



DIGITAL FORENSICS- ASSIGNMENT # 4

SUBMITTED BY: FARWAH HAMID (FA22-BCT-007)

SUBMITTED TO: SIR MUSTAFA KHATTAK

15TH MAY 2024

Contents

1. Examining a Forensic Image with Autopsy.....	5
Brief Summary:	5
Introduction:	5
Case information:.....	5
Tools used:	5
Procedure:	5
Data Artifacts and Analysis:	6
Conclusion:	6
2. Network Forensic using Wireshark.....	7
Brief Summary:	7
Introduction:	7
Tools Utilized:	7
Procedure:	7
Data Artifacts and Analysis:	8
Conclusion:	9
3. Rhino Hunt with Autopsy.	10
Brief Summary:	10
Introduction:	10
Tools Used:	10
Procedure:	10
Data Artifacts and Analysis:	11
Conclusion:	11
4. Rhino Hunt with Wireshark.....	12
Brief Summary:	12
Introduction:	12
Tools Used:	12
Procedure:	12
Data Artifacts and Analysis:	13
Conclusion:	13
5. Memory Analysis with Autopsy:.....	14
Brief Summary:	14
Introduction:	14

Tools Used:	14
Procedure:	14
Data Artifacts and Analysis:	15
Conclusion:	16
6. Memory forensics of LastPass and Keeper.	17
Brief Summary:	17
Tools Used:	17
Procedure:	17
Data Artifacts and Analysis:	18
Conclusion:	19
7. Capturing and examining the registry.	20
Brief Summary:	20
Tools Used:	20
Procedure:	20
Conclusion:	20
Data Artifacts and Analysis:	20
Conclusion:	20
8. Examining a windows disk image.	21
Brief Summary:	21
Case Information:	21
Tools Used:	21
Procedure:	21
Data artifacts and analysis:	22
Conclusion:	23
9. Email Forensics.	24
Brief Summary:	24
Introduction:	24
Tools Used:	24
Procedure:	24
Data Artifacts and Analysis:	25
Conclusion:	26
10. Android Studio Emulator.	27
Brief Summary:	27

Tools Used:	27
Procedure:	27
Data Artifacts and Analysis:	28
Conclusion:	29
11&14. Rooting Android Studio's Emulator.	30
Brief Summary:	30
Tools Used:	30
Procedure:	30
Data Artifacts and Analysis:	30
Conclusion:	31
12. Forensics Acquisition from Android.	32
Brief Summary:	32
Tools Used:	32
Procedure:	32
Data Artifacts and Analysis:	33
Conclusion:	34
13. Android Analysis with Autopsy.	35
Brief Summary:	35
Tools Used:	35
Procedure:	35
Data Artifacts and Analysis:	35
Conclusion:	36
15. iPhone Analysis with Autopsy.	37
Brief Summary:	37
Tools Used:	37
Procedure:	37
Data Artifacts and Analysis:	37
Conclusion:	38
16. Windows and Linux Machines.	39
Brief Summary:	39
Procedure:	39
Conclusion:	40
17. Velociraptor on Linux.	41

Brief Summary:	41
Tools Used:	41
Procedure:	41
Data Artifacts and Analysis:	42
Conclusion:	42
18. Investigating a PUP with Velociraptor.	43
Brief Summary:	43
Tools Used:	43
Procedure:	43
Data Artifacts and Analysis:	44
Conclusion:	44
19. Investigating a BOT with Velociraptor.	45
Brief Summary:	45
Tools Used:	45
Procedure:	45
Data Artifacts and Analysis:	46
Conclusion:	46
20. Investigating a Two-Stage RAT with Velociraptor.	47
Brief Summary:	47
Tools Used:	47
Procedure:	47
Data Artifacts and Analysis:	48
Conclusion:	48

1. Examining a Forensic Image with Autopsy.

Brief Summary:

This report outlines the procedures involved in examining a forensic image using autopsy. It involves installation of autopsy, downloading the case file, creation of case and finding the flag from within the downloaded image.

Introduction:

Autopsy is an open-source image used for analyzing hard drives, smartphones, virtual machine disk images, logical files, and other storage devices.

Case information:

Case name: Final1

Case number: 1

Evidence file: F200.E01

Tools used:

Autopsy.

Procedure:

1. Downloading the evidence file:

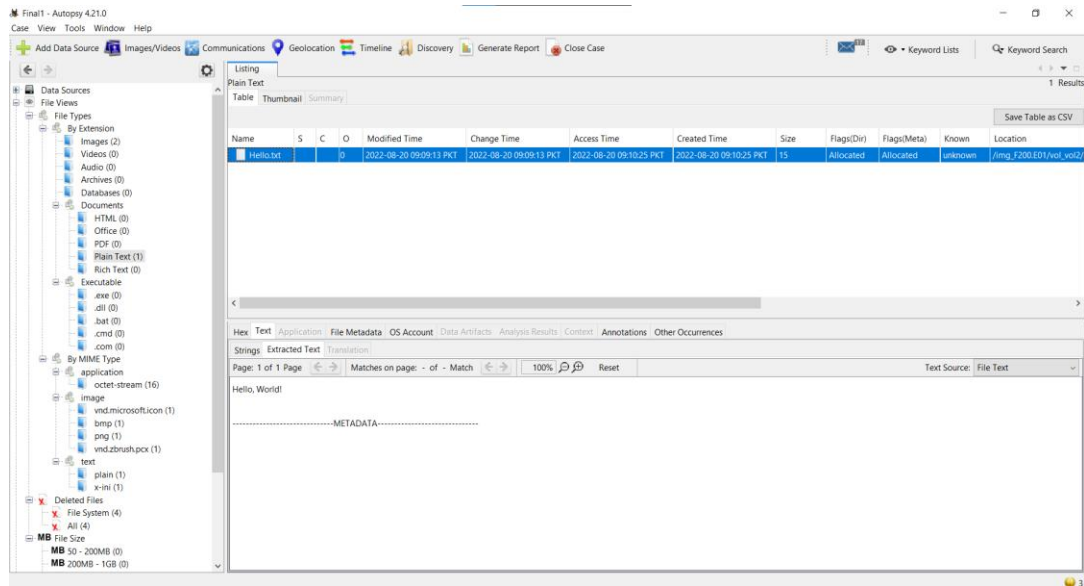
Downloading the evidence file **"F200.E01"**.

2. Creating a case in autopsy:

- Creating a new case named **"Final1"**.
- Select the **"Case Stuff"** as case location and assigned it as **"1"**.
- Add the evidence file **"F200.E01"** to the case.
- Configure the ingest option and select all the check boxes.
- Finish the case setup process.

3. Investigation:

- Explore the tree view pane in autopsy.
- Doing keyword search to find a message starting with **"The flag is"**.
- Capture the image containing the flag and saving it for documentation process.



Data Artifacts and Analysis:

Task	Artifact	Tools Used	Description	Flag
Flag 1	File containing message	Autopsy	Searching for “the flag is” in all the files content.	The flag is EVIDENCE .

Conclusion:

The examination of the forensics image “F200.E01” we find the flag “EVIDENCE”. Autopsy proved to be very helpful in doing digital forensics investigation.

2. Network Forensic using Wireshark.

Brief Summary:

This report outlines the methodology employed for conducting network forensics utilizing Wireshark. The investigation centered on analyzing network traffic captured in pcapng files to uncover sensitive data such as FTP and HTTP passwords, identify encrypted communication protocols, ascertain the encryption tool employed, and scrutinize APT attacker traffic.

Introduction:

Wireshark serves as a robust network protocol analyzer utilized for various purposes including network troubleshooting, protocol development, and education. In this scenario, Wireshark was utilized to scrutinize captured network traffic to extract pertinent information and pinpoint security vulnerabilities.

Tools Utilized:

Wireshark

Procedure:

1. Examination of Layers 1-4:

"**FTPlogin.pcapng**" file was downloaded and opened in Wireshark. Analysis encompassed filtering the packet list, packet details, and packet bytes to comprehend the **TCP handshake, MAC addresses, IP addresses, and TCP port numbers. FTP packets** were filtered to identify FTP login procedures and extract passwords.

2. Retrieval of an FTP Password:

Packet filtering was conducted utilizing the **FTP protocol**. The login process for a user named "**john**" was identified, and John's password was extracted from the captured packets.

3. Retrieval of an HTTP Password:

"**Httplogin.pcapng**" file was downloaded and opened in Wireshark. Packet filtering was performed using the **HTTP protocol**. HTTP POST requests were identified to extract usernames and passwords.

