Lab #1: Linux File Permissions – Department-Based Access Control

🎯 Objective

Set up departmental directories for a fictional company, assign users and groups accordingly, appropriately manage file permissions, and elevate the C-Level department's privileges to ensure they have full access across all folders using both traditional UNIX permissions and ACLs.

Scenario: Company Directory Setup for a Fictional Business

Company Name: CyberCorp

CyberCorp has **5 departments and 15 employees. Each department has 3 staff members**. Only the **C-Level department** should have full access to all files across all departments.

Departments & Employees

Department	Group Name	Staff Members
Marketing	marketing	Alice, Bob, Carol
HR	hr	David, Emily, Frank
IT	it	Grace, Henry, Ian
Sales	sales	Jack, Kelly, Liam
C-Level	clevel	CEO, CTO, CFO

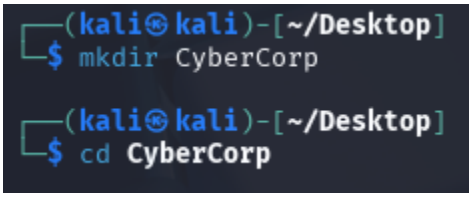
Step-by-Step Instructions / Summary

1. Create Base Directory on Desktop
2. Create Department Subdirectories
    a. Create Groups for Departments
    b. Create Users and Assign to Groups
    c. Check Group Assignments
3. Set Group Ownership of Directories
4. Set Directory Permissions (770)
5. Create 3 Files Per Department

    a. Marketing
    b. HR
    c. IT
    d. Sales
    e. C-Level
6. Add C-Level Group Access via ACL
    a. Verify C-Level Permissions with `getfacl`
7. Generate Directory Tree View with `tree`
8. `Screenshot submissions`

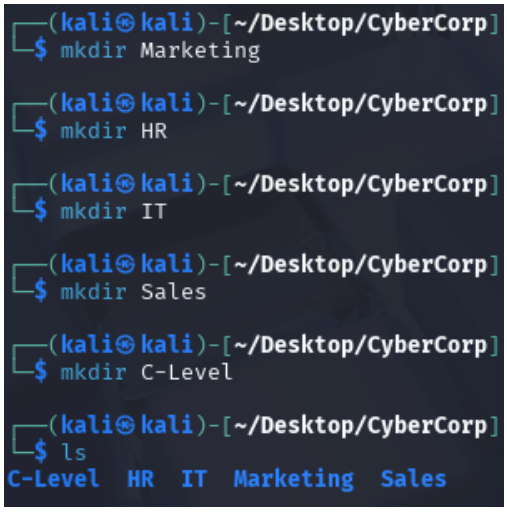
Commands and steps used for this lab:

1. Create base directory on desktop

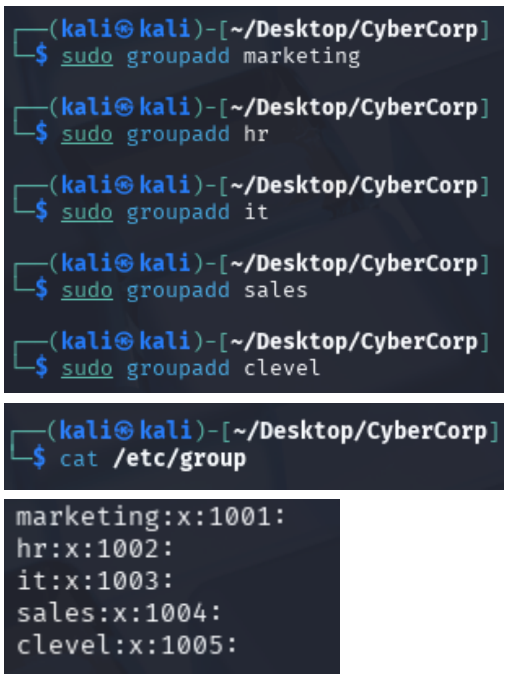
Making the base folder **mkdir CyberCorp**	

2. Create department subdirectories and groups

2.1. Making the subdirectories inside the CyberCorps folder

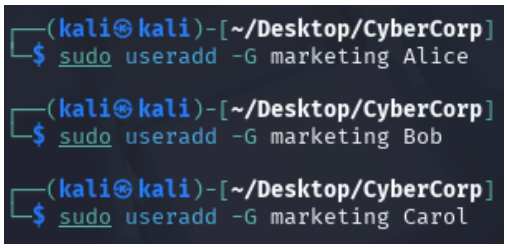
mkdir Marketing **mkdir HR** **mkdir IT** **mkdir Sales** **mkdir C-Level** **ls**	

2.2. Making the required groups for the collection of users later on

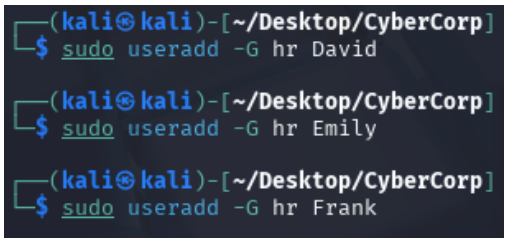
sudo groupadd marketing **sudo groupadd hr** **sudo groupadd it** **sudo groupadd sales** **sudo groupadd clevel** **cat /etc/group**	

3. **Create users and assign to groups**

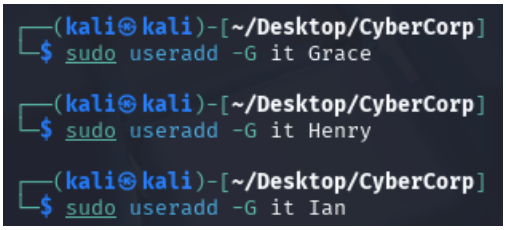
3.1. **Creating users for Marketing group**

sudo useradd -G marketing Alice **sudo useradd -G marketing Bob** **sudo useradd -G marketing Carol**	

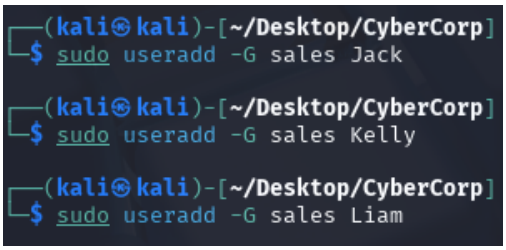
3.2. **Creating users for HR group**

sudo useradd -G hr David **sudo useradd -G hr Emily** **sudo useradd -G hr Frank**	

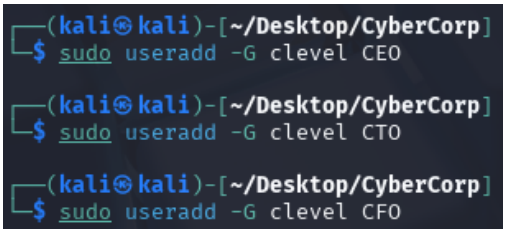
3.3. **Creating users for IT group**

sudo useradd -G it Grace **sudo useradd -G it Henry** **sudo useradd -G it Ian**	

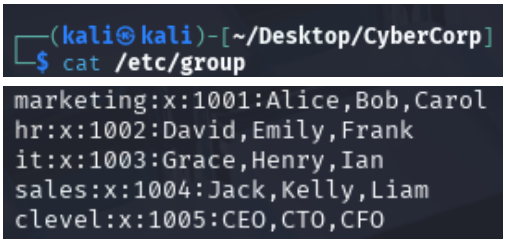
3.4. Creating users for Sales group

sudo useradd -G sales Jack **sudo useradd -G sales Kelly** **sudo useradd -G sales Liam**	

3.5. Creating users for C-Level group

sudo useradd -G clevel CEO **sudo useradd -G clevel CTO** **sudo useradd -G clevel CFO**	

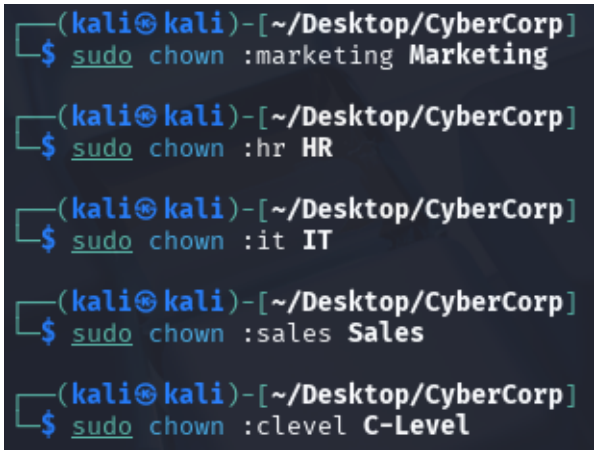
3.6. Check if all the created users assigned to the associated group

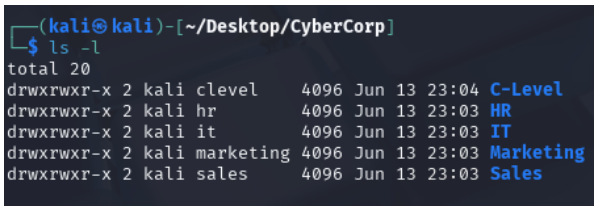
cat /etc/group	

4. Set group ownership and permissions

4.1. Setting the group ownership for each respective directory

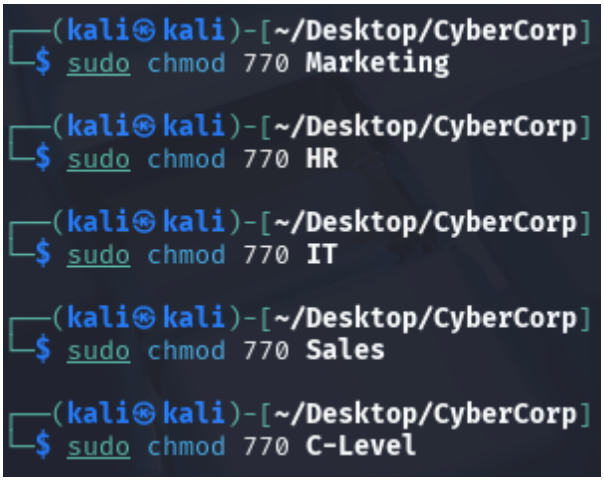
```
sudo chown :marketing Marketing
sudo chown :hr HR
sudo chown :it IT
sudo chown :sales Sales
sudo chown :clevel C-Level
ls -l
```

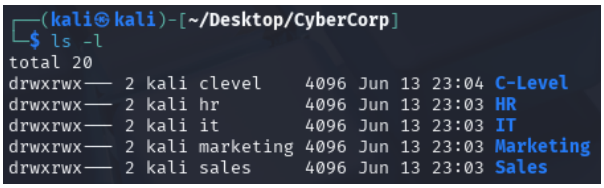




4.2. Setting the proper permissions for each group and respective directory

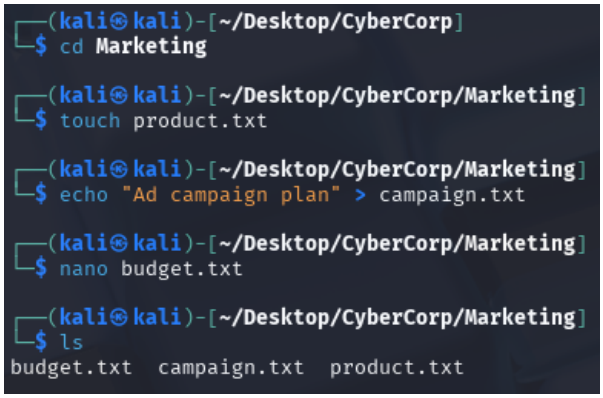
```
sudo chmod 770 Marketing
sudo chmod 770 HR
sudo chmod 770 IT
sudo chmod 770 Sales
sudo chmod 770 C-Level
ls -l
```



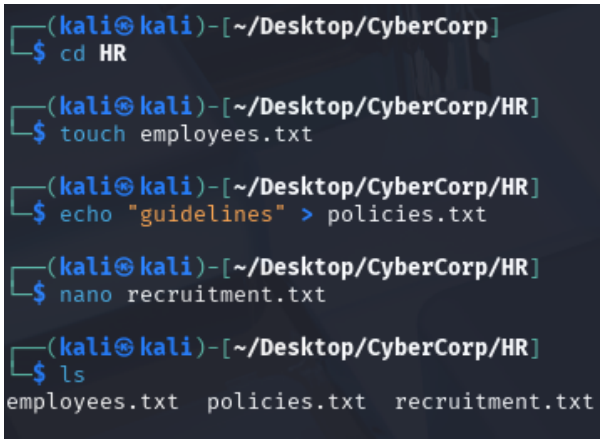


5. Create 3 files per department using different methods (Using touch, echo, nano)

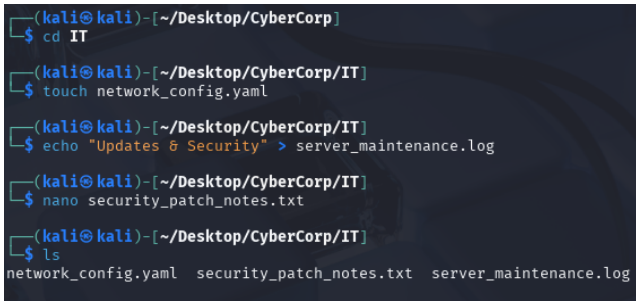
5.1. Creating the 3 files for the Marketing Directory

cd Marketing touch product.txt echo "Ad campaign plan" > campaign.txt nano budget.txt ls	

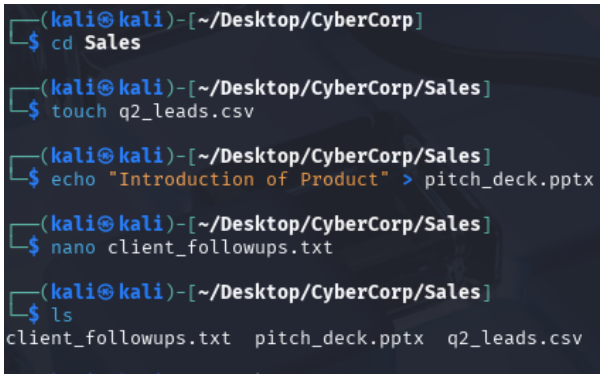
5.2. Creating the 3 files of the HR Directory

cd HR touch employees.txt echo "guidelines" > policies.txt nano recruitment.txt ls	

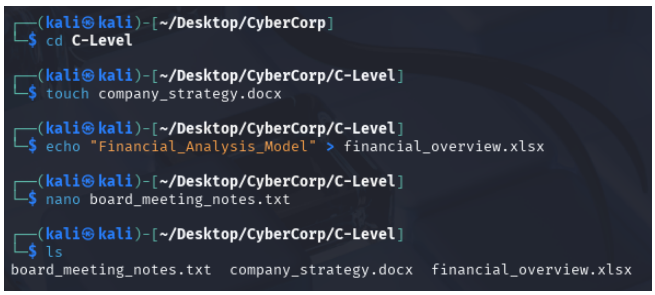
5.3. Creating the 3 files of the IT Directory

cd IT touch network_config.yaml echo "Updates & Security" > server_maintenance.log nano security_patch_notes.txt ls	

5.4. Creating the 3 files of the Sales directory

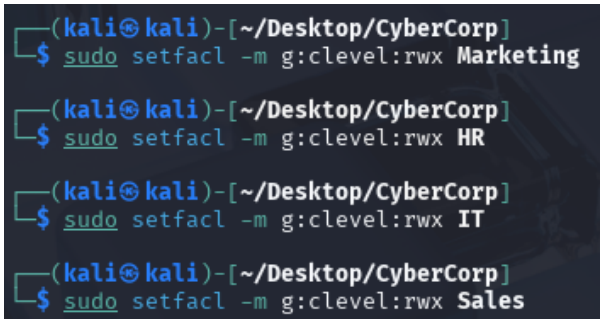
cd Sales touch q2_leads.csv echo "Introduction of Product" > pitch_deck.pptx nano client_followups.txt ls	

5.5. Creating the 3 files for C-Level

cd C-Level touch company_strategy.docx echo "Financial_Analysis_Model" > financial_overview.xlsx nano board_meeting_notes.txt ls	

6. Add C-Level group access to all folders

6.1. Setting the access for all C-Level groups

sudo setfacl -m g:clevel:rwx marketing sudo setfacl -m g:clevel:rwx hr sudo setfacl -m g:clevel:rwx it sudo setfacl -m g:clevel:rwx sales	

6.2. Verify the C-Level permissions

getfacl marketing
getfacl hr
getfacl it
getfacl sales
getfacl clevel

```
┌──(kali㉿kali)-[~/Desktop/CyberCorp]
└─$ getfacl Marketing
# file: Marketing
# owner: kali
# group: marketing
user::rwx
group::rwx
group:clevel:rwx
mask::rwx
other::---
```

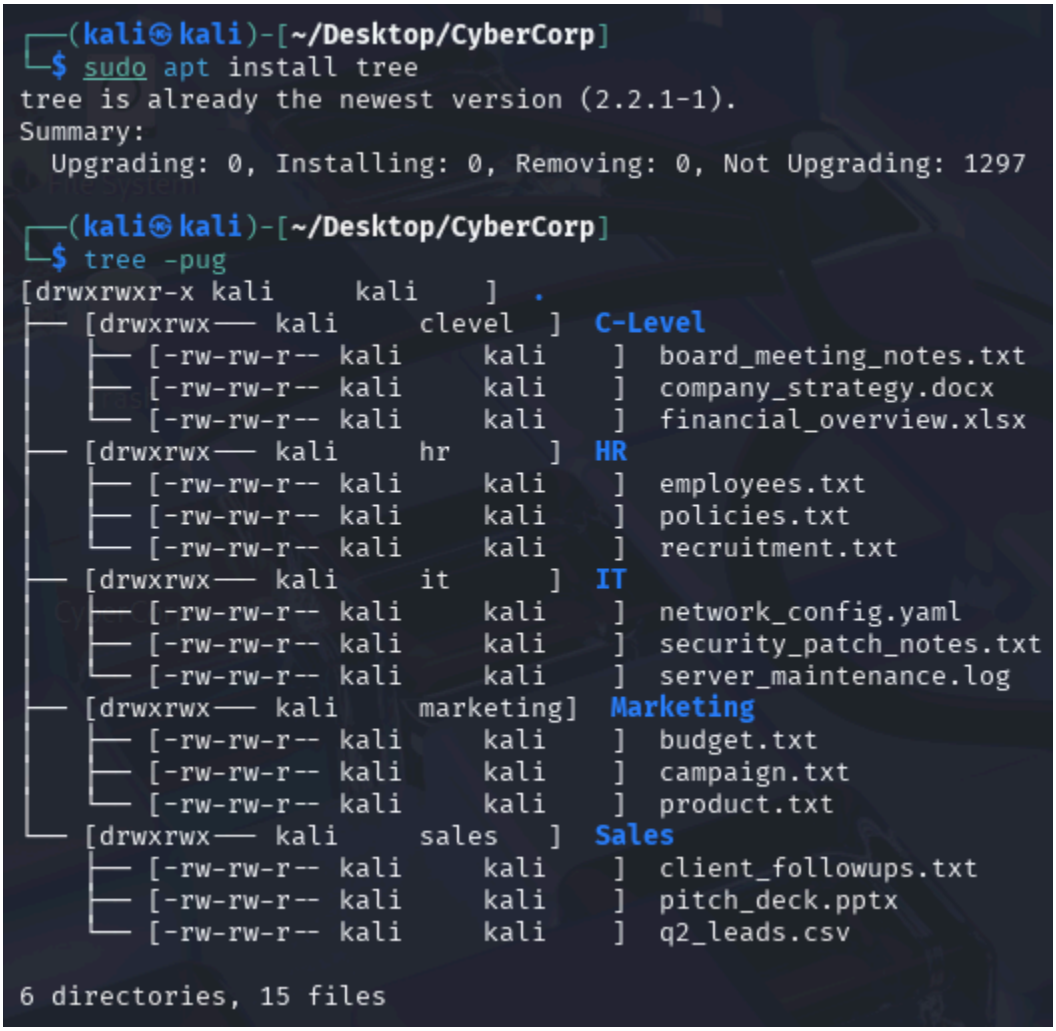
```
┌──(kali㉿kali)-[~/Desktop/CyberCorp]
└─$ getfacl HR
# file: HR
# owner: kali
# group: hr
user::rwx
group::rwx
group:clevel:rwx
mask::rwx
other::---
```

```
┌──(kali㉿kali)-[~/Desktop/CyberCorp]
└─$ getfacl IT
# file: IT
# owner: kali
# group: it
user::rwx
group::rwx
group:clevel:rwx
mask::rwx
other::---
```

```
┌──(kali㉿kali)-[~/Desktop/CyberCorp]
└─$ getfacl Sales
# file: Sales
# owner: kali
# group: sales
user::rwx
group::rwx
group:clevel:rwx
mask::rwx
other::---
```

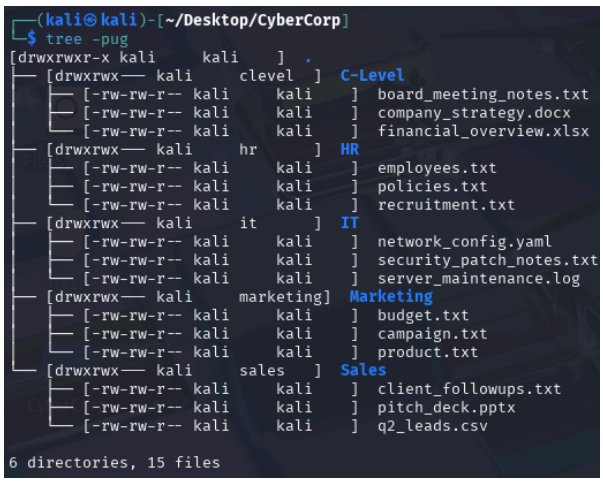
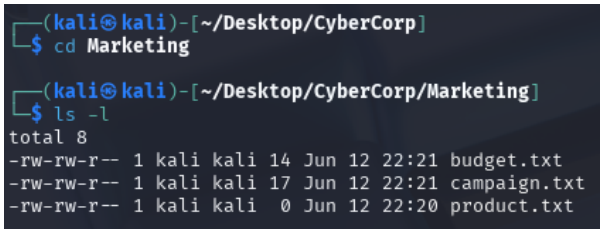
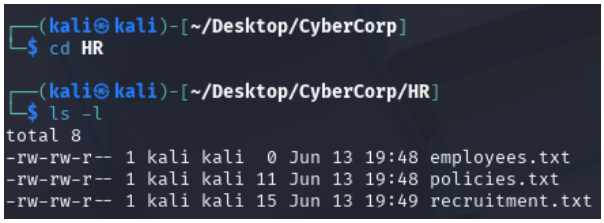
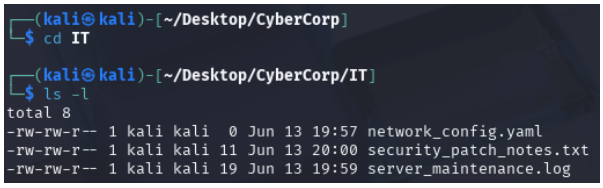
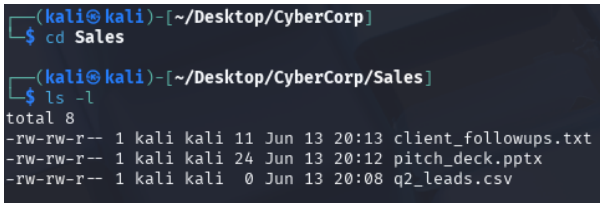
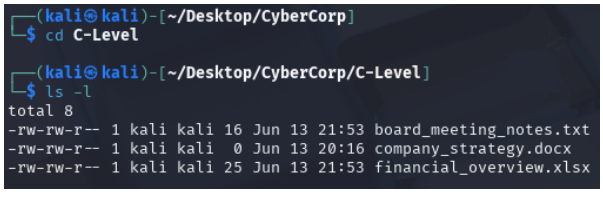
```
┌──(kali㉿kali)-[~/Desktop/CyberCorp]
└─$ getfacl C-Level
# file: C-Level
# owner: kali
# group: clevel
user::rwx
group::rwx
other::---
```

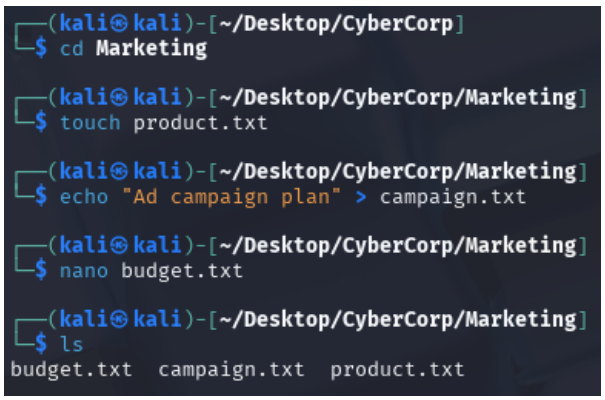
7. **Generate a directory tree view**
   - 7.1. **sudo apt install tree**
   - 7.2. **tree -pug**

