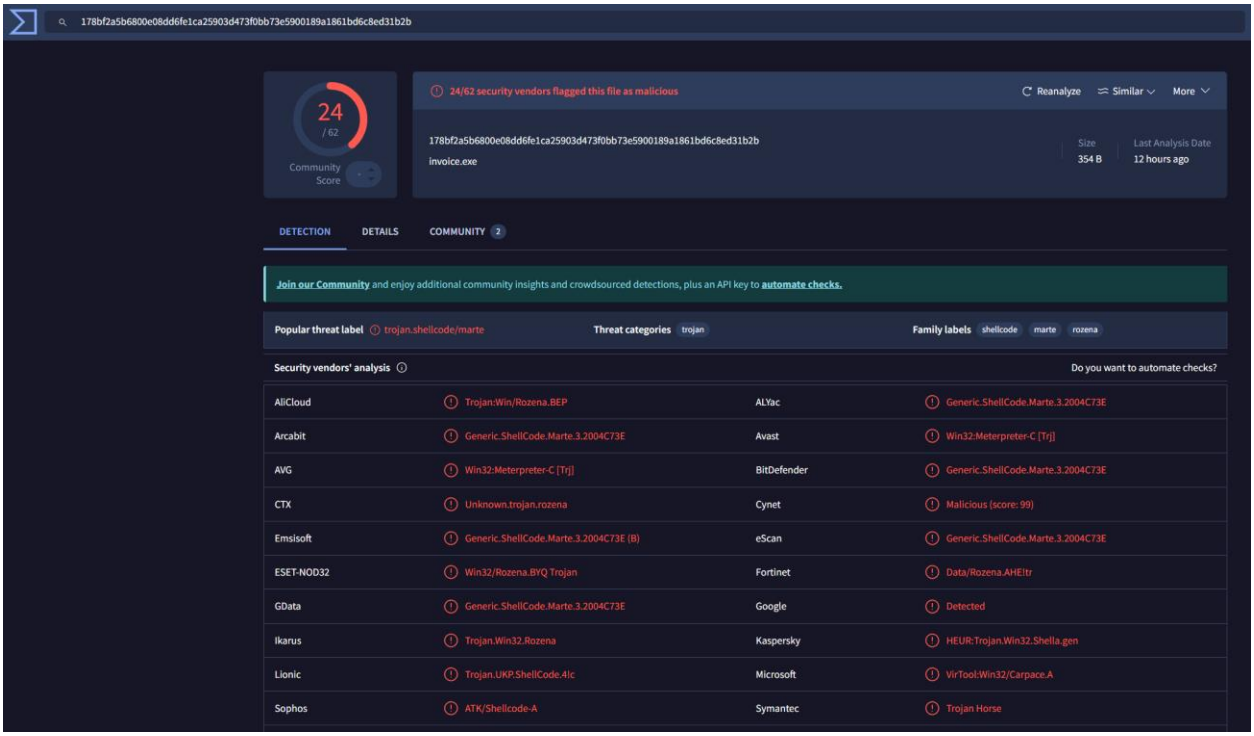


Figure 29. VirusTotal results showing invoice.exe flagged as malicious by 24/62 security vendors, with listed vendor detection names and threat labels

24. Submit the actual executable to VirusTotal. It is malicious? What would you classify the file as?

- **Analysis Performed:**
  - The examiner submitted the executable to VirusTotal and reviewed the overall detection ratio and threat label.
  - The examiner reviewed the Security vendors' analysis list to identify common detection names and malware family indicators (e.g., shellcode, Meterpreter, Rozena).
  - The examiner reviewed the Community / YARA results to identify signature-based matches and supporting context.
  - The examiner documented the THOR APT Scanner YARA matches, including rule names, rule sets, timestamps, and associated tags.
  - The examiner correlated the YARA matches with common offensive framework indicators (Meterpreter stagers and Cobalt Strike resources) to support classification.
- **Answer:**
  Based on VirusTotal results and corroborating YARA signature matches, the examiner determined the executable is malicious. Multiple security vendors flagged the file and identified indicators consistent with Metasploit Meterpreter stagers, and community YARA matches also referenced Cobalt Strike–related resources. The examiner would classify invoice.exe as a Trojan/backdoor stager (hacktool), most consistent with a Meterpreter stager and/or Cobalt Strike loader component, indicating it is likely intended to establish an initial foothold and facilitate remote access.