4. Analysis of TCP Stream:

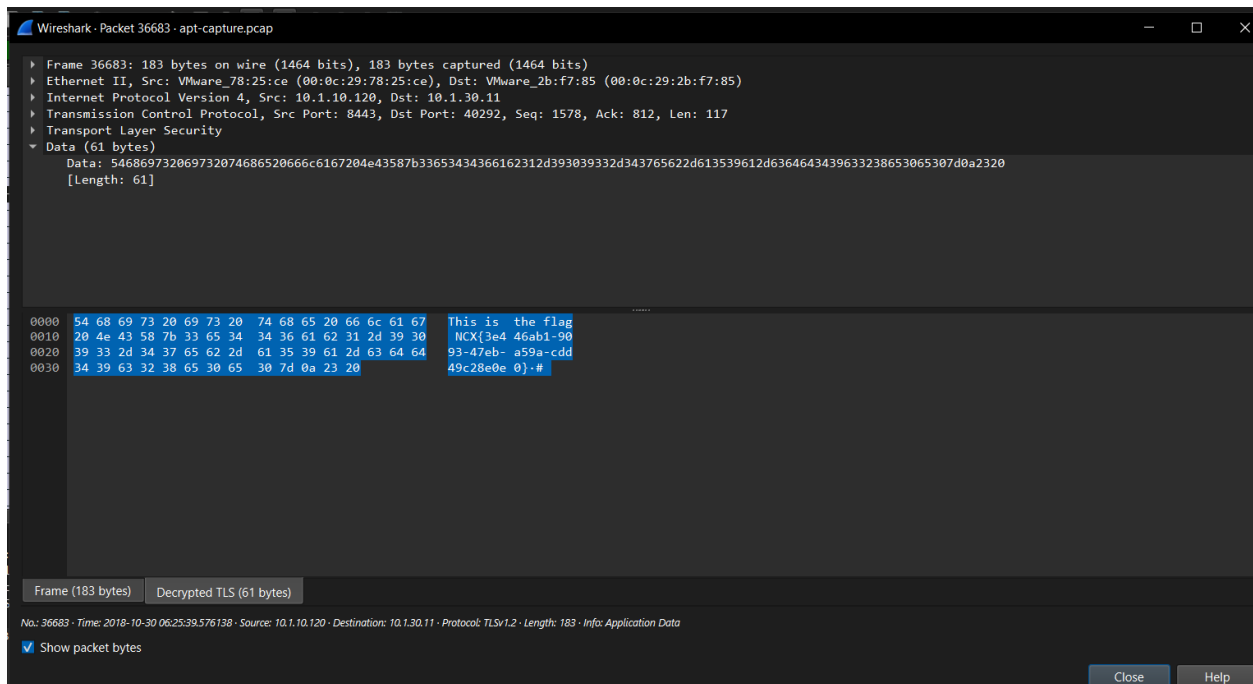
The first **POST request** was selected, and the TCP stream was followed to examine client-server communication. Efforts were made to identify passwords within the TCP stream.

5. Restoration of Packet Filter:

The filter was cleared to view all **HTTP packets**. HTTP replies were located and scrutinized to ascertain login success or failure.

6. Examination of APT Capture:

The "**apt-capture.pcap**" file containing APT attacker traffic was obtained. **Encrypted communication** protocols were identified. **The port number and encryption tool** were determined. The **sha1sum of the tool** was calculated. Communication was **decrypted** to uncover the private key and extract the flag. **Port knocking patterns** were analyzed, and the **sequence of ports** was identified. Shell sessions were investigated to identify the largest session and extract the last executed command.



Data Artifacts and Analysis:

Task	Artifact	Tools Used	Description	Flag
Finding FTP password	FTP Packets	Wireshark	Analyzed FTP Packets to extract user passwords	John's Password is "Flapper".
Finding HTTP password	HTTP Packets	Wireshark	Analyzed HTTP Packets to extract user passwords	Isaac's Password is "Flapper". Waldo's password is "VERYSCURE".
Encrypted Transmissions	Encrypted Packets	Wireshark	Analyzed encrypted packets to determine protocol	Server Port Number is 8443.
Tool Identification	Encrypted Packets	Wireshark	Determined the tool used for encryption	"Socat"
Hash Of Tool	Encrypted Packets	Wireshark	Calculated Sha1 sum of encrypted tool	Hash value = "da8ff40c6a60605c4d250 aae59912f1e60315c81"

Decrypting Communication	Encrypted Packets	Wireshark	Decrypted Communication to extract sensitive data	NCX{3e446ab1-9093-47eb-a59a-cdd49c28e0e0}
Port - Knocking	Network Traffic	Wireshark	Analyzed port knocking patterns	32001,33001,34001
Exploitation	Shell Session	Wireshark	Investigated shell sessions for exploitation	Stream ID = 75

Conclusion:

The analysis of network traffic utilizing Wireshark revealed notable security vulnerabilities, including insecure transmission of FTP and HTTP passwords, encrypted communication, port knocking activities, and attempts at shell exploitation. Using Wireshark's capabilities enabled the extraction of sensitive information and identification of potential security threats, telling the significance of network forensics in cybersecurity investigations.

3. Rhino Hunt with Autopsy.

Brief Summary:

This report details the steps taken to investigate a case involving the possession of illegal rhinoceros images using Autopsy. The investigation included verifying hash values, examining a disk image, identifying specific files, reading a diary, locating a missing hard drive, and identifying an email address associated with MIT.

Introduction:

The case involves the possession of illegal rhinoceroses' images, a serious offense according to a law passed in New Orleans in 2004. Evidence includes a disk image of a USB key and network traces provided by the network administrator. Autopsy, a digital forensics tool, was used for analysis.

Tools Used:

- Autopsy
- PowerShell

Procedure:

1. Verifying Hash Values:

- Used PowerShell to calculate **MD5 and SHA1 hash values** of the evidence file "**case1.zip**".
- Confirmed hash values matched expected values for data integrity.

2. Unzipping the Evidence File:

Extracted contents of "**case1.zip**" to access the disk image and network traces.

3. Creating an Autopsy Case:

- Opened Autopsy and created a new case named "**Final3**".
- Added disk image "**RHINOUSB.dd**" as a data source.
- Configured ingestion settings and completed setup.

4. Investigating the Evidence:

- Explored containers in Autopsy to view images and deleted files.
- Found image of a mother rhinoceros and her child.
- Examined deleted files, prioritizing "**application/msword**" files.
- Read diary file to uncover important information.
- Located missing hard drive within disk image.
- Identified real filename containing an email address associated with MIT.

Most of the rides we wanted to take were sold out, but we got to ride on a tall ship from 3-5, which is exactly what we wanted. I found this site that is full of surveys through some people who are now obsessed with the site. Rhino pictures illegal? Makes me sick. I "hid" the photos...hehehehe. Apparently, if there are less than 10 photos, it's no big deal. OK. Things are getting a little weird. I zapped the hard drive and then threw it into the Mississippi River. I'm gonna reformat my USB key after this entry, but try not to destroy the good stuff. I need to change the password on the gnome account that Jeremy gave me. I can probably just do that at Radio Shack.

Data Artifacts and Analysis:

Task	Artifact	Tools Used	Description	Flag
Verifying Hash Values	MD5 and SHA1 hash values	PowerShell	Calculated hash values to verify file authenticity	Hash values were matched
Investigating Deleted Files	Deleted files	Autopsy	Examined deleted files for valuable information	Found various deleted files including pictures of rhinoceros and crocodiles.
Reading the Diary	Diary file	Autopsy	Read diary to uncover pertinent information	Found location of hard drive and e-mail address
Locating Missing Hard Drive	Disk image analysis	Autopsy	Identified location of missing hard drive	The location of missing hard drive is "Mississippi river".
Email Address Identification	Recovered files	Autopsy	Identified real filename containing email address	Real filename is "f0103512.jpg".

Conclusion:

Autopsy investigation revealed important details regarding the possession of illegal rhinoceroses' images. Analysis of artifacts like deleted files, diary entries, and recovered emails provides valuable insights for further investigation and potential legal action.

4. Rhino Hunt with Wireshark.

Brief Summary:

This report explains how we used Wireshark to investigate the Rhino scenario involving illegal rhinoceros image activities. We looked at network traffic captured in log files to find **TELNET** and **FTP packets**, check **TELNET and FTP streams**, take out images from **FTP-DATA traffic**, and confirm **file hash values**.

Introduction:

In the Rhino scenario, we checked network traffic logs to find out about illegal rhinoceros image activities. We used Wireshark to do this, which helps to see what's happening on a network.

Tools Used:

- Wireshark
- PowerShell

Procedure:

1. Opening Rhino.log:

Opened "**rhino.log**" in Wireshark to see the network traffic.

2. Finding TELNET Packets:

Looked for TELNET packets in the traffic and saved them in a new file called "**telnet.pcap**".

3. Finding FTP Packets:

Searched for FTP packets and saved them in a new file called "**ftp.pcap**".

4. Examining TELNET Traffic:

Checked the "**telnet.pcap**" file and looked at the first packet's info to find a filename left as a message to John.

5. Examining FTP Traffic:

Did the same as with **TELNET traffic** to see what files were transferred. Found the filename of the third file transferred.

6. Extracting Images from FTP-DATA Traffic:

- In the original "**rhino.log**" file, we found **FTP-DATA** packets and saved them separately.
- Looked in these packets to find image data and saved it as "**rhino1.jpg**".
- Checked the image's integrity by checking its hash value.

7. Extracting ZIP Archive from FTP-DATA Traffic:

- Found and saved a **ZIP archive** from **FTP-DATA** streams.
- Unzipped the archive to get a password-protected image file.
- Checked the image's integrity by checking its hash value and found the flag.

```
PS D:\farwah uni\4th Semester\Digital Forensics\Final\case stuff\5th part> cd ..
PS D:\farwah uni\4th Semester\Digital Forensics\Final\case stuff> cd "4th part"
PS D:\farwah uni\4th Semester\Digital Forensics\Final\case stuff\4th part> ls

Directory: D:\farwah uni\4th Semester\Digital Forensics\Final\case stuff\4th part

Mode                LastWriteTime         Length Name
----                -
d-----          5/8/2024   10:10 PM             Z
-a-----          5/8/2024    1:45 AM       513751 FTP-DATA.pcap
-a-----          5/8/2024    1:24 AM        5252 FTP.pcap
-a-----          5/8/2024    1:56 AM       65703 rhino1.jpg
-a-----          4/26/2004    5:00 PM       230665 rhino2.jpg
-a-----          5/8/2024    1:22 AM       47317 telnet.pcap
-a-----          5/8/2024   10:05 PM       230566 Z.zip

PS D:\farwah uni\4th Semester\Digital Forensics\Final\case stuff\4th part> cd Z
PS D:\farwah uni\4th Semester\Digital Forensics\Final\case stuff\4th part\Z> ls

Directory: D:\farwah uni\4th Semester\Digital Forensics\Final\case stuff\4th part\Z

Mode                LastWriteTime         Length Name
----                -
-a-----          4/26/2004    5:00 PM       230665 rhino2.jpg

PS D:\farwah uni\4th Semester\Digital Forensics\Final\case stuff\4th part\Z> Get-FileHash -Algorithm MD5 .\rhino2.jpg
Algorithm Hash Path
-----
MD5 ED070202082EA4FD8F5488533A561B35 D:\farwah uni\4th Semester\Di...
```

Data Artifacts and Analysis:

Task	Artifact	Tools Used	Description	Flag
Analyzing TELNET Traffic	TELNET packets	Wireshark	Analyzed TELNET traffic for relevant information	Filename for message to John is "JOHNREADME".
Analyzing FTP Traffic	FTP packets	Wireshark	Analyzed FTP traffic for transferred filenames	Filename of third transferred file is "contraband.zip"
Extracting Images from FTP-DATA	FTP-DATA packets	Wireshark	Extracted images from FTP-DATA traffic	Rhino1.jpg
Verifying MD5 Hash of Images	Extracted image files	PowerShell	Verified integrity of extracted image files	Hash values are same.