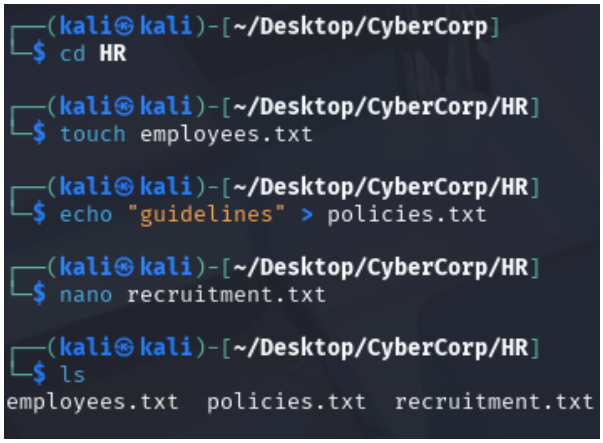


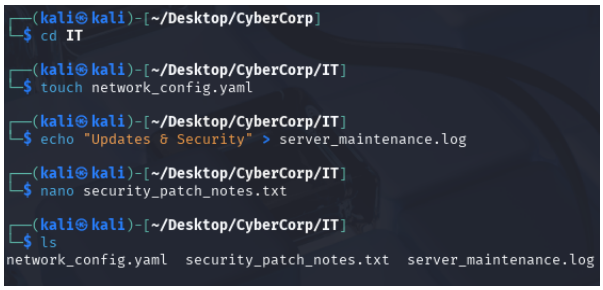
8. **Additional screenshot submissions**

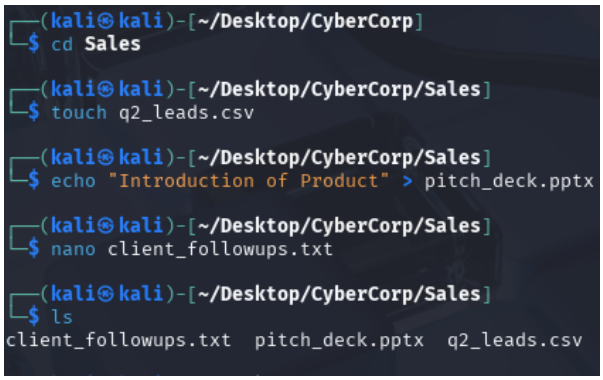
Screenshot	Description

	CyberCorp directory structure using tree
    	Each department folder with ls -l









File creation steps using touch, nano, echo

```
┌──(kali㉿kali)-[~/Desktop/CyberCorp]
└─$ cd C-Level

┌──(kali㉿kali)-[~/Desktop/CyberCorp/C-Level]
└─$ touch company_strategy.docx

┌──(kali㉿kali)-[~/Desktop/CyberCorp/C-Level]
└─$ echo "Financial_Analysis_Model" > financial_overview.xlsx

┌──(kali㉿kali)-[~/Desktop/CyberCorp/C-Level]
└─$ nano board_meeting_notes.txt

┌──(kali㉿kali)-[~/Desktop/CyberCorp/C-Level]
└─$ ls
board_meeting_notes.txt  company_strategy.docx  financial_overview.xlsx
```

Group creation

```
┌──(kali㉿kali)-[~/Desktop/CyberCorp]
└─$ sudo groupadd marketing

┌──(kali㉿kali)-[~/Desktop/CyberCorp]
└─$ sudo groupadd hr

┌──(kali㉿kali)-[~/Desktop/CyberCorp]
└─$ sudo groupadd it

┌──(kali㉿kali)-[~/Desktop/CyberCorp]
└─$ sudo groupadd sales

┌──(kali㉿kali)-[~/Desktop/CyberCorp]
└─$ sudo groupadd clevel
```

```
┌──(kali㉿kali)-[~/Desktop/CyberCorp]
└─$ cat /etc/group
```

```
marketing:x:1001:
hr:x:1002:
it:x:1003:
sales:x:1004:
clevel:x:1005:
```

User assignment

```
┌──(kali㉿kali)-[~/Desktop/CyberCorp]
└─$ sudo useradd -G marketing Alice

┌──(kali㉿kali)-[~/Desktop/CyberCorp]
└─$ sudo useradd -G marketing Bob

┌──(kali㉿kali)-[~/Desktop/CyberCorp]
└─$ sudo useradd -G marketing Carol
```

```
┌──(kali㉿kali)-[~/Desktop/CyberCorp]
└─$ sudo useradd -G hr David

┌──(kali㉿kali)-[~/Desktop/CyberCorp]
└─$ sudo useradd -G hr Emily

┌──(kali㉿kali)-[~/Desktop/CyberCorp]
└─$ sudo useradd -G hr Frank
```

Group creation and user assignment

<table>
<tr>
<td>

```
┌──(kali㉿kali)-[~/Desktop/CyberCorp]
└─$ sudo useradd -G it Grace

┌──(kali㉿kali)-[~/Desktop/CyberCorp]
└─$ sudo useradd -G it Henry

┌──(kali㉿kali)-[~/Desktop/CyberCorp]
└─$ sudo useradd -G it Ian
```

```
┌──(kali㉿kali)-[~/Desktop/CyberCorp]
└─$ sudo useradd -G sales Jack

┌──(kali㉿kali)-[~/Desktop/CyberCorp]
└─$ sudo useradd -G sales Kelly

┌──(kali㉿kali)-[~/Desktop/CyberCorp]
└─$ sudo useradd -G sales Liam
```

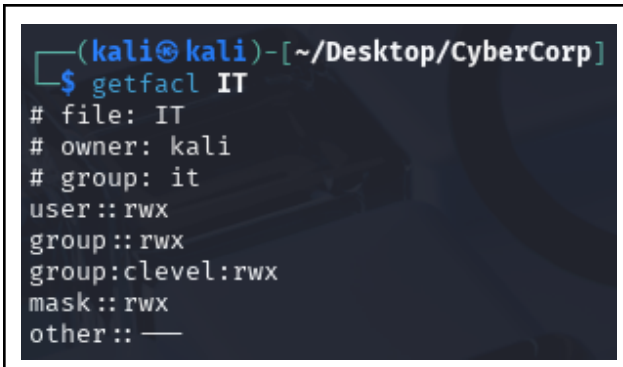
```
┌──(kali㉿kali)-[~/Desktop/CyberCorp]
└─$ sudo useradd -G clevel CEO

┌──(kali㉿kali)-[~/Desktop/CyberCorp]
└─$ sudo useradd -G clevel CTO

┌──(kali㉿kali)-[~/Desktop/CyberCorp]
└─$ sudo useradd -G clevel CFO
```

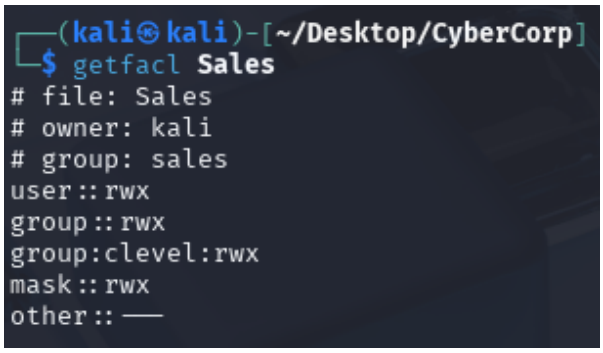
</td>
<td></td>
</tr>
<tr>
<td>

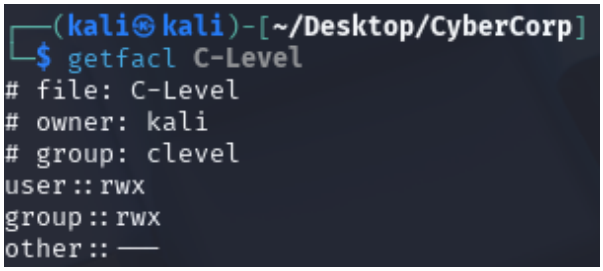
```
┌──(kali㉿kali)-[~/Desktop/CyberCorp]
└─$ getfacl Marketing
# file: Marketing
# owner: kali
# group: marketing
user::rwx
group::rwx
group:clevel:rwx
mask::rwx
other::---
```

```
┌──(kali㉿kali)-[~/Desktop/CyberCorp]
└─$ getfacl HR
# file: HR
# owner: kali
# group: hr
user::rwx
group::rwx
group:clevel:rwx
mask::rwx
other::---
```

</td>
<td>getfacl output to verify C-Level permissions</td>
</tr>
</table>







🧰 Tools & Skills Used

- **mkdir, ls, tree, touch, echo, nano**
- **useradd, groupadd, chown, chmod**
- **setfacl, getfacl**
- Linux filesystem permissions
- Group-based access control
- ACLs for extended permissions

🧠 Reflection & Takeaways

In this lab I learned how Linux file permission and group-based access control work.
I made a mistake from Step 4 where I named the base directory CyberCorps instead of CyberCorp. At the time it would've caused confusion with my structures and screenshots. I fixed it by deleting and recreating the users, groups, and directories to have proper consistency. I learned to use setfacl and getfacl to grant cross-department access for C-Level users, something I couldn't do with chmod alone.

Lab# 02 Wireshark Lab: Installation and Basic Network Scan

🎯 Objective

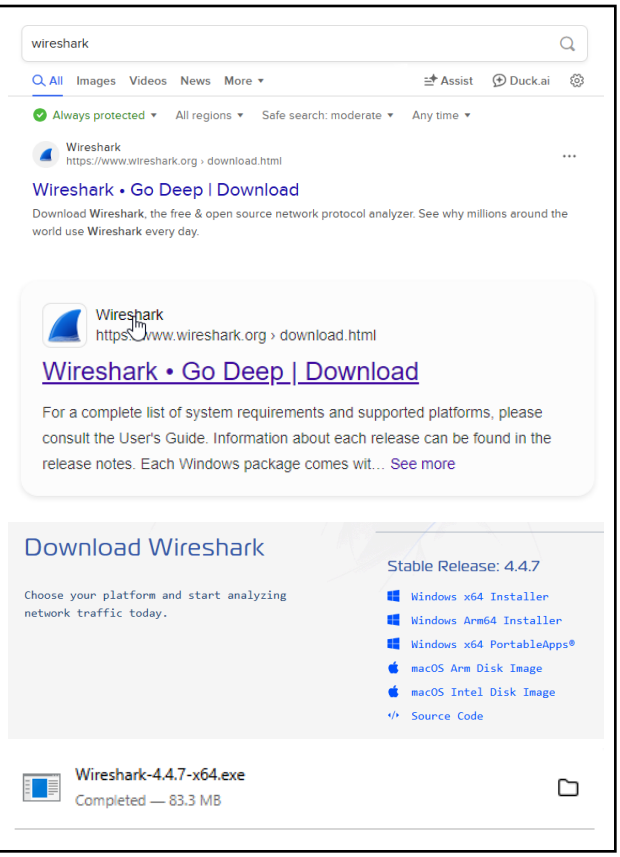
Students will install Wireshark on their Windows computers, perform a basic network scan, and apply filters to analyze network traffic.

Step-by-Step Instructions / Summary

- Part 1: Install Wireshark on Windows
- Part 2: Perform a Basic Network Scan
- Part 3: Apply Filters in Wireshark
- Part 4: Advanced Filters
- Part 5: Capture and Export

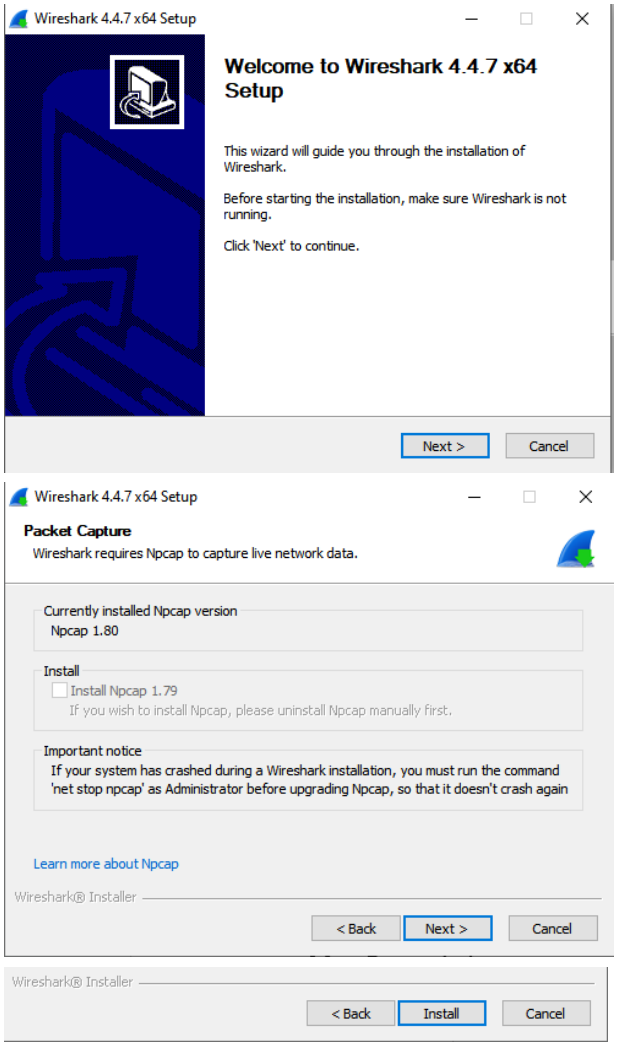
Steps and screenshots for this lab:

1. **Install Wireshark on Windows**
   - **1.1. Download Wireshark**

Going to the website to download Wireshark **Downloading the Windows Installer version**	

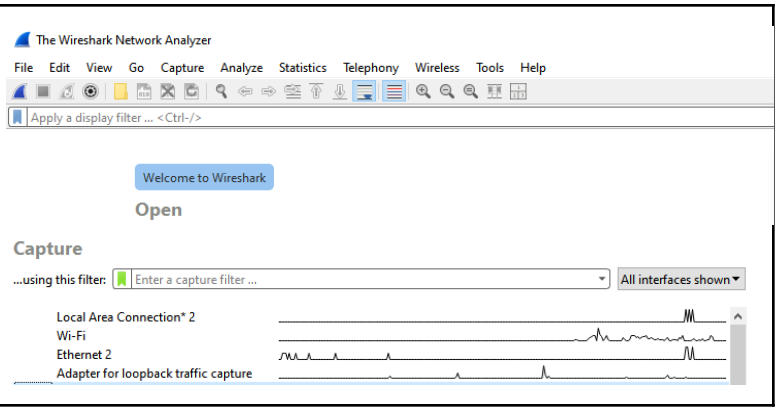
1.2. Install Wireshark

Running the installer
Accept all default components, includes WinPcap and NpCap



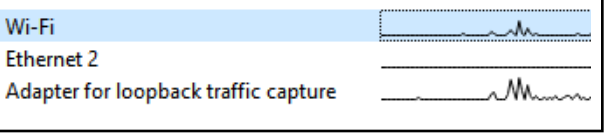
1.3. Verify Installation

Open Wireshark
Ensure all that Wireshark lists available network interfaces (WiFi, Ethernet)

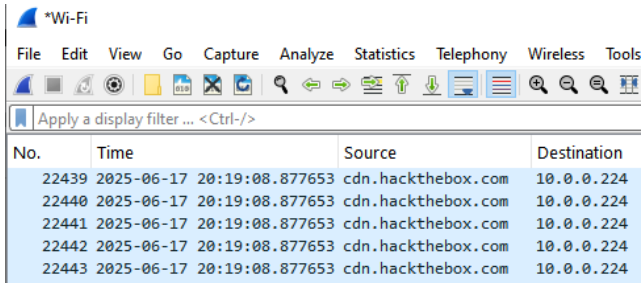


2. Perform a Basic Network Scan

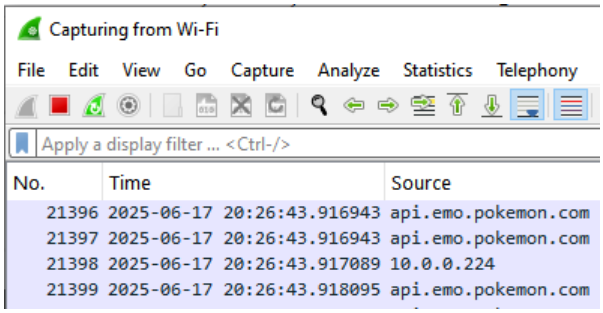
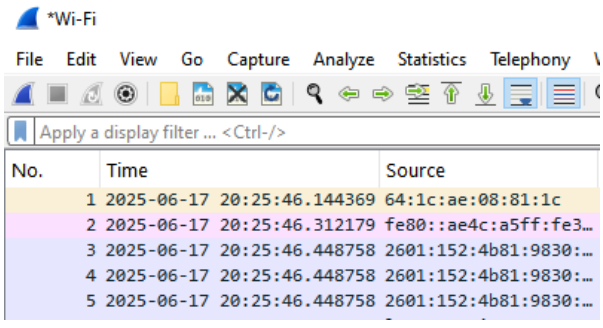
2.1. Select an interface

Selecting the Wi-FI interface	

2.2. Browse the internet

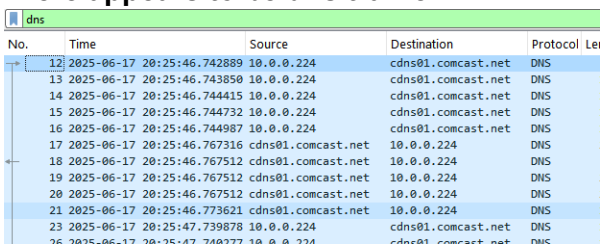
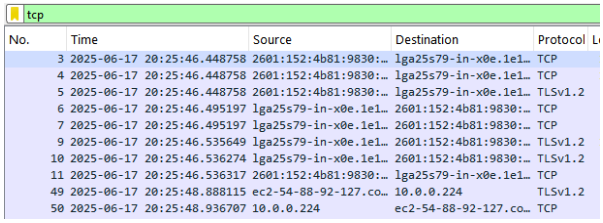
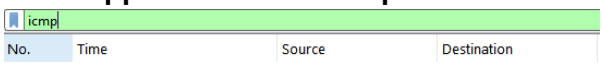
Press the shark fin **Open a website and visit websites like hackthebox.com**	

2.3. Stop the Capture

Press the red button to stop Wireshark from capture the packets	 **The screen once after the packets had stop capturing:** 

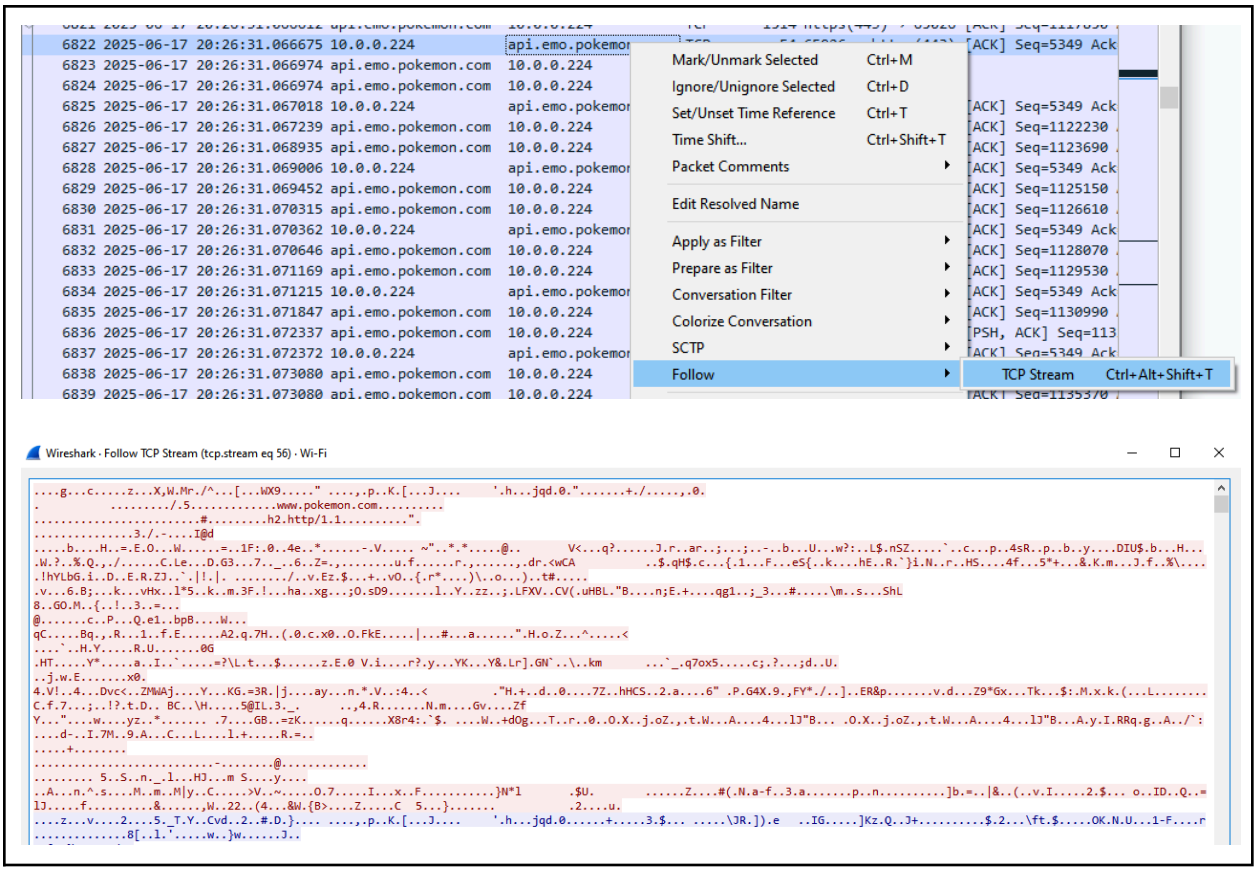
3. Apply Filters in Wireshark

3.1. Basic Filters

Filter http traffic **Filter dns traffic** **Filter tcp traffic** **Filter icmp traffic**	**There were no http traffic:**  **There appears to be dns traffic:**  **There appears to be tcp traffic:**  **There appears to be no icmp traffic:** 

3.2. Conservation Filters

Pressing right click and following the TCP stream

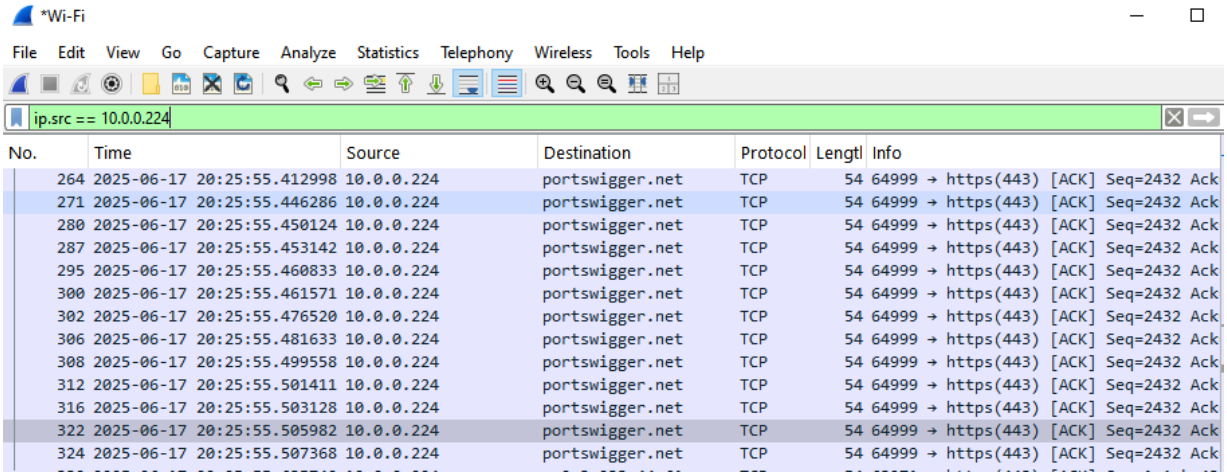


4. Advanced Filters

4.1. Filter A <-> Any

Apply a filter to show packets sent from your IP address to any destination.

