1. **Perform an experiment for port scanning with Nmap.**

Nmap (Network Mapper) is a versatile and powerful tool designed for network scanning. It identifies hosts, services, and open ports on a network, revealing potential vulnerabilities that attackers might exploit. Port scanning is a critical part of assessing network security, providing insights into exposed services and their associated risks.

How to Download Nmap

1. Visit the official Nmap website.
2. Select the appropriate installer for your operating system (Windows, macOS, or Linux).
3. Download the installer file.

Steps to Install Nmap

1. Run the downloaded installer file.
2. Follow the on-screen instructions to complete the setup.
3. Launch Nmap to verify successful installation.

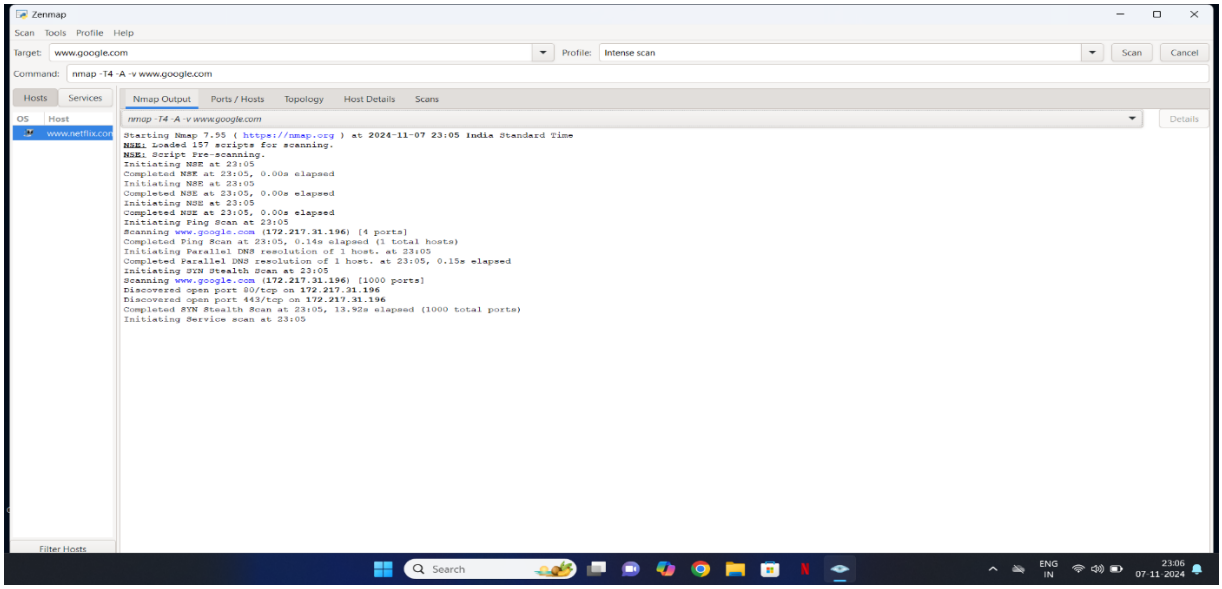
How to Use Nmap

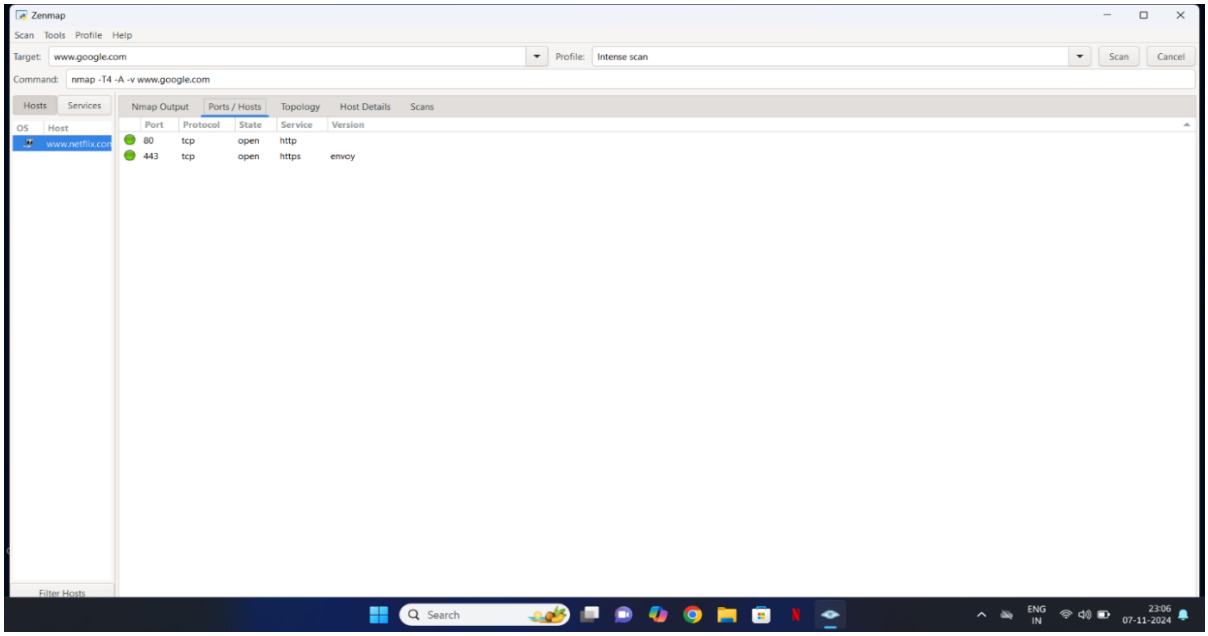
1. **Launch the Application**: Open Nmap on your system to get started.
2. **Enter the Target**: Input the target URL or IP address in the designated field.
3. **Choose a Scan Type**: Select **"Intense Scan"** from the list of available scan options. This mode provides detailed information on open ports, services, and operating systems.
4. **Run the Scan**: Click **"Scan"** to initiate the process.
5. **Review the Results**: Analyze the output, which includes:
    - **Ports/Hosts**: A list of open ports, associated services, and potential vulnerabilities.
    - **Topology**: A visual map of host connections and the network structure.
    - **Host Details**: Information on operating systems, uptime, and MAC addresses.
    - **Scan Details**: Details on protocols used and the scan completion time.

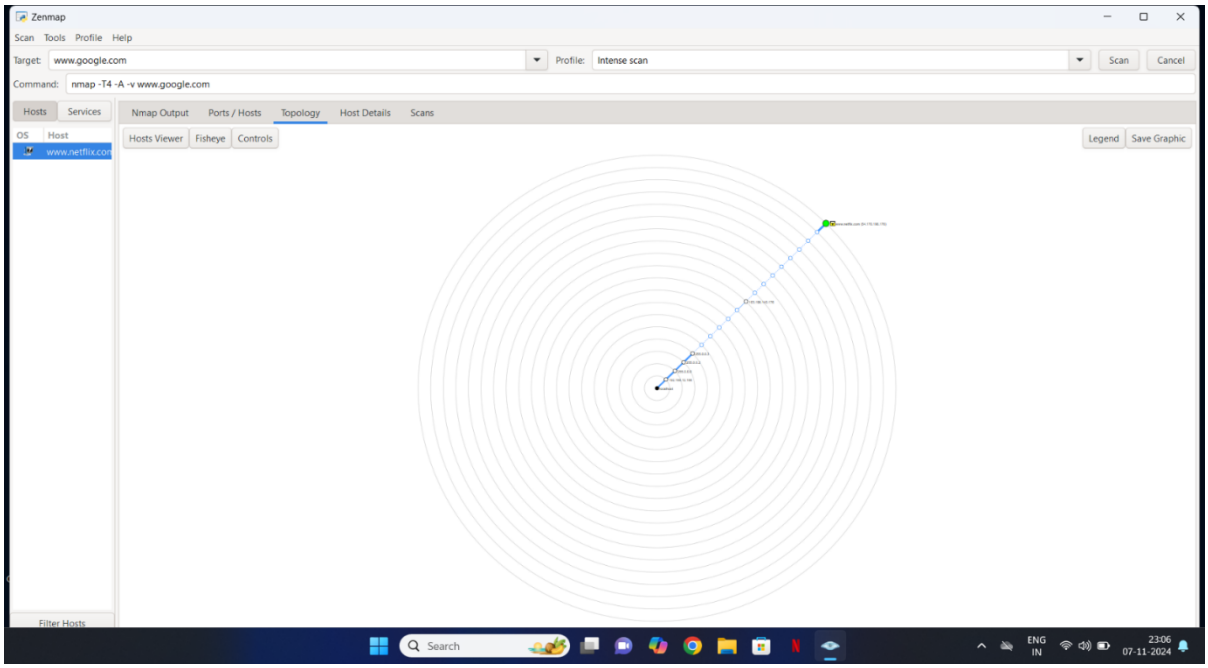
Important Observations

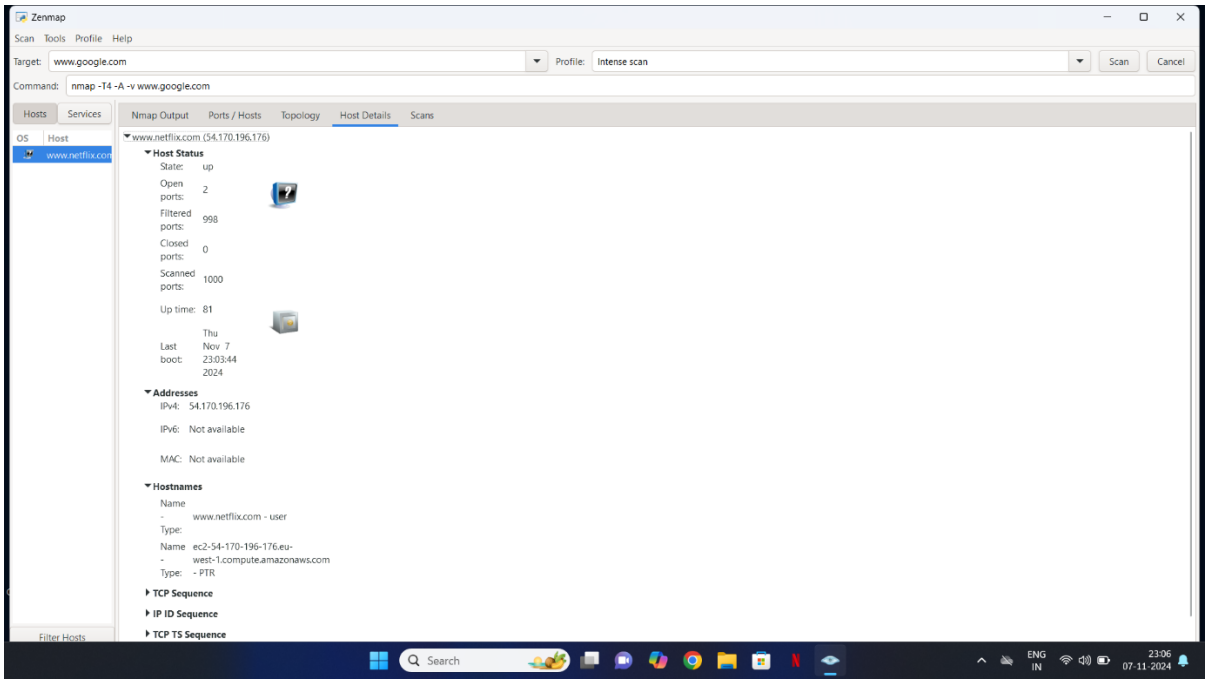
- **Common Ports**: Take note of commonly vulnerable ports, such as TCP ports **80**, **443**, and **22**.
- **Anomalies**: Look for any unusual or unexpected open ports or services that might indicate security risks.

Nmap's **Intense Scan mode** delivers an in-depth view of network structure and security vulnerabilities. By analyzing exposed ports and active services, you can identify potential weaknesses and take proactive measures to secure your network assets effectively.


Zenmap
Scan Tools Profile Help
Target: www.google.com
Profile: Intense scan
Scan Cancel
Command: nmap -T4 -A -v www.google.com
Hosts Services
OS Host
www.netflix.com
Nmap Output Ports / Hosts Topology Host Details Scans
nmap -T4 -A -v www.google.com
Details
Starting Nmap 7.95 (https://nmap.org) at 2024-11-07 23:05 India Standard Time
NSE: Loaded 157 scripts for scanning.
NSE: Script Pre-scanning.
Initiating NSE at 23:05
Completed NSE at 23:05, 0.00s elapsed
Initiating NSE at 23:05
Completed NSE at 23:05, 0.00s elapsed
Initiating NSE at 23:05
Completed NSE at 23:05, 0.00s elapsed
Initiating Ping Scan at 23:05
Scanning www.google.com (172.217.31.196) [4 ports]
Completed Ping Scan at 23:05, 0.14s elapsed (1 total hosts)
Initiating Parallel DNS resolution of 1 host. at 23:05
Completed Parallel DNS resolution of 1 host. at 23:05, 0.15s elapsed
Initiating SYN Stealth Scan at 23:05
Scanning www.google.com (172.217.31.196) [1000 ports]
Discovered open port 80/tcp on 172.217.31.196
Discovered open port 443/tcp on 172.217.31.196
Completed SYN Stealth Scan at 23:05, 13.92s elapsed (1000 total ports)
Initiating Service scan at 23:05
Filter Hosts
Search
ENG IN
23:06 07-11-2024


Zenmap
Scan Tools Profile Help
Target: www.google.com
Profile: Intense scan
Scan Cancel
Command: nmap -T4 -A -v www.google.com
Hosts Services
OS Host
www.netflix.com
Nmap Output Ports / Hosts Topology Host Details Scans
Port Protocol State Service Version
80 tcp open http
443 tcp open https envoy
Filter Hosts
Search
ENG IN
23:06 07-11-2024


Zenmap
Scan Tools Profile Help
Target: www.google.com
Profile: Intense scan
Scan
Cancel
Command: nmap -T4 -A -v www.google.com
Hosts
Services
OS
Host
www.netflix.com
Nmap Output
Ports / Hosts
Topology
Host Details
Scans
Hosts Viewer
Fisheye
Controls
Legend
Save Graphic
Filter Hosts
Search
ENG
IN
23:06
07-11-2024


Zenmap
Scan Tools Profile Help
Target: www.google.com
Profile: Intense scan
Scan
Cancel
Command: nmap -T4 -A -v www.google.com
Hosts
Services
OS
Host
www.netflix.com
Nmap Output
Ports / Hosts
Topology
Host Details
Scans
www.netflix.com (54.170.196.176)
Host Status
State: up
Open ports: 2
Filtered ports: 998
Closed ports: 0
Scanned ports: 1000
Up time: 81
Last boot: Thu Nov 7 23:03:44 2024
Addresses
IPv4: 54.170.196.176
IPv6: Not available
MAC: Not available
Hostnames
Name - Type: www.netflix.com - user
Name - Type: ec2-54-170-196-176.eu-west-1.compute.amazonaws.com - PTR
TCP Sequence
IP ID Sequence
TCP TS Sequence
Filter Hosts
Search
ENG
IN
23:06
07-11-2024

2. **Setup a honeypot and monitor the honeypot on the system.**

KFSensor is a specialized honeypot software designed to simulate vulnerable network services. By deploying KFSensor, organizations can detect and log unauthorized access attempts, gaining valuable insights into potential attacker behavior. This proactive approach helps identify weaknesses and fortify network security.

Downloading KFSensor

1. Navigate to the official KFSensor website.
2. Select the appropriate software version for your operating system.
3. Download the installer file and save it to your device.

Installing KFSensor

1. Open the downloaded file to begin the installation process.
2. Follow the prompts provided by the installer.
3. Launch KFSensor once the installation is complete to ensure it is set up correctly.

Getting Started with KFSensor

1. **Open the Program**: Launch KFSensor on your system.
2. **Set Up Monitoring**: Configure the network settings, specifying the IP addresses to be monitored.
3. **Select Services**: Choose the vulnerable services (e.g., FTP, HTTP, or SMTP) that you want to simulate.
4. **Begin Monitoring**: Activate the honeypot to start logging network activity and detecting unauthorized access.

Key Features for Monitoring and Analysis

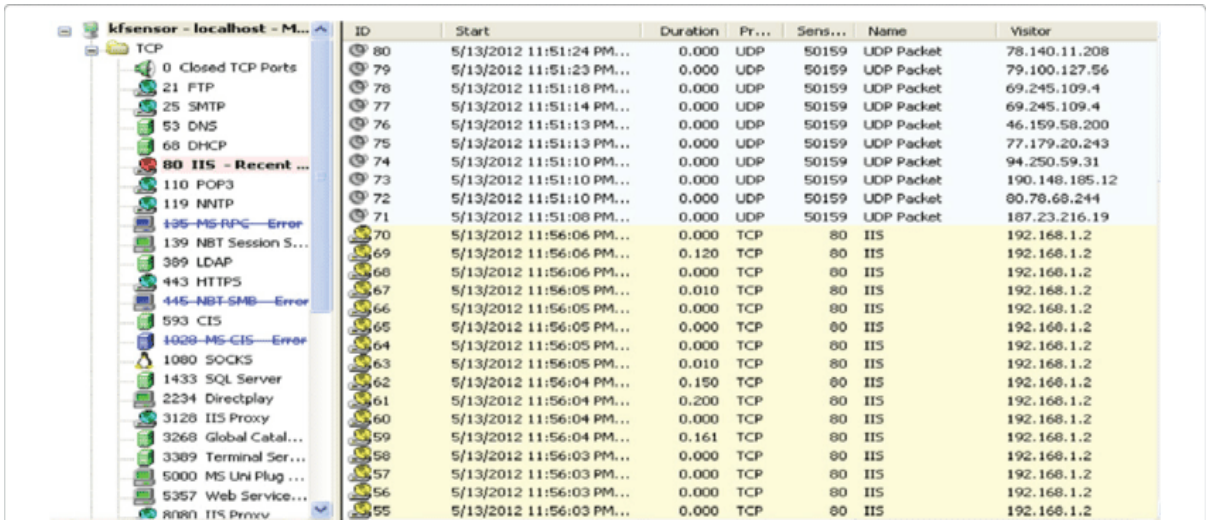
- **Port Simulation**: Create an environment that mimics open ports, such as those used for HTTP or FTP, to attract potential attackers.
- **Logging Access Attempts**: Record connection attempts, capturing details like attacker IPs and the tools used.
- **Real-Time Alerts**: Set up notifications for specific activities, such as brute force login attempts or suspicious IP activity.
- **Data Review**: Examine logs with detailed timestamps, accessed services, and attacker behaviors to identify intrusion methods.
- **Multi-Service Emulation**: Simulate several services to observe attacker interactions and study their techniques at different layers of the network.

Insights

- Monitor patterns in intrusion attempts, such as repeated access to specific IP ranges or services.
- Document trends in attack behavior to better prepare for potential threats.

Conclusion

KFSensor provides a safe and controlled platform to analyze unauthorized network activity. By simulating vulnerabilities, it reveals common attack methods, allowing organizations to strengthen their defenses and address potential security risks more effectively.