Conclusion:

Using Wireshark in the Rhino scenario helped us find out about illegal rhinoceros image activities. We found filenames and transferred files from TELNET and FTP traffic and got images from FTP-DATA traffic. We checked the files were okay by looking at their hash values.

5. Memory Analysis with Autopsy:

Brief Summary:

This report explains how we used Autopsy to extract useful information from a memory image. We installed 7-Zip, checked the image's integrity, launched Autopsy, created a new case, imported the memory image, and analyzed different things to find important clues like passwords and executables.

Introduction:

We wanted to analyze a memory image to find important information like passwords and executables. We used Autopsy and other tools like 7-Zip and Online Hash Calculator to do this.

Tools Used:

- Autopsy
- 7-Zip
- Online Hash Calculator

Procedure:

1. Installing 7-Zip:

Downloaded and installed 7-Zip from its website.

2. Downloading the Evidence File:

- Got the memory image file "**memdump.7z**" and used 7-Zip to unzip it, getting "**memdump.mem**".
- Checked the file's integrity using Online Hash Calculator.

3. Launching Autopsy:

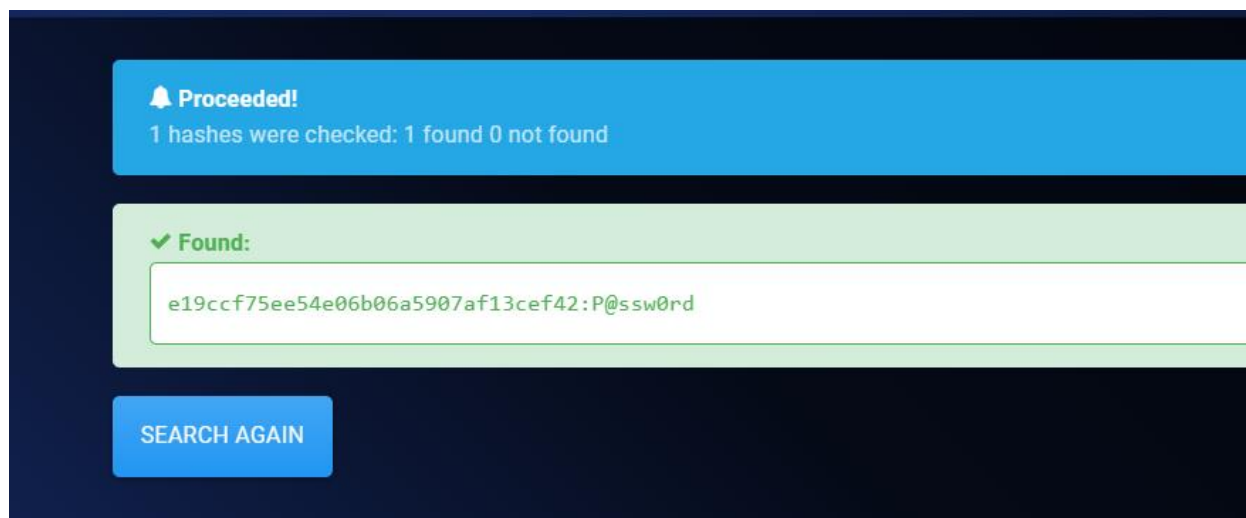
Opened Autopsy and turned on the Experimental module.

4. Creating a New Case:

Made a new case called "**Final5**" in Autopsy.

5. Importing the Memory Image:

- Added the memory image file "**memdump.mem**" to the case.
- Set it as a "Memory Image File (Volatility)".
- Set up the analysis process and looked at the **Module Output** section for important clues like passwords and executables.



Data Artifacts and Analysis:

Task	Artifact	Tools Used	Description	Flag
Consoles	Consoles section	Autopsy	Located Waldo's password in console logs	Waldo's password is "Apple123"
Hashdump	Hashdump section	Autopsy	Located Waldo's password hash in hashdump logs	Waldo's password hash is "cfeac129dc5e61b2eb9b27131fc7e2b"
LSAdump	LSAdump section	Autopsy	Identified default password in LSAdump logs	Default password is "P@ssw0rd"
Netscan	Netscan section	Autopsy	Identified executable listening on port 8080	Executable name is "ftpbasicsvr.exe"
Pslist	Pslist section	Autopsy	Identified running executable from pslist logs	Executable name is "FTK Imager.exe"
Shellbags	Shellbags section	Autopsy	Identified connected shared folder name	Shared folder name is "sawbowne"
Userassist	Userassist section	Autopsy	Identified dangerous executable from userassist logs	Executable name is "Poison Ivy 2.3.2.exe"
Probe Password	Hashdump section	Autopsy	Located probe account password hash	Probe account password is "P@ssw0rd".

Conclusion:

Analyzing the memory image with Autopsy gave us useful information about user activities, passwords, and executables.

6. Memory forensics of LastPass and Keeper.

Brief Summary:

This report talks about checking how LastPass and Keeper, two popular password managers, handle sensitive user data in memory. We installed some tools, set up accounts for both password managers, looked at their memory processes, and searched for sensitive data using Process Explorer and HxD.

Tools Used:

- Google Chrome
- Mozilla Firefox
- Process Explorer
- HxD Hex Editor

Procedure:

1. Installing Needed Software:

Installed Google Chrome, Mozilla Firefox, Process Explorer, and HxD Hex Editor on a Windows virtual machine.

2. Installing LastPass:

- Installed the LastPass browser extension in Google Chrome.
- Created a LastPass account with a disposable email address.
- Added a fake password entry for testing.

3. Exploring LastPass Memory Process:

- Opened Process Explorer and found the Chrome process with the LastPass extension.
- Noted the Process ID (PID) of the LastPass extension process.
- Opened HxD and checked the memory of the LastPass extension process.
- Searched for the test password **"P@ssw0rd"** in Unicode format.

4. Installing Keeper:

- Installed the Keeper browser extension in Mozilla Firefox.
- Created a Keeper account with a disposable email address.
- Added a login record with a password containing the string **"CCSF#"**.


5. Finding Keeper Memory Process:

- Looked at Firefox processes in Process Explorer.
- Looked for clues like DLL names with **"keeper security"** or **"form autofill"** or handles to File \Device\KsecDD.
- Searched for the string **"CCSF#"** in Unicode format in the identified Firefox process memory using HxD.

Handles	DLLs	Threads
Type	Name	
ALPC Port	\BaseNamedObjects\[CoreUI]-PID(4828)-TID(20944) d0a2497e-dd64-4489-bafa-5db8c6485...	
Desktop	\Default	
Directory	\KnownDlls	
Directory	\Sessions\18\BaseNamedObjects	
Event	\BaseNamedObjects\TermSrvReadyEvent	
Event	\KernelObjects\MaximumCommitCondition	
Event	\BaseNamedObjects\C::Users\Lenovo\AppData\Local\Microsoft\Windows\Explorer\iconcac...	
File	C:\Program Files\HxD	
File	C:\Windows\WinSxS\amd64_microsoft.windows.common-controls_6595b64144ccf1df_6.0.19...	
File	\Device\KsecDD	
File	\Device\CNG	
File	C:\Windows\System32\en-US\user32.dll.mui	
File	C:\Windows\WinSxS\amd64_microsoft.windows.c...controls.resources_6595b64144ccf1df_6...	
File	C:\Windows\WinSxS\amd64_microsoft.windows.c...controls.resources_6595b64144ccf1df_6...	
File	C:\Windows\Fonts\StaticCache.dat	
File	C:\Users\Lenovo\AppData\Local\Microsoft\Windows\Explorer\IconCacheToDelete\icnA0C1...	
File	C:\Windows\WinSxS\amd64_microsoft.windows.common-controls_6595b64144ccf1df_6.0.19...	
File	C:\Windows\WinSxS\amd64_microsoft.windows.common-controls_6595b64144ccf1df_6.0.19...	
File	C:\Users\Lenovo\AppData\Local\Microsoft\Windows\Explorer\IconCacheToDelete\icnA094...	
File	C:\Users\Lenovo\AppData\Local\Microsoft\Windows\Explorer\IconCacheToDelete\icnA094...	
File	C:\Users\Lenovo\AppData\Local\Microsoft\Windows\Explorer\IconCacheToDelete\icnA0C1...	
File	C:\Users\Lenovo\AppData\Local\Microsoft\Windows\Explorer\IconCacheToDelete\icnA0C0...	
CPU Usage: 16.17%		Commit Charge: 84.38%
Processes: 275		Physical Usage: 61.95%

HxD - [firefox.exe (3104)]

File Edit Search View Analysis Tools Window Help

 16 Windows (ANSI) hex

firefox.exe (3104)