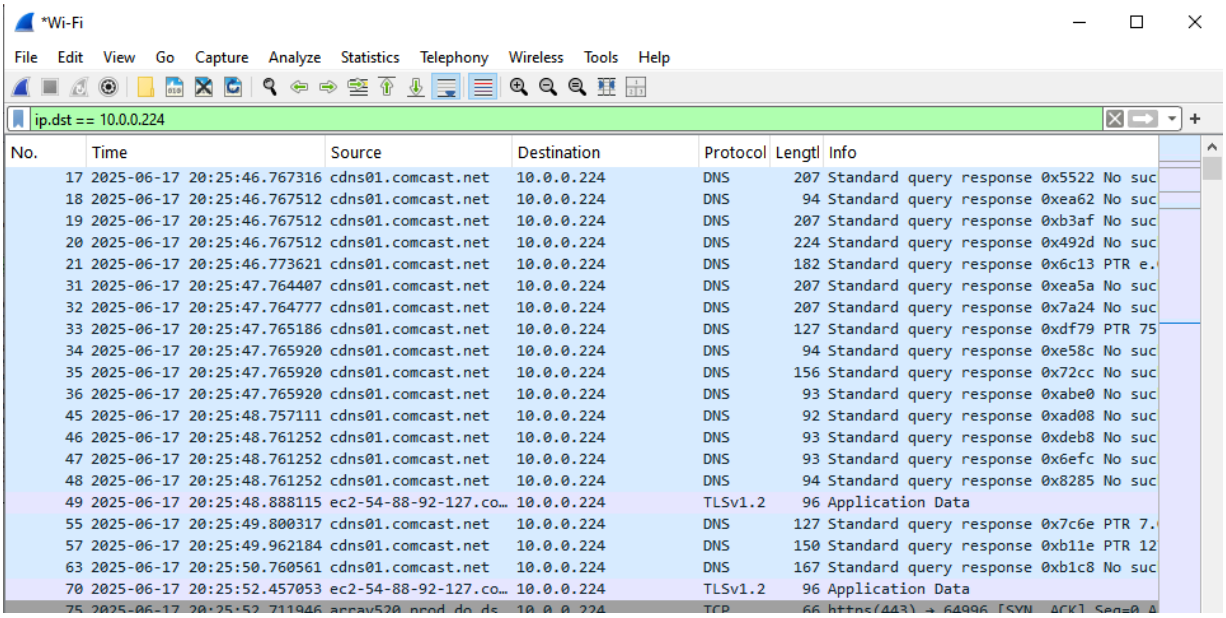
ip.src == 10.0.0.224



4.2. Filter Any <-> A

Apply a filter to show packets sent to your IP address from any source.

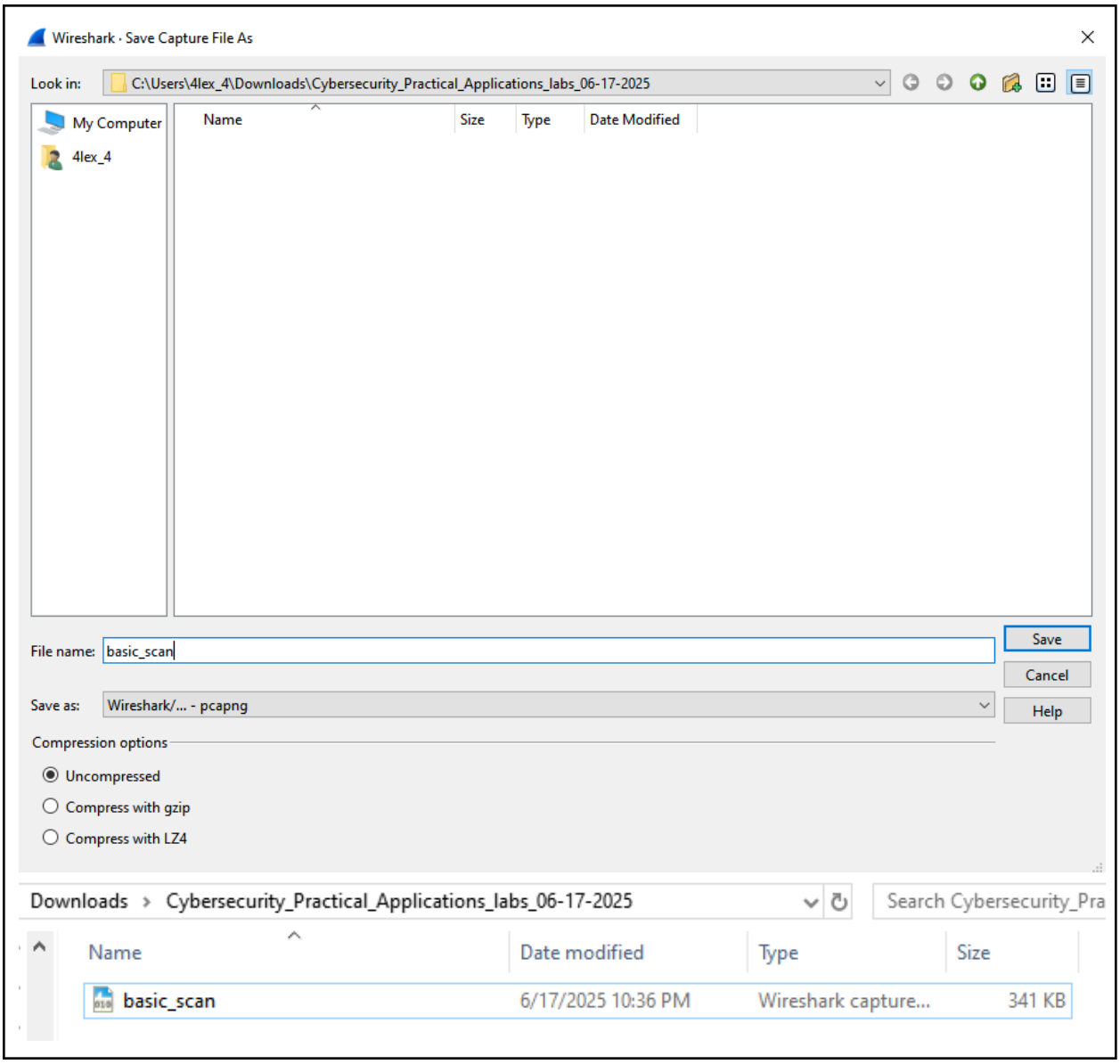
ip.dst == 10.0.0.224



5. Capture and Export

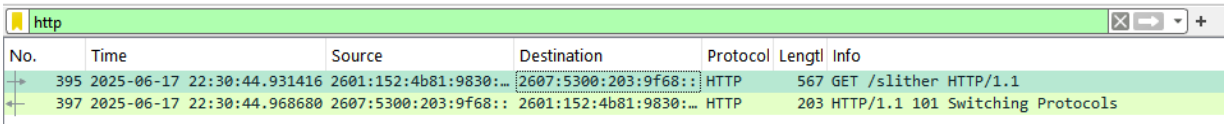
5.1. Export the capture

Export file and save as “basic_scan.pcapng”

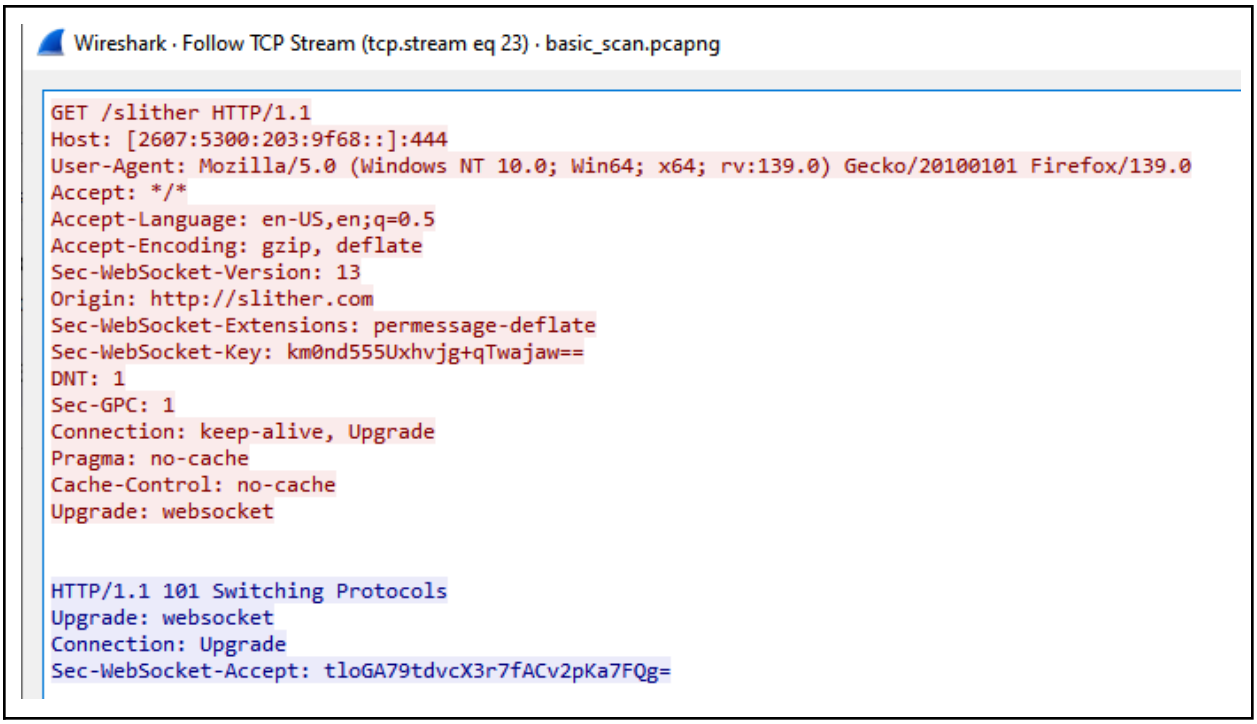


5.2. Take screenshots

http filter



Following the tcp



🧰 Tools & Skills Used

- **Wireshark**
- **Windows 10**
- **Packet Filtering**
- **Network Protocol Analysis**

🧠 Reflection & Takeaways

This lab helped me reinforce my prior knowledge with Wireshark. While I was familiar with Wireshark before with ctf competitions and other tasks. Revisiting this tool helps me identify small knowledge gaps with filtering syntax and capturing exports.
Due to performance issues with the vm, I opted to use my personal Windows 10 computer which gives me an optimal experience.

🔓 Lab#03 : Wireshark Lab: Capturing Cleartext Login with Local HTTP Server

🎯 Objectives:

Utilizing Wireshark on a local server to capture cleartext credentials.

Step-by-Step Instructions / Summary

- Part-1: Setup a local HTTP Server with Login Form
- Part-2: Start the server
- Part-3: Start Wireshark Capture
- Part-4: Filter in Wireshark

1. Setup a local HTTP Server with Login Form

1.1. Create a project folder

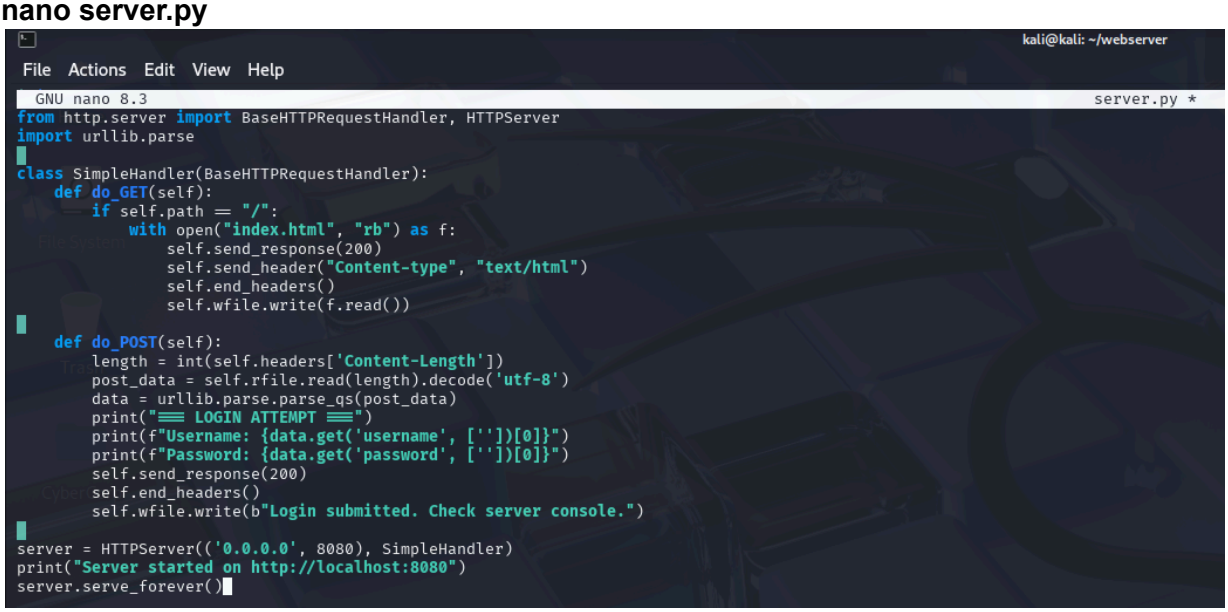
mkdir ~/webserver
cd ~/webserver

1.2. Create Fake Login Page (index.html)

nano index.html

1.3. Create Python Web Server Script (server.py)

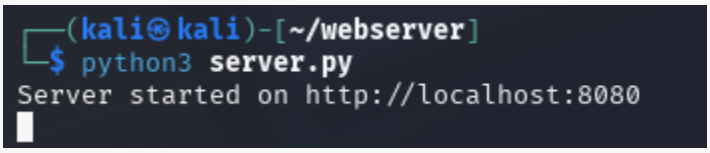
nano server.py



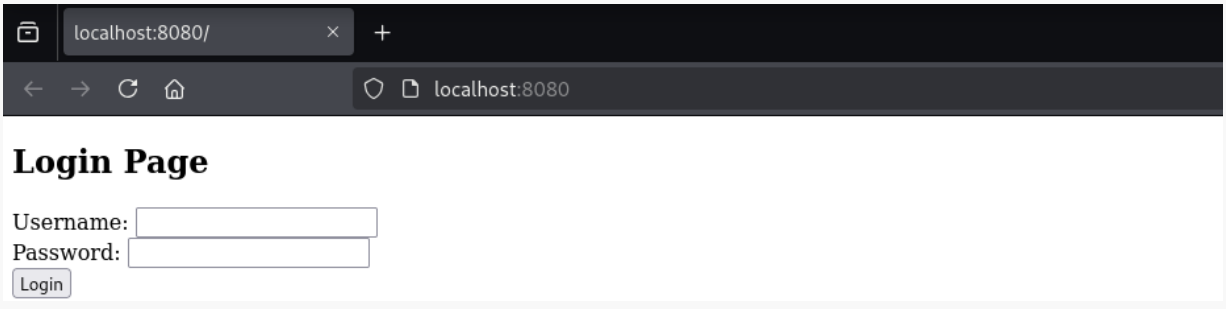
After the two files are saved, we'll use them for the login page.

2. Start the server

```
python3 server.py
```



We'll then go to use that login page and it pops up like this:



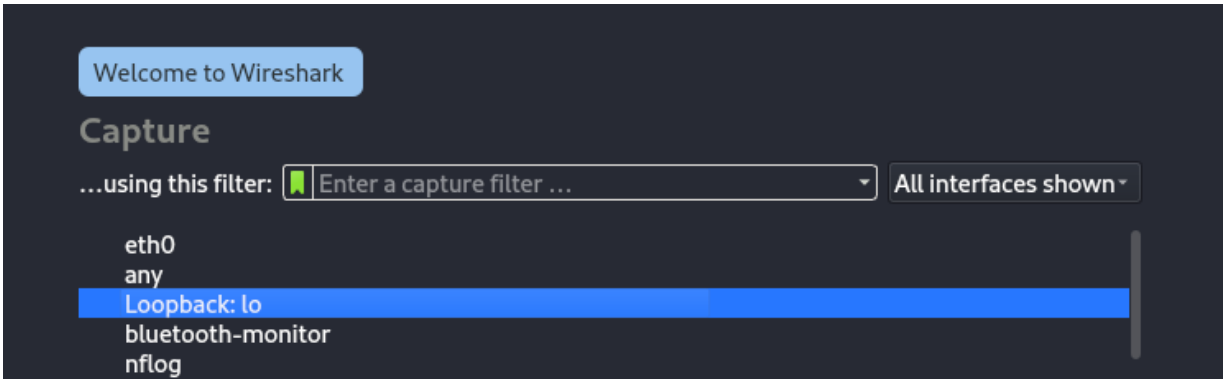
3. Start Wireshark Capture

3.1. Launch Wireshark

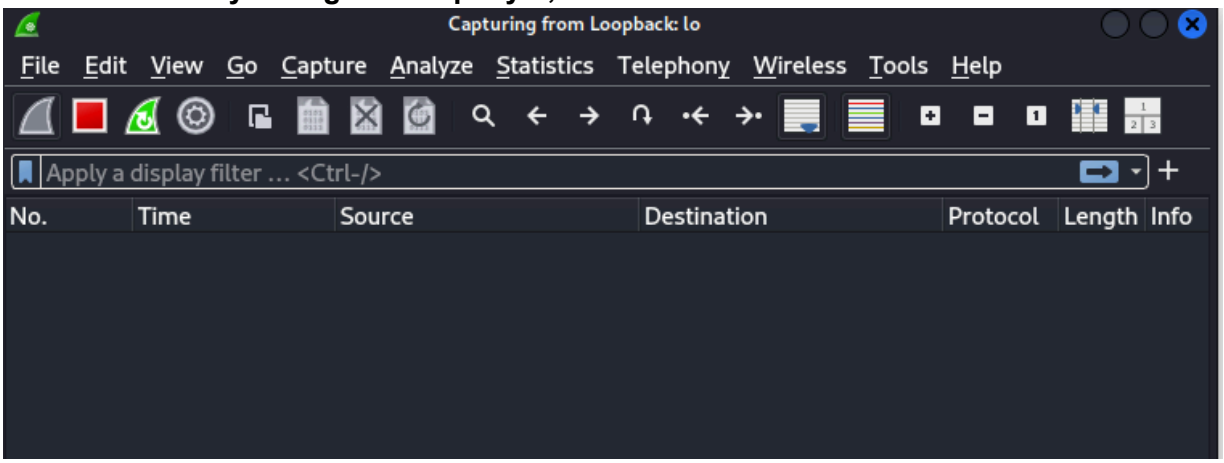
First is launching Wireshark through the terminal

```
┌──(kali㉿kali)-[~]
└─$ wireshark
 ** (wireshark:402178) 23:41:07.150492 [GUI ECHO] -- virtual const QPalette*
Qt6CTPlatformTheme::palette(QPlatformTheme::Palette) const QPlatformTheme::Sy
stemPalette
```

Afterwards we'll select the *IO interface* to capture the credentials from the webserver

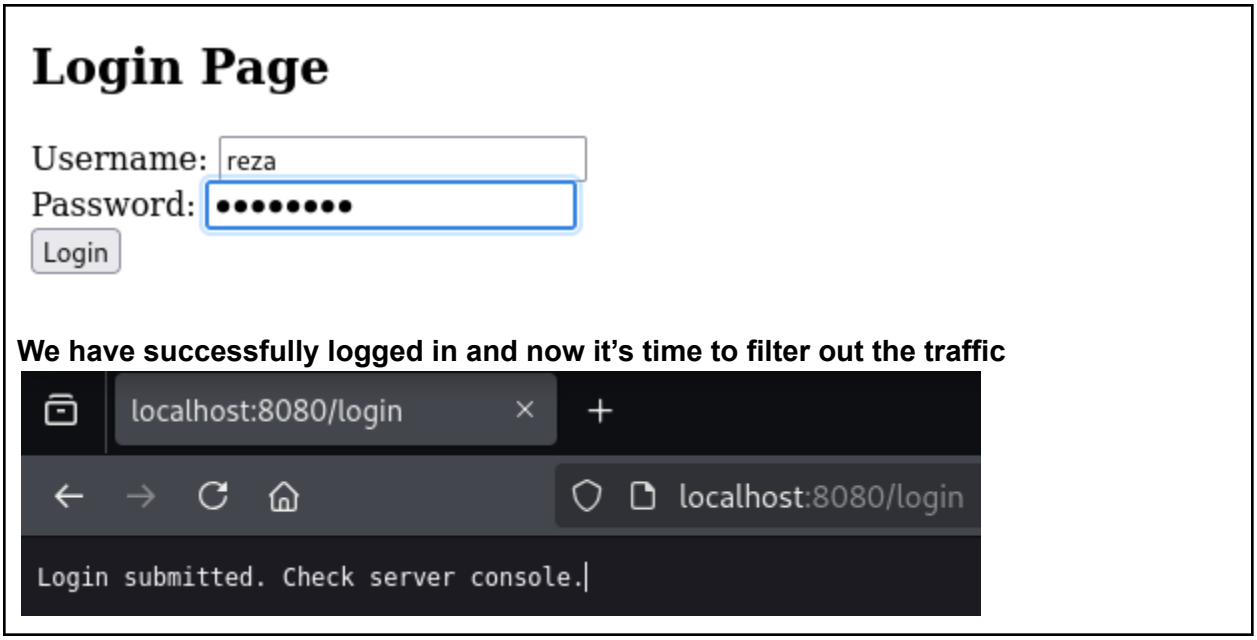


There is currently no login attempts yet, so we'll have to enter them to see the traffic



3.2. Visit the Login Page in Browser

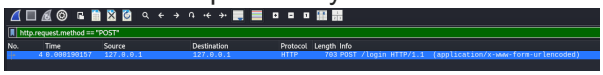
Coming back to the Login Page.
http://localhost:8080
This is the credentials that will be used:
username: reza
password: test@123



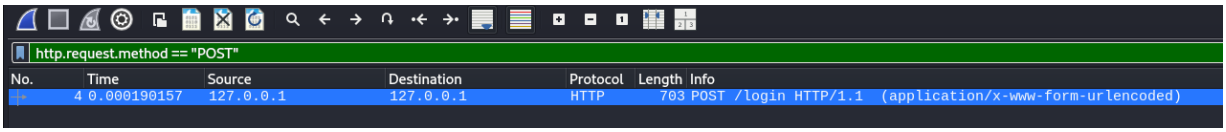
We have successfully logged in and now it's time to filter out the traffic

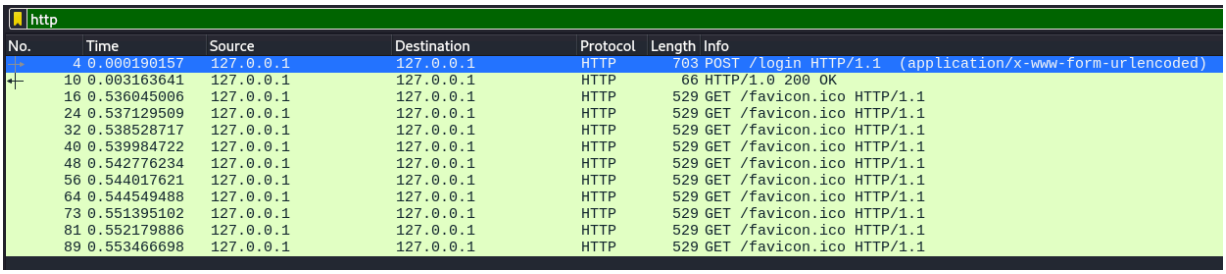
4. Filter in Wireshark

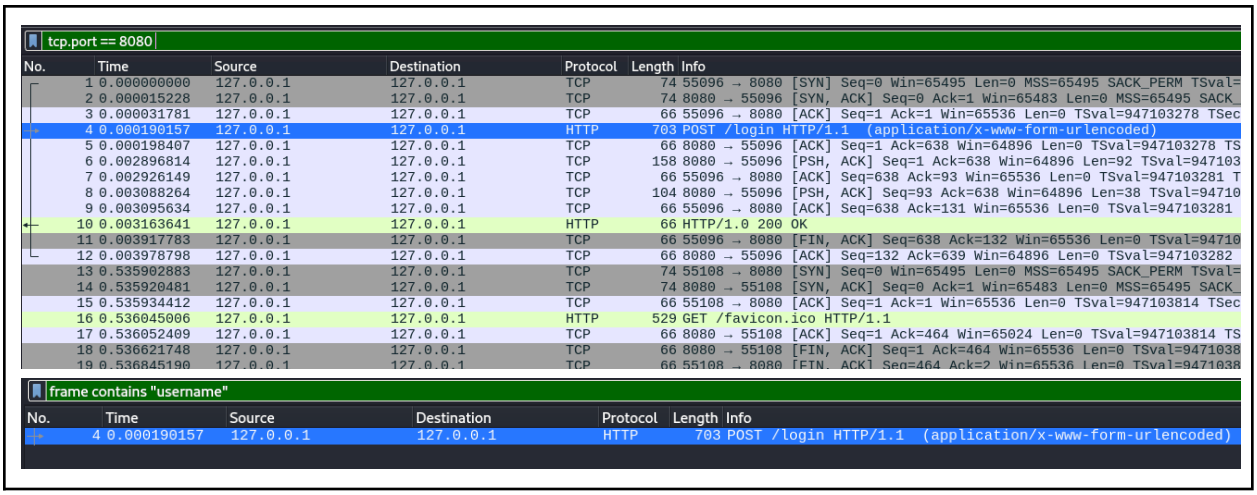
4.1. Using the following filters in order

Filter	What it does
http.request.method == "POST"	Shows POST requests only 
http	Shows all HTTP traffic
tcp.port == 8080	Shows all packets on port 8080
frame contains "username"	Searches for keyword "username" in frames

Filters in order:

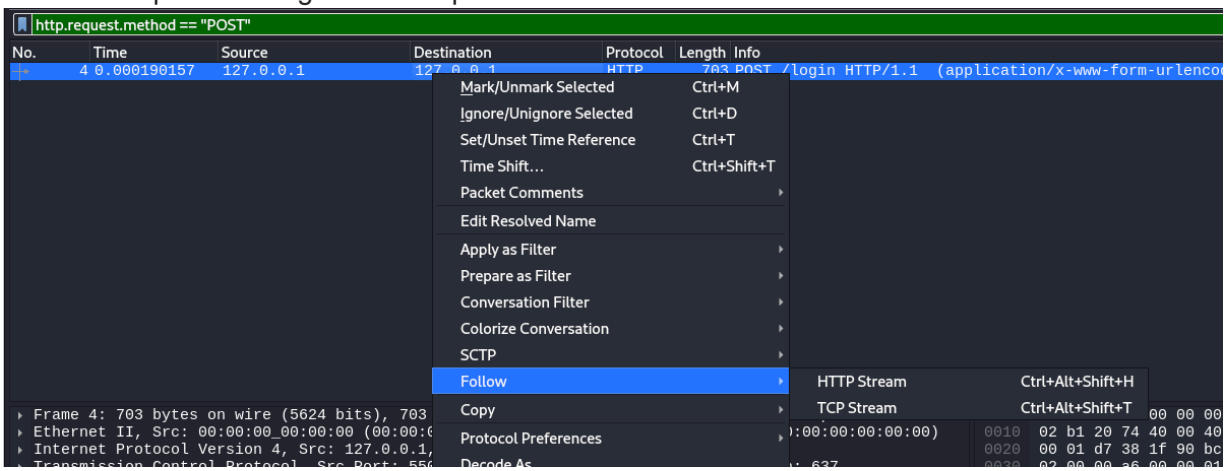




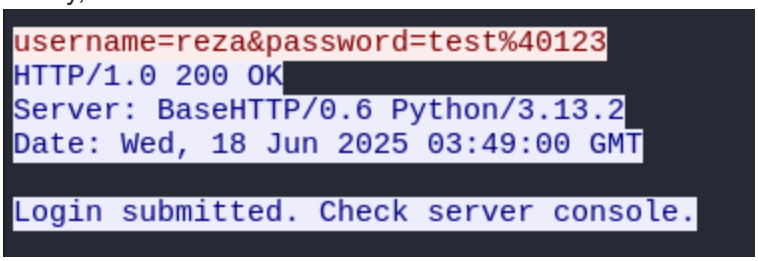


4.2. Follow the stream

The next step is following the POST packet from the follow > HTTP stream



Finally, we can find the credentials here!