ID	Start	Duration	Pr...	Sens...	Name	Visitor
80	5/13/2012 11:51:24 PM...	0.000	UDP	50159	UDP Packet	78.140.11.208
79	5/13/2012 11:51:23 PM...	0.000	UDP	50159	UDP Packet	79.100.127.56
78	5/13/2012 11:51:18 PM...	0.000	UDP	50159	UDP Packet	69.245.109.4
77	5/13/2012 11:51:14 PM...	0.000	UDP	50159	UDP Packet	69.245.109.4
76	5/13/2012 11:51:13 PM...	0.000	UDP	50159	UDP Packet	46.159.58.200
75	5/13/2012 11:51:13 PM...	0.000	UDP	50159	UDP Packet	77.179.20.243
74	5/13/2012 11:51:10 PM...	0.000	UDP	50159	UDP Packet	94.250.59.31
73	5/13/2012 11:51:10 PM...	0.000	UDP	50159	UDP Packet	190.148.185.12
72	5/13/2012 11:51:10 PM...	0.000	UDP	50159	UDP Packet	80.78.68.244
71	5/13/2012 11:51:08 PM...	0.000	UDP	50159	UDP Packet	187.23.216.19
70	5/13/2012 11:56:06 PM...	0.000	TCP	80	IIS	192.168.1.2
69	5/13/2012 11:56:06 PM...	0.120	TCP	80	IIS	192.168.1.2
68	5/13/2012 11:56:06 PM...	0.000	TCP	80	IIS	192.168.1.2
67	5/13/2012 11:56:05 PM...	0.010	TCP	80	IIS	192.168.1.2
66	5/13/2012 11:56:05 PM...	0.000	TCP	80	IIS	192.168.1.2
65	5/13/2012 11:56:05 PM...	0.000	TCP	80	IIS	192.168.1.2
64	5/13/2012 11:56:05 PM...	0.000	TCP	80	IIS	192.168.1.2
63	5/13/2012 11:56:05 PM...	0.010	TCP	80	IIS	192.168.1.2
62	5/13/2012 11:56:04 PM...	0.150	TCP	80	IIS	192.168.1.2
61	5/13/2012 11:56:04 PM...	0.200	TCP	80	IIS	192.168.1.2
60	5/13/2012 11:56:04 PM...	0.000	TCP	80	IIS	192.168.1.2
59	5/13/2012 11:56:04 PM...	0.161	TCP	80	IIS	192.168.1.2
58	5/13/2012 11:56:03 PM...	0.000	TCP	80	IIS	192.168.1.2
57	5/13/2012 11:56:03 PM...	0.000	TCP	80	IIS	192.168.1.2
56	5/13/2012 11:56:03 PM...	0.000	TCP	80	IIS	192.168.1.2
55	5/13/2012 11:56:03 PM...	0.000	TCP	80	IIS	192.168.1.2

Start Time	Pro...	Sens...	Name	Visitor	Received
10:59:12.598	TCP	8180	TCP Closed Port	datachem.de	
10:59:12.598	TCP	80	Port Scan War...	datachem.de	Possible Port Scan.[0D 0A 0D 0A]The visitor ha
10:59:12.608	TCP	81	IIS 81	datachem.de	GET /w00tw00t.at.ISC.SANS.DFind:) HTTP/1.1[
10:59:12.618	TCP	8080	IIS Proxy	datachem.de	GET /w00tw00t.at.ISC.SANS.DFind:) HTTP/1.1[
10:59:12.668	TCP	8081	TCP Closed Port	datachem.de	
10:59:12.668	TCP	8181	TCP Closed Port	datachem.de	
10:59:12.668	TCP	9090	TCP Closed Port	datachem.de	
10:59:12.598	TCP	80	IIS	datachem.de	GET /w00tw00t.at.ISC.SANS.DFind:) HTTP/1.1[
10:59:13.159	TCP	9090	TCP Closed Port	datachem.de	
10:59:13.159	TCP	8081	TCP Closed Port	datachem.de	
10:59:13.159	TCP	8181	TCP Closed Port	datachem.de	
10:59:13.159	TCP	8180	TCP Closed Port	datachem.de	
10:59:13.810	TCP	9090	TCP Closed Port	datachem.de	
10:59:13.810	TCP	8081	TCP Closed Port	datachem.de	
10:59:13.810	TCP	8180	TCP Closed Port	datachem.de	
10:59:13.810	TCP	8181	TCP Closed Port	datachem.de	

3. **Install Jcrypt/CryptTool and demonstrate Asymmetric, Symmetric crypto algorithm, Hash and Digital/PKI signatures.**

Introduction

Jcrypt/CryptTool is a valuable application for exploring cryptographic methods. By supporting various encryption algorithms and cryptographic functions, it provides insights into securing data while maintaining confidentiality, integrity, and authenticity.

Steps to Download and Install

To start using CryptTool, download it from the official website. Select the version that matches your operating system, save the installer, and follow the setup prompts. Once installed, launch the application to confirm successful installation.

Using Jcrypt/CryptTool

1.Symmetric-Encryption

Symmetric encryption uses a single key for both encryption and decryption.

- **Steps**: Select a symmetric algorithm (e.g., AES), input your plaintext (e.g., "Confidential Data"), and provide a key (e.g., "Secret123"). Encrypt the message to generate the output.
- **Example**:
  - **Input**: Confidential Data
  - **Key**: Secret123
  - **Output**: AES-encrypted text

2.Asymmetric-Encryption

This method uses a key pair (public and private) for secure communication.

- **Steps**: Select an asymmetric algorithm (e.g., RSA), generate a key pair, encrypt a message (e.g., "Hello World") with the public key, and decrypt it with the private key.
- **Example**:
  - **Input**: Hello World
  - **Output**: RSA-encrypted text

3.Hashing

Hashing generates a fixed-size hash value, ensuring data integrity.

- **Steps**: Choose a hashing algorithm (e.g., SHA-256), input your message (e.g., "Important Document"), and create the hash.
- **Example**:
  - **Input**: Important Document
  - **Output**: SHA-256 hash

4.Digital-Signatures

Digital signatures authenticate messages and verify their integrity.

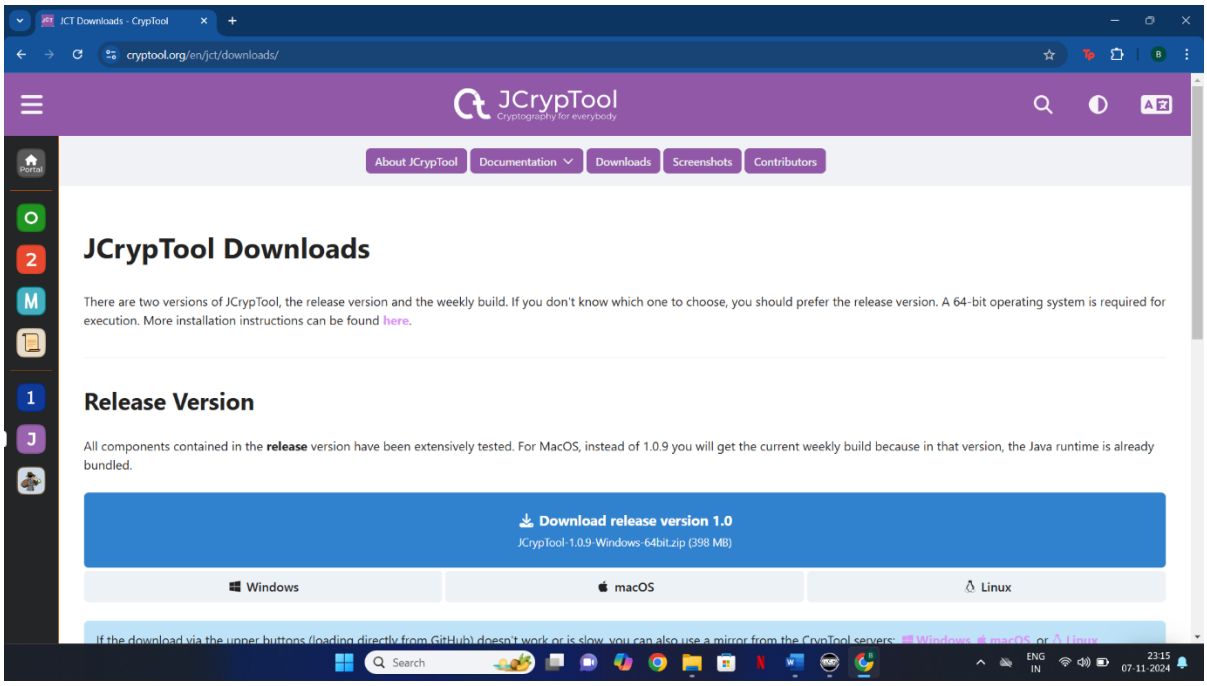
- **Steps**: Use an RSA key pair to sign a message (e.g., "Verify Sender") with the private key and verify it using the public key.
- **Example**:
  - **Message**: Verify Sender
  - **Output**: A digital signature verified using the public key

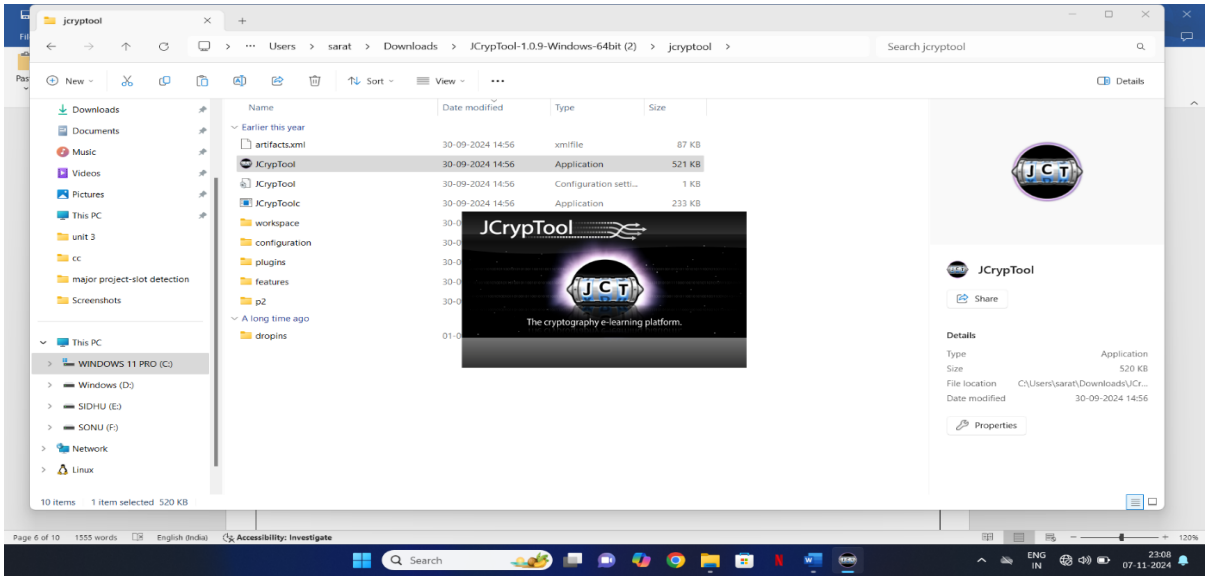
Key Insights

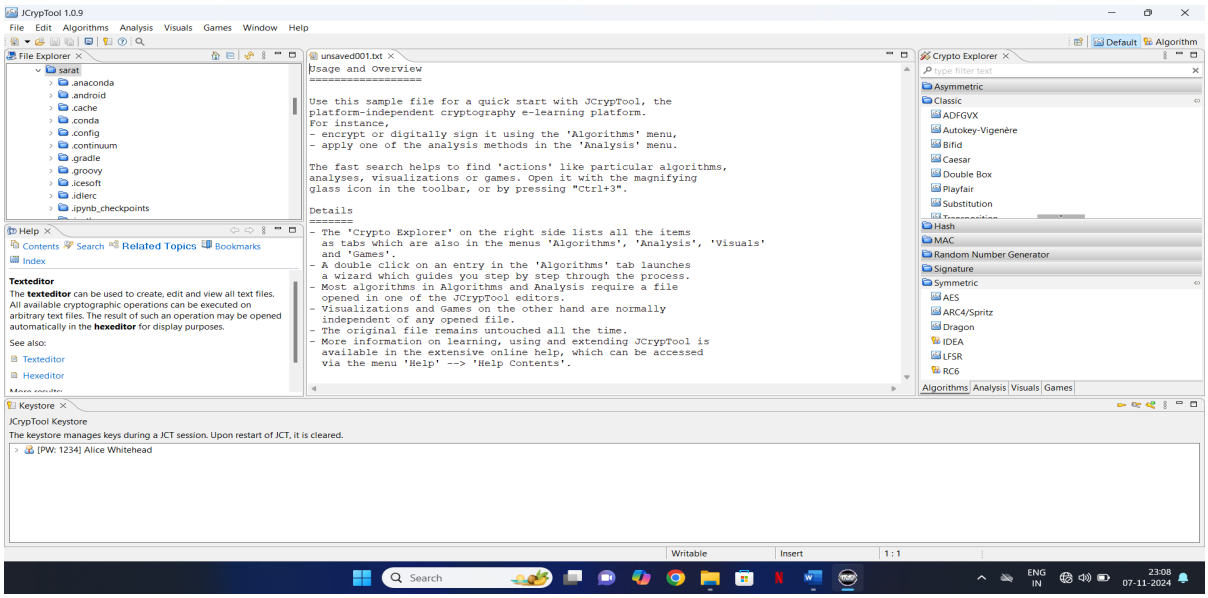
- Symmetric encryption is faster and suitable for large datasets.
- Asymmetric encryption enhances security by using key pairs for data exchange.
- Hashing ensures data consistency and integrity.
- Digital signatures confirm message authenticity and protect against tampering.

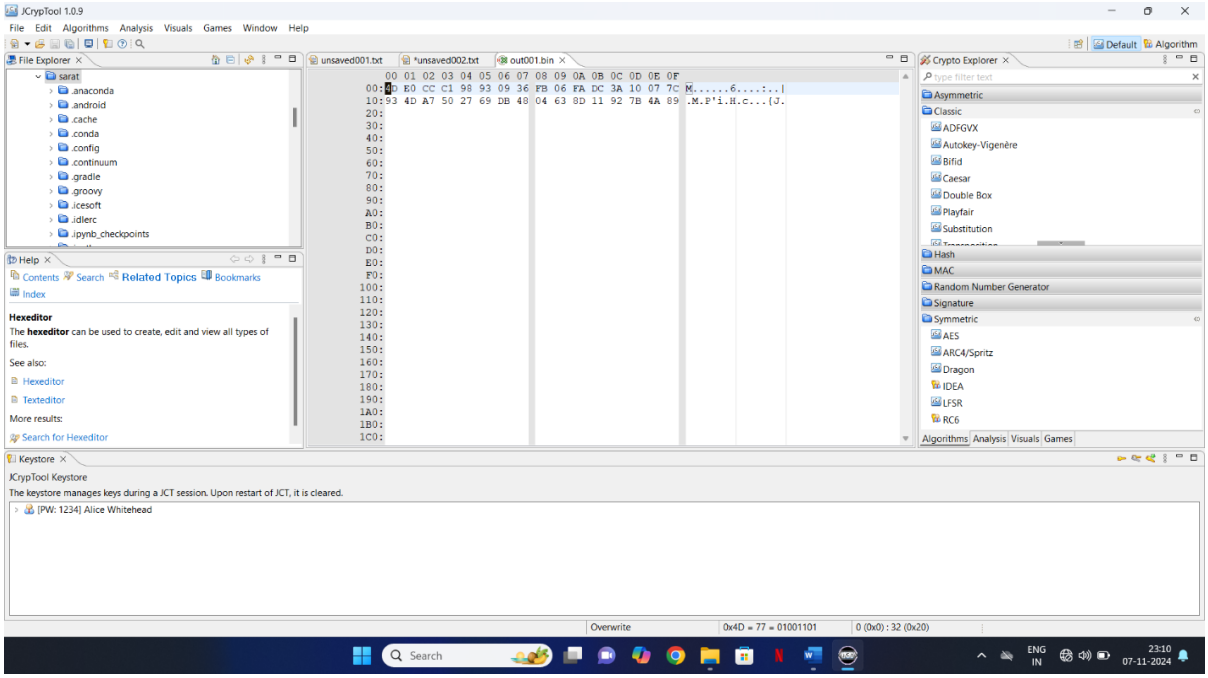
Conclusion

Jcrypt/CryptTool is an excellent platform for learning and applying cryptographic techniques. Its diverse functionalities provide users with a strong foundation in data security, ensuring that critical information remains protected in digital environments.






JCrypTool 1.0.9
File Edit Algorithms Analysis Visuals Games Window Help
File Explorer
sarat
.anaconda
.android
.cache
.conda
.config
.continuum
.gradle
.groovy
.icesoft
.idlerc
.ipynb_checkpoints
unsaved001.txt
Usage and Overview
Use this sample file for a quick start with JCrypTool, the
platform-independent cryptography e-learning platform.
For instance,
- encrypt or digitally sign it using the 'Algorithms' menu,
- apply one of the analysis methods in the 'Analysis' menu.
The fast search helps to find 'actions' like particular algorithms,
analyses, visualizations or games. Open it with the magnifying
glass icon in the toolbar, or by pressing "Ctrl+3".
Details
- The 'Crypto Explorer' on the right side lists all the items
as tabs which are also in the menus 'Algorithms', 'Analysis', 'Visuals'
and 'Games'.
- A double click on an entry in the 'Algorithms' tab launches
a wizard which guides you step by step through the process.
- Most algorithms in Algorithms and Analysis require a file
opened in one of the JCrypTool editors.
- Visualizations and Games on the other hand are normally
independent of any opened file.
- The original file remains untouched all the time.
- More information on learning, using and extending JCrypTool is
available in the extensive online help, which can be accessed
via the menu 'Help' --> 'Help Contents'.
Crypto Explorer
Asymmetric
Classic
ADFGVX
Autokey-Vigenère
Bifid
Caesar
Double Box
Playfair
Substitution
Hash
MAC
Random Number Generator
Signature
Symmetric
AES
ARC4/Spritz
Dragon
IDEA
LFSR
RC6
Algorithms Analysis Visuals Games
Help
Contents Search Related Topics Bookmarks
Index
Texteditor
The texteditor can be used to create, edit and view all text files.
All available cryptographic operations can be executed on
arbitrary text files. The result of such an operation may be opened
automatically in the hexeditor for display purposes.
See also:
Texteditor
Hexeditor
Keystore
JCrypTool Keystore
The keystore manages keys during a JCT session. Upon restart of JCT, it is cleared.
[PW: 1234] Alice Whitehead
Writable
Insert
1 : 1
Search
ENG
IN
23:08
07-11-2024


JCrypTool 1.0.9
File Edit Algorithms Analysis Visuals Games Window Help
File Explorer
sarat
.anaconda
.android
.cache
.conda
.config
.continuum
.gradle
.groovy
.icesoft
.idlerc
.ipynb_checkpoints
unsaved001.txt
*unsaved002.txt
out001.bin
00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
00: 4D E0 CC C1 98 93 09 36 FB 06 FA DC 3A 10 07 7C M.....6....:..|
10: 93 4D A7 50 27 69 DB 48 04 63 8D 11 92 7B 4A 89 .M.P'i.H.c...{J.
Crypto Explorer
Asymmetric
Classic
ADFGVX
Autokey-Vigenère
Bifid
Caesar
Double Box
Playfair
Substitution
Hash
MAC
Random Number Generator
Signature
Symmetric
AES
ARC4/Spritz
Dragon
IDEA
LFSR
RC6
Algorithms Analysis Visuals Games
Help
Contents Search Related Topics Bookmarks
Index
Hexeditor
The hexeditor can be used to create, edit and view all types of
files.
See also:
Hexeditor
Texteditor
More results:
Search for Hexeditor
Keystore
JCrypTool Keystore
The keystore manages keys during a JCT session. Upon restart of JCT, it is cleared.
[PW: 1234] Alice Whitehead
Overwrite
0x4D = 77 = 01001101
0 (0x0) : 32 (0x20)
Search
ENG
IN
23:10
07-11-2024

4. **Generate minimum 10 passwords of length 12 characters using openssl command.**

Introduction

OpenSSL is a widely adopted toolkit that offers robust cryptographic functions. One of its notable features is the ability to generate random strings, making it ideal for creating secure passwords and enforcing strong password policies in cybersecurity.