Offset (h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Decoded text
016BAD756560	43	6B	65	65	70	65	72	73	65	63	75	72	69	74	79	2E	Ckeepersecurity.
016BAD756570	63	6F	6D	25	32	39	00	00	00	00	FF	00	00	00	FF	00	com%29...ÿ...ÿ..
016BAD756580	00	BB	58	31	FC	7F	00	00	00	01	00	00	00	00	00	00	..Xlû.....
016BAD756590	40	89	78	A9	6B	01	00	00	01	01	01	01	00	E5	E5	E5	@tux@k.....âââ
016BAD7565A0	38	7C	1F	31	FC	7F	00	00	00	00	00	00	00	01	00	02	8 .lû.....
016BAD7565B0	01	E5	E5	E5	E5	E5	E5	E5	E5	E5	E5	E5	E5	E5	E5	E5	..âââââââââââââââ
016BAD7565C0	00	BB	58	31	FC	7F	00	00	01	00	00	00	00	00	00	00	..Xlû.....
016BAD7565D0	40	89	78	A9	6B	01	00	00	01	01	01	01	00	E5	E5	E5	@tux@k.....âââ
016BAD7565E0	E5	E5	E5	E5	E5	E5	E5	E5	E5	E5	E5	E5	E5	E5	E5	E5	..âââââââââââââââ
016BAD7565F0	00	E5	E5	E5	E5	E5	E5	E5	E5	E5	E5	E5	E5	E5	E5	E5	..âââââââââââââââ
016BAD756600	02	00	00	00	38	00	00	00	5E	70	61	72	74	69	74	698...^partiti
016BAD756610	6F	6E	4B	65	79	3D	25	32	38	68	74	74	70	73	25	32	onKey=%29https%2
016BAD756620	43	6B	65	65	70	65	72	73	65	63	75	72	69	74	79	2E	Ckeepersecurity.
016BAD756630	63	6F	6D	25	32	39	00	00	FF	FF	00	B6	F3	FF	00	68	com%29..ÿÿ..%ôÿ.h
016BAD756640	01	00	00	00	38	00	00	00	43	00	43	00	53	00	46	008...C.C.S.F.
016BAD756650	23	00	66	00	61	00	32	00	32	00	2D	00	30	00	30	00	fr.a.2.2.-.0.0.
016BAD756660	37	00	00	00	E5	E5	E5	E5	E5	E5	E5	E5	E5	E5	E5	E5	7...âââââââââââââ
016BAD756670	E5	E5	E5	E5	E5	E5	E5	E5	E5	E5	E5	E5	E5	E5	E5	E5	âââââââââââââââââ
016BAD756680	A0	31	D8	A9	6B	01	00	00	00	00	00	00	00	00	00	00	l0@k.....
016BAD756690	00	00	00	00	00	00	00	00	06	00	00	00	00	00	00	00
016BAD7566A0	80	18	59	AC	6B	01	00	00	00	00	00	00	00	00	00	00	€.Y-k.....
016BAD7566B0	00	00	00	00	00	00	00	00	06	00	00	00	00	00	00	00
016BAD7566C0	E0	18	59	AC	6B	01	00	00	00	00	00	00	00	00	00	00	â.Y-k.....
016BAD7566D0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
016BAD7566E0	40	19	59	AC	6B	01	00	00	00	00	00	00	00	00	00	00	@.Y-k.....
016BAD7566F0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
016BAD756700	40	19	59	AC	6B	01	00	00	00	00	00	00	00	00	00	00	@.Y-k.....
016BAD756710	00	00	00	00	00	00	00	00	06	CF	3B	AB	6B	01	00	00I;ek...
016BAD756720	F0	0D	6E	AD	6B	01	00	00	03	00	00	00	00	00	00	00	8.n.k.....
016BAD756730	00	00	00	00	00	00	00	00	06	CF	3B	AB	6B	01	00	00I;ek...
016BAD756740	40	19	59	AC	6B	01	00	00	00	00	00	00	00	00	00	00	@.Y-k.....

Data Artifacts and Analysis:

Task	Artifact	Tools Used	Description	Flag
LastPass Memory Analysis	Memory process	Process Explorer, HxD	Searched for test password string “P@ssw0rd”	The flag is “Excerpt”

Keeper Memory Analysis	Memory process	Process Explorer, HxD	Searched for string "CCSF#" in memory process	The flag is "43 00 43 00"
------------------------------	-------------------	--------------------------	---	----------------------------------

Conclusion:

The check on LastPass and Keeper password managers showed potential security risks in how they handle sensitive data in memory. By looking at memory processes, we found instances where plaintext passwords were present, which could be a threat to user security. More investigation and monitoring of these password managers are needed to ensure user data is protected.

7. Capturing and examining the registry.

Brief Summary:

This report talks about a digital forensics investigation where we looked at a memory image using a tool called Autopsy. The goal was to find important information from the memory that could be used in a legal case. The report explains what we found, the tools we used, how we got the memory image, how we looked at it, and the evidence we found.

Tools Used:

- Autopsy
- FTK imager
- Registry editor

Procedure:

- Get the memory image using the right tools, making sure not to change anything.
- Use Autopsy to look at the memory and find useful information.
- Write down everything we found, how we looked at the data, and any evidence we discovered.

Conclusion:

These digital forensics investigation shows how important it is to use special tools and methods to look at memory images. By using Autopsy, we can find important evidence from memory data, which can help in legal cases.

Data Artifacts and Analysis:

Task	Artifact	Tools Used	Description	Flag
Examining the Registry Image with Autopsy	Registry Image	Autopsy, FTK Imager	Imported registry image into Autopsy. Analyzed registry artifacts using Autopsy plugins.	NTUSER.DAT file

Conclusion:

Using Autopsy, we found important information about the illegal rhino images. We looked at different files, including deleted ones, a diary, and emails. These findings will help in further investigation and possible legal actions.

8. Examining a windows disk image.

Brief Summary:

This report explains how we examined a forensic image from the NIST Data Leakage Case. We downloaded evidence files, checked their hash value, analyzed them using Autopsy, and found important details to answer questions about the case.

Case Information:

- Case Name: NIST Data Leakage Case
- Case Number: F221
- Evidence Files: cfreds_2015_data_leakage_pc.7z.001, cfreds_2015_data_leakage_pc.7z.002, cfreds_2015_data_leakage_pc.7z.003

Tools Used:

- 7-Zip
- Autopsy

Procedure:

1. Downloading the Evidence Files:

Downloaded the files ending in ".001", ".002", and ".003" from the given URL.

2. Unzipping the Evidence File:

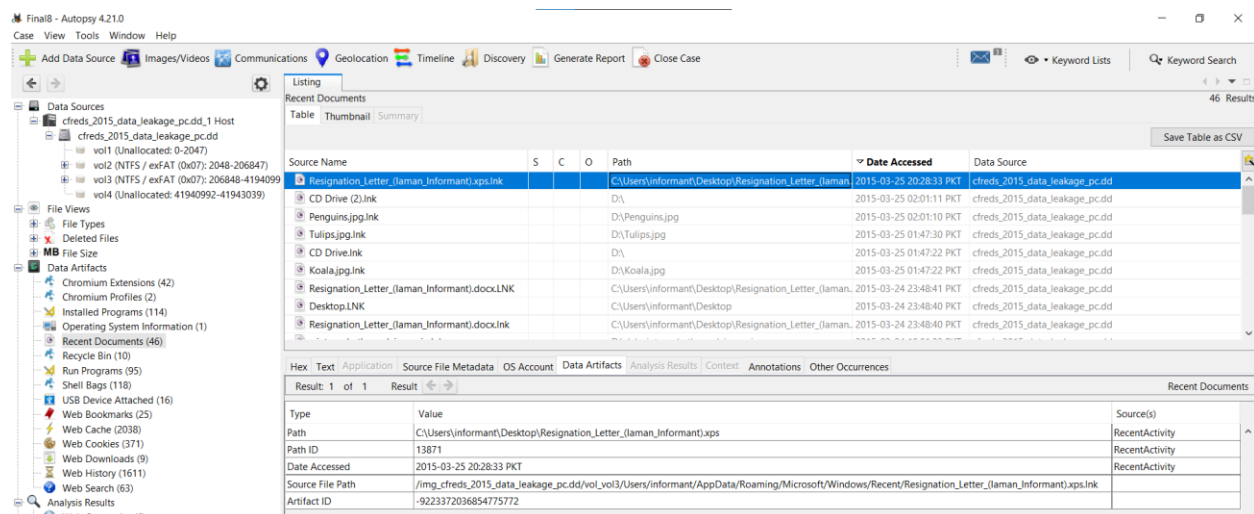
- Used 7-Zip to extract "**cfreds_2015_data_leakage_pc.dd**".
- Checked the file size to make sure it was extracted correctly.

3. Verifying the Hash (MD5):

- Opened PowerShell and went to the Downloads folder.
- Ran the command **Get-FileHash -Algorithm MD5 cfreds_2015_data_leakage_pc.dd** to get the MD5 hash value.
- Compared this hash value with the expected one to make sure the file was not changed.

4. Analyzing the Evidence with Autopsy:

- Opened Autopsy and started a new case called "**Final8**".
- Added the evidence image "**cfreds_2015_data_leakage_pc.dd**" to the case.
- Set up options to process the data.