🧰 Tools & Skills Used

- Python3 HTTP Server
- Kali Linux / Linux environment
- HTTP protocol inspection
- Credential sniffing techniques

🧠 Reflection & Takeaways

With this lab, it helped me sharpen my skill using Wireshark filters to capture cleartext credentials, a skill previously used for ctfs such as Cyberdefenders and PicoCTF. I remember specifically applying filters such as http.request.method == “POST” and searching for strings such as “username” reminded me how credentials can be stolen when encryption wasn’t implemented.

Lab #04a: Windows Event Viewer Exploration

Goal: Identify a failed login attempt and a system shutdown event using Event Viewer

🎯 Learning Objectives:

- Navigate Windows Event Viewer
- Analyze different types of logs (Application, Security, System)
- Locate and interpret a **failed login attempt**
- Locate and interpret a **system shutdown or reboot event**

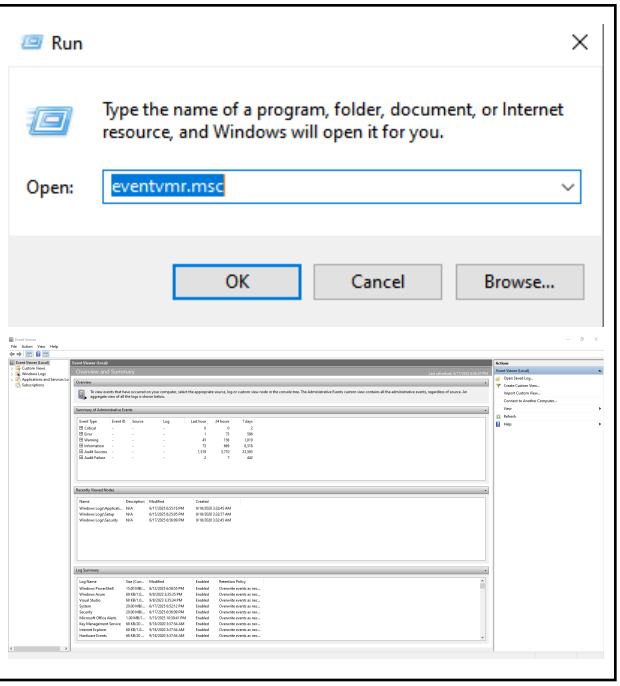
🛠️ Lab Environment Requirements:

- A Windows system (Windows 10 or 11 recommended)
- Admin privileges

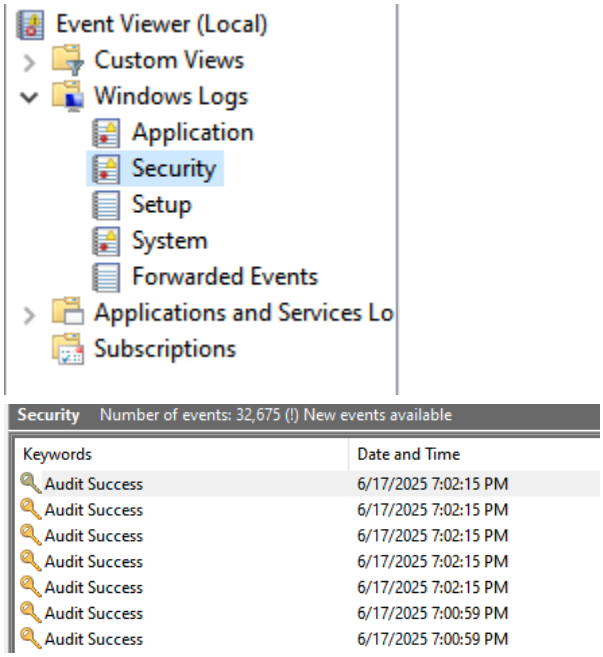
Step-by-Step Instructions / Summary

- Step 1: Open Event Viewer
- Step 2: Explore Log Types
- Step 3: Locate a Failed Login Attempt
- Step 4: Trigger a Failed Login
- Locate a Shutdown or Restart Event

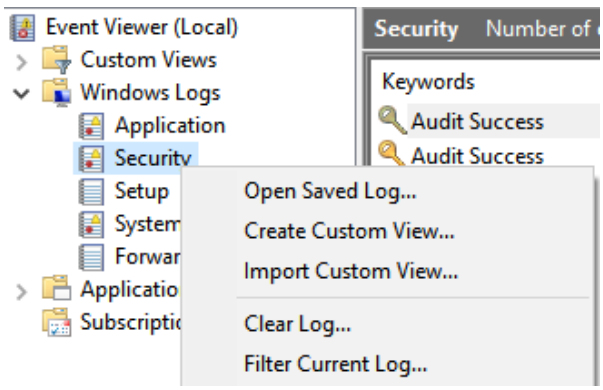
1. Open Event Viewer

Press Windows + R **type eventvwr.msc** **press Enter**	

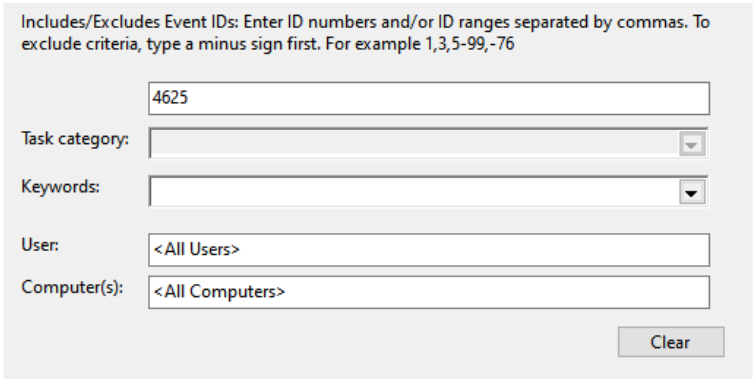
2. **Explore Log Types**
    a. **Expand the following in the left panel**

Press to expand the Windows Logs to show the following: Application, Security, and System.	

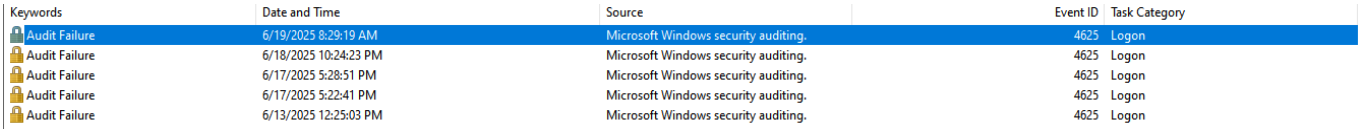
3. **Locate a Failed Login Attempt**

Right click security **Click Filter Current Log**	

Apply filter
Set Event IDs: 4625 (Failed login)

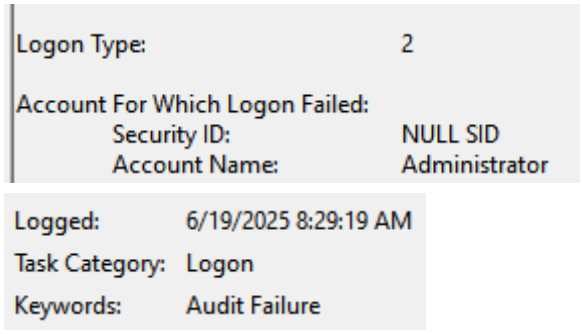


After the filter



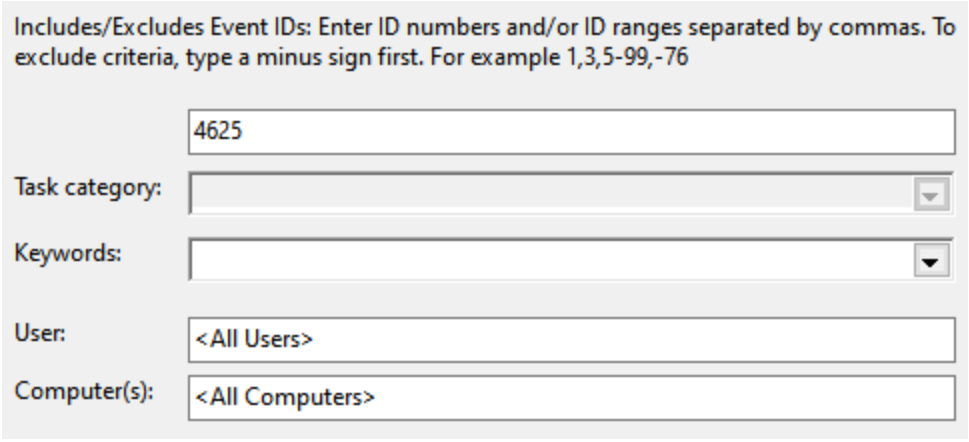
Keywords	Date and Time	Source	Event ID	Task Category
Audit Failure	6/19/2025 8:29:19 AM	Microsoft Windows security auditing.	4625	Logon
Audit Failure	6/18/2025 10:24:23 PM	Microsoft Windows security auditing.	4625	Logon
Audit Failure	6/17/2025 5:28:51 PM	Microsoft Windows security auditing.	4625	Logon
Audit Failure	6/17/2025 5:22:41 PM	Microsoft Windows security auditing.	4625	Logon
Audit Failure	6/13/2025 12:25:03 PM	Microsoft Windows security auditing.	4625	Logon

Required questions:
What was the **username** attempted?
Administrator
What is the **Logon Type**?
2
What is the **Failure Reason**?
Audit Failure



4. **Trigger a New Failed Login**
    a. **Log out and try to login with an incorrect password**
    b. **Repeat Step 3 to find new entry**

After signing out, I used a guest account and then utilized the run command to open Event Viewer and next I applied the filter for id 4625.

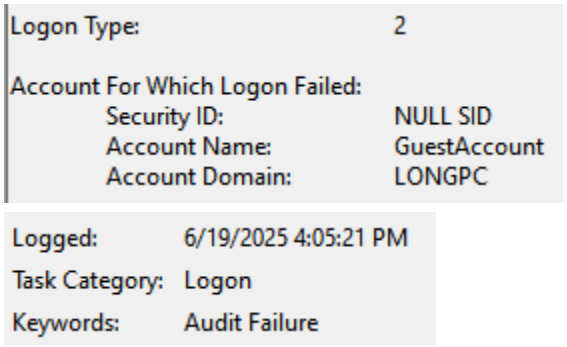


The listed events:

Filtered: Log: Security; Source: ; Event ID: 4625. Number of events: 14

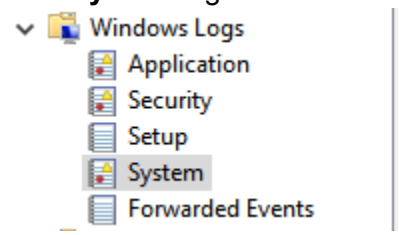
Keywords	Date and Time	Source	Event ID	Task Category
Audit Failure	6/19/2025 4:42:01 PM	Microsoft Windows security auditing.	4625	Logon
Audit Failure	6/19/2025 4:41:58 PM	Microsoft Windows security auditing.	4625	Logon
Audit Failure	6/19/2025 4:41:50 PM	Microsoft Windows security auditing.	4625	Logon
Audit Failure	6/19/2025 4:38:23 PM	Microsoft Windows security auditing.	4625	Logon
Audit Failure	6/19/2025 4:38:18 PM	Microsoft Windows security auditing.	4625	Logon
Audit Failure	6/19/2025 4:38:18 PM	Microsoft Windows security auditing.	4625	Logon
Audit Failure	6/19/2025 4:05:21 PM	Microsoft Windows security auditing.	4625	Logon

Required questions with different user:
What was the **username** attempted?
GuestAccount
What is the **Logon Type**?
2
What is the **Failure Reason**?
Audit Failure



5. Locate a Shutdown or Restart Event

Click **System** log:



After clicking the System log:

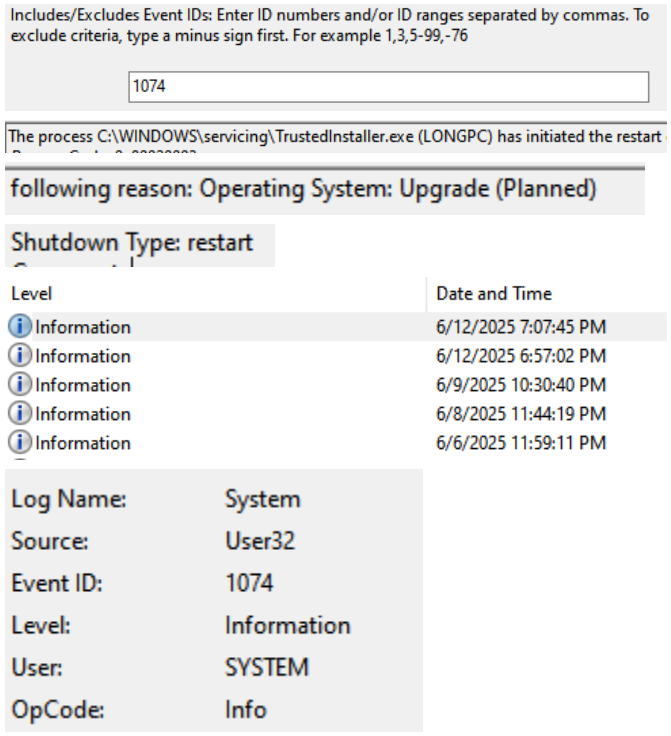
Level	Date and Time	Source	Event ID	Task Category
Warning	6/19/2025 5:08:08 PM	DNS Client Events	1014	(1014)
Warning	6/19/2025 5:01:25 PM	DistributedCOM	10016	None
Warning	6/19/2025 4:55:29 PM	DistributedCOM	10016	None
Warning	6/19/2025 4:48:18 PM	DistributedCOM	10016	None
Information	6/19/2025 4:46:59 PM	Service Control Manager	7040	None

Filter by Event IDs:
1074 (Planned shutdown/restart)

Required questions:
Was it planned or unexpected?
It was planned

Who initiated it?
It was initiated by SYSTEM

What was the reason?
Operating System: **Upgrade** (Planned)

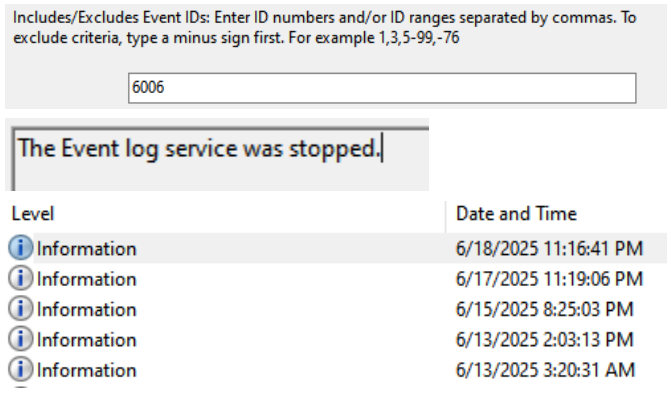


6006 (Event log service shutdown – system going down)

Required questions:
Was it planned or unexpected?
It was Planned (Clean shutdown)

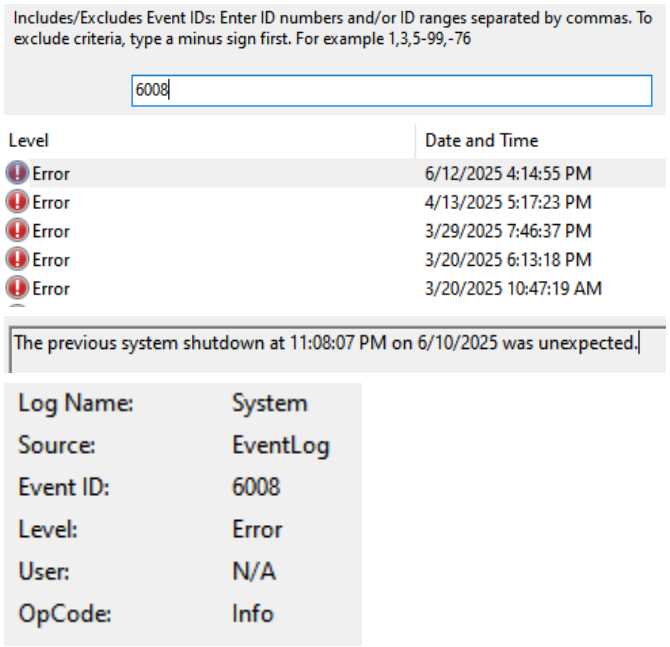
Who initiated it?
Not specified or N/A

What was the reason?
Event log service stopped due to restart/shutdown



	Log Name: System Source: EventLog Event ID: 6006 Level: Information User: N/A OpCode: Info

6008 (Unexpected shutdown)

Required questions: Was it planned or unexpected? It was not planned Who initiated it? Not specified or N/A What was the reason? User initiated shutdown	

🧰 Tools & Skills Used

- Windows Event Viewer
- Security & System Log Analysis
- Event ID Filtering

🧠 Reflection & Takeaways

This lab helped me refresh my skills in navigating and analyzing logs using Windows Event Viewer. I found the **Security** logs to be the most helpful, especially in identifying any failed login attempts. As for the **System** logs, they provide me information about shutdowns and restarts.

Monitoring logs is an essential skill in cybersecurity because they can help detect suspicious activity, identify insider threats, and understand system events. My takeaway of this lab is the importance to distinguish false negatives and false positives providing critical to threat detection and response.

Lab#05: Hash a File Using CMD

🎯 Objective

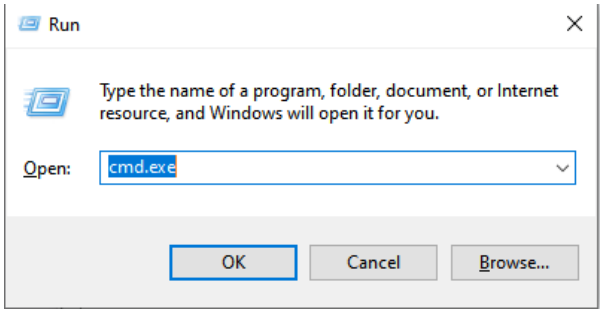
Using the certutil command from the Windows command line to hash a file and check its integrity.

Step-by-Step Instructions / Summary

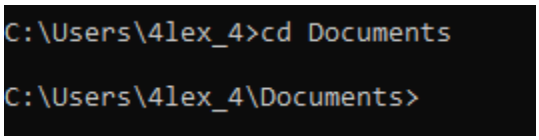
1. **Open Command Prompt**
2. **Navigate to the file location**
3. **Run certutil to hash the file**

Commands and steps used for this lab:

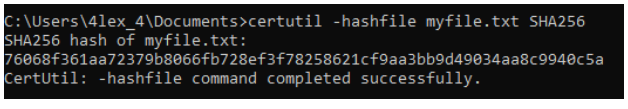
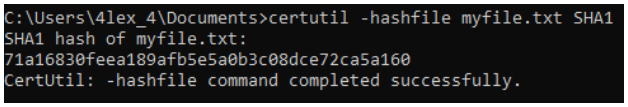
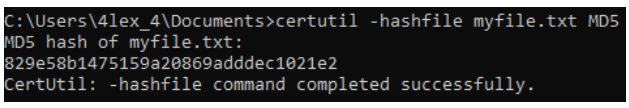
1. **Open Command Prompt**

Win + R, type cmd.exe **Press enter**	

2. **Navigate to the file location**

cd Documents	

3. **Run certutil to hash the file(SHA256, SHA1, MD5)**

Original **Hello there, this is a random text file!** **certutil -hashfile myfile.txt SHA256** **certutil -hashfile myfile.txt SHA1** **certutil -hashfile myfile.txt MD5** **Modified** ***Hello there, this is a random text file!*** **certutil -hashfile myfile.txt SHA256**	  

certutil -hashfile myfile.txt SHA1
certutil -hashfile myfile.txt MD5
	Here's the change after one character:
	`C:\Users\4lex_4\Documents>certutil -hashfile myfile.txt SHA256`
	`SHA256 hash of myfile.txt:`
	`00850dde2c53859df5071dac3a60341d443d134808421a818adc9cc3f037eaca`
	`CertUtil: -hashfile command completed successfully.`
	
	`C:\Users\4lex_4\Documents>certutil -hashfile myfile.txt SHA1`
	`SHA1 hash of myfile.txt:`
	`97636b0ef45b17049dee474a21650478e13016fd`
	`CertUtil: -hashfile command completed successfully.`
	
	`C:\Users\4lex_4\Documents>certutil -hashfile myfile.txt MD5`
	`MD5 hash of myfile.txt:`
	`71ea63fc549529b8489f07af9a51d4ac`
	`CertUtil: -hashfile command completed successfully.`

🧰 Tools & Skills Used

- **Windows Command Line, certutil, text file**
- **Skills: File hashing, file integrity checking**
- **Hash algorithms: SHA256, SHA1, MD5**

🧠 Reflection & Takeaways

In this lab I learned how to hash a file using certutil in this file. I made a few mistakes having the filename "myfile.txt" with an additional extension like "myfile.txt.txt." I quickly changed that and I added a few extra screenshots for the third step. The extra screenshots show how one character can alter the entire hash. Using hashes can ensure the file's integrity and is widely used in malware detection, digital forensics, and file validation.

Lab#06: UFW (Uncomplicated Firewall)

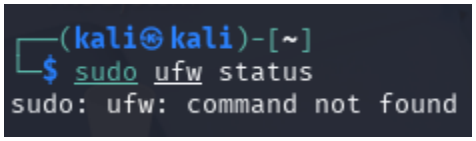
🎯 Objectives:

Setting the UFW and applying basic commands.