Steps to Download and Install

1.Downloading Open SSL

Visit the official OpenSSL download page and choose the appropriate version. For this guide, use **OpenSSL 3.2.1 Light**. Save the installer file to your device.

2.Installing Open SSL

Run the downloaded installer and follow the setup instructions. By default, install it in the directory C:\Program Files\OpenSSL-Win64. Once installation is complete, verify that OpenSSL is working correctly.

Setting Up the Environment Path

1. Adding OpenSSL to the System Path

- Navigate to Control Panel > System > Advanced System Settings > Environment Variables.
- Locate **Path** under **System Variables** and click **Edit**.
- Add the path to OpenSSL binaries, e.g., C:\Program Files\OpenSSL-Win64\bin, and save the changes.

2. Configuring PowerShell Path

- Repeat the steps above, but add the PowerShell path, e.g., C:\Windows\System32\WindowsPowerShell\v1.0.

Generating Secure Passwords

1. Open the Command Line

Launch PowerShell or Command Prompt to start generating passwords.

2. Generate Random Passwords

Use the following OpenSSL command to create a 12-character password:

Bash:

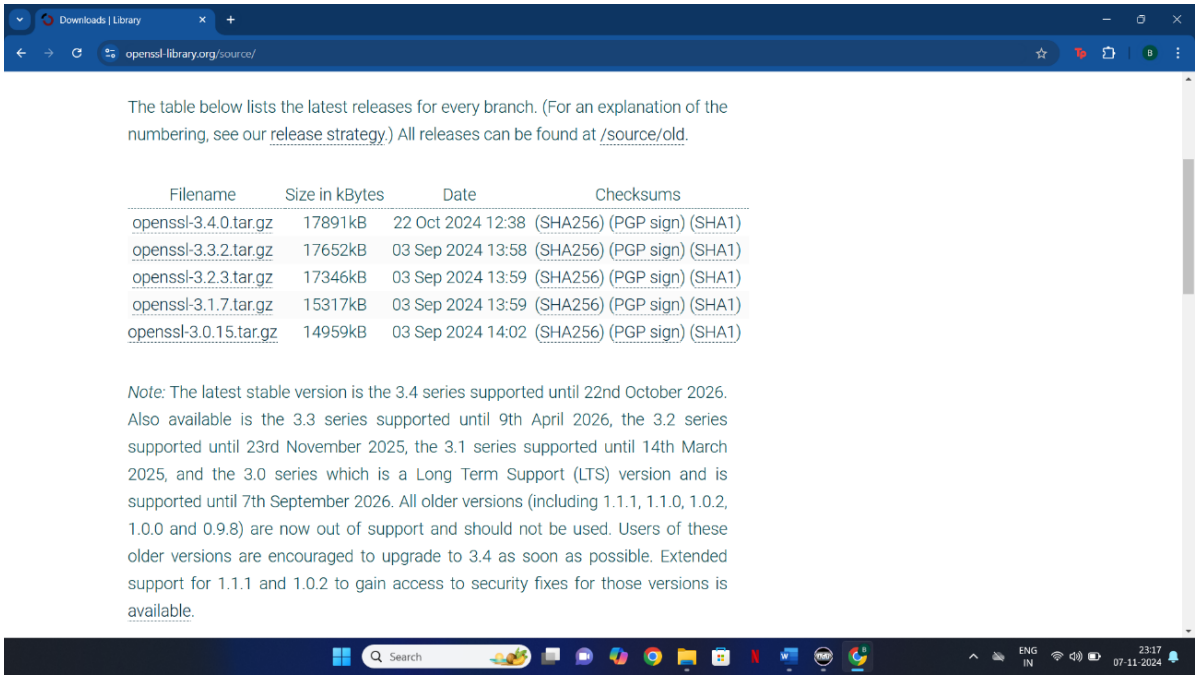
for \l in (1,110) do(openssl rand -base64 15 | openssl enc -base64 -A | powershell -command "$input.substring(0,12)")

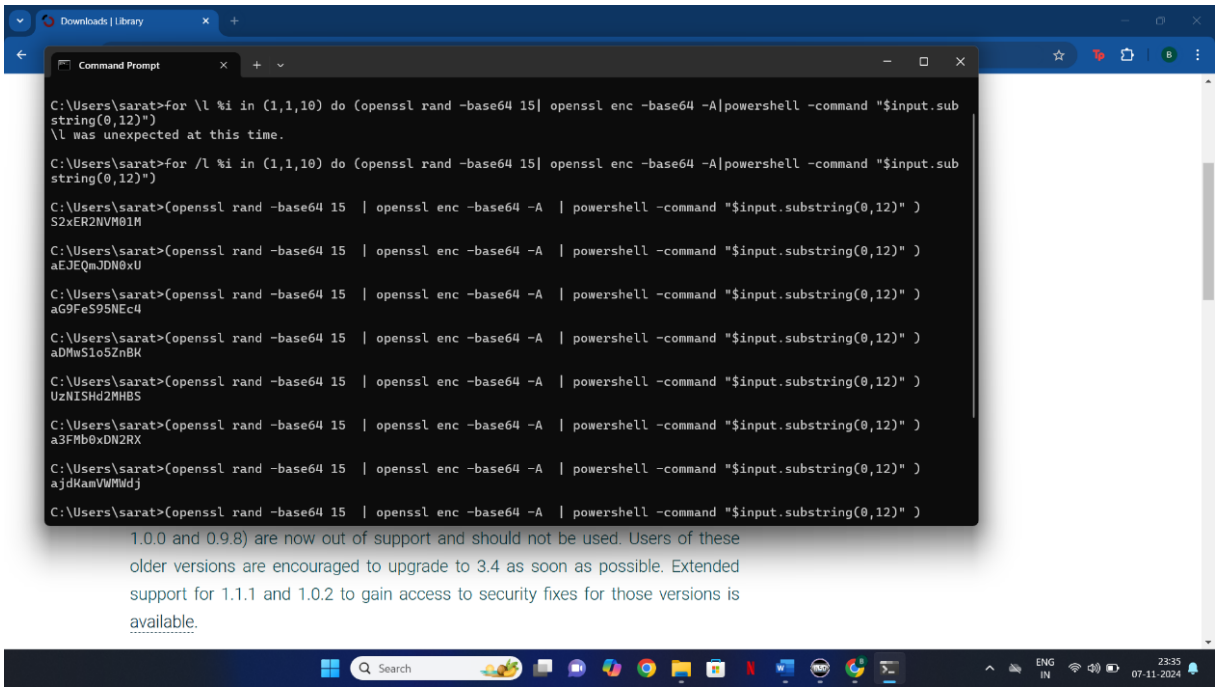
Key Insights

- Each password generated is random and unique, adhering to strong password requirements.
- The output consists of exactly 12 characters, ensuring consistency and security.

Conclusion

OpenSSL simplifies the generation of secure passwords through its cryptographic tools. By automating this process, the risk of predictable or weak passwords is minimized, significantly improving cybersecurity defenses.





5. Working with sniffers for monitoring network communication-Wireshark.

Introduction

Wireshark is a powerful network protocol analyzer used to capture and examine packet-level data on a network. This experiment demonstrates how unsecured HTTP traffic can expose sensitive information, such as login credentials, highlighting the need for HTTPS to secure communication channels.

Steps for Downloading and Installing Wireshark

1. Downloading Wireshark

- Visit the official Wireshark website.
- Select the version compatible with your operating system (Windows, macOS, or Linux).
- Download the installer.

2. Installing Wireshark

- Run the installer and follow the on-screen instructions to complete the installation.

Capturing Network Traffic with Wireshark

1. Launch Wireshark

- Open the Wireshark application after installation.

2. Select Network Interface

- Choose the appropriate network interface (e.g., Ethernet or WiFi) that you want to monitor.

3. Start Capturing Traffic

- Click on **Start** to begin capturing network traffic.

4. Access the Test Website

- Open a web browser and navigate to an unsecured HTTP website (e.g., testphp.login).

5. Enter Login Details

- Input a sample username and password to simulate a login attempt.

6. Stop the Capture

- After submitting the login details, go back to Wireshark and click **Stop** to end the capture.

7. Filter for HTTP Traffic

- In the Wireshark filter bar, type http and press Enter to filter the captured traffic and display only HTTP-related packets.

8. Search for Login Requests

- Look for packets that relate to the login process, specifically those with URLs like http://login.php or HTTP status codes like 1.1.

9. Follow the TCP Stream

- Select the HTTP packet related to the login page, right-click on it, and choose **Follow > TCP Stream**.

10. Analyze the TCP Stream

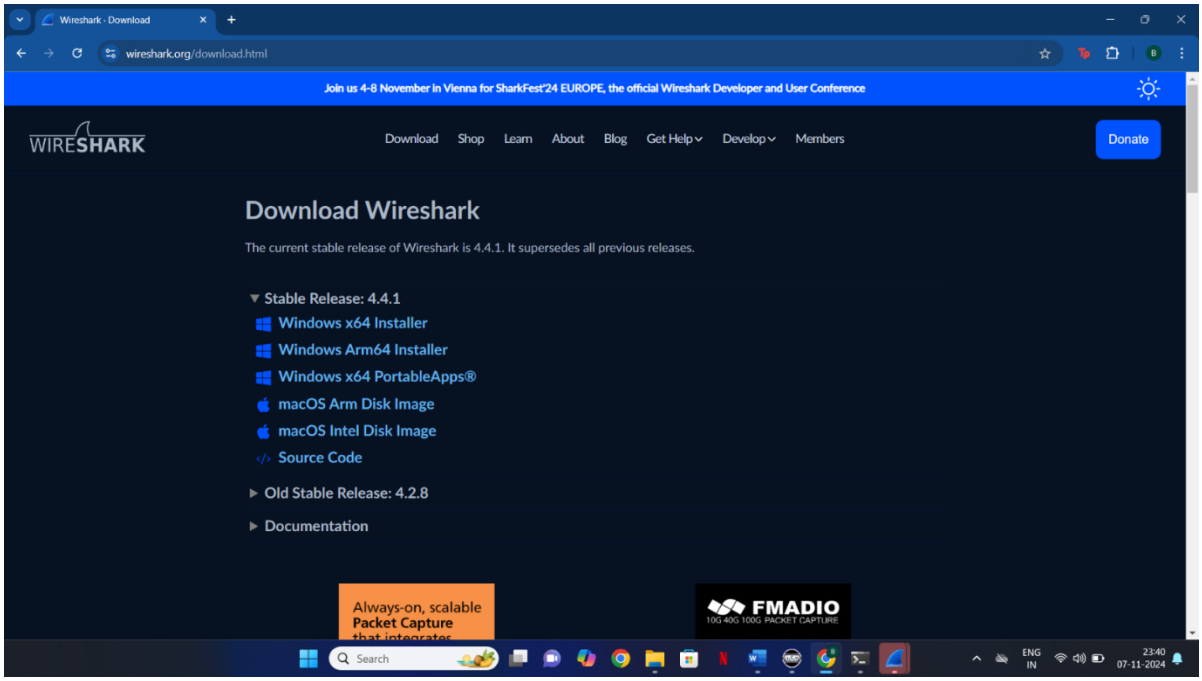
- In the TCP Stream window, observe the data being transmitted. The username and password entered on the website may be visible in plain text, demonstrating the risks of using HTTP.

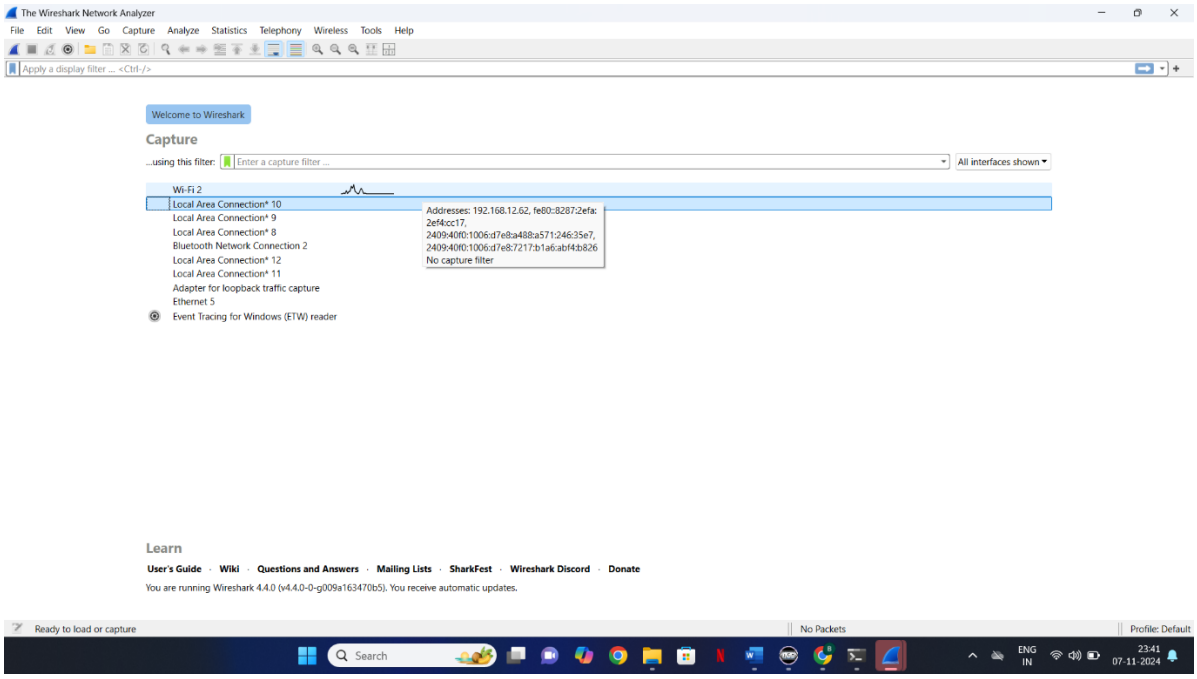
Key Insights

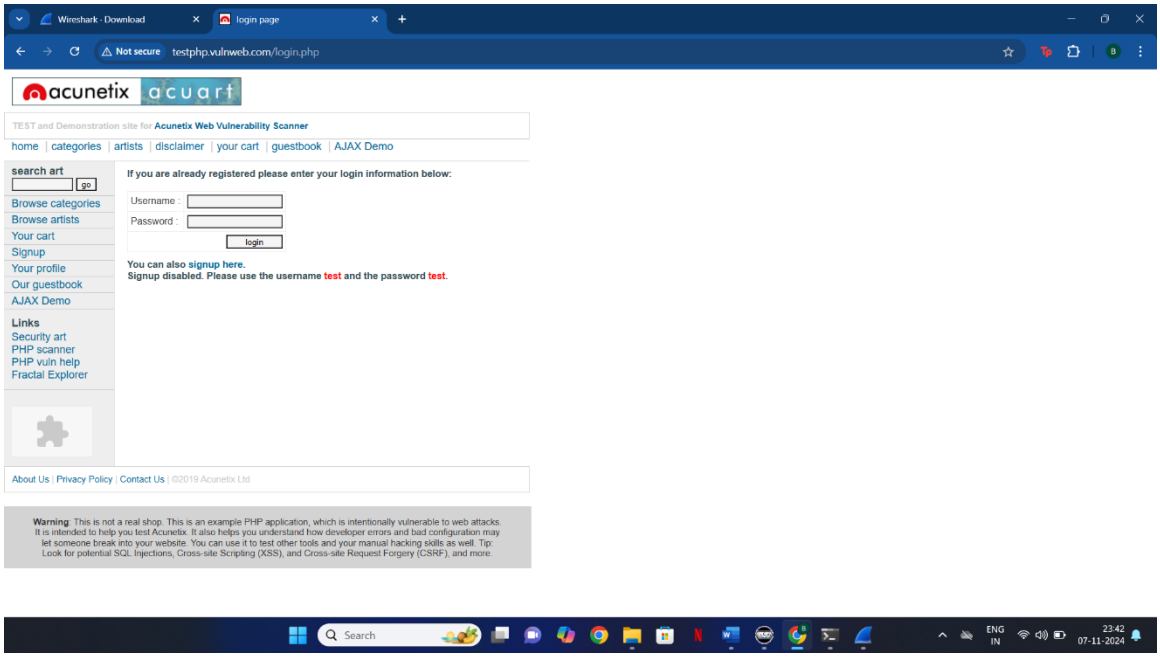
- Sensitive information, such as login credentials, can be easily intercepted on unsecured HTTP sites. This illustrates why HTTPS is crucial for protecting sensitive data during transmission.

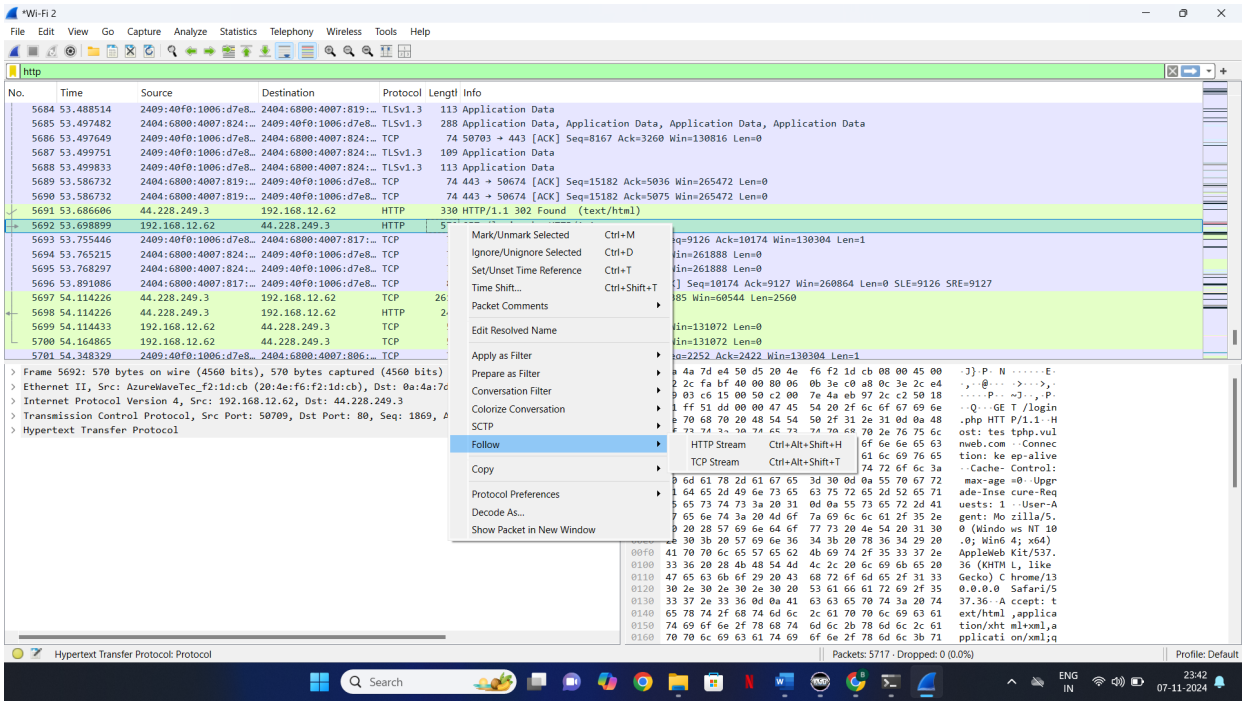
Conclusion

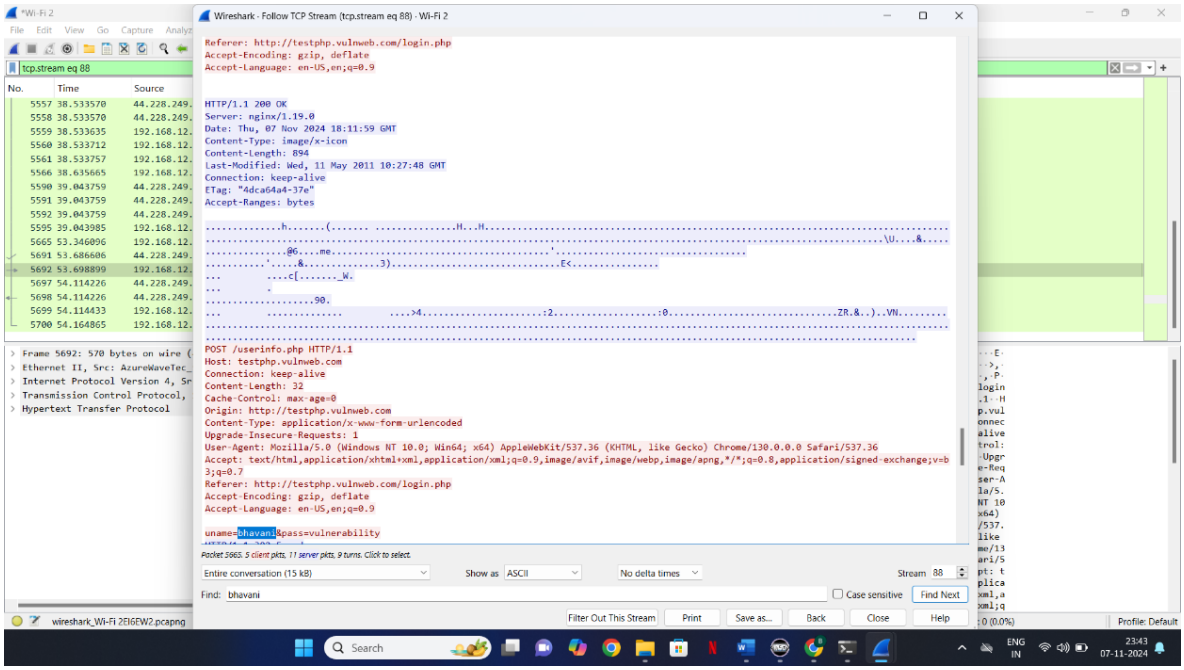
Wireshark effectively captures and displays network communication, highlighting the vulnerability of unsecured HTTP traffic. This experiment underscores the importance of using HTTPS to ensure secure data transmission and protect user privacy from potential interception.