- **Supporting Evidence:**

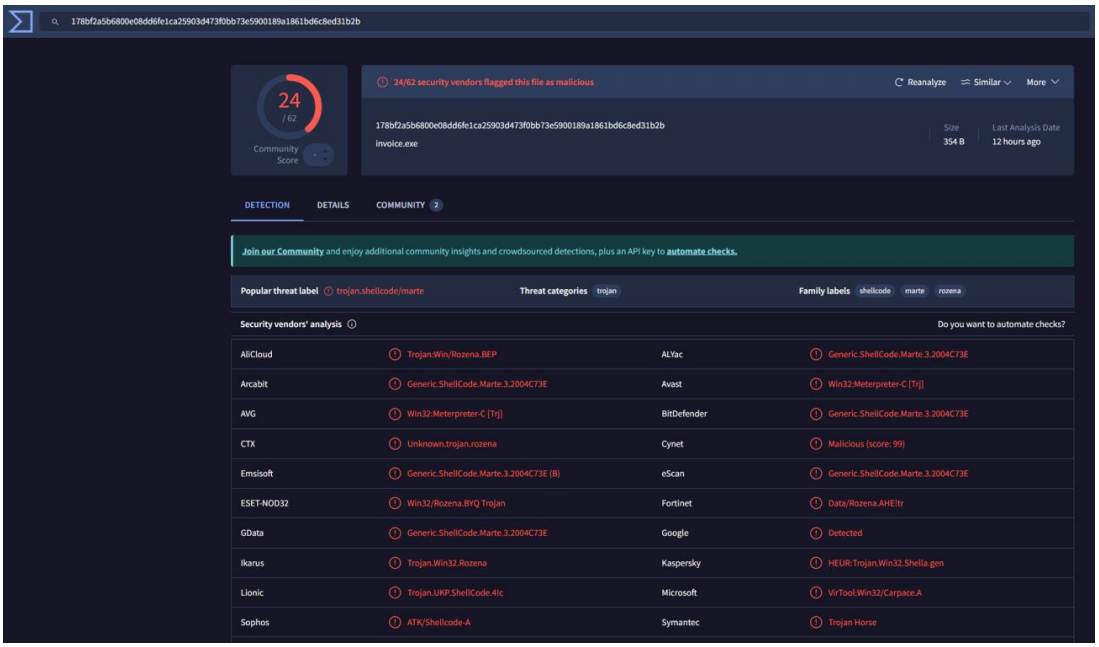


Figure 30. VirusTotal results showing invoice.exe flagged as malicious by 24/62 security vendors, with listed vendor detection names and threat labels