Data artifacts and analysis:

Task	Artifact	Tools Used	Description	Flag
Verifying the Hash (MD5)	cfreds_2015_data_leakage_pc.dd	PowerShell	Calculated MD5 hash value to ensure file integrity	Hash values were same
Most Recently Installed Program	Installed Programs	Autopsy	Examined installed programs to identify the most recent installation	Eraser 6.2.0.2962 v.6.2.2962
Most Recent Document	Recent Documents	Autopsy	Reviewed recent document access to determine the most recent document accessed	Resignation letter (laman_informant).xps.lnk
59 Times	Prefetch folder	Autopsy	Examined Prefetch folder to identify programs run 59 times	DLLHOST.EXE
Search	Web Search History	Autopsy	Investigated web search history to identify suspicious searches	Anti-forensic tools

Conclusion:

We successfully investigated the NIST Data Leakage Case using Autopsy and 7-Zip. We made sure the evidence files were not changed and used Autopsy to find important details about the case. Autopsy was very helpful in examining digital evidence and finding signs of data leakage.

9. Email Forensics.

Brief Summary:

This report explains how we investigated harassing emails sent to Lily Tuckrige, a teacher at Nitroba. We suspected a student from her Chemistry 109 class. The investigation involved checking email details, looking at network traffic, and examining web requests to find out who sent the emails.

Introduction:

Lily Tuckrige complained to the Nitroba IT department about harassing emails sent to her personal email account. We aimed to find out where the emails came from, who sent them, and if the sender was in her Chemistry 109 class.

Tools Used:

- Network Sniffer
- Wireshark

Procedure:

1. Checking Email Headers:

- Got the full email headers from Lily Tuckrige.
- Found the **IP address 140.247.62.34**, which was from a student dorm room at Nitroba.

2. Capturing Network Traffic:

- Set up a network sniffer on the dorm room's ethernet port.
- Logged all network packets to analyze the traffic.

3. Checking HTTP Requests:

- Found that "**willselfdestruct.com**" was used to send the harassing emails.
- Identified **IP address 69.25.94.22** accessing "**willselfdestruct.com**".
- Traced the suspect's **IP address as 192.168.15.4**.

4. Examining HTTP POST Requests:

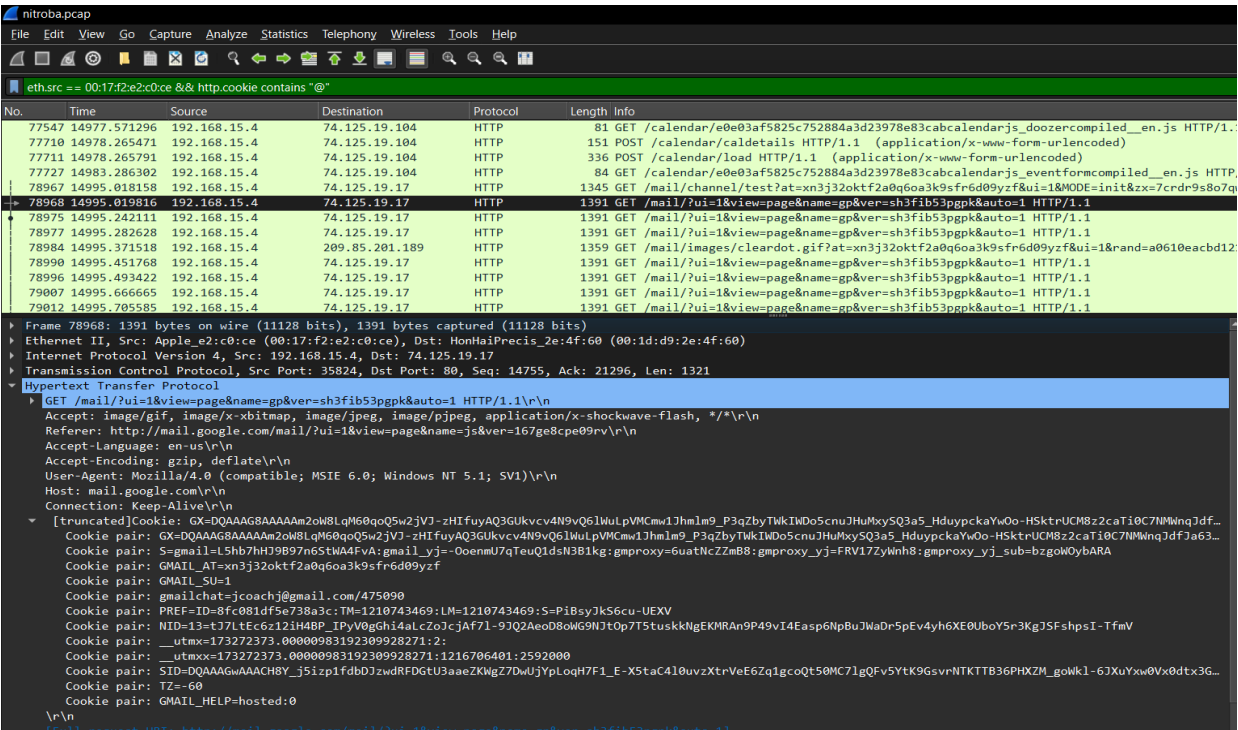
- Looked for **HTTP POST requests** between **69.25.94.22** and **192.168.15.4**.
- Found harassing messages in these HTTP POST requests.

5. Linking IP to Suspects:

- Identified **MAC address 00:17:f2:e2:c0:ce** linked to the suspect's Apple device.

6. Identifying Suspects:

- Used cookies to narrow down possible suspects.
- Found profiles linked to these cookies.
- Checked if the suspect was in Lily Tuckrige's Chemistry 109 class.



Teacher: Lily Tuckrige

Students:

- Amy Smith
- Burt Greedom
- Tuck Gorge
- Ava Book
- Johnny Coach
- Jeremy Ledvkin
- Nancy Colburne
- Tamara Perkins
- Esther Pringle
- Asar Misrad
- Jenny Kant

Data Artifacts and Analysis:

Task	Artifact	Tools Used	Description	Flag
Consoles	Consoles section	Autopsy	Located Waldo's password in console logs	Waldo's password is "Apple123"

Hashdump	Hashdump section	Autopsy	Located Waldo's password hash in hashdump logs	Waldo's password hash is "cfeac129dc5e61b2eb9b27131fc7e2b"
LSAdump	LSAdump section	Autopsy	Identified default password in LSAdump logs	Default password is "P@ssw0rd"
Netscan	Netscan section	Autopsy	Identified executable listening on port 8080	Executable name is "ftpbasicsvr.exe"
Pslist	Pslist section	Autopsy	Identified running executable from pslist logs	Executable name is "FTK Imager.exe"
Shellbags	Shellbags section	Autopsy	Identified connected shared folder name	Shared folder name is "sambowne"
Userassist	Userassist section	Autopsy	Identified dangerous executable from userassist logs	Executable name is "Poison Ivy 2.3.2.exe"
Probe Password	Hashdump section	Autopsy	Located probe account password hash	Probe account password is "P@ssw0rd".

Conclusion:

The investigation traced the harassing emails back to a Nitroba student dorm room. By analyzing email headers, network traffic, and HTTP requests, we identified the web service **"willselfdestruct.com"** was used. We found the suspect's IP and MAC addresses and narrowed down possible suspects. More investigation is needed to confirm the perpetrator's identity and their connection to Lily Tuckrige's Chemistry 109 class.

10. Android Studio Emulator.

Brief Summary:

This report explains how to set up an Android emulator using Android Studio and install the Qute terminal emulator app for security auditing. The steps include downloading Android Studio, creating a virtual Android device, fixing common problems, and installing Qute.

Tools Used:

- Android Studio
- Qute: Terminal Emulator

Procedure:

1. Installing Android Studio:

- Downloaded Android Studio from its official website.
- Installed Android Studio with the default settings.
- Opened Android Studio and started a new project with a **"Basic Views"** activity.

2. Creating an Emulated Android Device:

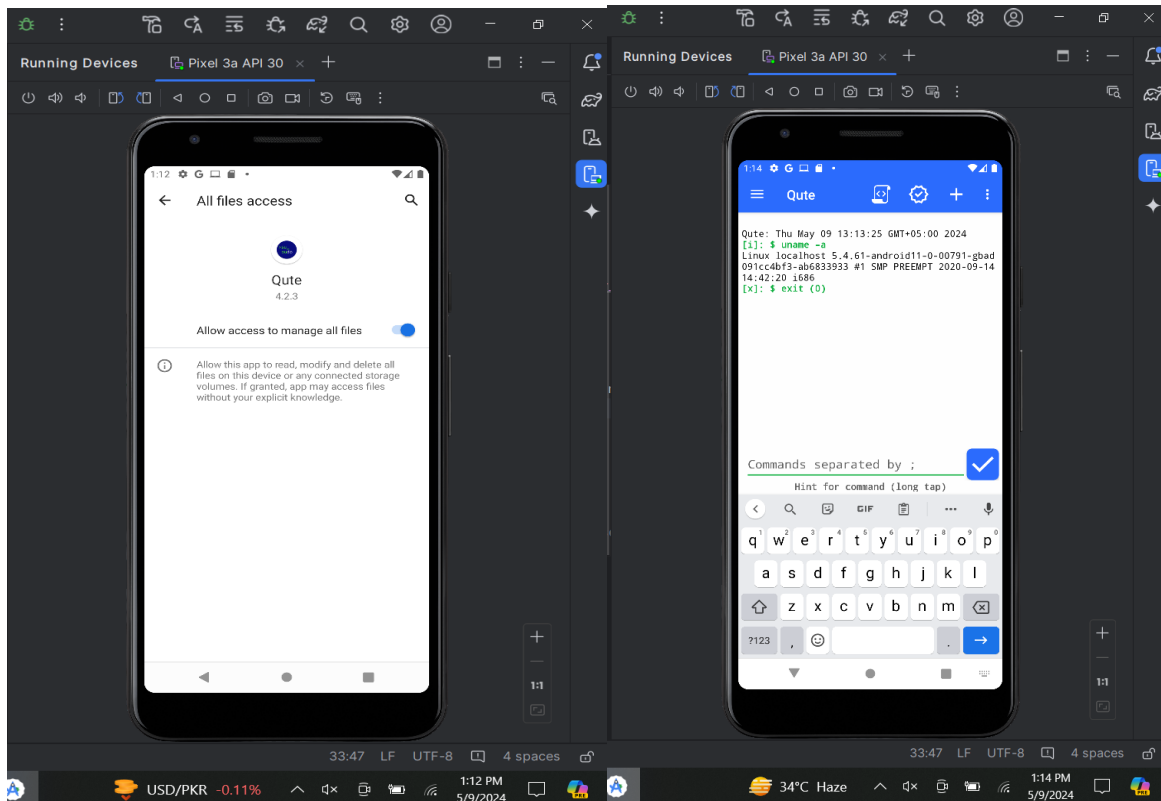
- Went to the Device Manager tab in Android Studio.
- Created a new virtual device using **"Pixel 3a"** hardware with Google Play support.
- Downloaded and installed the recommended system image.
- Changed advanced settings, like increasing internal storage.
- Started the virtual device using the Run button.