Wireshark - Download
login page
Not secure testphp.vulnweb.com/login.php
acunetix acuart
TEST and Demonstration site for Acunetix Web Vulnerability Scanner
home | categories | artists | disclaimer | your cart | guestbook | AJAX Demo
search art
go
Browse categories
Browse artists
Your cart
Signup
Your profile
Our guestbook
AJAX Demo
Links
Security art
PHP scanner
PHP vuln help
Fractal Explorer
If you are already registered please enter your login information below:
Username :
Password :
login
You can also signup here.
Signup disabled. Please use the username test and the password test.
About Us | Privacy Policy | Contact Us | ©2019 Acunetix Ltd
Warning: This is not a real shop. This is an example PHP application, which is intentionally vulnerable to web attacks.
It is intended to help you test Acunetix. It also helps you understand how developer errors and bad configuration may let someone break into your website. You can use it to test other tools and your manual hacking skills as well. Tip:
Look for potential SQL Injections, Cross-site Scripting (XSS), and Cross-site Request Forgery (CSRF), and more.
Search
ENG IN
23:42 07-11-2024


*Wi-Fi 2
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
http
No. Time Source Destination Protocol Length Info
5684 53.488514 2409:40f0:1006:d7e8... 2404:6800:4007:819:... TLSv1.3 113 Application Data
5685 53.497482 2404:6800:4007:824:... 2409:40f0:1006:d7e8... TLSv1.3 288 Application Data, Application Data, Application Data, Application Data
5686 53.497649 2409:40f0:1006:d7e8... 2404:6800:4007:824:... TCP 74 50703 → 443 [ACK] Seq=8167 Ack=3260 Win=130816 Len=0
5687 53.499751 2409:40f0:1006:d7e8... 2404:6800:4007:824:... TLSv1.3 109 Application Data
5688 53.499833 2409:40f0:1006:d7e8... 2404:6800:4007:824:... TLSv1.3 113 Application Data
5689 53.586732 2404:6800:4007:819:... 2409:40f0:1006:d7e8... TCP 74 443 → 50674 [ACK] Seq=15182 Ack=5036 Win=265472 Len=0
5690 53.586732 2404:6800:4007:819:... 2409:40f0:1006:d7e8... TCP 74 443 → 50674 [ACK] Seq=15182 Ack=5075 Win=265472 Len=0
5691 53.686606 44.228.249.3 192.168.12.62 HTTP 330 HTTP/1.1 302 Found (text/html)
5692 53.698899 192.168.12.62 44.228.249.3 HTTP
5693 53.755446 2409:40f0:1006:d7e8... 2404:6800:4007:817:... TCP
5694 53.765215 2404:6800:4007:824:... 2409:40f0:1006:d7e8... TCP
5695 53.768297 2404:6800:4007:824:... 2409:40f0:1006:d7e8... TCP
5696 53.891086 2404:6800:4007:817:... 2409:40f0:1006:d7e8... TCP
5697 54.114226 44.228.249.3 192.168.12.62 TCP
5698 54.114226 44.228.249.3 192.168.12.62 HTTP
5699 54.114433 192.168.12.62 44.228.249.3 TCP
5700 54.164865 192.168.12.62 44.228.249.3 TCP
5701 54.348329 2409:40f0:1006:d7e8... 2404:6800:4007:806:... TCP
Frame 5692: 570 bytes on wire (4560 bits), 570 bytes captured (4560 bits)
Ethernet II, Src: AzureWaveTec_f2:1d:cb (20:4e:f6:f2:1d:cb), Dst: 0a:4a:7d
Internet Protocol Version 4, Src: 192.168.12.62, Dst: 44.228.249.3
Transmission Control Protocol, Src Port: 50709, Dst Port: 80, Seq: 1869,
Hypertext Transfer Protocol
Mark/Unmark Selected Ctrl+M
Ignore/Unignore Selected Ctrl+D
Set/Unset Time Reference Ctrl+T
Time Shift... Ctrl+Shift+T
Packet Comments
Edit Resolved Name
Apply as Filter
Prepare as Filter
Conversation Filter
Colorize Conversation
SCTP
Follow
Copy
Protocol Preferences
Decode As...
Show Packet in New Window
HTTP Stream Ctrl+Alt+Shift+H
TCP Stream Ctrl+Alt+Shift+T
Hypertext Transfer Protocol: Protocol
Packets: 5717 · Dropped: 0 (0.0%)
Profile: Default
Search
ENG IN
23:42 07-11-2024


Wireshark - Follow TCP Stream (tcp.stream eq 88) - Wi-Fi 2
Referer: http://testphp.vulnweb.com/login.php
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
HTTP/1.1 200 OK
Server: nginx/1.19.0
Date: Thu, 07 Nov 2024 18:11:59 GMT
Content-Type: image/x-icon
Content-Length: 894
Last-Modified: Wed, 11 May 2011 10:27:48 GMT
Connection: keep-alive
ETag: "4dca64a4-37e"
Accept-Ranges: bytes
POST /userinfo.php HTTP/1.1
Host: testphp.vulnweb.com
Connection: keep-alive
Content-Length: 32
Cache-Control: max-age=0
Origin: http://testphp.vulnweb.com
Content-Type: application/x-www-form-urlencoded
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/130.0.0.0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Referer: http://testphp.vulnweb.com/login.php
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
uname=bhavani&pass=vulnerability
Entire conversation (15 kB)
Show as ASCII
No delta times
Stream 88
Find: bhavani
Case sensitive
Find Next
Filter Out This Stream
Print
Save as...
Back
Close
Help
Frame 5692: 570 bytes on wire
Ethernet II, Src: AzureWaveTec
Internet Protocol Version 4, Sr
Transmission Control Protocol,
Hypertext Transfer Protocol
wireshark_Wi-Fi 2EI6FW2.pcapng
Profile: Default
Search
ENG
IN
23:43
07-11-2024

6. **Using Snort, perform real time traffic analysis and packet logging.**

Introduction

SNORT is a powerful open-source Intrusion Detection System (IDS) and Intrusion Prevention System (IPS) that offers real-time network analysis and packet logging. It alerts network administrators to potential security threats by detecting a wide variety of network attacks, including Denial of Service (DoS), Distributed Denial of Service (DDoS), CGI attacks, buffer overflows, and stealth port scans. SNORT uses a rule-based system for attack detection and offers multiple operating modes for network security.

Download and Installation Instructions

1. **Download SNORT**
   - Visit the official SNORT website.
   - Download the latest version of SNORT for your operating system.
2. **Install SNORT**
   - Run the installer and follow the setup instructions to install SNORT.
   - Ensure SNORT is installed in the default directory (e.g., C:\snort).

Configuring the Path Variable for SNORT

1. **Access Environment Variables**
   - Go to **Control Panel** > **System** > **Advanced System Settings** > **Environment Variables**.
2. **Edit the Path Variable**
   - Under **System Variables**, locate and select **Path**, then click **Edit**.
   - Add the path to SNORT's **bin** directory (e.g., C:\snort\bin) to the list of path variables.
3. **Save Changes**
   - Click **OK** to save the changes and close all dialog boxes.

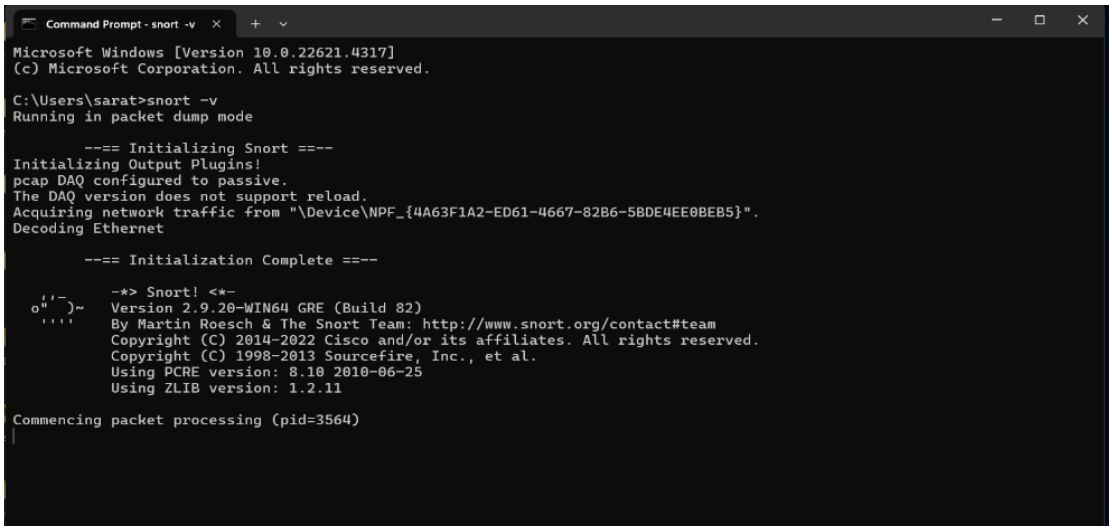
SNORT Operational Modes

1. Sniffer Mode

- **Command:**

```
snort -v
```

- **Description:** Displays TCP/IP packet headers in real-time, showing essential information such as source and destination addresses.

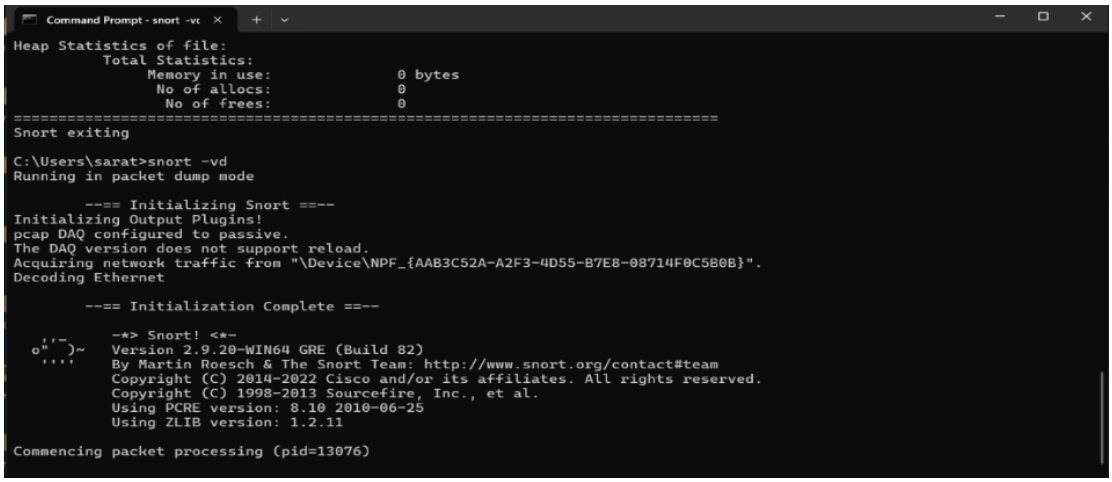


2. Extended Sniffer Mode:

- **Command:**

```
snort -vd
```

- **Description:** Shows TCP/IP and ICMP headers, as well as application data, providing more insight into packets in transit.

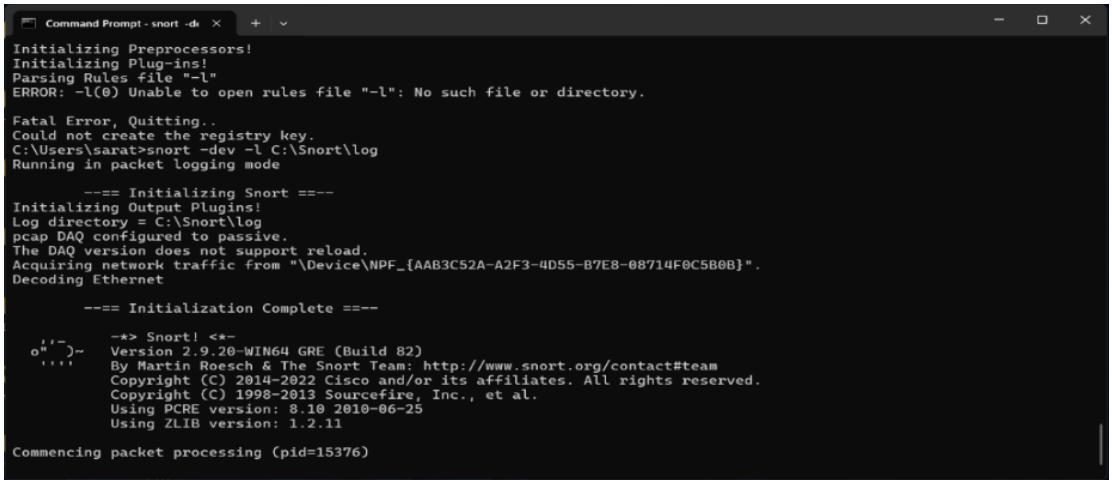


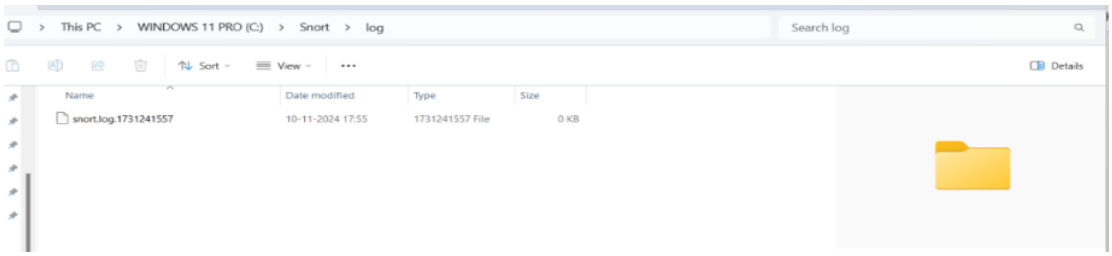
3. Packet Logger Mode

- **Command:**

```
snort -dev -l C:\snort\log
```

- **Description:** SNORT logs each packet it encounters to a specified directory (C:\snort\log). In this mode, it captures detailed packet information, creating a comprehensive log for offline analysis.
- **Setup:**
  Ensure that the directory C:\snort\log exists on the drive.



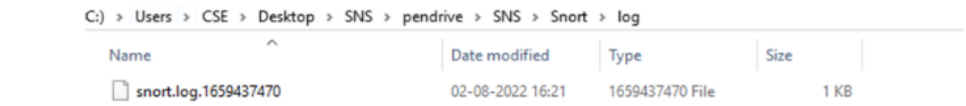


4. Network Intrusion Detection System (NIDS) Mode

- **Command:**

```
snort -c C:\snort\etc\snort.conf -l C:\snort\log
```

- **Description:** SNORT reads its configuration and rules from snort.conf, identifying and alerting on potentially malicious activities based on rule sets. It logs packets related to identified threats.



Key Observations

- In Sniffer Mode and Packet Logger Mode, observe how SNORT effectively captures and logs network packets for analysis, providing insights into network traffic.
- In Network IDS Mode (NIDS), focus on the alerts generated for specific malicious traffic patterns. Analyze how SNORT identifies rule-based attacks such as DoS, DDoS, or buffer overflows and verifies its ability to detect potential threats.

Conclusion

- SNORT's flexibility in operating modes allows for comprehensive monitoring and logging of network traffic in real time. Its rule-based detection capabilities play a vital role in proactive threat identification, making SNORT an essential tool for network security. It helps administrators safeguard systems by quickly identifying malicious activities and providing alerts.

7. **Perform email analysis using Autopsy tool.**

Autopsy Overview

Autopsy is an open-source digital forensics platform designed for investigating data from storage devices and email accounts. It plays a crucial role in email forensics by helping investigators find email artifacts, retrieve metadata, examine attachments, and analyze communication patterns. The tool offers a user-friendly, structured interface to facilitate the examination of evidence during forensic investigations.

Opening Autopsy and Creating a New Case

1. Launch the Autopsy application.
2. Select **New Case** to initiate a new investigation.

Case Setup

1. In the **New Case Information** dialog box:
    - **Case Name**: Enter a distinctive name for the case.
    - **Base Directory**: Choose the location to store the case files.
    - **Case Type**: Select the appropriate case type based on the investigation's focus.
2. Optionally, provide additional details:
    - **Case Number**: Assign a unique identifier.
    - **Examiner Details**: Enter the investigator's name, contact details, and organization.

Adding Data Sources and Configuring the Case

Adding a Data Source

1. Once the case is created, the **Add Data Source** window will appear.
2. Select **Host**: Choose the host system relevant to the data source.
3. Select **Data Source Type**: Choose the type of data source (e.g., disk image or local folder).
4. **Select Data Source**: Browse to and select the email data source.
5. **Configure Digest**: Set hashing options for data integrity verification (if required).
6. Click **Next** to complete the data source addition.

Analyzing Email Artifacts

Locating Email Artifacts

1. Navigate to the **Results** section to view email-related artifacts.
2. Search for folders such as **Email Messages** or **Communication**, where Autopsy may automatically categorize the email data.