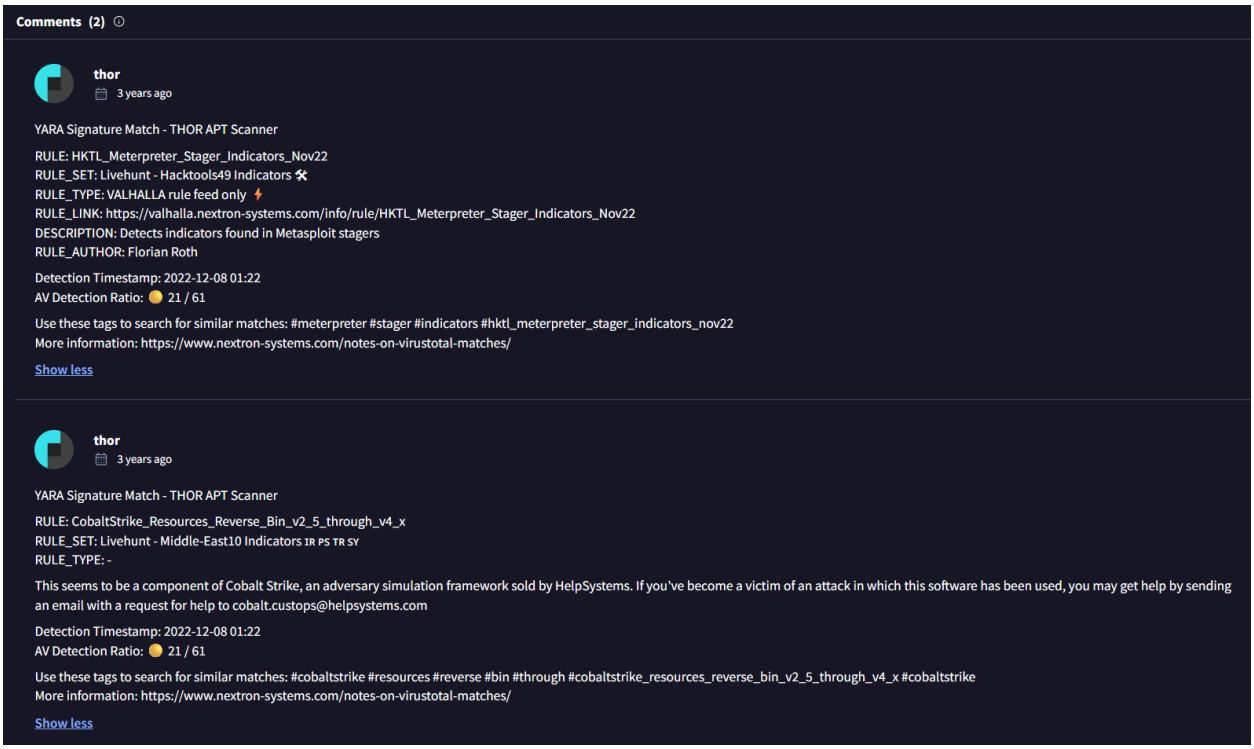


Figure 31. VirusTotal community YARA matches indicating Meterpreter stager and Cobalt Strike indicators.

25. What was the target?

- **Analysis Performed:**
  - The examiner then accessed “target.txt” to find out who the target was, as shown in Figure 32.
- **Answer:**
  The target was **bbunny@acmecorp.com**, as shown in Figure 32

- **Supporting Evidence:**

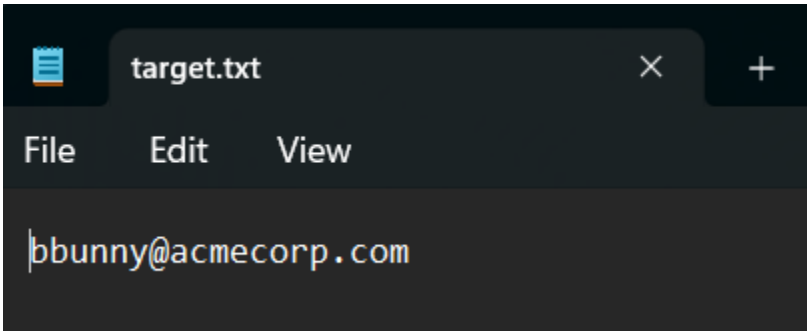


Figure 32. Text editor view of target.txt containing the email address bbunny@acmecorp.com

Conclusion

The examiner, Inor Wang, enjoyed this lab. There is no critique from me. Thank you.

References

Carvey, H. A. (2014). Windows forensic analysis toolkit: Advanced analysis techniques for Windows 8 (Fourth edition). Syngress.

Johansen, G., & Safari, an O. M. C. (2020). Digital Forensics and Incident Response—Second Edition.

Ligh, M. H., Case, A., Levy, J., & Walters, A. (2014). The art of memory forensics: Detecting malware and threats in Windows, Linux, and Mac memory. Wiley.

Malware forensics field guide for Windows systems digital forensics field guides. (2012). Syngress.

Oettinger, W., & Safari, an O. M. C. (2020). Learn Computer Forensics.

Reddy, N. (2019). Practical cyber forensics: An incident-based approach to forensic investigations. APress. https://doi.org/10.1007/978-1-4842-4460-9.

VeraCrypt Project. (2025). *VeraCrypt*. https://veracrypt.io/en/Downloads.html.