3. Installing Qute: Terminal Emulator:

- Opened Google Play on the virtual Android device.
- Logged in with a Google account.
- Searched for and installed **"Qute: Terminal Emulator"** from Google Play.

4. Executing Commands in Qute:

- Opened Qute: Terminal Emulator from the Android home screen.
- Agreed to the End User License Agreement (EULA) and Privacy Policy.
- Gave the app the necessary permissions.
- Ran the command **uname -a** to get system information.



Data Artifacts and Analysis:

Task	Artifact	Tools Used	Description	Flag
Installing Android Studio	Android Studio Installation	Manual Installation	Installed Android Studio with default options and created a new project	N/A
Creating an Emulated Android Device	Virtual Device Configuration	Android Studio	Created a virtual Android device with Google Play support	N/A
Installing Qute: Terminal Emulator	Qute Installation	Google Play	Installed "Qute: Terminal Emulator" app from Google Play	N/A
Executing Commands in Qute	Terminal Commands Execution	Qute: Terminal Emulator	Executed the command uname -a in Qute terminal to gather system information	Flag is "Linux"

Conclusion:

We successfully set up the Android emulator for security auditing. Android Studio was installed, a virtual Android device with Google Play support was created, and the Qute terminal emulator app was installed to run commands. Using Qute, we gathered system information, which is useful for security auditing.

11&14. Rooting Android Studio's Emulator.

Brief Summary:

This report explains how to create an Android emulator with root access using Android Studio. The goal is to set up a 32-bit x86 device with ARM translation libraries to ensure it works with older apps. We cover the steps to create the emulator, adjust its RAM and storage, start it, connect with ADB, and open a root shell to run commands.

Tools Used:

- Android Studio
- ADB (Android Debug Bridge)

Procedure:

1. Creating a Device:

- Opened Android Studio and went to the Device Manager.
- Chose "**Pixel 3a XL**" hardware from the Select Hardware box.
- Selected "**Android 11.0**" without Google API from the System Image box and downloaded it.
- Set **RAM to 4096 MB** and **internal storage to 2048 MB** in the Advanced Settings.
- Created the emulator device.

2. Starting the Emulator:

- Started the emulator from Android Studio.

3. Connecting with ADB:

- Ran the command `adb devices` in a Command Prompt.
- Checked that the emulator device was listed in the output.

4. Opening a Root Shell:

- Ran the commands `adb shell` and `su` in the Command Prompt to open a root shell.

5. Executing Commands for Flags:

- In the root shell, ran the command `ps` to find a specific line ending with "**ps**" and got the flag.
- Ran the command `id` to get another flag.

Data Artifacts and Analysis:

Task	Artifact	Tools Used	Flag
Creating a Device	Emulator Configuration	Android Studio	N/A

Start Your Emulator	Emulator Start	Android Studio	N/A
Connecting with ADB	ADB Connection	ADB	5554 Emulator Devices Listed
Opening a Root Shell	Root Shell	ADB	Root Shell
Executing Commands for Flags	Command Output	ADB	Ps = "root" Id = "su"

```

C:\Users\Lenovo\Downloads\platform-tools-latest-windows\platform-tools>.\adb -s emulator-5556 shell
generic_x86_64:/ $ su
generic_x86_64:/ # ps
USER      PID   PPID   VSZ   RSS  WCHAN          ADDR S  NAME
shell     2684   397 10755080 2672 __x64_sys+      0 S  sh
root      2686  2684 10755080 2656 __x64_sys+      0 S  sh
root      2688  2686 10758192 3384 0               0 R  ps
generic_x86_64:/ #

generic_x86_64:/ # id
uid=0(root) gid=0(root) groups=0(root),1004(input),1007(log),1011(adb),1015(sdcard_rw),1028(sdcard_r),3001(net_bt_admin),3002(net_bt),3003(inet),3006(net_bw_stats),3009(readproc),3011(uhid) context=u:r:su:s0
generic_x86_64:/ #

```

Conclusion:

We successfully set up an Android emulator with root access using Android Studio and ADB. The emulator was configured, started, and connected using ADB. We opened a root shell to run commands and retrieved the necessary flags for the project tasks.

12. Forensics Acquisition from Android.

Brief Summary:

This report explains how to collect and analyze data from an Android emulator with root access using Autopsy. We created messages and calls on the emulator, collected the data, transferred it to a Windows computer, and analyzed it with Autopsy.

Tools Used:

- a. Android Studio (for setting up the emulator)
- b. ADB (Android Debug Bridge)
- c. Autopsy
- d. 7-Zip

Procedure:

1. Starting the Emulator:

- e. Started the Android emulator with root access set up earlier.

2. Generating User Data:

- f. Sent messages and made phone calls on the emulator.
- g. Opened Messages on the emulator to receive and reply to SMS.
- h. Answered an incoming call on the emulator.

3. Opening a Root Shell:



- i. Used ADB to access a root shell on the emulator.
- j. Ran commands to collect user data from the /data directory.

4. Collecting User Data:

- k. Created a tar archive of user data and saved it on the emulator's storage.
- l. Transferred the tar archive to the Windows computer using ADB.

5. Analyzing Data with Autopsy:

- m. Opened Autopsy on the Windows computer.
- n. Created a new case named "Android" in Autopsy.
- o. Imported the Android data as a logical file source.
- p. Set up the options to use the Android Analyzer (aLEAPP).
- q. Checked the evidence in the Data Artifacts section of Autopsy.

Source Name	S	C	O	Message Type	Date/Time	Read	Phone Number	Text	Thread ID	Data Source
 mmssms.db			0	SMS messages	2024-05-10 13:41:42 PKT	1	707e6848-74a0-405f-813b-81f8175887d8	"HELLO PHONE"	1	LogicalFileSet1
 mmssms.db			0	SMS messages	2024-05-10 13:43:48 PKT	1	707e6848-74a0-405f-813b-81f8175887d8	"YOU LITERALLY TESTED MY PATIENCE!!!"	1	LogicalFileSet1

HexTextApplicationSource File MetadataOS AccountData ArtifactsAnalysis ResultsContextAnnotationsOther Occurrences

Result: 3 of 4Result↩↪Messages

From:2024-05-10 13:41:42 PKT
To:
CC:
Subject:

HeadersTextHTMLRTFAttachments (0)Accounts

Original Text

"HELLO PHONE"

HexTextApplicationSource File MetadataOS AccountData ArtifactsAnalysis ResultsContextAnnotationsOther Occurrences

Result: 4 of 4Displays file contentsMessages

From:2024-05-10 13:43:48 PKT
To:
CC:
Subject:

HeadersTextHTMLRTFAttachments (0)Accounts

Original Text

"YOU LITERALLY TESTED MY PATIENCE!!!"

Data Artifacts and Analysis:

Task	Artifact	Tools Used	Flag
Starting the Emulator	Emulator Startup	Android Studio	N/A
Putting Evidence on the Phone	SMS, Phone Call	ADB	N/A
Opening a Root Shell	Root Shell	ADB	N/A
Collecting User Data	Tar Archive	ADB	Tar Archive
Analyzing the Android Data with Autopsy	Autopsy Case, Data Artifacts	Autopsy	Flag is “mmssms.db”

Conclusion:

We successfully collected data from an Android emulator with root access and analyzed it using Autopsy. We simulated SMS exchanges and phone calls to generate user data, collected it from the emulator, and transferred it to a Windows computer. Using Autopsy, we analyzed the data and found useful information for forensic analysis.

13. Android Analysis with Autopsy.

Brief Summary:

This report explains how to analyze data from an Android device using Autopsy. The data was in a file called android_image2.tar.gz. We extracted it, imported it into Autopsy, and looked for details like phone calls, messages, and web searches. We also identified the most recently installed app and the website viewed at a specific time.

Tools Used:

- a. Autopsy
- b. 7-Zip
- c. Online Hash Calculator

Procedure:

1. Downloading the Evidence File:

- d. Downloaded the **android_image2.tar.gz** file from a provided link using a web browser.

2. Verifying the Hash Value:

- e. Calculating hash by using PowerShell command **"Get-FileHash -Algorithm MD5 .\android_image2.tar.gz"**.
- f. Checked that the calculated hash value matches the provided hash value to ensure file integrity.

3. Unzipping the Data:

- g. Deleted any existing **"data"** folder on the Windows desktop.
- h. Placed the **android_image2.tar.gz** file on the Windows desktop.
- i. Used 7-Zip to extract the contents, creating a **"data"** folder.

4. Analyzing the Android Data with Autopsy:

- j. Opened Autopsy on the Windows computer.
- k. Created a new case named **"Final13"** in Autopsy.
- l. Imported the data from the **"data"** folder as a logical file source.
- m. Set the ingest options to use the Android Analyzer (aLEAPP).
- n. Examined the evidence in the Data Artifacts section of Autopsy.