Reviewing Email Messages

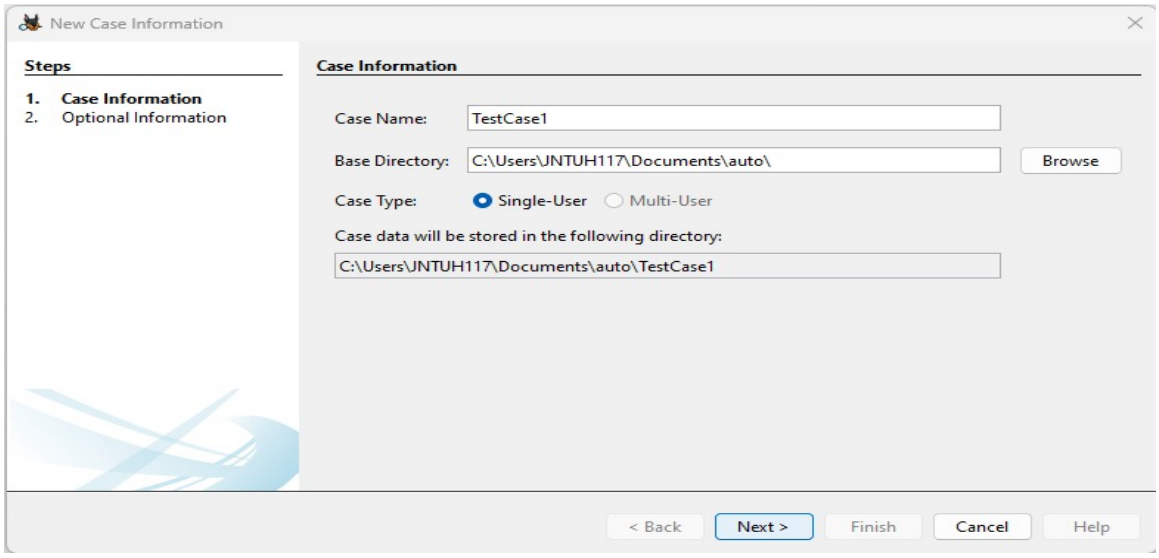
1. Browse through the listed emails.
2. For each email, you can:
    - **Examine Metadata**: Select an email to view details like sender, recipient, date, time, and subject.
    - **Review Content and Attachments**: Click on specific emails to examine the message body and any attached files.
    - **Analyze Communication Patterns**: Observe recurring contacts, frequent keywords, or suspicious email chains.

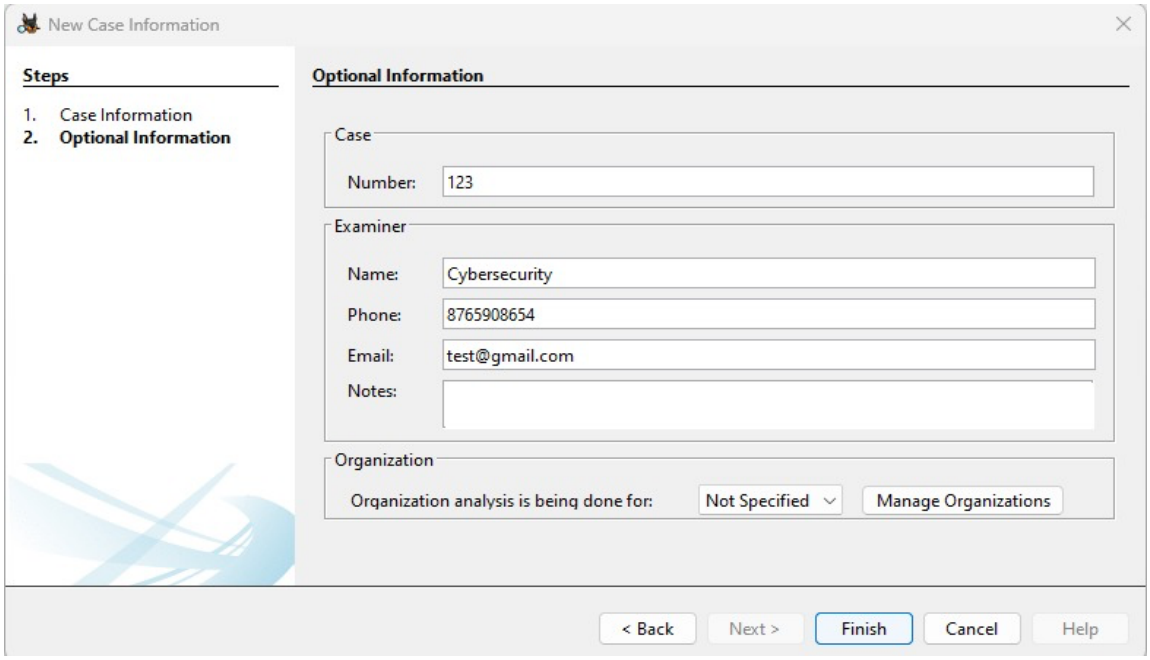
Key Observations

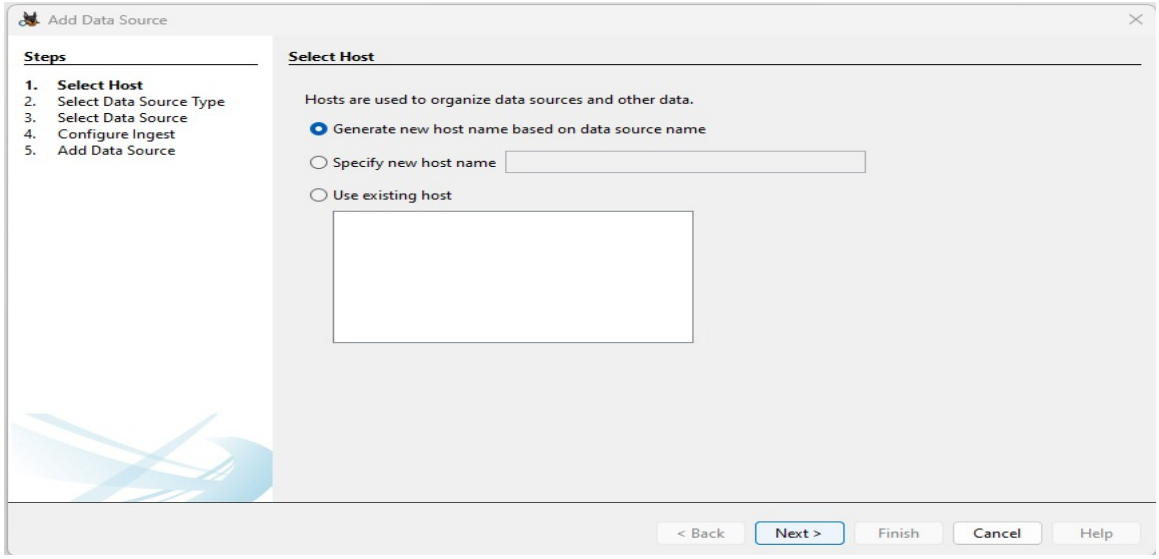
- Note any unusual communication, suspicious attachments, or metadata anomalies. Pay close attention to messages that may indicate phishing, spam, or other malicious activities.

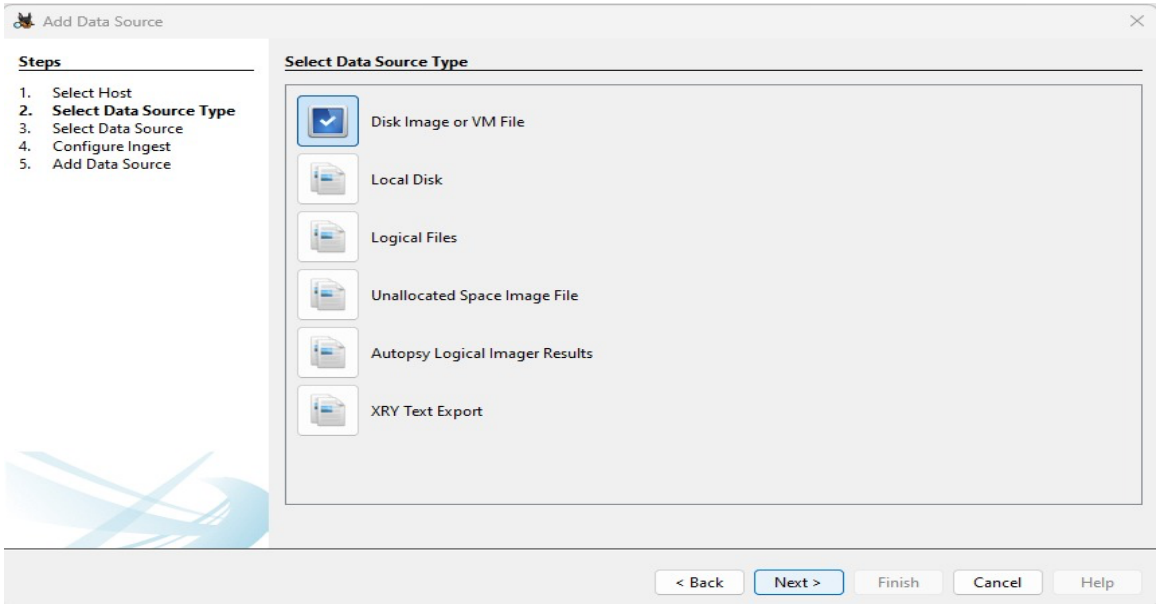
Conclusion

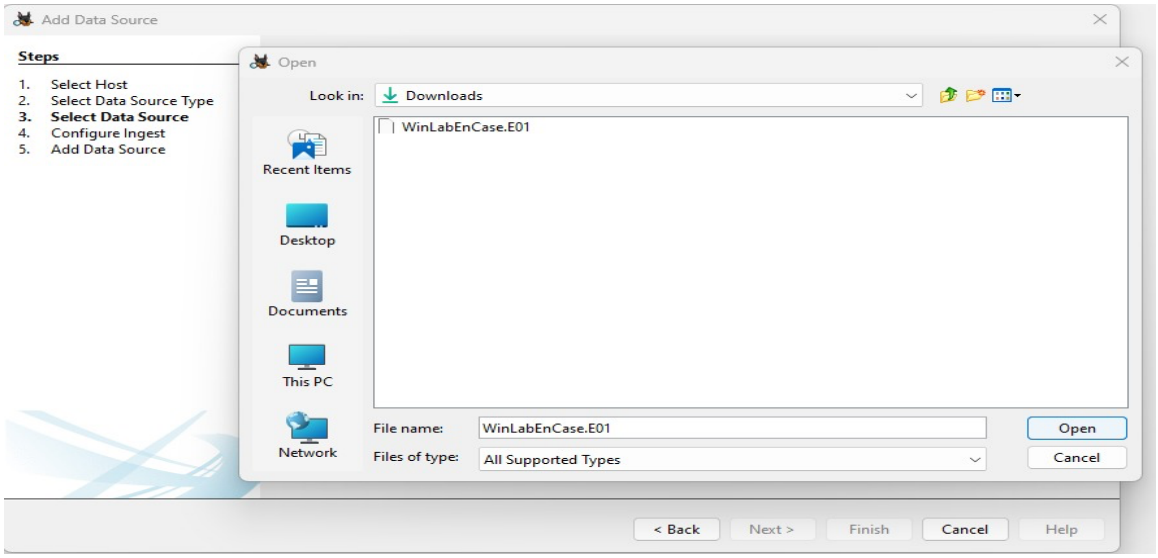
Autopsy provides a powerful framework for email analysis in digital forensics. It helps investigators efficiently examine email metadata, content, and attachments, making it a valuable tool for identifying potentially harmful emails and understanding the context of email exchanges in forensic investigations.

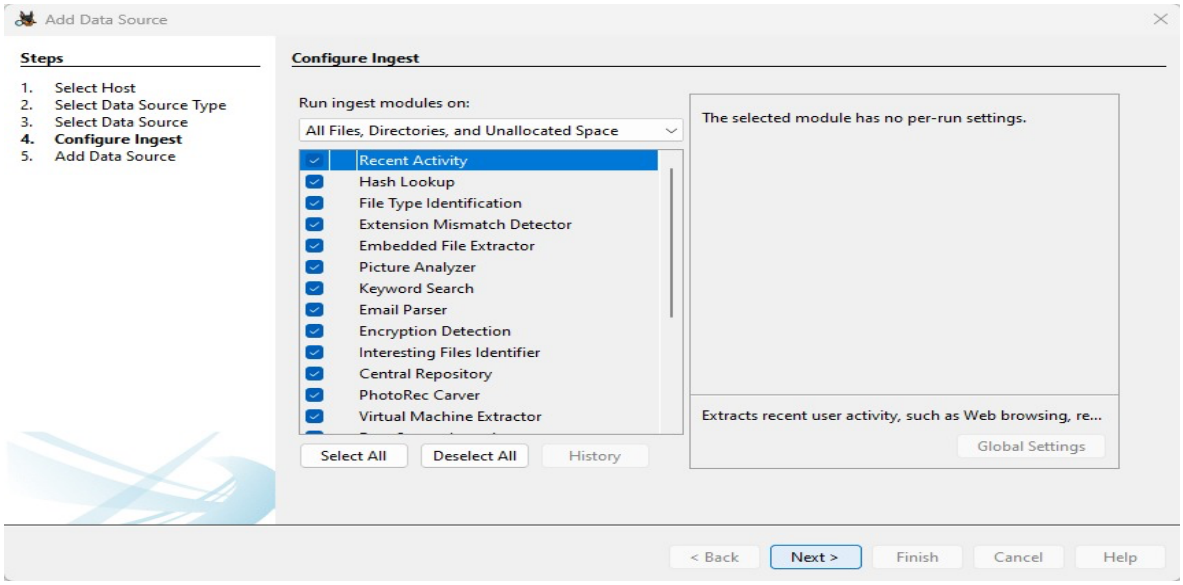


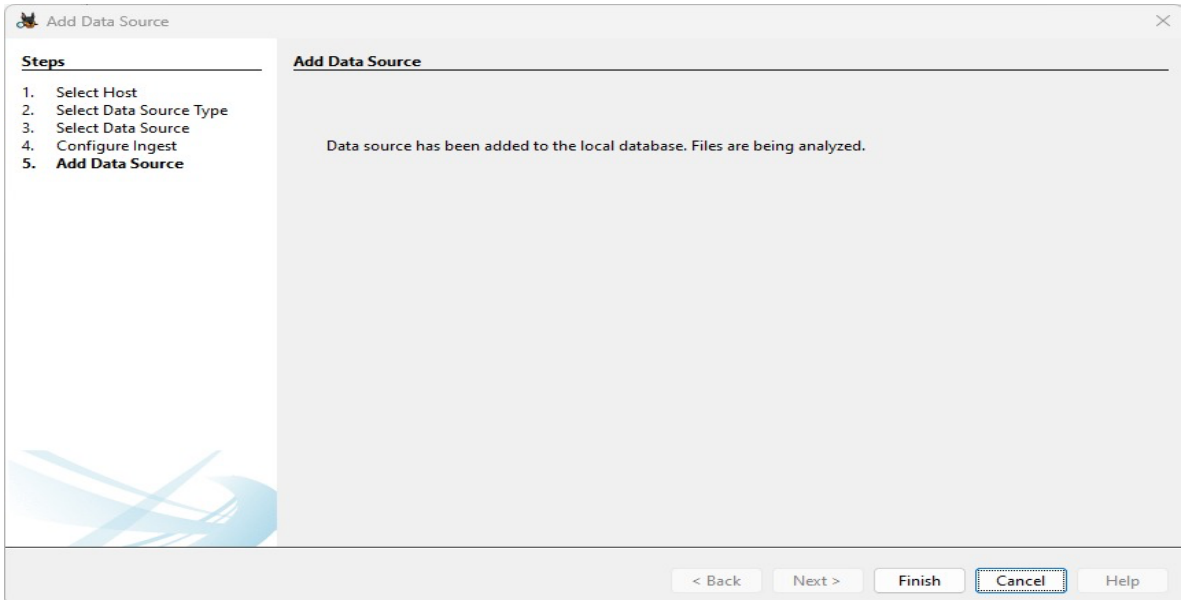

New Case Information
Steps
1. Case Information
2. Optional Information
Optional Information
Case
Number: 123
Examiner
Name: Cybersecurity
Phone: 8765908654
Email: test@gmail.com
Notes:
Organization
Organization analysis is being done for: Not Specified
Manage Organizations
< Back
Next >
Finish
Cancel
Help

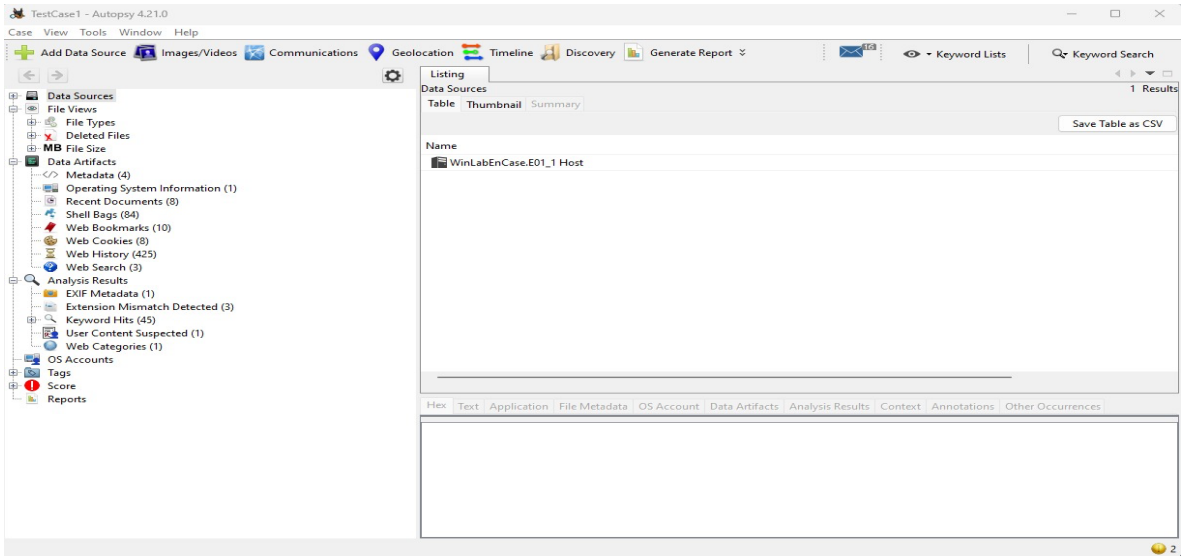

Add Data Source
Steps
1. Select Host
2. Select Data Source Type
3. Select Data Source
4. Configure Ingest
5. Add Data Source
Select Host
Hosts are used to organize data sources and other data.
Generate new host name based on data source name
Specify new host name
Use existing host
< Back
Next >
Finish
Cancel
Help