Step-by-Step Instructions / Summary

- Step 1: Install UFW
- Step 2: Basic Commands
- Step 3: Default Rules
- Step 4: Allowing Services/Ports
- Step 5: Denying Services/Ports
- Step 6: Allow or Deny by IP and Port
- Step 7: Deleting Rules
- Step 8: Advanced - Applications Profiles
- Step 9: Reload and Reset
- Step 10: Testing
- Step 11: Log and Monitor

1. **Installing UFW**

Check if UFW is installed sudo ufw status It appears to be not installed and the next step would be using the following command.	

sudo apt install ufw -y

```
┌──(kali㉿kali)-[~]
└─$ sudo apt install ufw -y
Installing:
  ufw

Suggested packages:
  rsyslog

Summary:
  Upgrading: 0, Installing: 1, Removing: 0, Not Upgrading: 1299
  Download size: 169 kB
  Space needed: 880 kB / 3,769 MB available

Get:1 http://kali.download/kali kali-rolling/main amd64 ufw all 0.36.2-9 [169 kB]
Fetched 169 kB in 1s (336 kB/s)
Preconfiguring packages ...
Selecting previously unselected package ufw.
(Reading database ... 407419 files and directories currently installed.)
Preparing to unpack .../archives/ufw_0.36.2-9_all.deb ...
Unpacking ufw (0.36.2-9) ...
Setting up ufw (0.36.2-9) ...
Creating config file /etc/ufw/before.rules with new version
Creating config file /etc/ufw/before6.rules with new version
Creating config file /etc/ufw/after.rules with new version
Creating config file /etc/ufw/after6.rules with new version
update-rc.d: We have no instructions for the ufw init script.
update-rc.d: It looks like a non-network service, we enable it.
Created symlink '/etc/systemd/system/multi-user.target.wants/ufw.service' → '/usr/lib/systemd/system/ufw.service'.
Processing triggers for kali-menu (2025.1.1) ...
Processing triggers for man-db (2.13.0-1) ...
```

2. **Basic Commands**
   a.
   b. Enable UFW

sudo ufw enable	```┌──(kali㉿kali)-[~] └─$ sudo ufw enable Firewall is active and enabled on system startup```

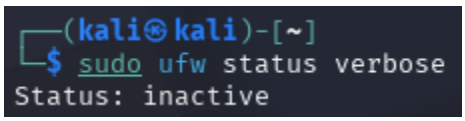
   c. Disable UFW

sudo ufw disable	```┌──(kali㉿kali)-[~] └─$ sudo ufw disable Firewall stopped and disabled on system startup```

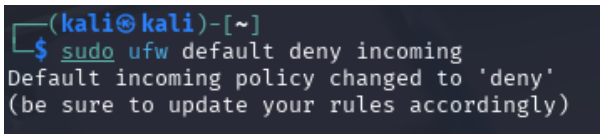
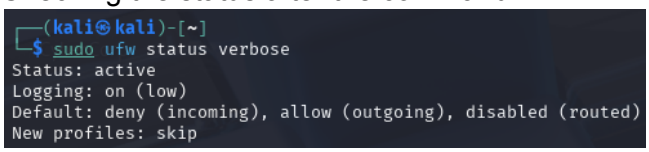
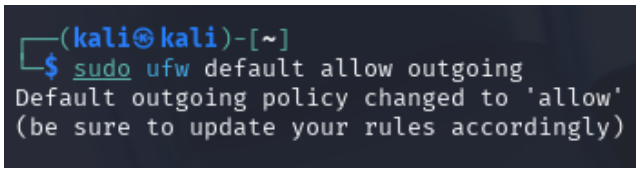
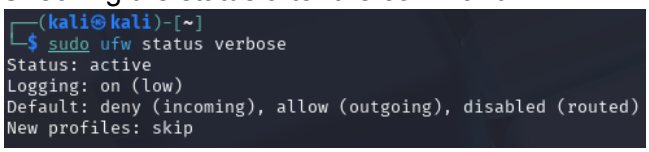
   d. Check Status

sudo ufw status	```┌──(kali㉿kali)-[~] └─$ sudo ufw status Status: inactive```

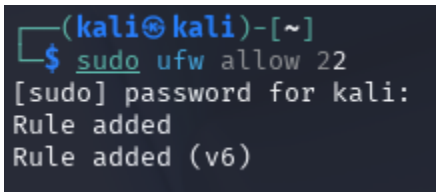
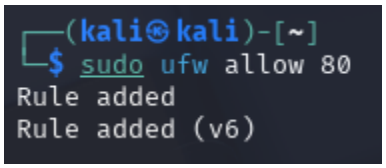
   e. View in Verbose Mode

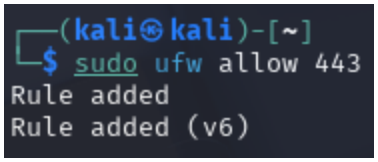
sudo ufw status verbose *If the ufw is active, it'll show if IPv6 is considered for each rule	

3. **Default Rules**
    a. Set Default Policies

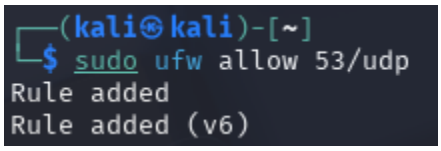
After changing back the ufw status active: sudo ufw default deny incoming *This command blocks all incoming connections by default unless specified to be allowed.	 Checking the status after the command: 
sudo ufw default allow outcoming *This command allows outgoing connections by default without restriction	 Checking the status after the command: 

4. **Allowing Services/Ports**
    a. **Allow specific ports**

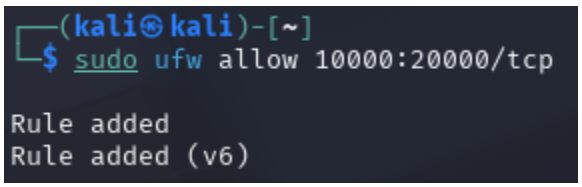
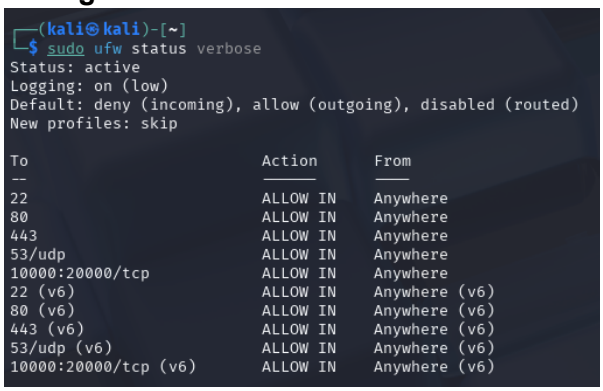
sudo ufw allow 22 #SSH **sudo ufw allow 80 #HTTP** **sudo ufw allow 443 #HTTPS**	 

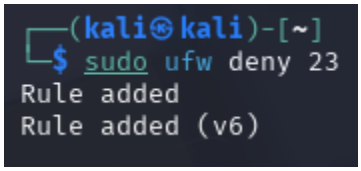
b. **Allow by Port and Protocol**

sudo ufw allow 53/udp # DNS over UDP	

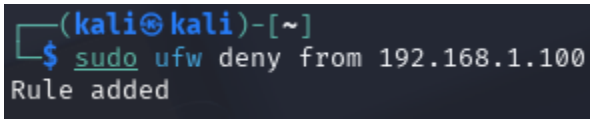
c. **Allow a Range of Ports**

sudo ufw allow 10000:20000/tcp	 **Listing all rules so far:** 

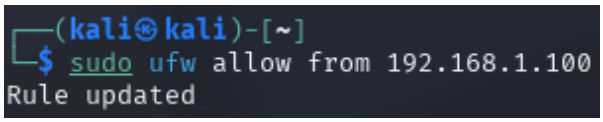
5. **Denying Services/Ports**
   a. **Deny a port**

sudo ufw deny 23 #Telnet	

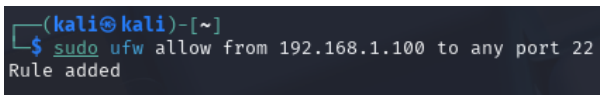
b. **Deny by IP**

sudo ufw deny from 192.168.1.100	

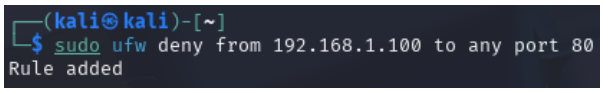
6. **Allow or Deny by IP and Port**
   a. **Allow from a specific IP**

sudo ufw allow from 192.168.1.100	

b. **Allow from IP from a specific port**

sudo ufw allow from 192.168.1.100 to any port 22	

c. **Deny from IP to port**

sudo ufw deny from 192.168.1.100 to any port 80	

7. **Deleting Rules**
   a. **Listing Numbered Rules**

sudo ufw status numbered

```
┌──(kali㉿kali)-[~]
└─$ sudo ufw status numbered
Status: active

     To                         Action      From
     --                         ------      ----
[ 1] 22                         ALLOW IN    Anywhere
[ 2] 80                         ALLOW IN    Anywhere
[ 3] 443                        ALLOW IN    Anywhere
[ 4] 53/udp                     ALLOW IN    Anywhere
[ 5] 10000:20000/tcp            ALLOW IN    Anywhere
[ 6] 23                         DENY IN     Anywhere
[ 7] Anywhere                   ALLOW IN    192.168.1.100
[ 8] 22                         ALLOW IN    192.168.1.100
[ 9] 80                         DENY IN     192.168.1.100
[10] 22 (v6)                    ALLOW IN    Anywhere (v6)
[11] 80 (v6)                    ALLOW IN    Anywhere (v6)
[12] 443 (v6)                   ALLOW IN    Anywhere (v6)
[13] 53/udp (v6)                ALLOW IN    Anywhere (v6)
[14] 10000:20000/tcp (v6)       ALLOW IN    Anywhere (v6)
[15] 23 (v6)                    DENY IN     Anywhere (v6)
```

b. Delete by Number

sudo ufw delete [number]

```
┌──(kali㉿kali)-[~]
└─$ sudo ufw delete 11
Deleting:
 allow 80
Proceed with operation (y|n)? y
Rule deleted (v6)
```

c. Delete by Rule

sudo ufw delete 11

```
┌──(kali㉿kali)-[~]
└─$ sudo ufw delete 11
Deleting:
 allow 80
Proceed with operation (y|n)? y
Rule deleted (v6)
```

After deleting the rule:

sudo ufw status numbered

```
┌──(kali㉿kali)-[~]
└─$ sudo ufw status numbered
Status: active

     To                         Action      From
     --                         ------      ----
[ 1] 22                         ALLOW IN    Anywhere
[ 2] 80                         ALLOW IN    Anywhere
[ 3] 443                        ALLOW IN    Anywhere
[ 4] 53/udp                     ALLOW IN    Anywhere
[ 5] 10000:20000/tcp            ALLOW IN    Anywhere
[ 6] 23                         DENY IN     Anywhere
[ 7] Anywhere                   ALLOW IN    192.168.1.100
[ 8] 22                         ALLOW IN    192.168.1.100
[ 9] 80                         DENY IN     192.168.1.100
[10] 22 (v6)                    ALLOW IN    Anywhere (v6)
[11] 443 (v6)                   ALLOW IN    Anywhere (v6)
[12] 53/udp (v6)                ALLOW IN    Anywhere (v6)
[13] 10000:20000/tcp (v6)       ALLOW IN    Anywhere (v6)
[14] 23 (v6)                    DENY IN     Anywhere (v6)
```

The removal shows that there are 14 rules now and 80 ALLOW IN rule was deleted

8. **Advanced - Applications Profiles**
    a. **List App Profiles**

sudo ufw app list
This command lists all application profiles from this directory "/etc/ufw/applications.d"

```
└─$ sudo ufw app list
Available applications:
  AIM
  Apache
  Apache Full
  Apache Secure
  Bonjour
  CIFS
  DNS
  Deluge
  IMAP
  IMAPS
  IPP
  KTorrent
  Kerberos Admin
  Kerberos Full
  Kerberos KDC
  Kerberos Password
  LDAP
  LDAPS
```

b. **Get App Info**

sudo ufw app info OpenSSH
This command displays detailed information about the predefined application profile used by the ufw firewall.

```
┌──(kali㉿kali)-[~]
└─$ sudo ufw app info OpenSSH

Profile: OpenSSH
Title: Secure shell server, an rshd replacement
Description: OpenSSH is a free implementation of the Secure Shell protocol.

Port:
  22/tcp
```

c. **Allow by App Name**

sudo ufw allow OpenSSH

Checking the firewall rules change:
sudo ufw status

```
┌──(kali㉿kali)-[~]
└─$ sudo ufw allow OpenSSH
[sudo] password for kali:
Rule added
Rule added (v6)
```

```
┌──(kali㉿kali)-[~]
└─$ sudo ufw status
Status: active

To                         Action      From
--                         ------      ----
OpenSSH                    ALLOW       Anywhere
```

9. **Reload and Reset**
   a. **Reload UFW**

sudo ufw reload
Refreshes the config file without shutting down the system.

```
┌──(kali㉿kali)-[~]
└─$ sudo ufw reload
Firewall reloaded
```

b. **Reset All Tools**

sudo ufw reset
With this command completely resets UFW to its default state

```
┌──(kali㉿kali)-[~]
└─$ sudo ufw reset
Resetting all rules to installed defaults. Proceed with operation (y|n)? y
Backing up 'user.rules' to '/etc/ufw/user.rules.20250626_192010'
Backing up 'before.rules' to '/etc/ufw/before.rules.20250626_192010'
Backing up 'after.rules' to '/etc/ufw/after.rules.20250626_192010'
Backing up 'user6.rules' to '/etc/ufw/user6.rules.20250626_192010'
Backing up 'before6.rules' to '/etc/ufw/before6.rules.20250626_192010'
Backing up 'after6.rules' to '/etc/ufw/after6.rules.20250626_192010'
```

10. Testing

a. Install a Web Server and setting up the server

sudo apt update

```
┌──(kali㉿kali)-[~]
└─$ sudo apt update
Hit:1 http://http.kali.org/kali kali-rolling InRelease
1295 packages can be upgraded. Run 'apt list --upgradable' to see them.
```

sudo apt install apache2 -y

```
┌──(kali㉿kali)-[~]
└─$ sudo apt install apache2 -y
apache2 is already the newest version (2.4.63-1).
```

sudo systemctl start apache2

b. Enable and Configure UFW (Only SSH and HTTP)

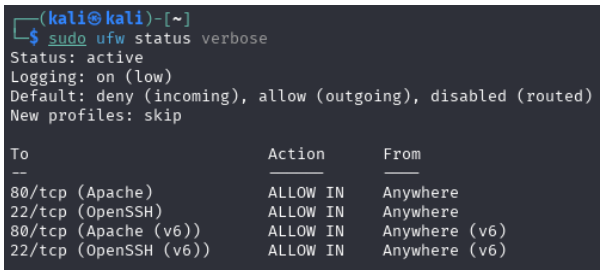
Allows SSH
sudo ufw allow OpenSSH

```
┌──(kali㉿kali)-[~]
└─$ sudo ufw allow OpenSSH
Rules updated
Rules updated (v6)
```

Allows HTTP traffic (port 80)
sudo ufw allow Apache

```
┌──(kali㉿kali)-[~]
└─$ sudo ufw allow Apache
Rule updated
Rule updated (v6)
```

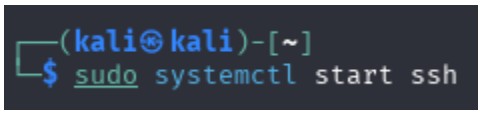
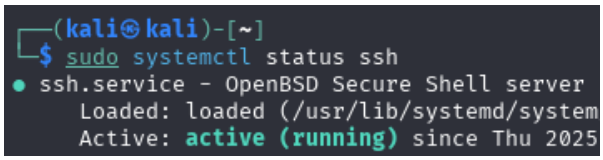
Checking the rules after additional rules:
sudo ufw status verbose

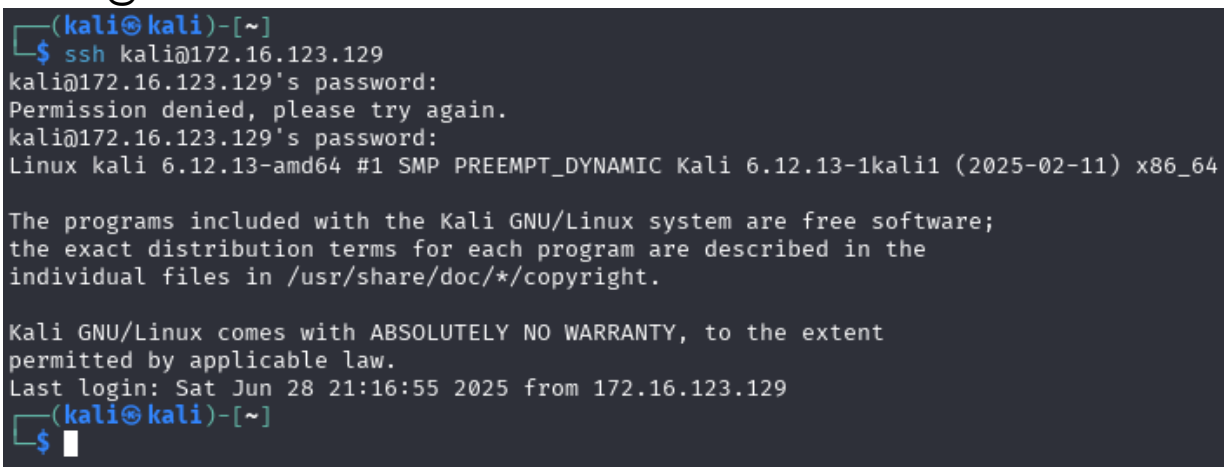
c. Access from Another System

i. Testing allowed

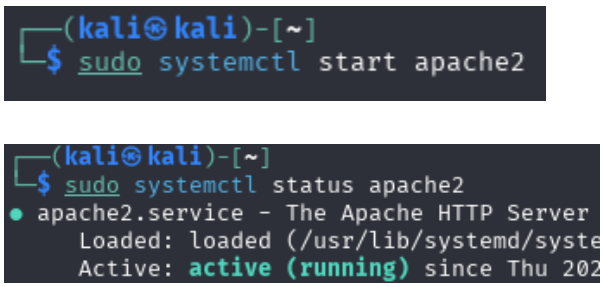
Starting SSH service

Starting the ssh service: **sudo systemctl start ssh** **Checking the ssh service status** **sudo systemctl status ssh**	 

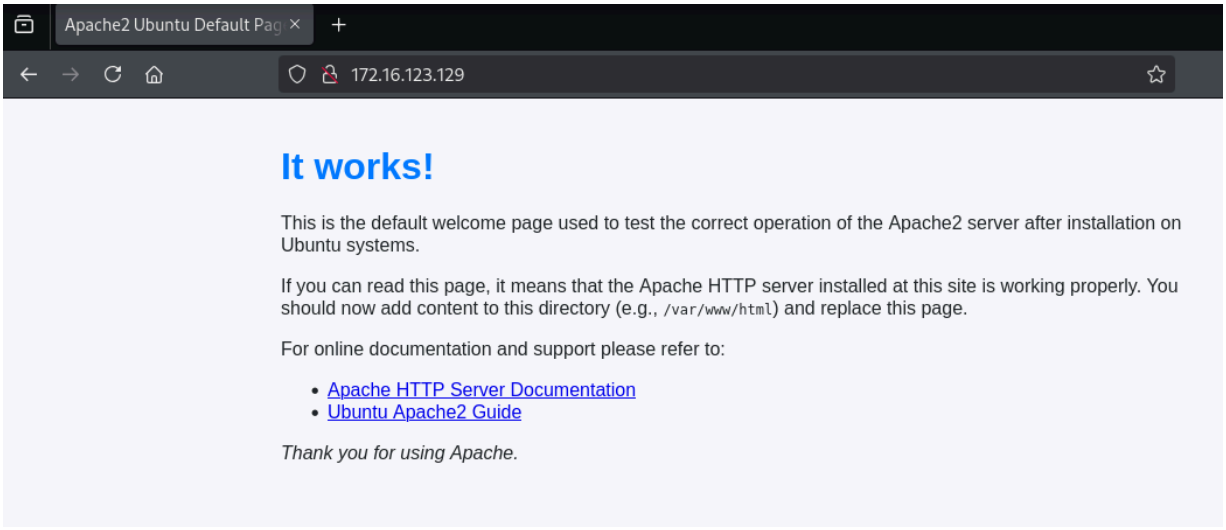
Testing SSH (allowed)
ssh kali@172.16.123.129



Starting Apache Service

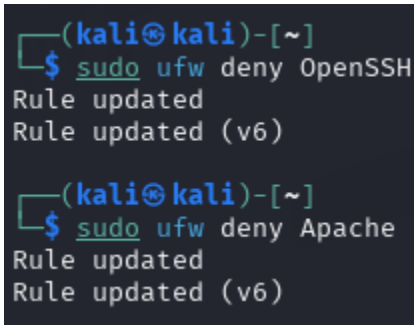
Starting the Apache service: **sudo systemctl start apache2** **Checking the apache service status** **sudo systemctl status apache2**	

Testing HTTP (Allowed)
Visiting: http://172.12.168



ii. Testing denied

Adding rules to deny the ports

sudo ufw deny OpenSSH **sudo ufw deny Apache** **Showing the rules that were added:** **sudo ufw status verbose**	

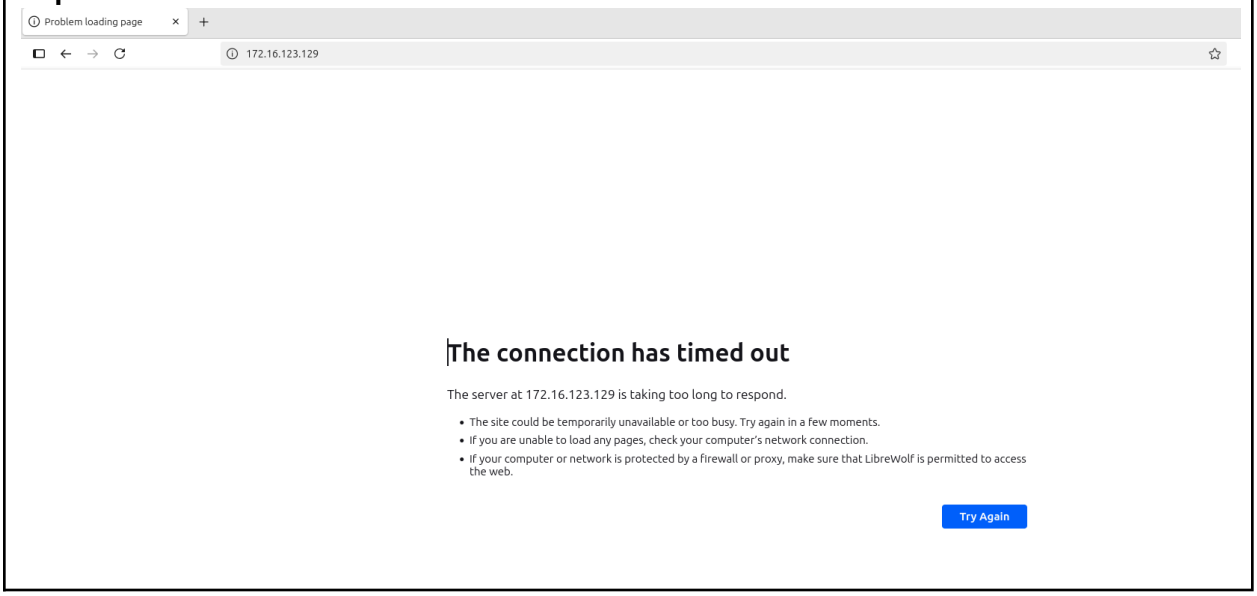
```
┌──(kali㉿kali)-[~]
└─$ sudo ufw status verbose
Status: active
Logging: on (low)
Default: deny (incoming), allow (outgoing), disabled (routed)
New profiles: skip

To                         Action      From
--                         ------      ----
22/tcp (OpenSSH)           DENY IN     Anywhere
80/tcp (Apache)            DENY IN     Anywhere
22/tcp (OpenSSH (v6))      DENY IN     Anywhere (v6)
80/tcp (Apache (v6))       DENY IN     Anywhere (v6)
```

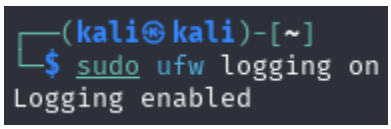
The services are denied after rules were added

SSH services denied (Host OS):
ssh kali@172.16.123.129

```
tempadmin@AlexPC447:~$ ssh kali@172.16.123.129
ssh: connect to host 172.16.123.129 port 22: Connection timed out
```

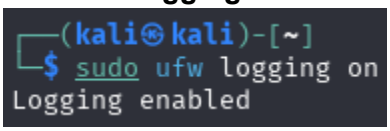
Apache Services denied (Host OS):
http://172.16.123.129



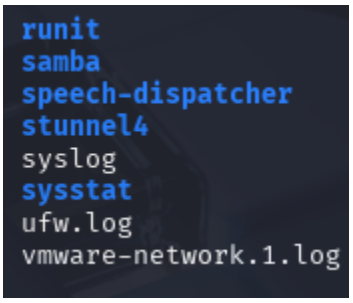
sudo ufw logging on	

11. Log and Monitor

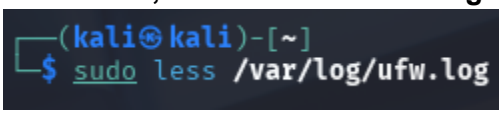
sudo ufw logging on

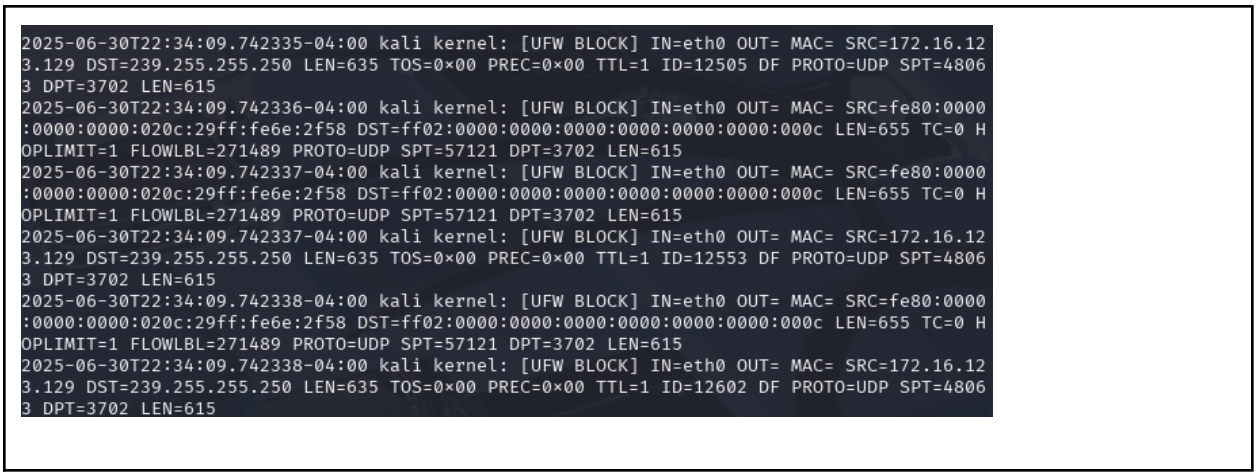


I used this command since the ufw logging enabled didn't create the ufw.log file.
systemctl start ufw.service



Afterwards, it's time to look through the logs





🧰 Tools & Skills Used

Tools:

- **Operating Systems:** Kali, Linux Mint (Host OS)
- **Core Services & Daemons:** Ufw (Uncomplicated Firewall), apache2, sshd
- **Command-Line Utilities:** sudo, systemctl, ssh, web browser

Skills:

- **Firewall Configuration, Log Analysis, System Troubleshooting**

🧠 Reflection & Takeaways

I learned the basics of ufw using both deny and allow connections. I had a hard time trying to show a firewall blocking my connections with another machine. I used the correct kali IP address to help me get the proper results. Another problem I had was enabling the ufw logging. I turned on the logging, but it appeared that it didn't work. So, I started the ufw.service again and I was able to capture logs.