Data Artifacts and Analysis:

Task	Artifact	Tools Used	Description	Flag
Most Recently Installed App	Installed Apps	Autopsy	Identified the most recently installed app in the Android data	Com.u360mobile.usna

Website Viewed at Specific Time	Web Browsing History	Autopsy	Determined the website viewed at 14:52:39 PDT on Oct 8, 2022	https://yahoo.com/
---------------------------------	----------------------	---------	--	---

Conclusion:

We successfully analyzed data from the Android device using Autopsy. By importing and examining the data, we found important information about installed apps and web browsing activity. We identified the most recently installed app and the website viewed at a specific time, which is useful for forensic investigations.

15. iPhone Analysis with Autopsy.

Brief Summary:

This report explains how to analyze data from an iPhone using Autopsy. The data came from a file called **"2021 CTF - iOS.zip."** We extracted the data and used Autopsy to examine it, looking for things like phone calls, messages, and web searches.

Tools Used:

- a. Autopsy
- b. 7-Zip
- c. PowerShell

Procedure:

1. Downloading the Evidence File:

- Downloaded the "2021 CTF - iOS.zip" file from a webpage using a web browser.

2. Verifying the Hash Value:

- d. Used PowerShell to calculate the SHA-256 hash of the **"2021 CTF - iOS.zip"** file.
- e. Checked that the calculated hash matched the provided hash to ensure the file's integrity.

3. Analyzing the iPhone Data with Autopsy:

- f. Opened Autopsy on a Windows computer.
- g. Created a new case named **"iPhone"** in Autopsy.
- h. Imported the data from the **"2021 CTF - iOS"** folder as a logical file source.
- i. Set the ingest options to use the iOS Analyzer (aLEAPP).

Data Artifacts and Analysis:

Task	Artifact	Description	Flag
Phone Number	Device Paired via Bluetooth / Phone Number in 541 Area Code	Identified the device paired via Bluetooth or phone number in the 541-area code	Eli's Apple Watch / +15854178420
Latitude	GPS Last Known Location / Wireless Networks	Determined the most northern latitude of the phone using GPS data or wireless network information	44.490257170
SMS	Messages	Identified the code sent by SIGNAL to the phone via SMS messages	191-116

Signal Contact	Program Notifications	Identified the person named Johnathan who sent Snapchat (pikaboo) messages to the phone	Jonathan Chipps
----------------	-----------------------	---	-----------------

Conclusion:

We successfully analyzed the iPhone data using Autopsy. By examining the data in Autopsy, we found important information like phone numbers, GPS data, text messages, and app notifications. This information is helpful for forensic investigations.

16.Windows and Linux Machines.

Brief Summary:

This guide helps you set up two virtual computers on Virtual box: one with Windows 10 and the other with Kali Linux. It explains each step clearly.

Procedure:

1. Installing the iso file of windows10:

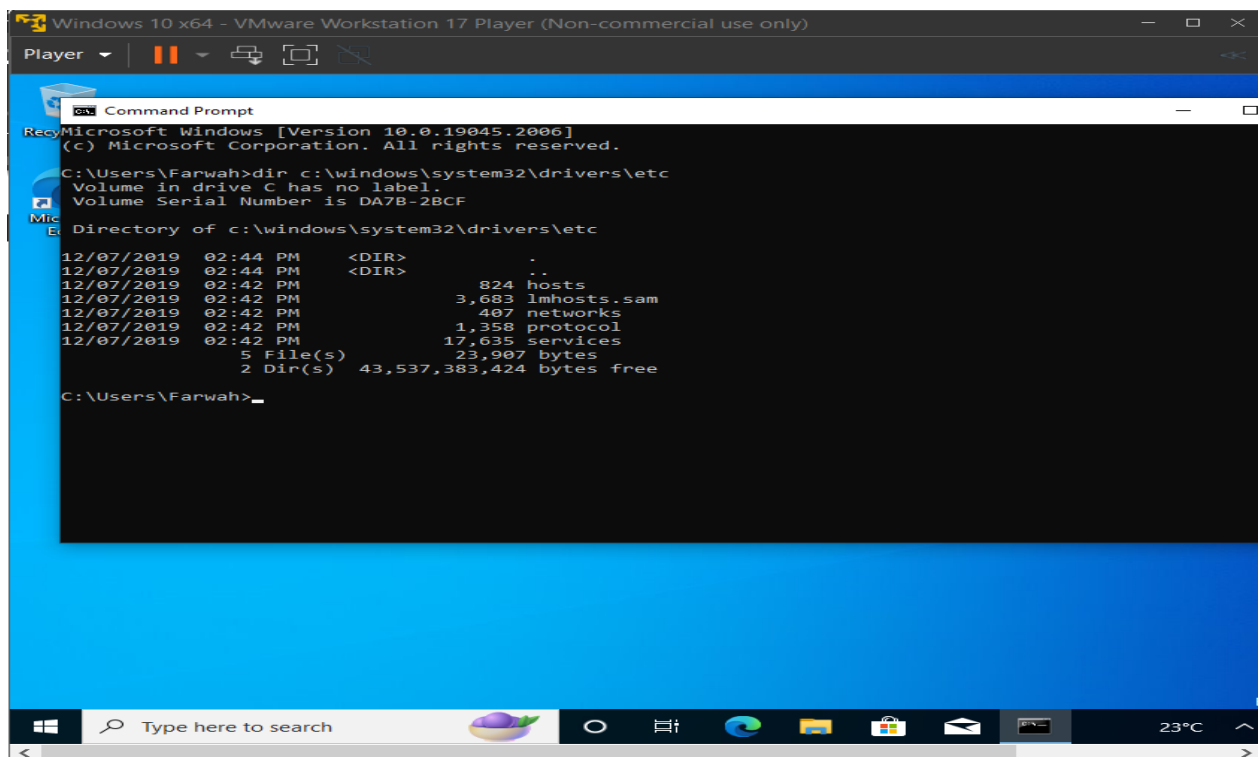
- Importing the disk image in virtual box to add it as a virtual machine.

2. Installing the iso file of Kali Linux:

- Importing the disk image in virtual box to add it as a virtual machine.

3. Recording Success:

- a. On the Windows computer, use Command Prompt to run **dir c:\windows\system32\drivers\etc** to find the Windows flag. **Flag = Hosts.**
- b. On the Linux computer, use SSH to run **lsb_release -a** to find the Linux flag. **Flag = Kali.**



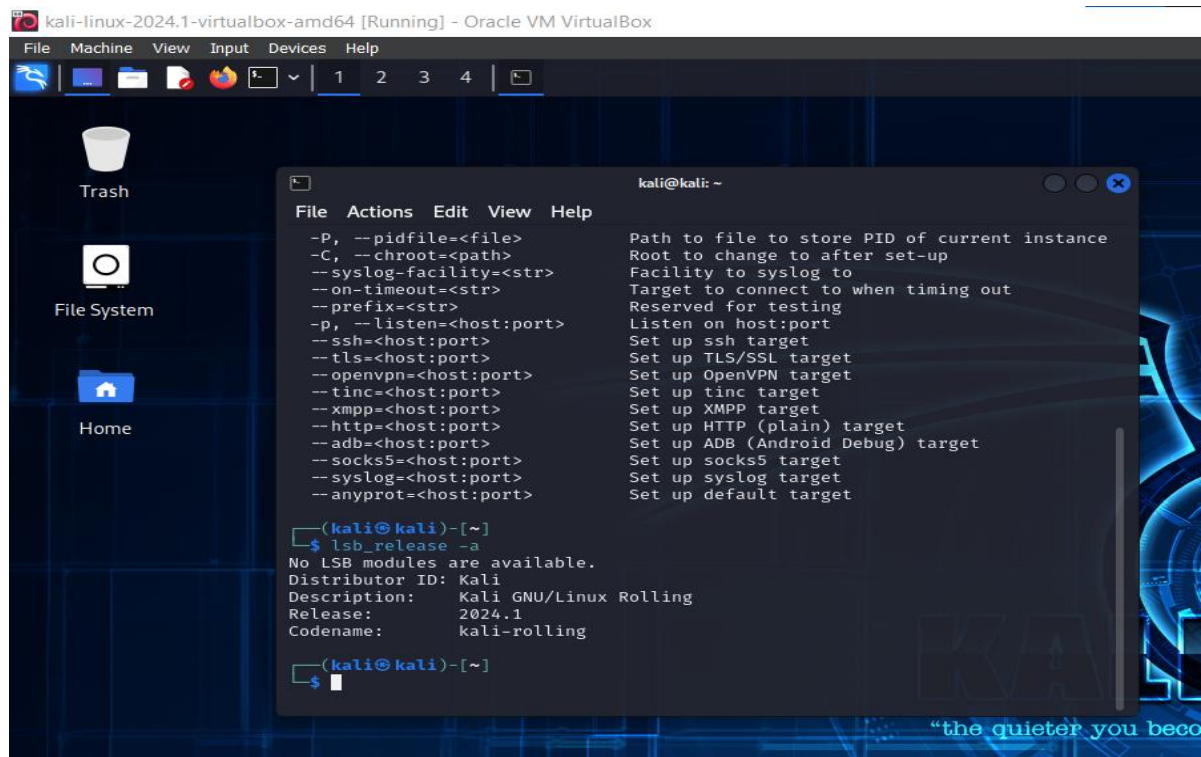
```
Windows 10 x64 - VMware Workstation 17 Player (Non-commercial use only)
Player
Command Prompt
Microsoft Windows [Version 10.0.19045.2006]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Farwah>dir c:\windows\system32\drivers\etc
Volume in drive C has no label.
Volume Serial Number is DA7B-2BCF

Directory of c:\windows\system32\drivers\etc

12/07/2019  02:44 PM    <DIR>          .
12/07/2019  02:44 PM    <DIR>          ..
12/07/2019  02:42 PM                824 hosts
12/07/2019  02:42 PM            3,683 lmhosts.sam
12/07/2019  02:42 PM                497 networks
12/07/2019  02:42 PM            1,358 protocol
12/07/2019  02:42 PM           17,635 services
                5 File(s)        23,907 bytes
                2 Dir(s)  43,537,383,424 bytes free

C:\Users\Farwah>
```

Conclusion:

By following these steps, we set up the windows 10 and kali Linux virtual machines on your virtual box.