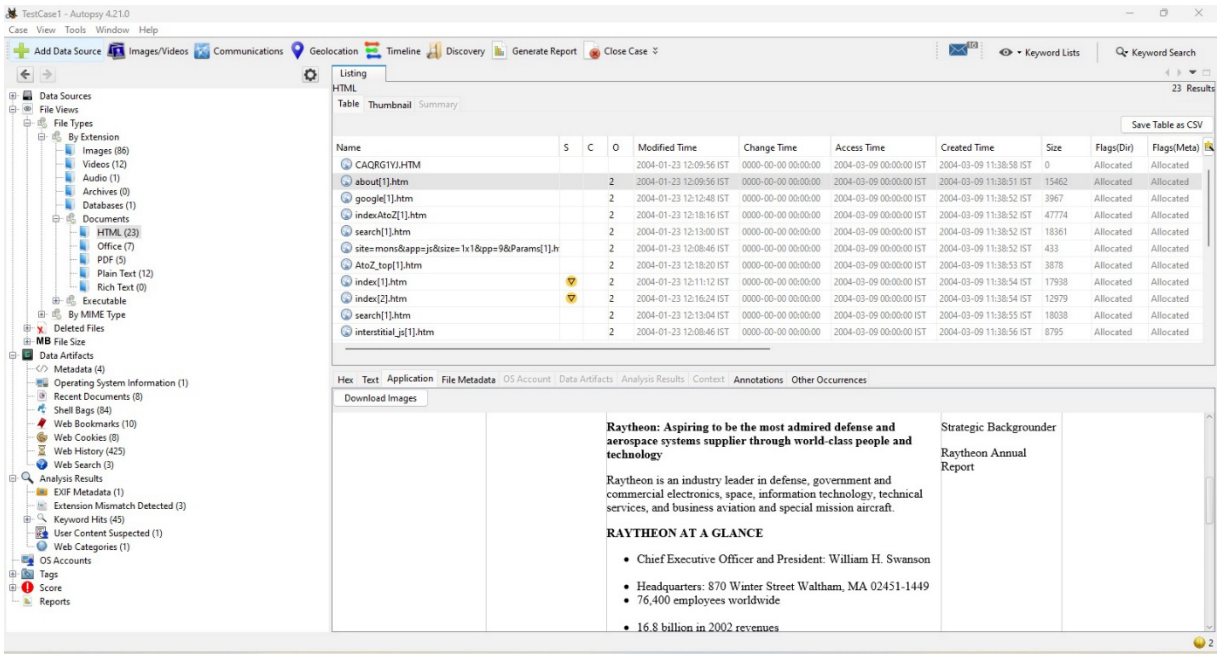

Add Data Source
Steps
1. Select Host
2. Select Data Source Type
3. Select Data Source
4. Configure Ingest
5. Add Data Source
Select Data Source Type
Disk Image or VM File
Local Disk
Logical Files
Unallocated Space Image File
Autopsy Logical Imager Results
XRY Text Export
< Back
Next >
Finish
Cancel
Help


Add Data Source
Steps
1. Select Host
2. Select Data Source Type
3. Select Data Source
4. Configure Ingest
5. Add Data Source
Open
Look in: Downloads
WinLabEnCase.E01
Recent Items
Desktop
Documents
This PC
Network
File name: WinLabEnCase.E01
Files of type: All Supported Types
Open
Cancel
< Back
Next >
Finish
Cancel
Help


Add Data Source
Steps
1. Select Host
2. Select Data Source Type
3. Select Data Source
4. Configure Ingest
5. Add Data Source
Configure Ingest
Run ingest modules on:
All Files, Directories, and Unallocated Space
Recent Activity
Hash Lookup
File Type Identification
Extension Mismatch Detector
Embedded File Extractor
Picture Analyzer
Keyword Search
Email Parser
Encryption Detection
Interesting Files Identifier
Central Repository
PhotoRec Carver
Virtual Machine Extractor
Select All
Deselect All
History
The selected module has no per-run settings.
Extracts recent user activity, such as Web browsing, re...
Global Settings
< Back
Next >
Finish
Cancel
Help


Add Data Source
Steps
1. Select Host
2. Select Data Source Type
3. Select Data Source
4. Configure Ingest
5. Add Data Source
Add Data Source
Data source has been added to the local database. Files are being analyzed.
< Back
Next >
Finish
Cancel
Help


TestCase1 - Autopsy 4.21.0
Case View Tools Window Help
Add Data Source Images/Videos Communications Geolocation Timeline Discovery Generate Report Keyword Lists Keyword Search
Data Sources
File Views
File Types
Deleted Files
File Size
Data Artifacts
Metadata (4)
Operating System Information (1)
Recent Documents (8)
Shell Bags (84)
Web Bookmarks (10)
Web Cookies (8)
Web History (425)
Web Search (3)
Analysis Results
EXIF Metadata (1)
Extension Mismatch Detected (3)
Keyword Hits (45)
User Content Suspected (1)
Web Categories (1)
OS Accounts
Tags
Score
Reports
Listing
Data Sources
1 Results
Table Thumbnail Summary
Save Table as CSV
Name
WinLabEnCase.E01_1 Host
Hex Text Application File Metadata OS Account Data Artifacts Analysis Results Context Annotations Other Occurrences


TestCase1 - Autopsy 4.21.0
Case View Tools Window Help
Add Data Source Images/Videos Communications Geolocation Timeline Discovery Generate Report Close Case Keyword Lists Keyword Search
Data Sources
File Views
File Types
By Extension
Images (86)
Videos (12)
Audio (1)
Archives (0)
Databases (1)
Documents
HTML (23)
Office (7)
PDF (5)
Plain Text (12)
Rich Text (0)
Executable
By MIME Type
Deleted Files
File Size
Data Artifacts
Metadata (4)
Operating System Information (1)
Recent Documents (8)
Shell Bags (84)
Web Bookmarks (10)
Web Cookies (8)
Web History (425)
Web Search (3)
Analysis Results
EXIF Metadata (1)
Extension Mismatch Detected (3)
Keyword Hits (45)
User Content Suspected (1)
Web Categories (1)
OS Accounts
Tags
Score
Reports
Listing
HTML
23 Results
Table Thumbnail Summary
Save Table as CSV
Name S C O Modified Time Change Time Access Time Created Time Size Flags(Dir) Flags(Meta)
CAQRG1YJ.HTM 2004-01-23 12:09:56 IST 0000-00-00 00:00:00 2004-03-09 00:00:00 IST 2004-03-09 11:38:58 IST 0 Allocated Allocated
about[1].htm 2 2004-01-23 12:09:56 IST 0000-00-00 00:00:00 2004-03-09 00:00:00 IST 2004-03-09 11:38:51 IST 15462 Allocated Allocated
google[1].htm 2 2004-01-23 12:12:48 IST 0000-00-00 00:00:00 2004-03-09 00:00:00 IST 2004-03-09 11:38:52 IST 3967 Allocated Allocated
indexAtoZ[1].htm 2 2004-01-23 12:18:16 IST 0000-00-00 00:00:00 2004-03-09 00:00:00 IST 2004-03-09 11:38:52 IST 47774 Allocated Allocated
search[1].htm 2 2004-01-23 12:13:00 IST 0000-00-00 00:00:00 2004-03-09 00:00:00 IST 2004-03-09 11:38:52 IST 18361 Allocated Allocated
site=mons&app=js&size=1x1&pp=9&Params[1].h 2 2004-01-23 12:08:46 IST 0000-00-00 00:00:00 2004-03-09 00:00:00 IST 2004-03-09 11:38:52 IST 433 Allocated Allocated
AtoZ_top[1].htm 2 2004-01-23 12:18:20 IST 0000-00-00 00:00:00 2004-03-09 00:00:00 IST 2004-03-09 11:38:53 IST 3878 Allocated Allocated
index[1].htm 2 2004-01-23 12:11:12 IST 0000-00-00 00:00:00 2004-03-09 00:00:00 IST 2004-03-09 11:38:54 IST 17938 Allocated Allocated
index[2].htm 2 2004-01-23 12:16:24 IST 0000-00-00 00:00:00 2004-03-09 00:00:00 IST 2004-03-09 11:38:54 IST 12979 Allocated Allocated
search[1].htm 2 2004-01-23 12:13:04 IST 0000-00-00 00:00:00 2004-03-09 00:00:00 IST 2004-03-09 11:38:55 IST 18038 Allocated Allocated
interstitial_js[1].htm 2 2004-01-23 12:08:46 IST 0000-00-00 00:00:00 2004-03-09 00:00:00 IST 2004-03-09 11:38:56 IST 8795 Allocated Allocated
Hex Text Application File Metadata OS Account Data Artifacts Analysis Results Context Annotations Other Occurrences
Download Images
Raytheon: Aspiring to be the most admired defense and aerospace systems supplier through world-class people and technology
Raytheon is an industry leader in defense, government and commercial electronics, space, information technology, technical services, and business aviation and special mission aircraft.
RAYTHEON AT A GLANCE
Chief Executive Officer and President: William H. Swanson
Headquarters: 870 Winter Street Waltham, MA 02451-1449
76,400 employees worldwide
16.8 billion in 2002 revenues
Strategic Backgrounder
Raytheon Annual Report

8. **Perform Registry analysis and get boottime logging using process monitor tool**

Process Monitor Overview

Process Monitor is a powerful monitoring tool for Windows that provides real-time tracking of file system, registry, process, and thread activity. It offers robust features such as non-destructive filtering, event properties, detailed logging, and boot-time monitoring, making it a critical tool for system troubleshooting and malware detection. By capturing and analyzing various system activities, including registry operations, file access, process creation, and network requests, Process Monitor helps in understanding system behavior and diagnosing issues.

Download and Installation

1. Download Process Monitor from the Sysinternals website.
2. Extract the files to a directory of your choice.

Understanding Process Monitor Files

The folder will contain the following key files:

- **Eula.txt**: License agreement.
- **procmon.chm**: Help file and documentation.
- **Procmon.exe**: Executable file to launch Process Monitor.
- **Procmon64.exe** and **Procmon64a.exe**: 64-bit versions of Process Monitor.

Running Process Monitor

1. Launch **procmon.exe** to start the tool.
2. Process Monitor begins capturing events immediately by default, displaying all system operations in real-time.

Process Monitor Interface and Features

Event Types Captured

Process Monitor captures five main categories of events:

- **Registry**: Monitors registry operations.
- **Filesystem**: Tracks file access, including read and write operations.
- **Network**: Observes network requests and activity.
- **Processes**: Monitors process creation, termination, and other related actions.
- **Profiling Events**: Provides periodic snapshots of process activity.

Event Columns

Each event is displayed with the following details:

- **Time of Day**: Timestamp of when the event occurred.
- **Process Name**: The name of the process initiating the event.
- **PID**: The process ID associated with the event.
- **Operation**: The action taken (e.g., registry modification, file read/write).
- **Path**: The object path related to the event, such as file or registry path.
- **Result**: The outcome of the event (e.g., SUCCESS, NAME NOT FOUND).
- **Detail**: Additional information about the event, providing context.

Enabling/Disabling Capture

- **Control Capture**: Click the magnifying glass icon in the toolbar to toggle event capturing. If a red "X" appears, capture is disabled.
- **Selective Capture**: Click the icons representing each event type (e.g., Registry, Filesystem) to selectively enable or disable the capture of specific event classes.

Procedure for Registry Analysis and Boot Time Capture

Registry Analysis

1. Open Process Monitor and ensure the **Registry** event type is enabled.
2. Perform actions that modify the registry, such as installing or uninstalling software.
3. Filter the registry events to focus on operations like **RegSetValue** or **RegQueryValue**, which show modifications to registry entries.

Boot Time Logging

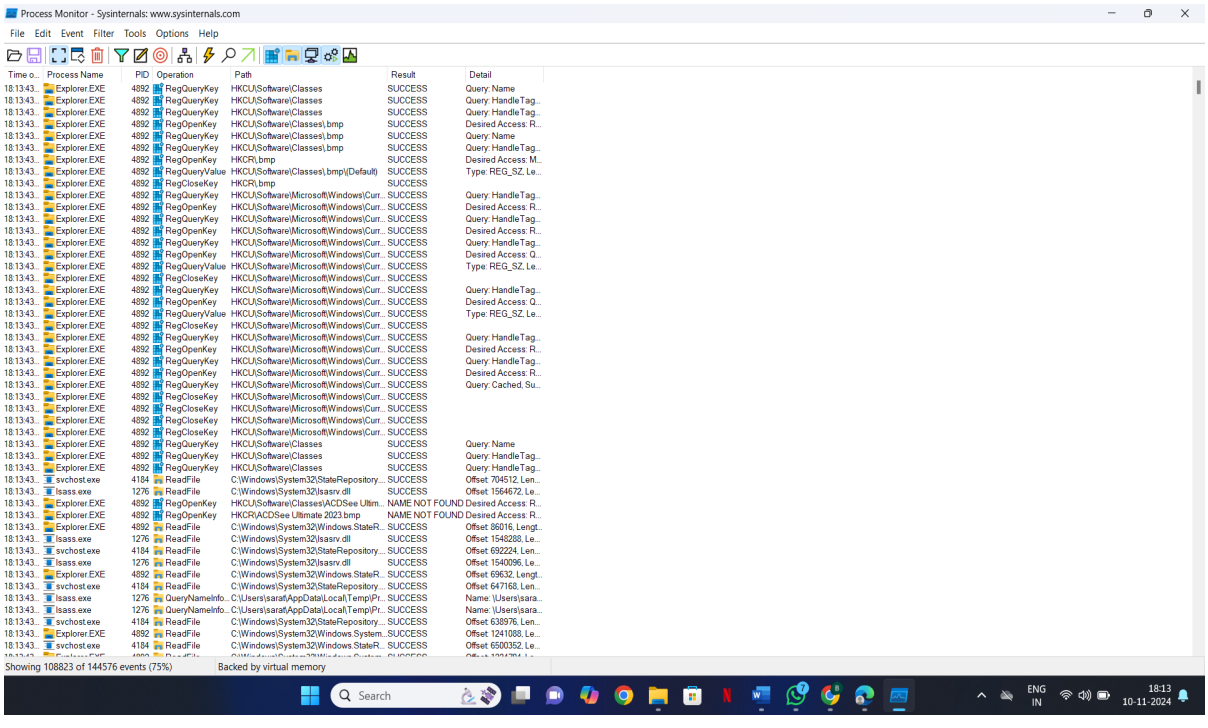
1. To capture boot-time activity, enable boot logging by going to **Options > Enable Boot Logging** in the toolbar.
2. Restart the system to allow Process Monitor to capture the boot process.
3. After rebooting, open Process Monitor and review the boot log to observe the sequence of system events and timestamps, which can highlight performance bottlenecks or system issues during startup.

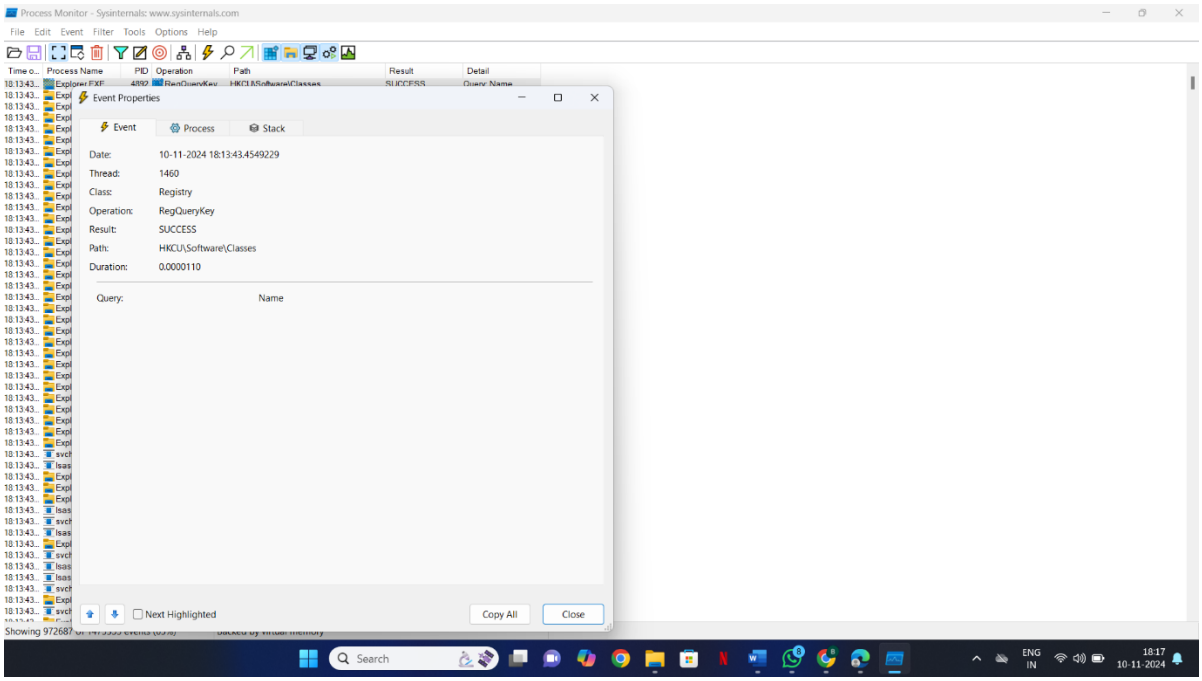
Key Observations

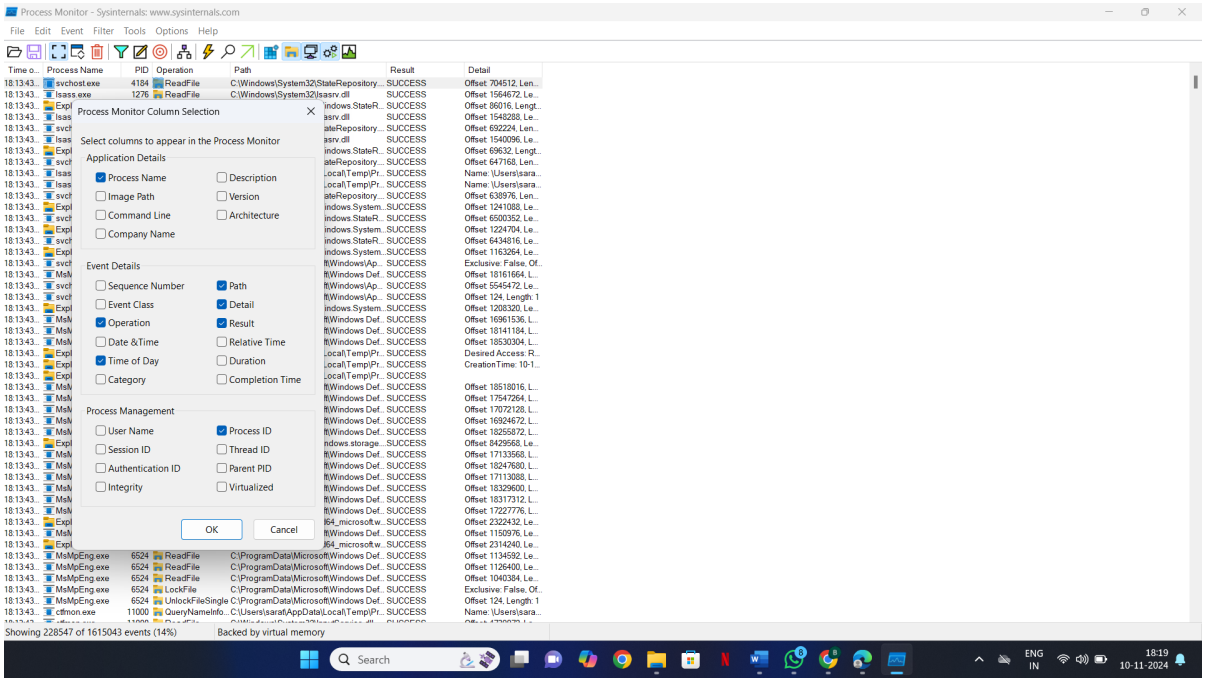
- Track the types and frequency of registry operations during normal system activity and boot time.
- Review captured events to identify critical timestamps and patterns in boot operations that can provide insights into system performance, delays, or potential issues.

Conclusion

Process Monitor provides detailed real-time analysis of registry operations and boot-time events, making it a crucial tool for system diagnostics. It enables in-depth troubleshooting, helps detect malware, and offers valuable insights into system behavior and performance.