Lab#07: Host-Based Firewall Configuration using UFW

🎯 Objective

Learn how to configure a host-based firewall using ufw (Uncomplicated Firewall) to allow or deny specific traffic and test it from a remote machine.

🧰 Lab Requirements:

- **Kali Linux** (Server role)
- **Parrot OS / Kali Linux** (Client role)
- ufw installed on both systems
- Network connectivity (Ping and SSH should initially work)

Step-by-Step Instructions / Summary

- 🔧**Part 1: Initial Setup and Connectivity**
  - Step 1: Get the IP address of Server
  - Step 2: Ping Test from Client
- 🔐**Part 2: Install and Enable UFW on Server**
  - Step 3: Install UFW (if not present)
  - Step 4: Enable UFW
  - Step 5: Check Current Status and Rules
- 🧱**Part 3: Block All Incoming Except SSH**
  - Step 6: Set Default Deny Policy
  - Step 7: Allow SSH
- 🔄**Part 4: Test Firewall Blocking from Client**
  - Step 8: Try to Ping from Client (should fail)
- 🎯 **Part 5: Allow HTTP (Simulated Web Server Test)**
  - Step 10: Install Apache2 on Server
  - Step 11: Allow HTTP
  - Step 12: Access Web Server from Client
- 🔍 **Part 6: View and Delete Rules**
  - Step 13: View Rules with Numbers
  - Step 14: Delete a Rule
- 💡 **Bonus: Enable Logging**

Steps and and screenshots for this labs:

🔧Part 1: Initial Setup and Connectivity

Get the IP Address of Server (Kali)
ip a

```
┌──(kali㉿kali)-[~]
└─$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:6e:2f:58 brd ff:ff:ff:ff:ff:ff
    inet 172.16.123.129/24 brd 172.16.123.255 scope global dynamic noprefixroute eth0
       valid_lft 954sec preferred_lft 954sec
    inet6 fe80::20c:29ff:fe6e:2f58/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
```

Ping Test from Client (Parrot)
ping 172.16.123.129

```
┌──[user@parrot]─[~]
└─$ ping 172.16.123.129
PING 172.16.123.129 (172.16.123.129) 56(84) bytes of data.
64 bytes from 172.16.123.129: icmp_seq=1 ttl=64 time=220 ms
64 bytes from 172.16.123.129: icmp_seq=2 ttl=64 time=1.14 ms
64 bytes from 172.16.123.129: icmp_seq=3 ttl=64 time=111 ms
64 bytes from 172.16.123.129: icmp_seq=4 ttl=64 time=1.08 ms
```

🔐Part 2: Install and Enable UFW on Server

Check any packages needed to be installed
sudo apt update

```
┌──(kali㉿kali)-[~]
└─$ sudo apt update
[sudo] password for kali:
Hit:1 http://http.kali.org/kali kali-rolling InRelease
1118 packages can be upgraded. Run 'apt list --upgradable' to see them.
```

Installs ufw (if not present)
sudo apt install ufw -y

```
┌──(kali㉿kali)-[~]
└─$ sudo apt install ufw -y
ufw is already the newest version (0.36.2-9).
Summary:
  Upgrading: 0, Installing: 0, Removing: 0, Not Upgrading: 1118
```

Enable UFW
sudo ufw enable

```
┌──(kali㉿kali)-[~]
└─$ sudo ufw enable
Firewall is active and enabled on system startup
```

Check Current Status and Rules
sudo ufw status verbose

```
┌──(kali㉿kali)-[~]
└─$ sudo ufw status verbose
Status: active
Logging: on (low)
Default: deny (incoming), allow (outgoing), disabled (routed)
New profiles: skip
```

🧱Part 3: Block All Incoming Except SSH

Set Default Deny Policy
sudo ufw default deny incoming
sudo ufw default allow outgoing

```
┌──(kali㉿kali)-[~]
└─$ sudo ufw default deny incoming
Default incoming policy changed to 'deny'
(be sure to update your rules accordingly)

┌──(kali㉿kali)-[~]
└─$ sudo ufw default allow outgoing
Default outgoing policy changed to 'allow'
(be sure to update your rules accordingly)
```

Allow SSH
sudo ufw allow ssh

```
┌──(kali㉿kali)-[~]
└─$ sudo ufw allow ssh
Rule added
Rule added (v6)
```

Check the rules that were added
sudo ufw status verbose

```
┌──(kali㉿kali)-[~]
└─$ sudo ufw status verbose
Status: active
Logging: on (low)
Default: deny (incoming), allow (outgoing), disabled (routed)
New profiles: skip

To                         Action      From
--                         ------      ----
22/tcp                     ALLOW IN    Anywhere
22/tcp (v6)                ALLOW IN    Anywhere (v6)
```

🔄Part 4: Test Firewall Blocking from Client

Client (Parrot) pings to Server (Kali)
ping 172.16.123.129

After many moments of waiting, it still appears the request to be blocked

```
┌─[user@parrot]─[~]
└──╼ $ping 172.16.123.129
PING 172.16.123.129 (172.16.123.129) 56(84) bytes of data.
```

Try to SSH into Server (should work)
This should succeed because port 22 was allowed
ssh kali@172.16.123.129

```
┌──[user@parrot]─[~]
└─$ ssh kali@172.16.123.129
The authenticity of host '172.16.123.129 (172.16.123.129)' can't be established.
ED25519 key fingerprint is SHA256:DAZoZL/NurKT+1qjNEc0uAnCPLGq42jvwMvWruRB+QQ.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '172.16.123.129' (ED25519) to the list of known hosts
.
kali@172.16.123.129's password:
Linux kali 6.12.13-amd64 #1 SMP PREEMPT_DYNAMIC Kali 6.12.13-1kali1 (2025-02-11)
 x86_64

The programs included with the Kali GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Kali GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Mon Jun 30 21:54:48 2025 from 172.16.123.129
┌──(kali㉿kali)-[~]
└─$
```

🎯 Part 5: Allow HTTP (Simulated Web Server Test)

Install Apache2 on Server

sudo apt install apache2 -y

```
┌──(kali㉿kali)-[~]
└─$ sudo apt install apache2 -y
apache2 is already the newest version (2.4.64-1).
Upgrading:
  e2fsprogs   keyutils      libcurl3t64-gnutls  libhogweed6t64  libsasl2-modules  libxml2-utils
  gnutls-bin  krb5-locales  libext2fs2t64       libldap-common  libss2            logsave

Summary:
  Upgrading: 12, Installing: 0, Removing: 0, Not Upgrading: 1106
  Download size: 0 B / 2,627 kB
  Space needed: 53.2 kB / 718 MB available
```

sudo systemctl start apache2

```
┌──(kali㉿kali)-[~]
└─$ sudo systemctl start apache2
[sudo] password for kali:
```

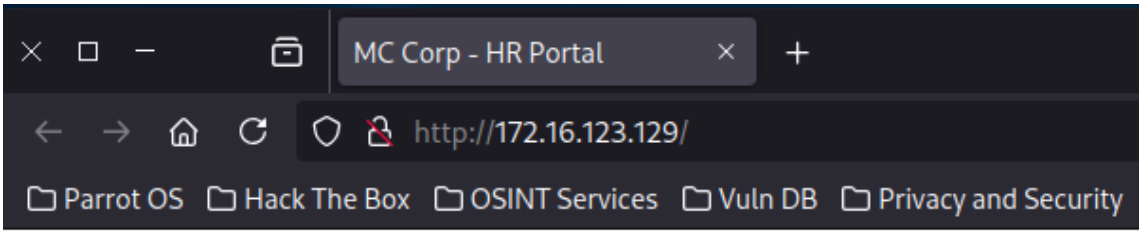
Check if service is running
sudo systemctl status apache2

```
┌──(kali㉿kali)-[~]
└─$ sudo systemctl status apache2
● apache2.service - The Apache HTTP Server
     Loaded: loaded (/usr/lib/systemd/system/apache2.service; enabled; preset: disabled)
     Active: active (running) since Tue 2025-07-29 23:30:09 EDT; 2h 4min ago
 Invocation: 18cd895c842642539ce3868942974f94
       Docs: https://httpd.apache.org/docs/2.4/
   Main PID: 334353 (apache2)
      Tasks: 6 (limit: 4502)
     Memory: 14.2M (peak: 25.9M)
        CPU: 1.289s
     CGroup: /system.slice/apache2.service
             ├─334353 /usr/sbin/apache2 -k start
             ├─376595 /usr/sbin/apache2 -k start
             ├─376596 /usr/sbin/apache2 -k start
             ├─376597 /usr/sbin/apache2 -k start
             ├─376598 /usr/sbin/apache2 -k start
             └─376599 /usr/sbin/apache2 -k start

Jul 29 23:30:09 kali systemd[1]: Starting apache2.service - The Apache HTTP Server ...
Jul 29 23:30:09 kali systemd[1]: Started apache2.service - The Apache HTTP Server.
Jul 30 00:15:21 kali systemd[1]: Reloading apache2.service - The Apache HTTP Server ...
Jul 30 00:15:21 kali systemd[1]: Reloaded apache2.service - The Apache HTTP Server.
```

Allow HTTP
sudo ufw allow http

```
┌──(kali㉿kali)-[~]
└─$ sudo ufw allow http
Rule added
Rule added (v6)
```

Access Web Server from Client
http://172.16.123.129/



Welcome to MC Corp HR Portal

This is a secure internal HR site.

🔍 Part 6: View and Delete Rules

View Rules with Numbers
sudo ufw status numbered

```
┌──(kali㉿kali)-[~]
└─$ sudo ufw status numbered
[sudo] password for kali:
Status: active

     To                         Action      From
     --                         ------      ----
[ 1] 22/tcp                     ALLOW IN    Anywhere
[ 2] 80/tcp                     ALLOW IN    Anywhere
[ 3] 22/tcp (v6)                ALLOW IN    Anywhere (v6)
[ 4] 80/tcp (v6)                ALLOW IN    Anywhere (v6)
```

Delete a Rule
removes http access
sudo ufw delete allow http

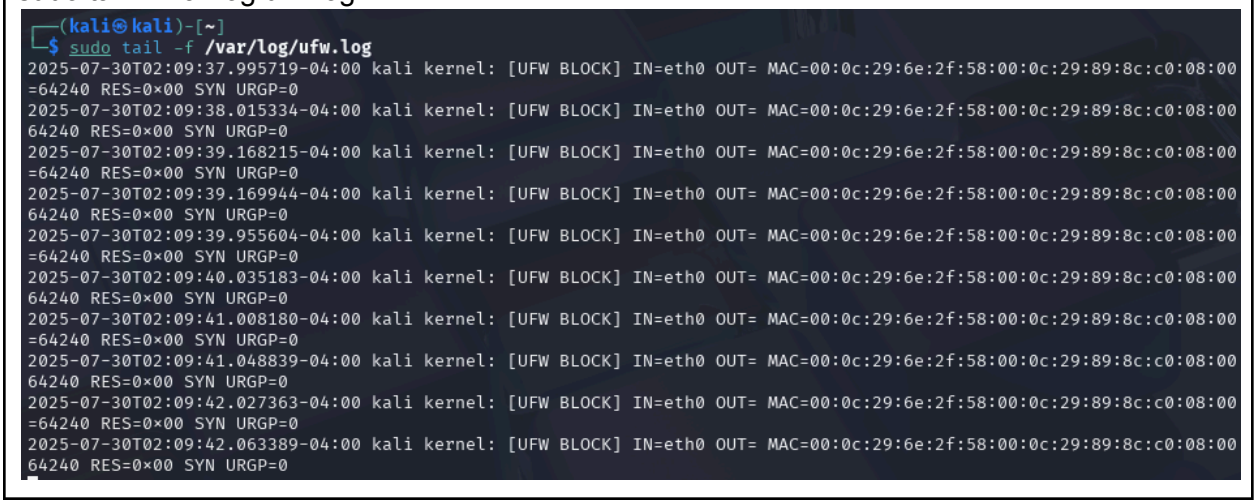
```
┌──(kali㉿kali)-[~]
└─$ sudo ufw delete allow http
Rule deleted
Rule deleted (v6)

┌──(kali㉿kali)-[~]
└─$ sudo ufw status
Status: active

To                         Action      From
--                         ------      ----
22/tcp                     ALLOW       Anywhere
22/tcp (v6)                ALLOW       Anywhere (v6)
```

💡 Bonus: Enable Logging

sudo ufw logging on

```
┌──(kali㉿kali)-[~]
└─$ sudo ufw logging on
Logging enabled
```

sudo tail -f /var/log/ufw.log



🧰 Tools & Skills Used

- **Operating Systems:** Kali Linux (Server), Parrot OS (Client)
- **Firewall Utility:** ufw (Uncomplicated Firewall)
- **Networking Tools:** ping (ICMP), ssh (TCP/22), ip a
- **System & Service Management:** sudo, apt, systemctl, tail
- **Core Skills:** Host-Based Firewall Configuration, Network Security, Rule Management, Port-Based Filtering (TCP), Protocol-Based Filtering (ICMP), Service Installation (Apache2), Log Monitoring

🧠 Reflection & Takeaways

This lab was a practical and effective demonstration of the "deny by default" security principle, which is a cornerstone of network security. By configuring the firewall to block all incoming traffic first and then explicitly allowing only necessary services, I was able to create a secure and controlled environment.

My key takeaways from this lab are:

1. **The Importance of a Default Deny Policy:** The most critical step was setting `ufw default deny incoming`. This immediately hardens the server by ensuring that no unintended or malicious traffic can get through. It forces a deliberate and conscious decision for every service that needs to be exposed.
2. **Firewalls are Protocol-Specific:** A key lesson was seeing the `ping` (ICMP protocol) fail while the `ssh` (TCP port 22) and web server (TCP port 80) connections succeeded. This clearly illustrates that firewall rules are not a blanket "on or off" switch; they provide

granular control over specific ports and protocols, allowing an administrator to create precise security policies.

3. **Reduced Attack Surface:** By only opening ports 22 and 80, I significantly reduced the server's attack surface. An attacker running a port scan against this machine would only see these two services, leaving other potential vulnerabilities on different ports hidden and inaccessible. This is a fundamental step in preventing unauthorized access and reconnaissance.

Lab#08: Web Server Defacement – Incident Response Simulation Lab

🎯 Objective

Students will:

- Investigate a defaced website hosted locally on Apache.
- Identify and remove the malicious file.
- Restore the original website.
- Patch and harden the system post-incident.

🔧 Step-by-Step Instructions with Explanations

🔹 **Step 1: Install and Configure Apache Web Server**

🔹 **Step 2: Deploy the Original Website**

🔹 **Step 3: Simulate a Defacement Attack**

🔍 **Step 4: Investigate the Incident**

🧹 **Step 5: Remove the Malicious File & Restore the Original**

🔐 **Step 6: Patch & Harden the System**

Step 1: Install and Configure Apache Web Server

sudo apt update

```
┌──(kali㉿kali)-[~]
└─$ sudo apt update
[sudo] password for kali:
Get:1 http://kali.download/kali kali-rolling InRelease [41.5 kB]
Get:2 http://kali.download/kali kali-rolling/main amd64 Packages [21.0 MB]
Ign:2 http://kali.download/kali kali-rolling/main amd64 Packages
Get:3 http://kali.download/kali kali-rolling/main amd64 Contents (deb) [51.4 MB]
Get:4 http://kali.download/kali kali-rolling/contrib amd64 Packages [117 kB]
Get:5 http://kali.download/kali kali-rolling/contrib amd64 Contents (deb) [327 kB]
Get:6 http://kali.download/kali kali-rolling/non-free amd64 Packages [198 kB]
Get:7 http://kali.download/kali kali-rolling/non-free amd64 Contents (deb) [911 kB]
Get:8 http://kali.download/kali kali-rolling/non-free-firmware amd64 Packages [10.8 kB]
Get:9 http://kali.download/kali kali-rolling/non-free-firmware amd64 Contents (deb) [26.7 kB]
Get:2 http://kali.download/kali kali-rolling/main amd64 Packages [21.0 MB]
Fetched 72.6 MB in 1min 6s (1,101 kB/s)
1328 packages can be upgraded. Run 'apt list --upgradable' to see them.
```

sudo apt install apache2

```
┌──(kali㉿kali)-[~]
└─$ sudo apt install apache2
Upgrading:
  apache2  apache2-bin  apache2-data  apache2-utils

Summary:
  Upgrading: 4, Installing: 0, Removing: 0, Not Upgrading: 1324
  Download size: 1,998 kB
  Space needed: 11.3 kB / 3,432 MB available
```

After installation, check if the service is running:
sudo systemctl status apache2

```
┌──(kali㉿kali)-[~]
└─$ sudo systemctl status apache2
● apache2.service - The Apache HTTP Server
     Loaded: loaded (/usr/lib/systemd/system/apache2.service; enabled; preset: disabled)
     Active: active (running) since Tue 2025-07-29 23:30:09 EDT; 1min 9s ago
 Invocation: 18cd895c842642539ce3868942974f94
       Docs: https://httpd.apache.org/docs/2.4/
   Main PID: 334353 (apache2)
      Tasks: 6 (limit: 4502)
     Memory: 13.6M (peak: 13.8M)
        CPU: 82ms
     CGroup: /system.slice/apache2.service
             ├─334353 /usr/sbin/apache2 -k start
             ├─334356 /usr/sbin/apache2 -k start
             ├─334357 /usr/sbin/apache2 -k start
             ├─334358 /usr/sbin/apache2 -k start
             ├─334359 /usr/sbin/apache2 -k start
             └─334360 /usr/sbin/apache2 -k start

Jul 29 23:30:09 kali systemd[1]: Starting apache2.service - The Apache HTTP Server ...
Jul 29 23:30:09 kali systemd[1]: Started apache2.service - The Apache HTTP Server.
```

Step 2: Deploy the Original Website

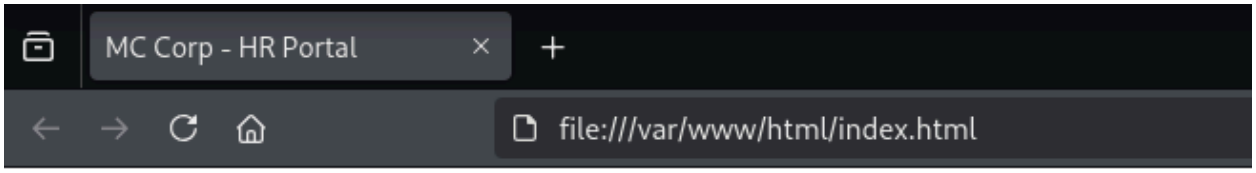
Navigate to web root:
cd /var/www/html
sudo rm index.html

Creating the new web page
Explanation: This simulates a legitimate site hosted on an internal web server.
sudo nano index.html

```
┌──(kali㉿kali)-[~]
└─$ cd /var/www/html

┌──(kali㉿kali)-[/var/www/html]
└─$ sudo rm index.html

┌──(kali㉿kali)-[/var/www/html]
└─$ sudo nano index.html
```

Creating the new web page

```
GNU nano 8.3
<!DOCTYPE html>
<html>
<head><title>MC Corp - HR Portal</title></head>
<body>
<h1>Welcome to MC Corp HR Portal</h1>
<p>This is a secure internal HR site.</p>
</body>
</html>
```



Welcome to MC Corp HR Portal

This is a secure internal HR site.