17.Velociraptor on Linux.

Brief Summary:

Velociraptor is a tool for checking computers and finding out what happened if there's a problem. This report shows how to set up Velociraptor on a Linux server to watch a Windows computer.

Tools Used:

- Web browser
- Terminal or SSH
- Velociraptor

Procedure:

1. Installing Velociraptor on Linux:

- Finding the Latest Version:
- Found the newest version of Velociraptor on GitHub.
- Got the right installer for Linux and Windows.
- Preparing the Server:
- Made a folder for Velociraptor and got the Linux installer.
- Made the installer ready to use and made a server setup file.
- Editing the Config File:
- Changed the setup file to use the server's IP address.
- Creating the Administrator User:
- Made an admin user to control Velociraptor.
- Starting the Velociraptor Server:
- Started Velociraptor with the setup we made.

2. Adding a Windows Client:

- Preparing a Client Config File:
- Changed the server setup to use self-signed SSL.
- Preparing a Windows Client Installer:
- Made a setup file for the client and got the Windows installer.
- Put the setup file in the installer.
- Moving Client to the Windows Machine:
- Sent the installer to the Windows computer using WinSCP.
- Installing the Windows Client:
- Ran the installer on the Windows computer to put Velociraptor on it.

Data Artifacts and Analysis:

Task	Artifact	Tools Used	Description	Flag
Velociraptor Installation	Linux server configuration	Web browser, Terminal	Installed and configured Velociraptor on Debian Linux server	Successful setup
	Windows client installer	Web browser, Terminal	Repackaged Windows client installer with necessary configurations for monitoring Windows endpoint	Installer created
Windows Client Addition	Client configuration file	Terminal	Edited client configuration file to specify server IP address	Configuration done
	Windows client installer	Terminal, WinSCP	Installed Velociraptor client on Windows machine for endpoint monitoring	Client installed
Server Name	GUI Home Page	velociraptor	Located and identified flag on the home page of the Velociraptor GUI	"velociraptor"
Agent Name	Velociraptor Client	velociraptor	Located and identified flag on the Velociraptor client page in the GUI	"Labels"
Registry Information	Velociraptor GUI	velociraptor	Retrieved registry information using Velociraptor GUI	"Keyboard"
Exploring the File System	Velociraptor GUI	velociraptor	Explored file system using Velociraptor GUI	"NTUSER.DAT"
DestPort	Velociraptor GUI	velociraptor	Retrieved destination port information using Velociraptor GUI	"8000"

Conclusion:

We set up Velociraptor on a Debian Linux server and a Windows client. The server is ready to use, and the client is watching the Windows computer. Setting up Velociraptor needs a few steps, like making files and installing programs, but it's important for checking computers and keeping them safe.

18. Investigating a PUP with Velociraptor.

Brief Summary:

The project is about pretending a virus is on a Windows computer and seeing what happens using Velociraptor on a Linux server.

Tools Used:

- Web browser
- Velociraptor
- Notepad
- Command Prompt

Procedure:

1. Infecting the Windows Machine:

- Disabling Windows Defender:
- Stopped Windows Defender to let the virus do its thing.
- Installing the Malware:
- Got the virus and put it on the Windows computer.
- Ran the virus and put it where it can do damage.
- Simulating Virus Running:
- Restarted the Windows computer to make the virus start.

2. Investigating with Velociraptor:

- Connecting to the Windows Computer:
- Used Velociraptor to look at what's happening on the infected computer.
- Checking Autoruns:
- Looked at what starts when the computer turns on to find where the virus is.
- Found the virus's info in a report.
- Checking the Virus File:
- Looked at the virus file to find out what it is.
- Scanned for similar viruses.

3. Fixing the Problem:

- Using Velociraptor to Fix:
- Went into the infected computer using Velociraptor.
- Used commands to stop the virus, delete bad stuff, and remove the virus.

Data Artifacts and Analysis:

Task	Artifact	Tools Used	Description	Flag
Windows Malware Infection	PUP executable	Web browser, Windows	Installed and executed PUP on Windows machine	"YOUR MACHINE IS PWNERD"
	Autoruns CSV report	Velociraptor, Notepad	Analyzed Autoruns report to locate Run key used for PUP	"PUP4"
	MD5 hash of PUP EXE	Velociraptor	Retrieved MD5 hash of PUP executable used for malware execution	MD5 hash found Flag = "a53"
	Yara Scan Results	Velociraptor	Conducted Yara scan to identify additional malware files on Windows machine	Malware files found Flag = "PWNERD"
Remediation	Command Execution Results	Velociraptor, Command Prompt	Executed remediation commands and verified their effects on the Windows machine	Remediation verified Flag = "completed successfully"

Conclusion:

We infected a Windows computer with a fake virus and used Velociraptor on a Debian Linux server to fix it. Velociraptor helped us find the virus and remove it. Velociraptor is a good tool for finding viruses and stopping them from causing more problems.

19. Investigating a BOT with Velociraptor.

Brief Summary:

The project is about pretending a virus is on a Windows computer and seeing what happens using Velociraptor on a Linux server.

Tools Used:

- Web browser
- Velociraptor
- Wireshark
- Command Prompt

Procedure:

1. Infecting the Windows Machine:

- Disabling Windows Defender:
- Stopped Windows Defender to let the virus do its thing.
- Installing the Malware:
- Got the virus and put it on the Windows computer.
- Ran the virus and put it where it can do damage.

2. Investigating with Velociraptor:

- Connecting to the Windows Computer:
- Used Velociraptor to look at what's happening on the infected computer.
- Checking Network Traffic:
- Looked at what the computer is sending and receiving.
- Identified some weird stuff going on.

3. More Investigation and Fixing:

- Checking DNS:
- Looked at what websites the computer is talking to.
- Found some suspicious website names.
- Finding the Virus:
- Looked for the virus on the computer and found where it is hiding.
- Finding the Bad Stuff:
- Looked for bad things the virus did on the computer.
- Found some bad tasks the virus created.
- Fixing the Problem:
- Used Velociraptor to stop the bad tasks and get rid of the virus.

Data Artifacts and Analysis:

Task	Artifact	Tools Used	Description	Flag
Windows Malware Infection	Malware executable	Web browser, Windows	Installed and executed PUP on Windows machine	Malware executed
	Network traffic capture	Velociraptor, Wireshark	Captured and analyzed network traffic to identify malware communication patterns	Beaconing activity Flag = "Mosilla"
	DNS cache analysis	Velociraptor	Analyzed DNS cache to identify C & C domain used by malware	C & C domain found Flag = "TTL"
	Beaconing EXE	Velociraptor	Identified malware file responsible for beaconing activities	Securitytest.exe Flag = "Upload"
	Sysmon event logs	Velociraptor, Command Prompt	Analyzed Sysmon event logs to identify scheduled task persistence mechanism	Flag = "DcamLaunch"
Hunts and Analysis	Hunt results	Velociraptor	Created and executed hunts to detect and analyze suspicious activities on the Windows client	Suspicious activity Flag = "OSPath"
Remediation	Command Execution Results	Command Prompt	Executed remediation commands to delete scheduled tasks and remove malware-related directories	Remediation verified

Conclusion:

We infected a Windows computer with a fake virus and used Velociraptor on a Debian Linux server to fix it. Velociraptor helped us find the virus and remove it. Velociraptor is a good tool for finding viruses and stopping them from causing more problems.

20. Investigating a Two-Stage RAT with Velociraptor.

Brief Summary:

We used Velociraptor to check for a virus on the Windows computer from a Linux Velociraptor server.

Tools Used:

- Velociraptor
- Sysmon
- 7-Zip

Procedure:

1. Installing Sysmon on the Windows Machine:

- Used Velociraptor to put Sysmon on the Windows computer.

2. Infecting the Windows Machine:

- Got the virus file "pup5.zip" from the internet.
- Opened the file and ran it on the Windows computer.

3. Investigating with Velociraptor:

i. Checking Network Connections:

- Looked at what the computer was connecting to online.
- Found a port the virus was using.

ii. Checking Autoruns:

- Looked at what the computer was automatically running.
- Found the virus program in the list.

iii. Checking Creation Time:

- Looked at when the virus program was made.
- Found out when the virus program was created.

iv. Checking Prefetch:

- Looked at data about how often programs were used.
- Found data about the virus program being used.

v. Checking Sysmon Logs:

- Looked at logs of what happened on the computer.
- Found the event when the virus program was run.

Data Artifacts and Analysis:

Task	Artifact	Tools Used	Description	Evidence Found
Network Connections	Network connections	Velociraptor	Identified port of "shellbind" process	Port number = 4444
Autoruns	Autoruns	Velociraptor	Noted path of "shellbind.exe"	File path = C:\\shellbind.exe
Creation Time	Creation time	Velociraptor	Noted creation timestamp	5/11/24 6:13 PM
Prefetch	Prefetch data	Velociraptor	Noted LastRunTimes	5/11/24 6:13 PM Flag = "Hash"
Sysmon logs	Sysmon event logs	Velociraptor	Identified relevant events	Company = "Igor Pavlov"

Conclusion:

We investigated the virus on the Windows computer using Velociraptor on the Linux server. We found out a lot about the virus and what it did on the computer.