Process Monitor - Sysinternals: www.sysinternals.com
File Edit Event Filter Tools Options Help
Event Properties
Event
Process
Stack
Date: 10-11-2024 18:13:43.4549229
Thread: 1460
Class: Registry
Operation: RegQueryKey
Result: SUCCESS
Path: HKCU\Software\Classes
Duration: 0.0000110
Query: Name
Next Highlighted
Copy All
Close
Search
ENG IN
18:17 10-11-2024


Process Monitor - Sysinternals: www.sysinternals.com
File Edit Event Filter Tools Options Help
Process Monitor Column Selection
Select columns to appear in the Process Monitor
Application Details
Process Name
Description
Image Path
Version
Command Line
Architecture
Company Name
Event Details
Sequence Number
Path
Event Class
Detail
Operation
Result
Date &Time
Relative Time
Time of Day
Duration
Category
Completion Time
Process Management
User Name
Process ID
Session ID
Thread ID
Authentication ID
Parent PID
Integrity
Virtualized
OK
Cancel
Showing 228547 of 1615043 events (14%)
Backed by virtual memory
Search
ENG IN
18:19 10-11-2024

9. Perform File type detection using Autopsy tool

Autopsy is an open-source digital forensics tool that offers powerful capabilities for analyzing and identifying files, including the detection of file types. By examining file headers and extensions, Autopsy can reveal renamed or hidden files that may otherwise go unnoticed. This functionality is essential for forensic investigations, allowing investigators to quickly flag files that need further scrutiny.

Starting a New Forensic Case

1. Open the **Autopsy** application.
2. Click **New Case** to begin a new investigation.

Configuring the Case

- In the **New Case Information** window, fill in the following:
    - **Case Name**: Provide a unique name for the case.
    - **Base Directory**: Specify the directory where the case data will be stored.
    - Optionally, you can enter additional details such as **Case Number** and **Examiner Information** (name, phone, email, organization).

Adding Data Sources for File Type Detection

1. After setting up the case, the **Add Data Source** window will appear.
2. **Select Data Source Type**: Choose the type of data source (e.g., disk image or local directory).
3. **Select Data Source**: Browse to the location of the dataset or storage device to be analyzed and add it to the case.
4. Optionally, configure **Digest** to enable data integrity verification using hashing.
5. Click **Next** to load the selected data source into Autopsy.

File Type Detection Workflow
Running the File Type Analysis

1. After the data source is loaded, Autopsy will begin an initial analysis to automatically detect and categorize file types by examining file headers and extensions.
2. To view the detected file types:
    - Navigate to **Results** and expand the **File Types** section.
    - Files will be sorted into categories like **images**, **documents**, **executables**, and **archives**.

Reviewing Detected File Types

1. Browse through the file categories to locate any files of interest, such as hidden or renamed **executables** or **images**.
2. Select individual files to inspect detailed properties such as:
    - **MIME type**
    - **File extension**
    - **File path**
    - **Hash value**

Identifying Potential Anomalies

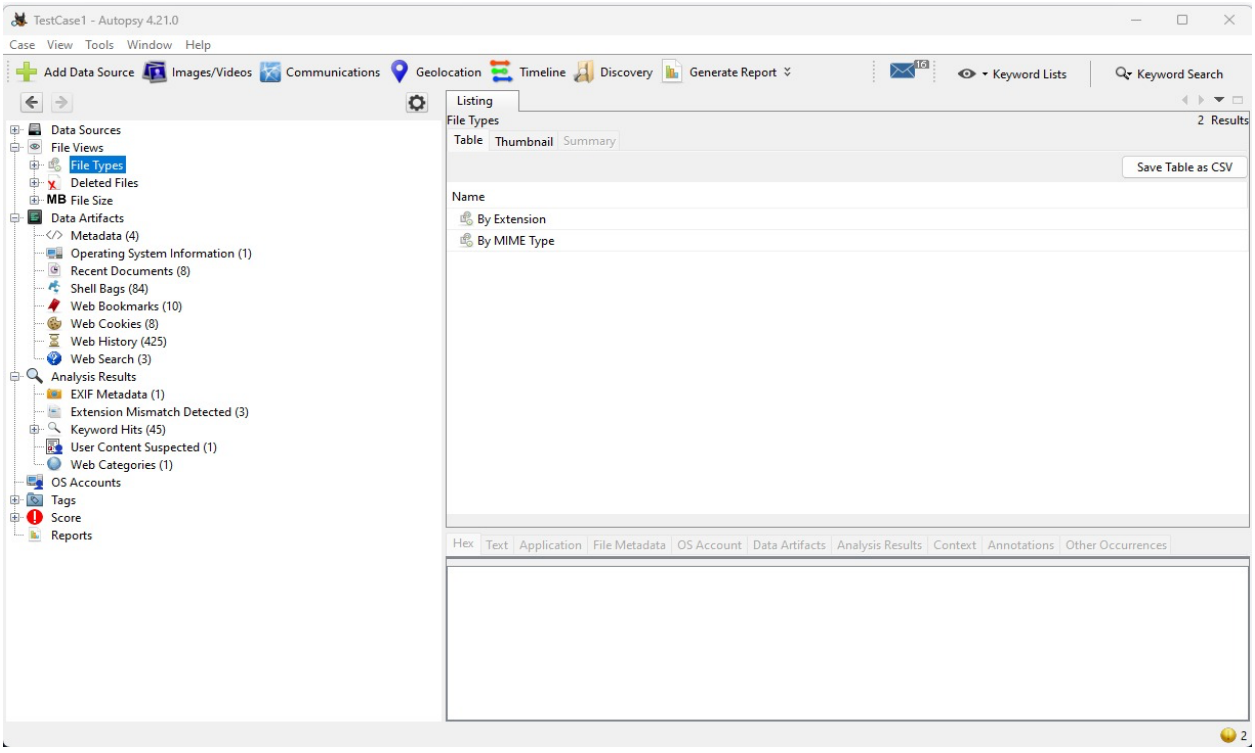
- Pay attention to files where there is a mismatch between the file extension and the MIME type. This could indicate that a file has been intentionally renamed or disguised.
- Flag files with unusual extensions or those that seem out of place in their directory for further investigation.

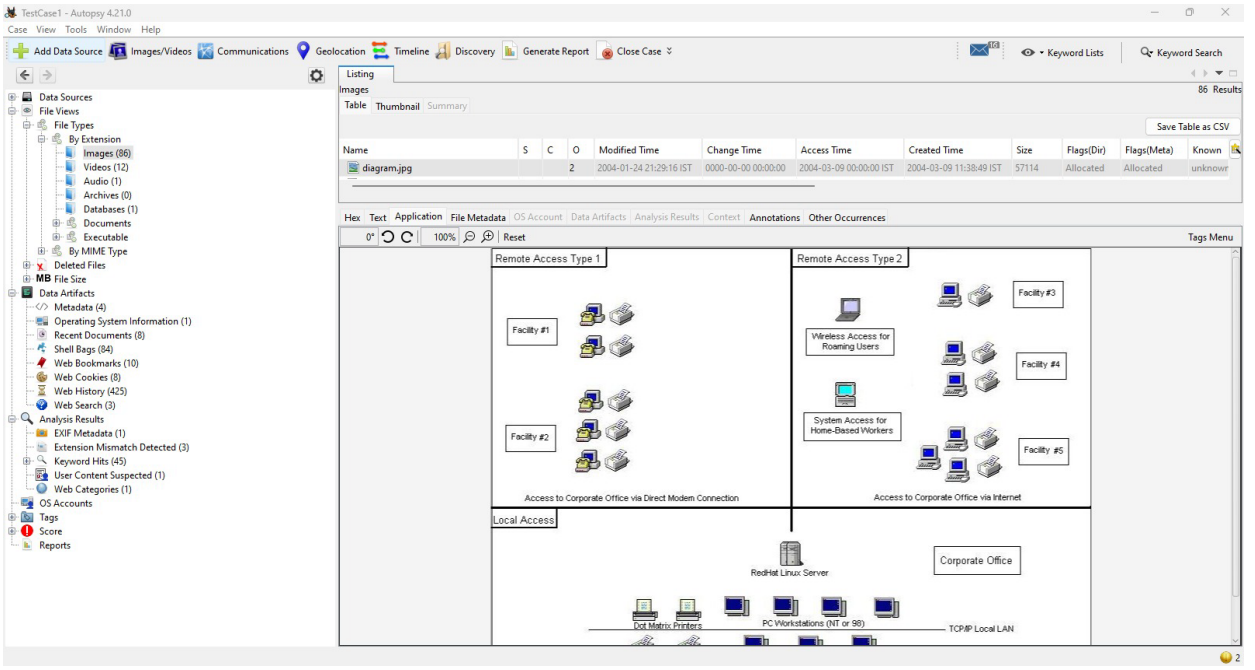
Key Observations

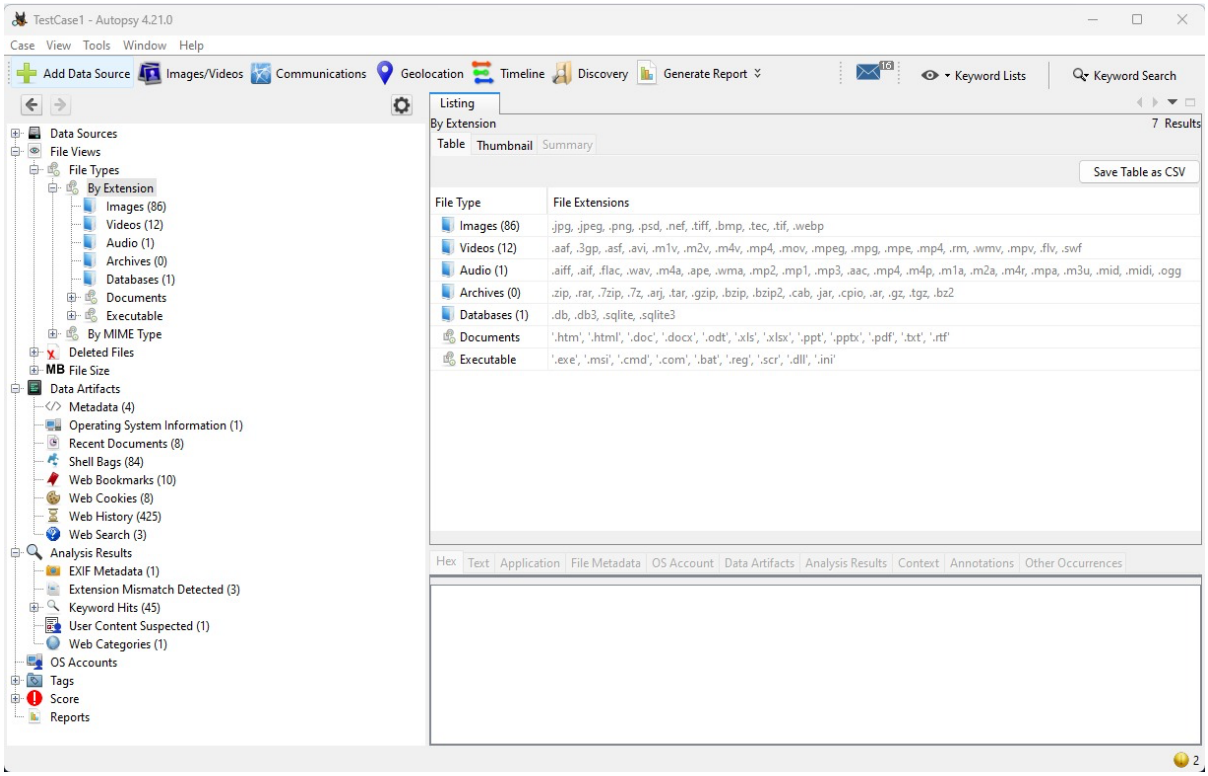
- Track instances where files have mismatched extensions and MIME types, as these may signal attempts at obfuscation or concealment.
- Hidden executables or files located in unexpected directories are strong indicators of potentially malicious activity or data manipulation.

Conclusion

Autopsy's file type detection capabilities provide forensic investigators with a reliable method to categorize and identify suspicious or hidden files. This functionality is vital for uncovering files that could play a significant role in an investigation, aiding in the efficient identification of evidence in forensic cases.






TestCase1 - Autopsy 4.21.0
Case View Tools Window Help
Add Data Source
Images/Videos
Communications
Geolocation
Timeline
Discovery
Generate Report
Keyword Lists
Keyword Search
Data Sources
File Views
File Types
By Extension
Images (86)
Videos (12)
Audio (1)
Archives (0)
Databases (1)
Documents
Executable
By MIME Type
Deleted Files
File Size
Data Artifacts
Metadata (4)
Operating System Information (1)
Recent Documents (8)
Shell Bags (84)
Web Bookmarks (10)
Web Cookies (8)
Web History (425)
Web Search (3)
Analysis Results
EXIF Metadata (1)
Extension Mismatch Detected (3)
Keyword Hits (45)
User Content Suspected (1)
Web Categories (1)
OS Accounts
Tags
Score
Reports
Listing
By Extension
7 Results
Table
Thumbnail
Summary
Save Table as CSV
File Type
File Extensions
Images (86)
.jpg, .jpeg, .png, .psd, .nef, .tiff, .bmp, .tec, .tif, .webp
Videos (12)
.aaf, .3gp, .asf, .avi, .m1v, .m2v, .m4v, .mp4, .mov, .mpeg, .mpg, .mpe, .mp4, .rm, .wmv, .mpv, .flv, .swf
Audio (1)
.aiff, .aif, .flac, .wav, .m4a, .ape, .wma, .mp2, .mp1, .mp3, .aac, .mp4, .m4p, .m1a, .m2a, .m4r, .mpa, .m3u, .mid, .midi, .ogg
Archives (0)
.zip, .rar, .7zip, .7z, .arj, .tar, .gzip, .bzip, .bzip2, .cab, .jar, .cpio, .ar, .gz, .tgz, .bz2
Databases (1)
.db, .db3, .sqlite, .sqlite3
Documents
'.htm', '.html', '.doc', '.docx', '.odt', '.xls', '.xlsx', '.ppt', '.pptx', '.pdf', '.txt', '.rtf'
Executable
'.exe', '.msi', '.cmd', '.com', '.bat', '.reg', '.scr', '.dll', '.ini'
Hex
Text
Application
File Metadata
OS Account
Data Artifacts
Analysis Results
Context
Annotations
Other Occurrences
2

10. Write a python program to convert lower case to upper case .

Server:

```
import socket
def start_server():
    # Create a socket object
    server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

    # Bind the socket to localhost and a port
    server_socket.bind(('localhost', 12345))
    server_socket.listen(1)
    print("Server is listening on port 12345...")

    # Accept a connection
    client_socket, addr = server_socket.accept()
    print("Connected by", addr)

    # Receive the modified message from the client
    message = client_socket.recv(1024).decode()
    message=message.upper()
    print("Capitalizied Message:", message)

    # Close the connection
    client_socket.close()

# Run the server
start_server()
```

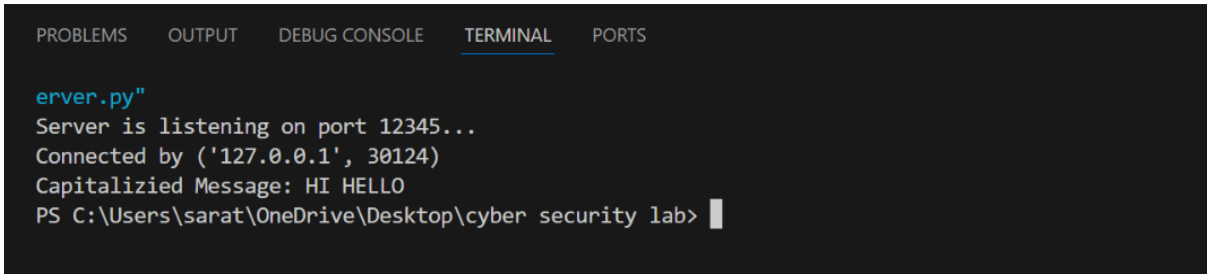
Client:

```
import socket
def send_message():
    #message
    message = "hi hello"

    # Create a socket object
    client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

    # Connect to the server
    client_socket.connect(('localhost', 12345))

    # Send the modified message
    client_socket.send(message.encode())
    print("Sent Message:",message)

    # Close the connection
    client_socket.close()
```

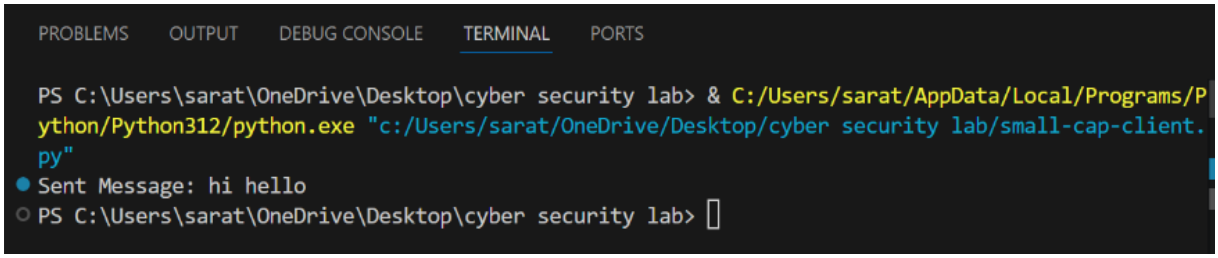
# Run the client
send_message()

Output:

Server:



Client:



11. Write a program to implement MD5 Algorithm.

Server:

```
import socket
import hashlib
def start_server():
    # Create a socket object
    server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

    # Bind the socket to localhost and a port
    server_socket.bind(('localhost', 12345))
    server_socket.listen(1)
    print("Server is listening on port 12345...")

    # Accept a connection
    client_socket, addr = server_socket.accept()
    print("Connected by", addr)

    # Receive the modified message from the client
    message = client_socket.recv(1024).decode()
    res=hashlib.md5(message.encode())
    print(f"MD5 hash: {res.hexdigest()}")
    # Close the connection
    client_socket.close()

# Run the server
start_server()
```

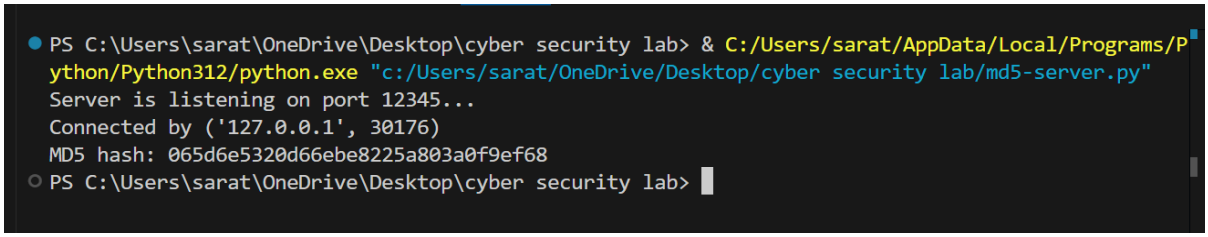
Client:

```
import socket
def send_message():

    #message
    message = "hi hello"

    # Create a socket object
    client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

    # Connect to the server
    client_socket.connect(('localhost', 12345))

    # Send the modified message
    client_socket.send(message.encode())
    print("Sent Message:",message)

    # Close the connection
    client_socket.close()
```

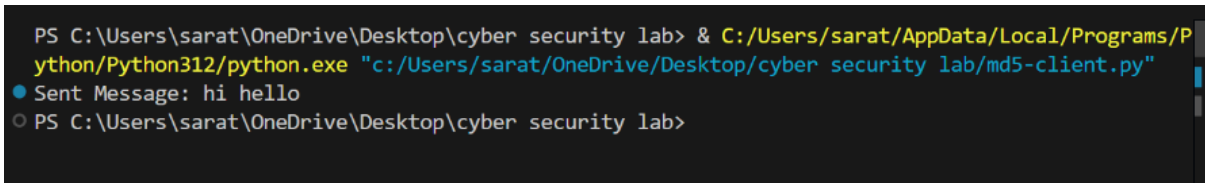
# Run the client
send_message()

Output:

Server



Client



12. Write a program to implement Caesar Cipher.

Server:

```
import socket

def caesar_decipher(text, shift):
    result = ""
    for char in text:
        if char.islower():
            result += chr((ord(char) - shift - 97) % 26 + 97)
        else:
            result += char
    return result

def start_server():
    server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    server_socket.bind(('localhost', 65432))  # Bind to all interfaces
    server_socket.listen(1)
    print("Server is listening on port 65432...")

    conn, addr = server_socket.accept()
    print(f"Connected by {addr}")

    while True:
        data = conn.recv(1024)
        if not data:
            break
        encrypted_message, shift = data.decode().split(',')
        shift = int(shift)
        original_message = caesar_decipher(encrypted_message, shift)
        print(f"Original message from client: {original_message}")

    conn.close()

if __name__ == "__main__":
    start_server()
```

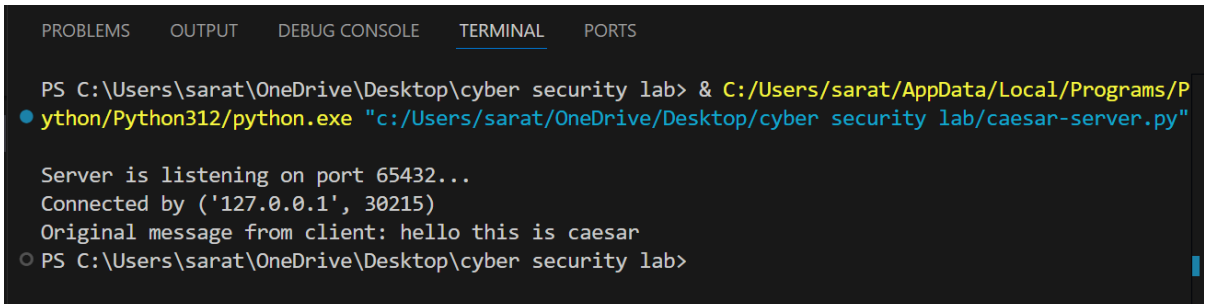
Client:

```
import socket

def caesar_cipher(text, shift):
    result = ""
    for char in text:
        if char.islower():
            result += chr((ord(char) + shift - 97) % 26 + 97)
        else:
```

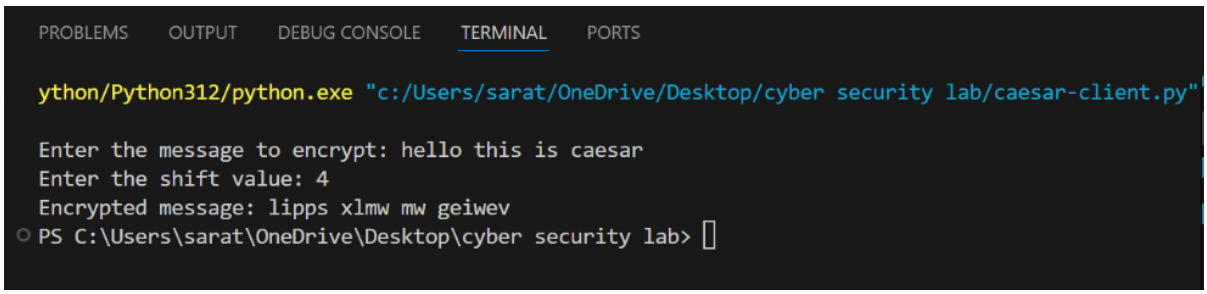
```
            result += char
    return result

def send_message(message, shift):
    client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    client_socket.connect(("localhost", 65432))

    encrypted_message = caesar_cipher(message, shift)
    print(f"Encrypted message: {encrypted_message}")

    data = f"{encrypted_message},{shift}"
    client_socket.sendall(data.encode())

    client_socket.close()

if __name__ == "__main__":
    message = input("Enter the message to encrypt: ")
    shift = int(input("Enter the shift value: "))
    send_message(message, shift)
```

Output:

Server



Client



13. Write a program to implement DES Algorithm.

Server:

```
import socket
from Crypto.Cipher import DES
from Crypto.Util.Padding import unpad

# Key must be 8 bytes
DES_KEY = b'8bytekey'

def des_decrypt(ciphertext, key):
    cipher = DES.new(key, DES.MODE_ECB)
    decrypted_data = unpad(cipher.decrypt(ciphertext), DES.block_size)
    return decrypted_data.decode('utf-8')

def start_server():
    server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    server_socket.bind(('localhost', 65432))
    server_socket.listen(1)
    print("Server is listening on port 65432...")

    conn, addr = server_socket.accept()
    print(f"Connected by {addr}")

    while True:
        data = conn.recv(1024)
        if not data:
            break
        decrypted_message = des_decrypt(data, DES_KEY)
        print(f"Original message from client: {decrypted_message}")

    conn.close()

if __name__ == "__main__":
    start_server()
```

Client:

```
import socket
from Crypto.Cipher import DES
from Crypto.Util.Padding import pad

# Key must be 8 bytes
DES_KEY = b'8bytekey'
```

```
def des_encrypt(text, key):
    cipher = DES.new(key, DES.MODE_ECB)
    padded_text = pad(text.encode('utf-8'), DES.block_size)
    encrypted_data = cipher.encrypt(padded_text)
    return encrypted_data

def send_message(message):
    client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    client_socket.connect(("localhost", 65432))

    encrypted_message = des_encrypt(message, DES_KEY)
    print(f"Encrypted message (in bytes): {encrypted_message}")

    client_socket.sendall(encrypted_message)

    client_socket.close()

if __name__ == "__main__":
    message = input("Enter the message to encrypt: ")
    send_message(message)
```

Output:

Server

```
PS C:\Users\sarat\OneDrive\Desktop\cyber security lab> & C:/Users/sarat/AppData/Local/Programs/Python/Python312/python.exe "c:/Users/sarat/OneDrive/Desktop/cyber security lab/des-server.py"
Server is listening on port 65432...
Connected by ('127.0.0.1', 30269)
Original message from client: this is des
PS C:\Users\sarat\OneDrive\Desktop\cyber security lab>
```

Client

```
PS C:\Users\sarat\OneDrive\Desktop\cyber security lab> & C:/Users/sarat/AppData/Local/Programs/Python/Python312/python.exe "c:/Users/sarat/OneDrive/Desktop/cyber security lab/des-client.py"
Enter the message to encrypt: this is des
Encrypted message (in bytes): b'd\x1bj\xcd\x8e\x00\x06\xbe\x15m\xe4 \xdc\x03\x7f\x8a'
PS C:\Users\sarat\OneDrive\Desktop\cyber security lab>
```

14. Write a program to implement Brute Force algorithm

```
import itertools
import string
import random

def brute_force_attack(target_password, max_length):
    characters = {"0","1"}
    attempts = 0

    for password_length in range(1, max_length + 1):
        for guess in itertools.product(characters, repeat=password_length):
            attempts += 1
            guess_password = ''.join(guess)
            if guess_password == target_password:
                print(f"Password found: {guess_password} in {attempts} attempts")
                return guess_password
    print("Password not found")
    return None
def rand(p):
    key1=""
    for i in range(p):
        temp = str(random.randint(0, 1))
        key1 += temp
    return key1
```

Output:

```
PS C:\Users\sarat\OneDrive\Desktop\cyber security lab> & C:/Users/sarat/AppData/Local/Programs/P
ython/Python312/python.exe "c:/Users/sarat/OneDrive/Desktop/cyber security lab/bruteforce.py"
target password is : 0011010001
Password found: 0011010001 in 1837 attempts
PS C:\Users\sarat\OneDrive\Desktop\cyber security lab>
```

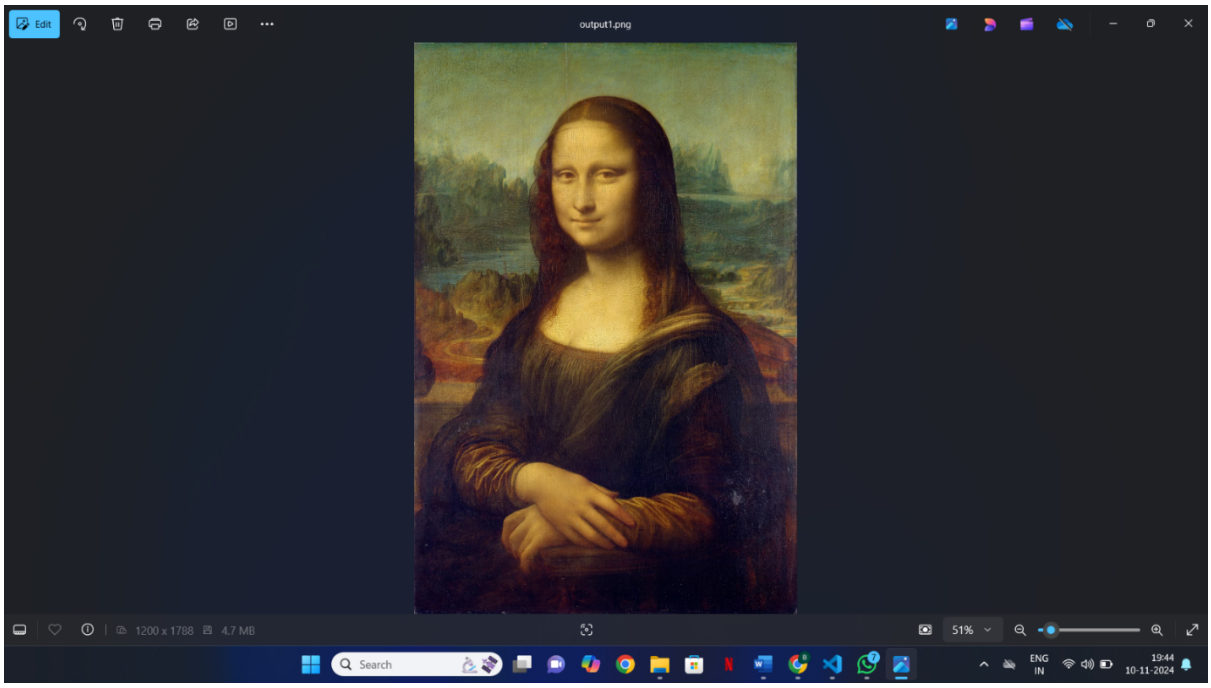
15. Write program to implement Image-Steganography.

Modifying LSB bits:

```
from PIL import Image
import numpy as np

def load_image(image_path):
    image = Image.open(image_path)
    return np.array(image)

def encode_message(image, message):
    binary_message = ''.join(format(ord(char), '08b') for char in message)
    binary_message += '1111111111111110'  # Delimiter to mark the end of the message

    data_index = 0
    for values in image:
        for pixel in values:
            for color in range(3):  # Iterate through RGB values
                if data_index < len(binary_message):
                    pixel[color] = int(bin(pixel[color])[2:9] + binary_message[data_index], 2)
                    data_index += 1

    return image

def save_image(image, output_path):
    encoded_image = Image.fromarray(image)
    encoded_image.save(output_path)

def decode_message(encoded_image):
    binary_data = ""
    for values in encoded_image:
        for pixel in values:
            for color in range(3):  # Iterate through RGB values
                binary_data += bin(pixel[color])[2:][-1]  # Extract the least significant bit

    # Split by 8-bits
    all_bytes = [binary_data[i: i+8] for i in range(0, len(binary_data), 8)]

    decoded_message = ""
    for byte in all_bytes:
        if byte == '11111111':
            break
        decoded_message += chr(int(byte, 2))

    return decoded_message
```

```
# Example usage:
image_path = 'black.png'
output_path = 'output1.png'
message = "hello this is a secret"

# Load the image
image = load_image(image_path)

# Encode the message into the image
encoded_image = encode_message(image, message)

# Save the encoded image
save_image(encoded_image, output_path)

# Load the encoded image
encoded_image = load_image(output_path)

# Decode the message from the image
decoded_message = decode_message(encoded_image)
print("Decoded message:", decoded_message)
```

Output:

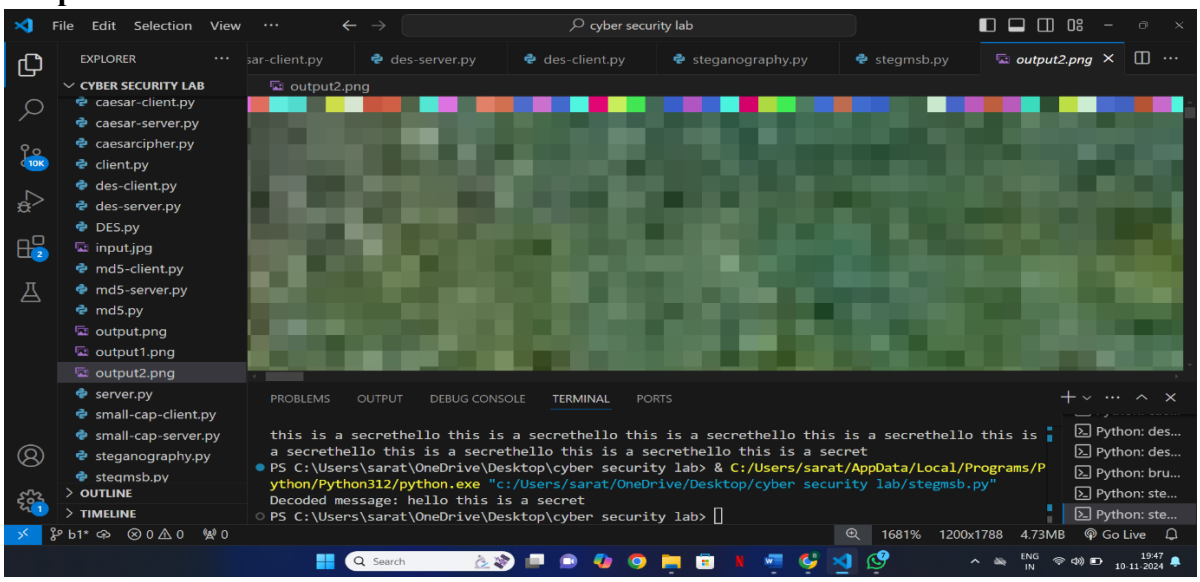


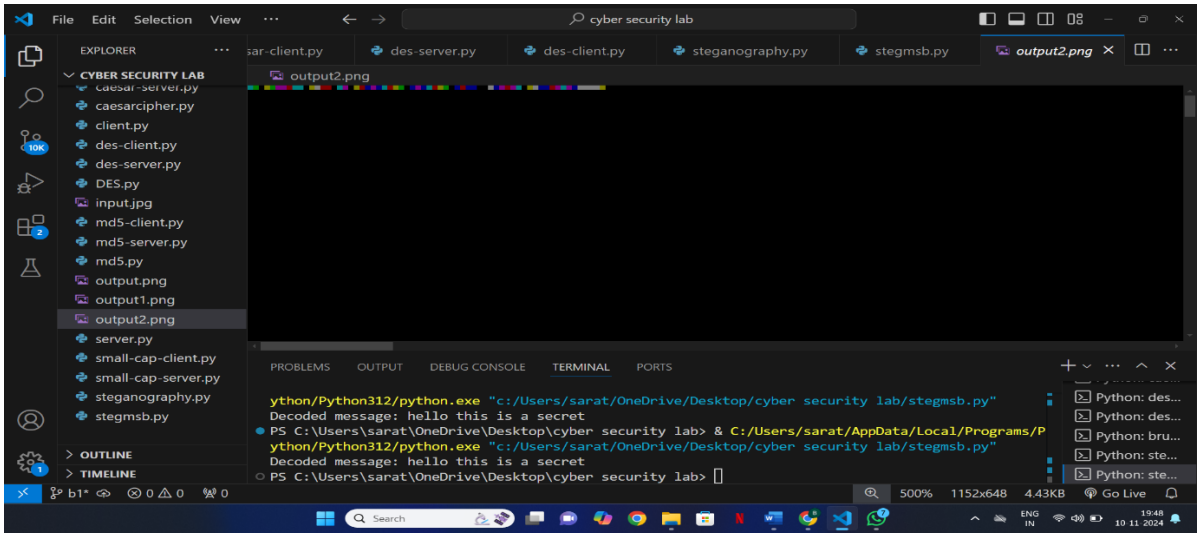
Modifying MSB Bits:

```
from PIL import Image
import numpy as np

def load_image(image_path):
    image = Image.open(image_path)
    return np.array(image)

def encode_message(image, message):
    binary_message = ''.join(format(ord(char), '08b') for char in message)
    binary_message += '1111111111111110'  # Delimiter to mark the end of the message

    data_index = 0
    for values in image:
        for pixel in values:
            for color in range(3):  # Iterate through RGB values
                if data_index < len(binary_message):
                    # Get the current pixel color value in binary form, ensuring it's 8 bits long
                    pixel_binary = bin(pixel[color])[2:].zfill(8)
                    # Replace the most significant bit with the current bit of the message
                    modified_binary = binary_message[data_index] + pixel_binary[1:]
                    # Convert the modified binary value back to an integer
                    pixel[color] = int(modified_binary, 2)
                    data_index += 1

    return image

def save_image(image, output_path):
    encoded_image = Image.fromarray(image)
    encoded_image.save(output_path)

def decode_message(encoded_image):
    binary_data = ""
    for values in encoded_image:
        for pixel in values:
            for color in range(3):  # Iterate through RGB values
                # Extract the most significant bit
                binary_data += bin(pixel[color])[2:].zfill(8)[0]

    # Split by 8-bits
    all_bytes = [binary_data[i: i+8] for i in range(0, len(binary_data), 8)]

    decoded_message = ""
    for byte in all_bytes:
        if byte == '11111111':
            break
        decoded_message += chr(int(byte, 2))
```

```
    return decoded_message

# Example usage:
image_path = 'black.png'
output_path = 'output2.png'
message = "hello this is a secret"

# Load the image
image = load_image(image_path)

# Encode the message into the image
encoded_image = encode_message(image, message)

# Save the encoded image
save_image(encoded_image, output_path)

# Load the encoded image
encoded_image = load_image(output_path)

# Decode the message from the image
decoded_message = decode_message(encoded_image)
print("Decoded message:", decoded_message)
```

Output:





16. Write a program to implement psn ratio of images.

```
import cv2
import numpy as np

# Load and convert images to grayscale, resizing imageB if needed
imageA = cv2.imread(r'C:\Users\sarat\OneDrive\Desktop\cyber security lab\one.jpeg', cv2.IMREAD_GRAYSCALE)
imageB = cv2.imread(r'C:\Users\sarat\OneDrive\Desktop\cyber security lab\two.jpeg', cv2.IMREAD_GRAYSCALE)

if imageA is not None and imageB is not None:
    imageB = cv2.resize(imageB, (imageA.shape[1], imageA.shape[0]))
    mse = np.mean((imageA - imageB) ** 2)
    psnr = 20 * np.log10(255 / np.sqrt(mse)) if mse != 0 else float('inf')
    print("PSNR:", psnr, "dB")
else:
    print("Error: Could not load images.")
```

Output:

```
PS C:\Users\sarat\OneDrive\Desktop\cyber security lab> & C:/Users/sarat/AppData/Local/Programs/Python/Python312/python.exe "c:/Users/sarat/OneDrive/Desktop/cyber security lab/psn.py"
PSNR: 27.887656098072867 dB
PS C:\Users\sarat\OneDrive\Desktop\cyber security lab>
```