Step 3: Simulate a Defacement Attack

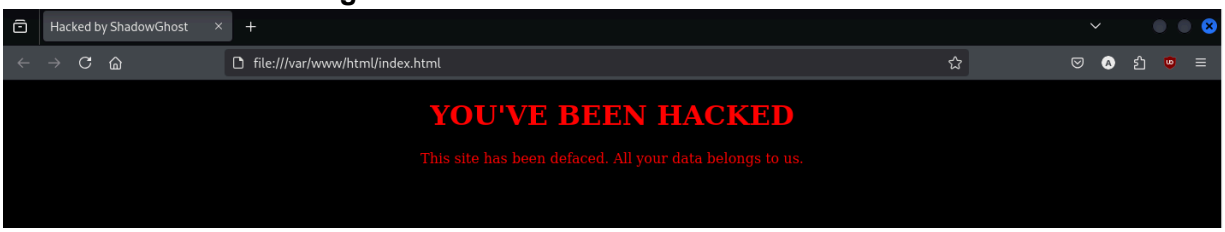
Replace the page with defaced content:
sudo mv index.html index_backup.html
sudo nano index.html

```
┌──(kali㉿kali)-[/var/www/html]
└─$ sudo mv index.html index_backup.html

┌──(kali㉿kali)-[/var/www/html]
└─$ ls
index_backup.html  index.nginx-debian.html

┌──(kali㉿kali)-[/var/www/html]
└─$ sudo nano index.html
```

The content being replaced:

```
GNU nano 8.3
<!DOCTYPE html>
<html>
<head><title>Hacked by ShadowGhost</title></head>
<body style="background-color:black; color:red; text-align:center;">
<h1>YOU'VE BEEN HACKED</h1>
<p>This site has been defaced. All your data belongs to us.</p>
</body>
</html>
```

The result after the change:



Step 4: Investigate the Incident

Check the bash history
history | grep index.html

```
┌──(kali㉿kali)-[/var/www/html]
└─$ history | grep index.html
  141  nano index.html
  306  sudo nano /var/www/html/index.html
  490  sudo rm index.html
  491  sudo nano index.html
  492  sudo mv index.html index_backup.html
  494  sudo nano index.html
```

Check the Apache logs
sudo cat /var/log/apache2/access.log | tail -n 50

```
┌──(kali㉿kali)-[/var/www/html]
└─$ sudo cat /var/log/apache2/access.log | tail -n 50
172.16.123.129 - - [29/Jul/2025:19:10:52 -0400] "GET / HTTP/1.1" 200 1609 "-" "Mozi
172.16.123.129 - - [29/Jul/2025:19:10:52 -0400] "POST / HTTP/1.1" 200 1609 "-" "Moz
172.16.123.129 - - [29/Jul/2025:19:10:52 -0400] "OPTIONS / HTTP/1.1" 200 181 "-" "M
172.16.123.129 - - [29/Jul/2025:19:10:52 -0400] "GET /robots.txt HTTP/1.1" 404 456
172.16.123.129 - - [29/Jul/2025:19:10:52 -0400] "OPTIONS / HTTP/1.1" 200 181 "-" "M
172.16.123.129 - - [29/Jul/2025:19:10:52 -0400] "PROPFIND / HTTP/1.1" 405 524 "-" "
172.16.123.129 - - [29/Jul/2025:19:10:52 -0400] "GET /.git/HEAD HTTP/1.1" 404 456 "
172.16.123.129 - - [29/Jul/2025:19:10:52 -0400] "PROPFIND / HTTP/1.1" 405 524 "-" "
172.16.123.129 - - [29/Jul/2025:19:10:52 -0400] "GET /nmaplowercheck1753830652 HTTP
e.html)"
```

Use diff to compare original vs defaced:
sudo diff index_backup.html index.html

```
┌──(kali㉿kali)-[/var/www/html]
└─$ sudo diff index_backup.html index.html
3,6c3,6
< <head><title>MC Corp - HR Portal</title></head>
< <body>
< <h1>Welcome to MC Corp HR Portal</h1>
< <p>This is a secure internal HR site.</p>
---
> <head><title>Hacked by ShadowGhost</title></head>
> <body style="background-color:black; color:red; text-align:center;">
> <h1>YOU'VE BEEN HACKED</h1>
> <p>This site has been defaced. All your data belongs to us.</p>
8a9
>
```

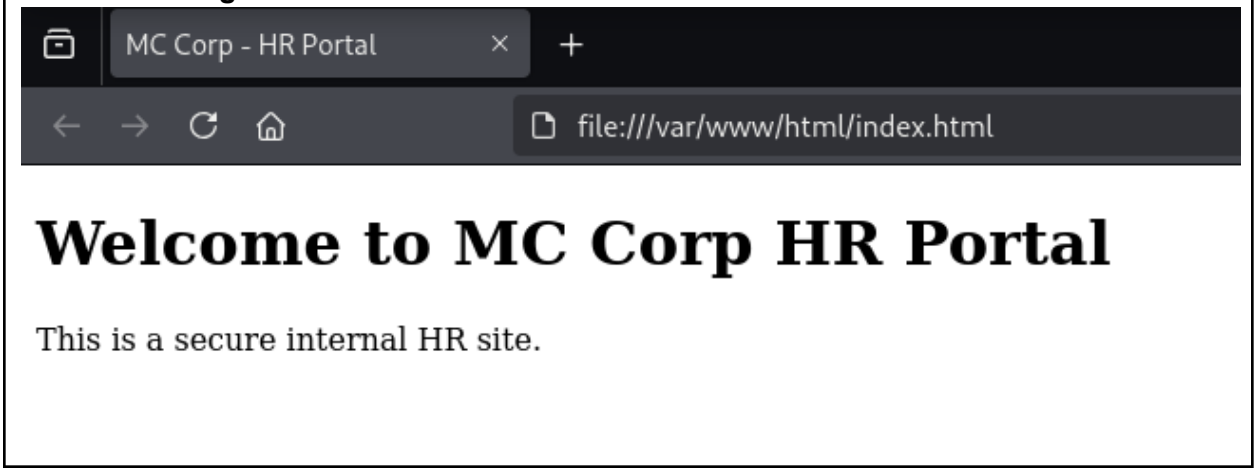
Step 5: Remove the Malicious File & Restore the Original

sudo mv index_backup.html index.html

```
┌──(kali㉿kali)-[/var/www/html]
└─$ sudo mv index_backup.html index.html
```

Explanation: This brings the site back to the legitimate version.

After refreshing the browser:



Step 6: Patch & Harden the System

Update all packages
sudo apt update && sudo apt upgrade -y

```
┌──(kali㉿kali)-[/var/www/html]
└─$ sudo apt update && sudo apt upgrade -y
Hit:1 http://http.kali.org/kali kali-rolling InRelease
1324 packages can be upgraded. Run 'apt list --upgradable' to see them.
The following packages were automatically installed and are no longer required:
  icu-devtools   libglapi-mesa  libpython3.12-minimal  python3-aioconsole     python
  libflac12t64   libicu-dev     libpython3.12-stdlib   python3-dunamai        python
  libfuse3-3     liblbfgsb0     libpython3.12t64       python3-nfsclient      python
  libgeos3.13.0  libpoppler145  libutempter0           python3-packaging-whl  python
Use 'sudo apt autoremove' to remove them.

Upgrading:
  7zip                              libcaja-extension1              libqt5webengin
  adduser                           libcamel-1.2-64t64              libqt5webengin
```

Restrict Apache file permissions
sudo chown -R root:root /var/www/html
sudo chmod -R 755 /var/www/html
ls -l /var/www/html

```
┌──(kali㉿kali)-[~]
└─$ sudo chown -R root:root /var/www/html
[sudo] password for kali:

┌──(kali㉿kali)-[~]
└─$ sudo chmod -R 755 /var/www/html

┌──(kali㉿kali)-[~]
└─$ ls -l /var/www/html
total 8
-rwxr-xr-x 1 root root 174 Jul 29 23:37 index.html
```

Enable and configure the UFW firewall
sudo apt install ufw -y

```
┌──(kali㉿kali)-[/var/www/html]
└─$ sudo apt install ufw -y
ufw is already the newest version (0.36.2-9).
The following packages were automatically installed and are no longer required:
  icu-devtools  libicu-dev  liblbfgsb0  python3.12-tk
Use 'sudo apt autoremove' to remove them.
```

sudo ufw allow ‘Apache Full’

```
┌──(kali㉿kali)-[/var/www/html]
└─$ sudo ufw allow 'Apache Full'
Rule added
Rule added (v6)
```

sudo ufw enable

```
┌──(kali㉿kali)-[/var/www/html]
└─$ sudo ufw enable
Firewall is active and enabled on system startup
```

Install fail2ban to block brute-force attempts
sudo apt install fail2ban -y

```
┌──(kali㉿kali)-[/var/www/html]
└─$ sudo apt install fail2ban -y
The following packages were automatically installed and are no longer required:
  icu-devtools  libicu-dev  liblbfgsb0  python3.12-tk
Use 'sudo apt autoremove' to remove them.

Upgrading:
  libcap2-bin

Installing:
  fail2ban

Installing dependencies:
  python3-systemd
```

🧰 Tools & Skills Used

Tools used

- OS: Kali Linux or Ubuntu (or any Debian-based Linux VM)
- Packages: apache2
- Tools: grep, diff, history, log files

🧠 Reflection & Takeaways

This incident response simulation was a valuable, hands-on exercise that reinforced several critical cybersecurity principles. It demonstrated the full lifecycle of an incident, from initial investigation to final system hardening.

My key takeaways from this lab are:

1. **The Power of Foundational Tools:** A successful investigation doesn't always require complex forensic software. In this scenario, fundamental Linux command-line tools like history, cat, grep, and diff were sufficient to effectively investigate the incident, identify

the malicious changes, and confirm the root cause. This highlights the importance of mastering the basics.

2. **Logging is the Cornerstone of Incident Response:** The investigation would have been nearly impossible without access to the bash_history and Apache access.log files. This lab was a clear reminder that without comprehensive and accessible logs, a security analyst has no visibility into what occurred on a system, making effective incident response incredibly difficult.
3. **Recovery is Only as Good as Your Preparation:** The restoration of the website was simple and fast *only because* a backup of the original index.html file existed. This emphasizes that an effective recovery strategy depends entirely on proactive preparation, such as having a robust and regularly tested data backup plan.
4. **Security Doesn't End at Remediation:** Simply restoring the file is not enough. The final hardening steps—updating all packages, restricting file permissions, and enabling a firewall with fail2ban—are what truly secure the system against future attacks. This demonstrates the critical importance of moving from a reactive to a proactive security posture after an incident.

🦠 Lab#09: Malware Containment and Eradication

🎯 Objective

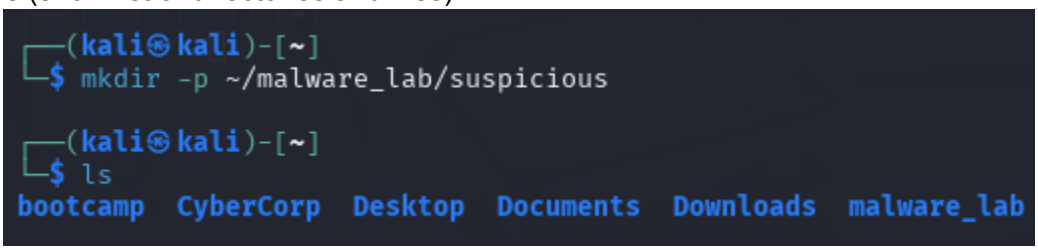
Simulate a detected malware file on a Linux system, isolate the threat, hash and preserve evidence, remove the file safely, and verify system integrity.

Step-by-Step Instructions / Summary

- **Step 1: Create a Simulated Malware File**
- **Step 2: Identify the Suspicious File**
- **Step 3: Isolate the File (Containment)**
- **Step 4: Hash the File (Preserve Evidence)**
- **Step 5: Compress & Archive File for Reporting**
- **Step 6: Remove the Malware Safely**
- **Step 7: Log Your Actions**
- **Step 8: Verify System Integrity (Basic)**

Step 1: Create a Simulated Malware File

Using this command to simulate a "malicious" file:
mkdir -p ~/malware_lab/suspicious
ls (show list of directories and files)



Creating the malicious file
echo "This file simulates a trojan downloader." > ~/malware_lab/suspicious/update_patch.bin

```
┌──(kali㉿kali)-[~]
└─$ echo "This file simulates a trojan downloader." > ~/malware_lab/suspicious/update_patch.bin

┌──(kali㉿kali)-[~]
└─$ cd malware_lab

┌──(kali㉿kali)-[~/malware_lab]
└─$ ls
suspicious
```

```
┌──(kali㉿kali)-[~/malware_lab/suspicious]
└─$ ls
update_patch.bin
```

Step 2: Identify the Suspicious File

Checking the type of file is created
file ~/malware_lab/suspicious/update_patch.bin

```
┌──(kali㉿kali)-[~]
└─$ file ~/malware_lab/suspicious/update_patch.bin
/home/kali/malware_lab/suspicious/update_patch.bin: ASCII text
```

Checks the file's metadata: permissions, creation data, and ownership

```
┌──(kali㉿kali)-[~]
└─$ stat ~/malware_lab/suspicious/update_patch.bin
  File: /home/kali/malware_lab/suspicious/update_patch.bin
  Size: 41              Blocks: 8          IO Block: 4096   regular file
Device: 8,1     Inode: 914180      Links: 1
Access: (0664/-rw-rw-r--)  Uid: ( 1000/    kali)   Gid: ( 1000/    kali)
Access: 2025-07-29 22:06:52.495926255 -0400
Modify: 2025-07-29 22:02:34.849695445 -0400
Change: 2025-07-29 22:02:34.849695445 -0400
 Birth: 2025-07-29 22:02:34.849695445 -0400
```

Step 3: Isolate the File (Containment)

Make a directory to move the malicious file to isolate from other folders
sudo mkdir -p /quarantine

```
┌──(kali㉿kali)-[~]
└─$ sudo mkdir -p /quarantine

┌──(kali㉿kali)-[~]
└─$ ls /
```

```
proc  quarantine
```

Moving the malicious file to the quarantine directory
sudo mv ~/malware_lab/suspicious/update_patch.bin /quarantine/

```
┌──(kali㉿kali)-[~]
└─$ cd /quarantine

┌──(kali㉿kali)-[/quarantine]
└─$ ls
update_patch.bin
```

Step 4: Hash the File (Preserve Evidence)

Hashing the file to preserve the evidence:
This commands creates a file, the "tee" makes it so the user writes as root, the additional redirect ">" and "/dev/null" suppresses the duplicates output.

sudo sha256sum update_patch.bin | sudo tee hash_update_patch.txt > /dev/null

```
┌──(kali㉿kali)-[/quarantine]
└─$ sudo sha256sum update_patch.bin | sudo tee hash_update_patch.txt > /dev/null
```

Check if the hash file is created:
ls -l hash_update_patch.txt
cat hash_update_patch.txt

```
┌──(kali㉿kali)-[/quarantine]
└─$ sudo sha256sum update_patch.bin | sudo tee hash_update_patch.txt > /dev/null

┌──(kali㉿kali)-[/quarantine]
└─$ ls -l  hash_update_patch.txt
-rw-r--r-- 1 root root 83 Jul 29 22:27 hash_update_patch.txt

┌──(kali㉿kali)-[/quarantine]
└─$ cat hash_update_patch.txt
fe86b6b629c09b44c98e1e95626521abff2b39cd19644b0726f721aa2b8eda8a  update_patch.bin
```

Step 5: Compress & Archive File for Reporting

This command packages the malware sample and the hash together for reporting to a threat intel team or malware lab.
sudo tar -czvf update_patch_quarantined.tar.gz update_patch.bin hash_update_patch.txt

```
┌──(kali㉿kali)-[/quarantine]
└─$ sudo tar -czvf update_patch_quarantined.tar.gz update_patch.bin hash_update_patch.txt
update_patch.bin
hash_update_patch.txt

┌──(kali㉿kali)-[/quarantine]
└─$ ls
hash_update_patch.txt  update_patch.bin  update_patch_quarantined.tar.gz
```

Step 6: Remove the Malware Safely

WIth this command it overwrites and deletes the file, making recovery difficult. A safe malware removal option.

shred -u update_patch.bin

```
┌──(kali㉿kali)-[/quarantine]
└─$ sudo shred -u update_patch.bin

┌──(kali㉿kali)-[/quarantine]
└─$ ls
hash_update_patch.txt  update_patch_quarantined.tar
```

Step 7: Log Your Actions

Creating a log file of actions that happened
nano ~/malware_lab/response_log.txt

```
┌──(kali㉿kali)-[/quarantine]
└─$ nano ~/malware_lab/response_log.txt
```

```
File  Actions  Edit  View  Help
  GNU nano 8.3
- Suspicious file discovered: update_patch.bin
- Moved to /quarantine
- SHA-256 hash generated
- Archived file and hash
- Malware safely deleted using shred
```

```
┌──(kali㉿kali)-[/quarantine]
└─$ cat  ~/malware_lab/response_log.txt
- Suspicious file discovered: update_patch.bin
- Moved to /quarantine
- SHA-256 hash generated
- Archived file and hash
- Malware safely deleted using shred
```

Step 8: Verify System Integrity (Basic)

Scan home directory for malicious files:
find ~ -name "*.sh"

```
┌──(kali㉿kali)-[/quarantine]
└─$ find ~ -name "*.sh"

┌──(kali㉿kali)-[/quarantine]
└─$
```

Checks for any hidden files
ls -la ~ | grep "^\."

```
┌──(kali㉿kali)-[/quarantine]
└─$ ls -la ~ | grep "^\."

┌──(kali㉿kali)-[/quarantine]
└─$
```

Installing and running an Antivirus
sudo apt install clamav -y

```
┌──(kali㉿kali)-[/quarantine]
└─$ sudo apt install clamav -y
Installing:
  clamav

Installing dependencies:
  clamav-base  clamav-freshclam  libclamav12

Suggested packages:
  libclamunrar  clamav-doc  libclamunrar11

Summary:
  Upgrading: 0, Installing: 4, Removing: 0, Not Upgrading: 1291
  Download size: 15.1 MB
  Space needed: 69.5 MB / 3,737 MB available
```

sudo clamscan -r ~/

```
┌──(kali㉿kali)-[/quarantine]
└─$ sudo clamscan -r ~/
LibClamAV Error: cli_loaddbdir: No supported database files found in /var/lib/clamav
ERROR: Can't open file or directory

----------- SCAN SUMMARY -----------
Known viruses: 0
Engine version: 1.4.2
Scanned directories: 0
Scanned files: 0
Infected files: 0
Data scanned: 0.00 MB
Data read: 0.00 MB (ratio 0.00:1)
Time: 0.005 sec (0 m 0 s)
Start Date: 2025:07:29 23:00:15
End Date:   2025:07:29 23:00:15
```

🧰 Tools & Skills Used

Tools used:

- Kali Linux or Ubuntu VM
- Terminal access with sudo rights
- Tools used: sha256sum, file, stat, ls, rm, tar

Skills used

- Containment
- Evidence preservation (hashing, logging)
- File system analysis
- Safe malware removal

🧠 Reflection & Takeaways

This lab taught me the importance of containment and evidence preservation. The first step is not to delete the malware first, but to move it to a safe "quarantine" directory to stop the potential harm of the malware spreading and isolating from important files. I later use the process of hashing and logging. The process of hashing (sha256sum) is used to document the unique digital footprint to ensure its integrity of the file. Afterwards, the "shred -u” command is used which differs from the usual “rm” command by making recovery nearly impossible and emphasizes the principle of “secure eradication.” This concludes my lab.

Lab#10:Apache2 vs Nginx + Full Demos

🎯 Objectives:

Setting up Apache2 and Nginx servers with the addition of a few demos.

Step-by-Step Instructions / Summary

Demo 1: Apache2 Setup.

1. Install Apache2
2. Start Apache2
3. Open Web Browser
4. Replace Default Page

Demo 2: Nginx Setup

1. Install Nginx
2. Start Nginx
3. Open Web Browser
4. Replace Default Page

🧠 1. Introduction: What Is a Web Server?

Definition:
A web server is a program that delivers content (like HTML pages, images, etc.) to users over the web.

🏛️ 2. What Is Apache2?

Feature	Apache2
Release Year	1995 (Apache Software Foundation)
Architecture	**Process-driven**: forks a new process per request
Config Files	/etc/apache2/apache2.conf, .htaccess

PHP Support	Built-in via mod_php
Use Case	Flexible hosting with dynamic modules
Default Port	80

🧩 **Use Apache2 when**:

You want fine-grained control with .htaccess
You need built-in PHP support
You expect moderate traffic

🚀 3. What Is Nginx?

Feature	Nginx (Engine-X)
Release Year	2004 (Igor Sysoev)
Architecture	**Event-driven**: handles thousands of connections
Config Files	/etc/nginx/nginx.conf, /etc/nginx/sites-available/
PHP Support	Requires PHP-FPM
Use Case	High-performance static & proxy server
Default Port	80

🧩 **Use Nginx when**:

You expect **high concurrency** traffic
You need **reverse proxy** or load balancing
You prioritize **speed** and **efficiency**

⚔️ 4. Side-by-Side Visual Comparison

Feature	Apache2	Nginx
Architecture	Process-based	Event-based
Performance (Static)	Slower	Faster
.htaccess Support	✅ Yes	❌ No
Reverse Proxy	✅ Basic support	✅ Excellent built-in
PHP Integration	mod_php	PHP-FPM
Configuration	More granular, verbose	Simpler, centralized

🧪 5. DEMO 1: Apache2

1. 📦Install Apache2

sudo apt update
sudo apt install apache2 -y

```
┌──(kali㉿kali)-[~]
└─$ sudo apt update
[sudo] password for kali:
Get:1 http://kali.download/kali kali-rolling InRelease [41.5 kB]
Get:2 http://kali.download/kali kali-rolling/main amd64 Packages [21.0 MB]
Get:3 http://kali.download/kali kali-rolling/main amd64 Contents (deb) [51.4
MB]
Get:4 http://kali.download/kali kali-rolling/contrib amd64 Packages [120 kB]
Get:5 http://kali.download/kali kali-rolling/contrib amd64 Contents (deb) [32
7 kB]
Get:6 http://kali.download/kali kali-rolling/non-free amd64 Packages [197 kB]
Get:7 http://kali.download/kali kali-rolling/non-free amd64 Contents (deb) [9
11 kB]
Get:8 http://kali.download/kali kali-rolling/non-free-firmware amd64 Packages
 [10.6 kB]
Get:9 http://kali.download/kali kali-rolling/non-free-firmware amd64 Contents
 (deb) [26.4 kB]
Fetched 74.0 MB in 49s (1,525 kB/s)
1301 packages can be upgraded. Run 'apt list --upgradable' to see them.
```

```
┌──(kali㉿kali)-[~]
└─$ sudo apt install apache2 -y
apache2 is already the newest version (2.4.63-1).
apache2 set to manually installed.
Summary:
  Upgrading: 0, Installing: 0, Removing: 0, Not Upgrading: 1301
```

2. ▶️Start Apache:

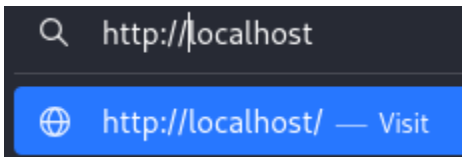
sudo systemctl start apache2
sudo systemctl enable apache2

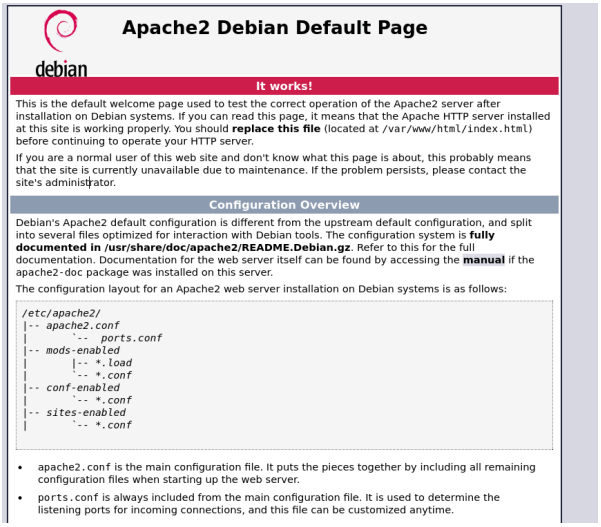
```
┌──(kali㉿kali)-[~]
└─$ sudo systemctl start apache2

┌──(kali㉿kali)-[~]
└─$ sudo systemctl enable apache2
Synchronizing state of apache2.service with SysV service script with /usr/lib
/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable apache2
Created symlink '/etc/systemd/system/multi-user.target.wants/apache2.service'
 → '/usr/lib/systemd/system/apache2.service'.
```

*Optional checking the status of apache2
sudo systemctl status apache2

```
┌──(kali㉿kali)-[~]
└─$ sudo systemctl status apache2
● apache2.service - The Apache HTTP Server
     Loaded: loaded (/usr/lib/systemd/system/apache2.service; enabled; prese>
     Active: active (running) since Tue 2025-06-24 19:17:17 EDT; 4min 6s ago
 Invocation: 7e268186a124486d80ec720aa560c015
       Docs: https://httpd.apache.org/docs/2.4/
   Main PID: 450562 (apache2)
      Tasks: 6 (limit: 4502)
     Memory: 21.2M (peak: 21.2M)
        CPU: 670ms
     CGroup: /system.slice/apache2.service
             ├─450562 /usr/sbin/apache2 -k start
             ├─450565 /usr/sbin/apache2 -k start
             ├─450566 /usr/sbin/apache2 -k start
             ├─450567 /usr/sbin/apache2 -k start
             ├─450568 /usr/sbin/apache2 -k start
             └─450569 /usr/sbin/apache2 -k start
```

3. 🌐Open Web Browser:

Visit: http://localhost or http://<your-ip> Expected output: 🟢 **"Welcome to Nginx!"**	 After entering the url it'll send this page

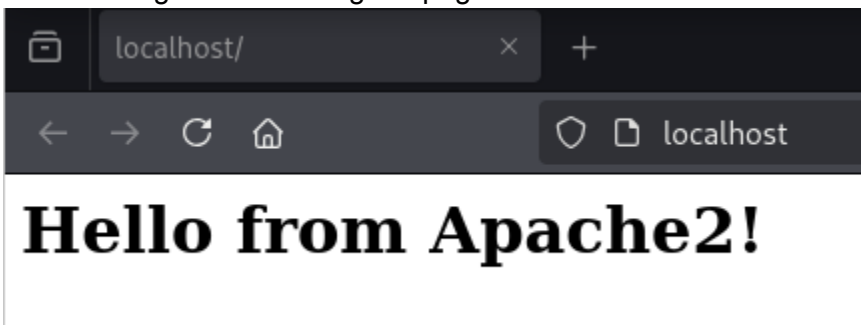


4. Replace Default Page:

```
echo "<h1>Hello from Nginx!</h1>" | sudo tee
/var/www/html/index.nginx-debian.html
```

```
┌──(kali㉿kali)-[~]
└─$ sudo echo "<h1>Hello from Apache2! </h1>" | sudo tee /var/www/html/index.html
<h1>Hello from Apache2! </h1>
```

After revising and refreshing the page:



🧪 6. DEMO 2: Nginx

1. Install Nginx

```
sudo apt update
```

```
┌──(kali㉿kali)-[~]
└─$ sudo apt update
[sudo] password for kali:
Get:1 http://kali.download/kali kali-rolling InRelease [41.5 kB]
Get:2 http://kali.download/kali kali-rolling/main amd64 Packages [21.0 MB]
Get:3 http://kali.download/kali kali-rolling/main amd64 Contents (deb) [51.4 MB]
Get:4 http://kali.download/kali kali-rolling/contrib amd64 Packages [120 kB]
Get:5 http://kali.download/kali kali-rolling/contrib amd64 Contents (deb) [327 kB]
Get:6 http://kali.download/kali kali-rolling/non-free amd64 Packages [197 kB]
Get:7 http://kali.download/kali kali-rolling/non-free amd64 Contents (deb) [911 kB]
Get:8 http://kali.download/kali kali-rolling/non-free-firmware amd64 Packages [10.6
kB]
Fetched 74.0 MB in 8s (9,446 kB/s)
1304 packages can be upgraded. Run 'apt list --upgradable' to see them.
```

sudo install nginx -y

```
┌──(kali㉿kali)-[~]
└─$ sudo apt install nginx -y
Upgrading:
  nginx  nginx-common

Summary:
  Upgrading: 2, Installing: 0, Removing: 0, Not Upgrading: 1302
  Download size: 718 kB
  Space needed: 0 B / 3,759 MB available

Get:1 http://kali.download/kali kali-rolling/main amd64 nginx amd64 1.26.3-3 [609 kB
]
Get:2 http://mirror.math.princeton.edu/pub/kali kali-rolling/main amd64 nginx-common
 all 1.26.3-3 [109 kB]
Fetched 718 kB in 0s (2,021 kB/s)
Preconfiguring packages ...
(Reading database ... 408017 files and directories currently installed.)
Preparing to unpack .../nginx_1.26.3-3_amd64.deb ...
Unpacking nginx (1.26.3-3) over (1.26.3-2) ...
Preparing to unpack .../nginx-common_1.26.3-3_all.deb ...
Unpacking nginx-common (1.26.3-3) over (1.26.3-2) ...
Setting up nginx-common (1.26.3-3) ...
nginx.service is a disabled or a static unit not running, not starting it.
Setting up nginx (1.26.3-3) ...
Not attempting to start NGINX, port 80 is already in use.
Processing triggers for kali-menu (2025.1.1) ...
Processing triggers for man-db (2.13.0-1) ...
```

2. Start Nginx

sudo systemctl stop apache2

```
┌──(kali㉿kali)-[~]
└─$ sudo systemctl stop apache2
```

sudo systemctl status apache2

```
┌──(kali㉿kali)-[~]
└─$ sudo systemctl status apache2
○ apache2.service - The Apache HTTP Server
```

```
Loaded: loaded (/usr/lib/systemd/system/apache2.service; enabled; preset: disabled)
```

Now, the following would work:

sudo systemctl start nginx

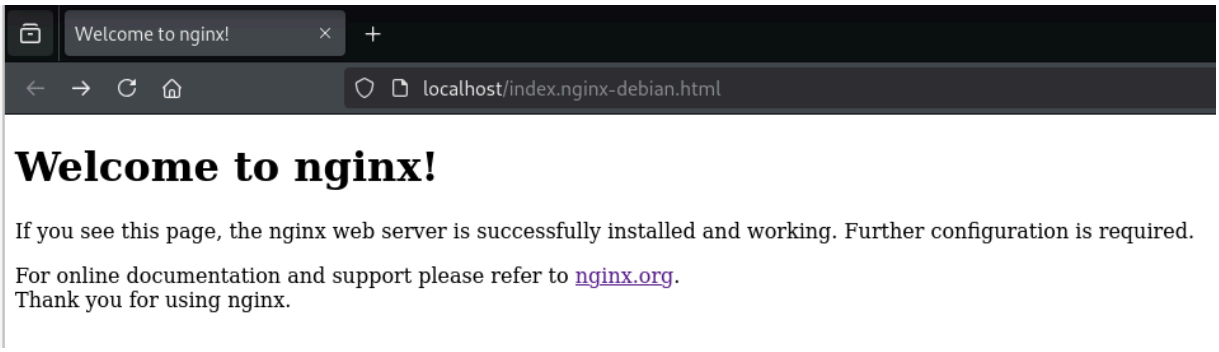
```
┌──(kali㉿kali)-[~]
└─$ sudo systemctl start nginx
```

sudo systemctl enable nginx

```
┌──(kali㉿kali)-[~]
└─$ sudo systemctl enable nginx
Synchronizing state of nginx.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable nginx
Created symlink '/etc/systemd/system/multi-user.target.wants/nginx.service' → '/usr/lib/systemd/system/nginx.service'.
```

3. Open Web Browser

http://localhost/index.nginx-debian.html

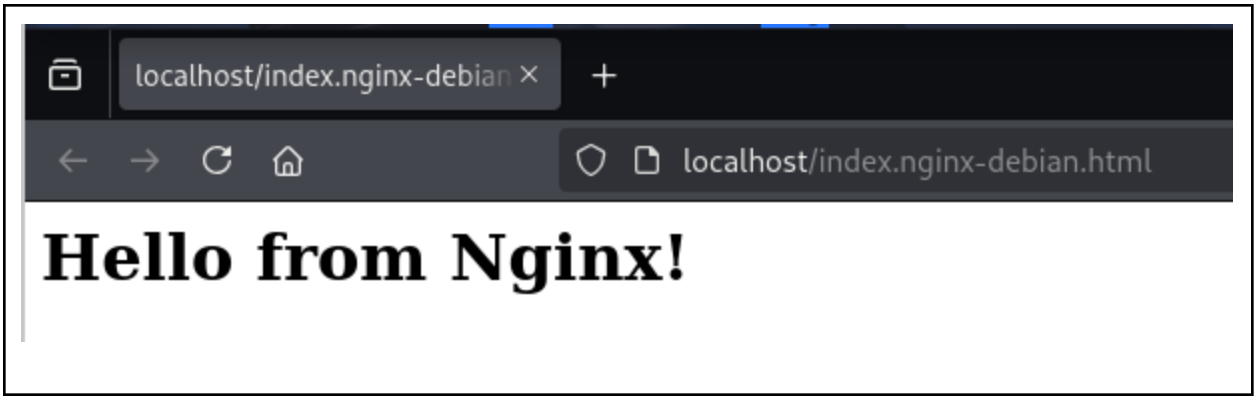


4. Replace Default Page

```
echo "<h1>Hello from Nginx!</h1>" | sudo tee
/var/www/html/index.nginx-debian.html
```

```
┌──(kali㉿kali)-[/etc/nginx]
└─$ echo "<h1>Hello from Nginx! </h1>" | sudo tee /var/www/html/index.nginx-debian.html
<h1>Hello from Nginx! </h1>
```

http://localhost/index.nginx-debian.html



5. 🔄 Switching Between Apache2 and Nginx (Same Port)

a. Run Apache2

sudo systemctl stop nginx

```
┌──(kali㉿kali)-[~]
└─$ sudo systemctl stop nginx
```

sudo systemctl start apache2

```
┌──(kali㉿kali)-[~]
└─$ sudo systemctl start apache2
```

sudo systemctl status apache2 (Optional)

```
┌──(kali㉿kali)-[~]
└─$ sudo systemctl status apache2
● apache2.service - The Apache HTTP Server
     Loaded: loaded (/usr/lib/systemd/system/apache2.service; disabled; preset: disabled)
     Active: active (running) since Wed 2025-06-25 20:27:19 EDT; 6s ago
```

b. Run Nginx

sudo systemctl stop apache2

```
┌──(kali㉿kali)-[~]
└─$ sudo systemctl stop apache2
```

sudo systemctl start nginx

```
┌──(kali㉿kali)-[~]
└─$ sudo systemctl start nginx
```

sudo systemctl status nginx (Optional)

```
┌──(kali㉿kali)-[~]
└─$ sudo systemctl status nginx
● nginx.service - A high performance web server and a reverse proxy server
     Loaded: loaded (/usr/lib/systemd/system/nginx.service; disabled; preset: disabled)
     Active: active (running) since Wed 2025-06-25 19:25:16 EDT; 10s ago
```

👩‍🏫 Classroom Activity Ideas

Installs both Apache and Nginx

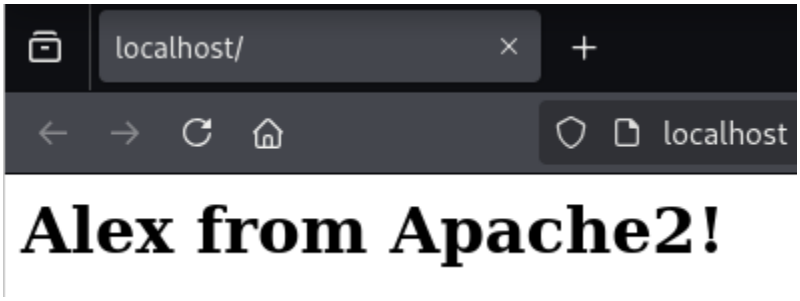
```
┌──(kali㉿kali)-[~]
└─$ sudo apt install apache2 -y
apache2 is already the newest version (2.4.63-1).
apache2 set to manually installed.
Summary:
  Upgrading: 0, Installing: 0, Removing: 0, Not Upgrading: 1301
```

```
┌──(kali㉿kali)-[~]
└─$ sudo apt install nginx -y
Upgrading:
  nginx  nginx-common

Summary:
  Upgrading: 2, Installing: 0, Removing: 0, Not Upgrading: 1302
  Download size: 718 kB
  Space needed: 0 B / 3,759 MB available

Get:1 http://kali.download/kali kali-rolling/main amd64 nginx amd64 1.26.3-3 [609 kB
]
Get:2 http://mirror.math.princeton.edu/pub/kali kali-rolling/main amd64 nginx-common
 all 1.26.3-3 [109 kB]
Fetched 718 kB in 0s (2,021 kB/s)
Preconfiguring packages ...
(Reading database ... 408017 files and directories currently installed.)
Preparing to unpack .../nginx_1.26.3-3_amd64.deb ...
Unpacking nginx (1.26.3-3) over (1.26.3-2) ...
Preparing to unpack .../nginx-common_1.26.3-3_all.deb ...
Unpacking nginx-common (1.26.3-3) over (1.26.3-2) ...
Setting up nginx-common (1.26.3-3) ...
nginx.service is a disabled or a static unit not running, not starting it.
Setting up nginx (1.26.3-3) ...
Not attempting to start NGINX, port 80 is already in use.
Processing triggers for kali-menu (2025.1.1) ...
Processing triggers for man-db (2.13.0-1) ...
```

Replaces default pages with their name

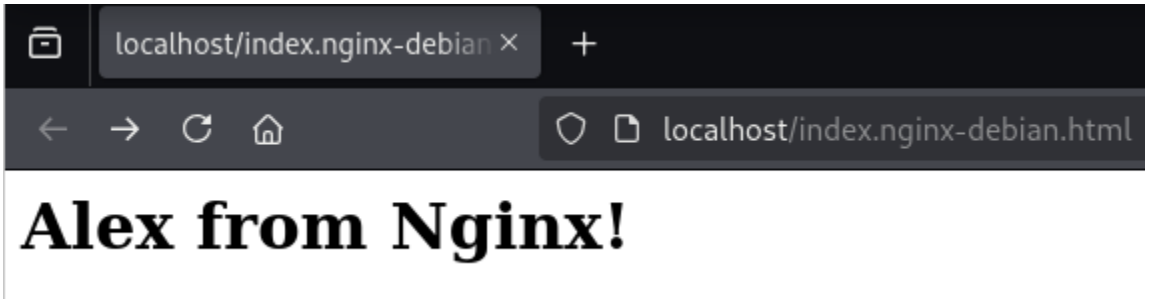
Screenshots for Apache2:

```
┌──(kali㉿kali)-[~]
└─$ echo "<h1>Alex from Apache2! </h1" | sudo tee /var/www/html/index.html

<h1>Alex from Apache2! </h1
```



Screenshots for Nginx:

```
┌──(kali㉿kali)-[/etc/nginx]
└─$ echo "<h1>Alex from Nginx! </h1>" | sudo tee /var/www/html/index.nginx-debian.html

<h1>Alex from Nginx! </h1>
```



Takes screenshots as proof

Compares performance using ab (Apache Benchmark):
sudo apt install apache2-utils

```
┌──(kali㉿kali)-[~]
└─$ sudo apt install apache2-utils

apache2-utils is already the newest version (2.4.63-1).
apache2-utils set to manually installed.
The following packages were automatically installed and are no longer required:
  dsniff  ettercap-common  ettercap-graphical  libapache2-mod-php  libluajit-5.1-2  libluajit-5.1-common  libnids1.21t64  python3-pefile  python3-qrcode
Use 'sudo apt autoremove' to remove them.

Summary:
  Upgrading: 0, Installing: 0, Removing: 0, Not Upgrading: 1300
```

ab -n 100 -c 10 http://localhost/
These commands run 100 requests with a concurrency of 10
to see how each server handles load.

```
┌──(kali㉿kali)-[~]
└─$ ab -n 100 -c 10 http://localhost/

This is ApacheBench, Version 2.3 <$Revision: 1923142 $>
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/
Licensed to The Apache Software Foundation, http://www.apache.org/

Benchmarking localhost (be patient).....done


Server Software:        Apache/2.4.63
Server Hostname:        localhost
Server Port:            80

Document Path:          /
Document Length:        30 bytes

Concurrency Level:      10
Time taken for tests:   0.721 seconds
Complete requests:      100
Failed requests:        0
Total transferred:      27600 bytes
HTML transferred:       3000 bytes
Requests per second:    138.79 [#/sec] (mean)
Time per request:       72.050 [ms] (mean)
Time per request:       7.205 [ms] (mean, across all concurrent requests)
Transfer rate:          37.41 [Kbytes/sec] received

Connection Times (ms)
              min  mean[+/-sd] median   max
Connect:        0    1   2.1      0      13
Processing:     1   12  65.3      3     653
Waiting:        1   10  52.9      2     529
Total:          1   13  65.4      3     653

Percentage of the requests served within a certain time (ms)
  50%      3
  66%      3
  75%      4
  80%      9
  90%     34
  95%     35
  98%     36
  99%    653
 100%    653 (longest request)
```

🧰 Tools & Skills Used

Tools:

- Apache2, Nginx
- tee, systemctl, Apache Benchmark (ab)
- Text Editors (nano)

Skills

- Installing Software
- Web Server Management
- Testing and Verifying Servers
- Configuration Management
- Server Troubleshooting

🧰 Real-Life Application

Apache2 is used by WordPress, older CMSs, shared hosting
Nginx powers Netflix, Dropbox, Instagram (as reverse proxy)

🧠 Reflection & Takeaways

This lab helped me setup both apache2 and nginx servers. I initially had a problem with the index.html when switching the nginx server. It only showed the apache2 server configuration, but I figured it out and used this localhost: http://localhost/index.nginx-debian.html. Afterwards it allowed me to go to the server I needed.

Lab #11: Nmap

Each lab exercise will introduce students to fundamental **Nmap** commands, teaching them how to scan networks, detect open ports, identify services, and gather basic host information. Students must **take a screenshot** of their results and **submit it on Blackboard**.

🎯 Objective

Each lab exercise is designed to introduce students to the fundamentals of Nmap, helping them develop essential network scanning skills.

Step-by-Step Instructions / Summary

By completing this lab, students will:

1. **Understand Network Scanning Basics –** Learn how **Nmap** is used for discovering live hosts, scanning ports, and identifying services.
2. **Perform Host Discovery –** Identify active devices on a network using **ping scans.**
3. **Scan for Open Ports –** Understand how to detect open ports and determine potential entry points.
4. **Perform Targeted Port Scanning –** Learn how to scan specific ports instead of scanning an entire system.
5. **Detect Running Services and Versions –** Use Nmap to determine what services are running on open ports.
6. **Identify Operating Systems –** Use **OS detection techniques** to analyze remote systems.
7. **Execute Stealthy Scans – Perform SYN (stealth)** scans to bypass firewalls and detection systems.
8. **Conduct Aggressive Scans – Utilize Nmap's aggressive scanning mode** to gather detailed host information.
9. **Scan Multiple Targets Simultaneously –** Learn how to scan **multiple IP addresses** efficiently.
10. **Save and Analyze Scan Results –** Store scan data for documentation, reporting, and future analysis.

Steps and screenshots for this lab:

Lab 1: Basic Host Discovery (Ping Scan)

Checking the ip address ip a

```
┌──(kali㉿kali)-[~]
└─$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:6e:2f:58 brd ff:ff:ff:ff:ff:ff
    inet 172.16.123.129/24 brd 172.16.123.255 scope global dynamic noprefixroute eth0
       valid_lft 947sec preferred_lft 947sec
    inet6 fe80::20c:29ff:fe6e:2f58/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
```

Objective: Learn how to identify live hosts on a network.
nmap -sn 172.16.123.129/24

```
┌──(kali㉿kali)-[~]
└─$ nmap -sn 172.16.123.129/24
Starting Nmap 7.95 ( https://nmap.org ) at 2025-07-29 19:05 EDT
Nmap scan report for AlexPC447 (172.16.123.1)
Host is up (0.00034s latency).
MAC Address: 00:50:56:C0:00:08 (VMware)
Nmap scan report for 172.16.123.2
Host is up (0.00031s latency).
MAC Address: 00:50:56:E1:46:7A (VMware)
Nmap scan report for 172.16.123.254
Host is up (0.00020s latency).
MAC Address: 00:50:56:FF:84:F4 (VMware)
Nmap scan report for 172.16.123.129
Host is up.
Nmap done: 256 IP addresses (4 hosts up) scanned in 2.07 seconds
```

Lab 2: Simple Port Scanning -

Objective: Scan a target to find open ports.
nmap 172.16.123.129

```
┌──(kali㉿kali)-[~]
└─$ nmap 172.16.123.129
Starting Nmap 7.95 ( https://nmap.org ) at 2025-07-29 19:04 EDT
Nmap scan report for 172.16.123.129
Host is up (0.0000090s latency).
Not shown: 998 closed tcp ports (reset)
PORT   STATE SERVICE
22/tcp open  ssh
80/tcp open  http

Nmap done: 1 IP address (1 host up) scanned in 0.14 seconds
```

Lab 3: Scanning Specific Ports

Objective: Scan for specific ports instead of scanning all.
Task: Scan for **ports 22 (SSH), 80 (HTTP), and 443 (HTTPS)** and submit a screenshot.

nmap -p 22,80,443 172.16.123.129

```
┌──(kali㉿kali)-[~]
└─$ nmap -p 22,80,443 172.16.123.129
Starting Nmap 7.95 ( https://nmap.org ) at 2025-07-29 19:03 EDT
Nmap scan report for 172.16.123.129
Host is up (0.000047s latency).

PORT    STATE  SERVICE
22/tcp  open   ssh
80/tcp  open   http
443/tcp closed https

Nmap done: 1 IP address (1 host up) scanned in 0.15 seconds
```

Lab 4: Full Port Scan (All 65,535 Ports)

Objective: Perform an exhaustive port scan.
nmap -p- 172.16.123.129

```
┌──(kali㉿kali)-[~]
└─$ nmap -p- 172.16.123.129
Starting Nmap 7.95 ( https://nmap.org ) at 2025-07-29 19:02 EDT
Nmap scan report for 172.16.123.129
Host is up (0.0000060s latency).
Not shown: 65533 closed tcp ports (reset)
PORT   STATE SERVICE
22/tcp open  ssh
80/tcp open  http

Nmap done: 1 IP address (1 host up) scanned in 1.12 seconds
```

Lab 5: Service and Version Detection

Objective: Identify what services are running on open ports.
nmap -sV 172.16.123.129

```
┌──(kali㉿kali)-[~]
└─$ nmap -sV 172.16.123.129
Starting Nmap 7.95 ( https://nmap.org ) at 2025-07-29 19:01 EDT
Nmap scan report for 172.16.123.129
Host is up (0.000019s latency).
Not shown: 998 closed tcp ports (reset)
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 10.0p2 Debian 5 (protocol 2.0)
80/tcp open  http    Apache httpd 2.4.63 ((Debian))
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 20.20 seconds
```

Lab 6: OS Detection

Objective: Determine the operating system of a target.
nmap -O 172.16.123.129

```
┌──(kali㉿kali)-[~]
└─$ nmap -O 172.16.123.129
Starting Nmap 7.95 ( https://nmap.org ) at 2025-07-29 19:05 EDT
Nmap scan report for 172.16.123.129
Host is up (0.0038s latency).
Not shown: 998 closed tcp ports (reset)
PORT   STATE SERVICE
22/tcp open  ssh
80/tcp open  http
Device type: general purpose
Running: Linux 2.6.X|5.X
OS CPE: cpe:/o:linux:linux_kernel:2.6.32 cpe:/o:linux:linux_kernel:5 cpe:/o:linux:linux_kernel:6
OS details: Linux 2.6.32, Linux 5.0 - 6.2
Network Distance: 0 hops

OS detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 1.79 seconds
```

Lab 7: Stealth Scan (SYN Scan)

Objective: Use a stealthy scan to bypass firewalls.
nmap -sS 172.16.123.129

```
┌──(kali㉿kali)-[~]
└─$ nmap -sS 172.16.123.129
Starting Nmap 7.95 ( https://nmap.org ) at 2025-07-29 19:07 EDT
Nmap scan report for 172.16.123.129
Host is up (0.0000070s latency).
Not shown: 998 closed tcp ports (reset)
PORT   STATE SERVICE
22/tcp open  ssh
80/tcp open  http

Nmap done: 1 IP address (1 host up) scanned in 0.15 seconds
```

Lab 8: Aggressive Scan

Objective: Perform an all-in-one aggressive scan.
nmap -A 172.16.123.129

```
┌──(kali㉿kali)-[~]
└─$ nmap -A 172.16.123.129
Starting Nmap 7.95 ( https://nmap.org ) at 2025-07-29 19:08 EDT
Nmap scan report for 172.16.123.129
Host is up (0.00010s latency).
Not shown: 998 closed tcp ports (reset)
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 10.0p2 Debian 5 (protocol 2.0)
80/tcp open  http    Apache httpd 2.4.63 ((Debian))
|_http-title: Apache2 Ubuntu Default Page: It works
|_http-server-header: Apache/2.4.63 (Debian)
Device type: general purpose
Running: Linux 2.6.X|5.X
OS CPE: cpe:/o:linux:linux_kernel:2.6.32 cpe:/o:linux:linux_kernel:5 cpe:/o:linux:linux_kernel:6
OS details: Linux 2.6.32, Linux 5.0 - 6.2
Network Distance: 0 hops
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 9.95 seconds
```

Lab 9: Scanning Multiple Targets

Objective: Scan multiple IP addresses at once.
nmap -A 172.16.123.129 172.16.123.129

```
┌──(kali㉿kali)-[~]
└─$ nmap -A 172.16.123.129 172.16.123.129
Starting Nmap 7.95 ( https://nmap.org ) at 2025-07-29 19:10 EDT
Nmap scan report for 172.16.123.129
Host is up (0.000072s latency).
Not shown: 998 closed tcp ports (reset)
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 10.0p2 Debian 5 (protocol 2.0)
80/tcp open  http    Apache httpd 2.4.63 ((Debian))
|_http-title: Apache2 Ubuntu Default Page: It works
|_http-server-header: Apache/2.4.63 (Debian)
Device type: general purpose
Running: Linux 2.6.X|5.X
OS CPE: cpe:/o:linux:linux_kernel:2.6.32 cpe:/o:linux:linux_kernel:5 cpe:/o:linux:linux_kernel:6
OS details: Linux 2.6.32, Linux 5.0 - 6.2
Network Distance: 0 hops
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

```
Nmap scan report for 172.16.123.129
Host is up (0.000086s latency).
Not shown: 998 closed tcp ports (reset)
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 10.0p2 Debian 5 (protocol 2.0)
80/tcp open  http    Apache httpd 2.4.63 ((Debian))
|_http-title: Apache2 Ubuntu Default Page: It works
|_http-server-header: Apache/2.4.63 (Debian)
Device type: general purpose
Running: Linux 2.6.X|5.X
OS CPE: cpe:/o:linux:linux_kernel:2.6.32 cpe:/o:linux:linux_kernel:5 cpe:/o:linux:linux_kernel:6
OS details: Linux 2.6.32, Linux 5.0 - 6.2
Network Distance: 0 hops
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 2 IP addresses (2 hosts up) scanned in 16.03 seconds
```

Lab 10: Saving Scan Results

Objective: Save scan results for later analysis.
nmap -oN myscan.txt 172.16.123.129

```
┌──(kali㉿kali)-[~]
└─$ nmap -oN myscan.txt 172.16.123.129
Starting Nmap 7.95 ( https://nmap.org ) at 2025-07-29 19:12 EDT
Nmap scan report for 172.16.123.129
Host is up (0.0000070s latency).
Not shown: 998 closed tcp ports (reset)
PORT   STATE SERVICE
22/tcp open  ssh
80/tcp open  http

Nmap done: 1 IP address (1 host up) scanned in 13.11 seconds
```

After the text file is made:

```
┌──(kali㉿kali)-[~]
└─$ cat myscan.txt
# Nmap 7.95 scan initiated Tue Jul 29 19:12:53 2025 as: /usr/lib/nmap/nmap --privileged -oN myscan.txt 172.16.123.129
Nmap scan report for 172.16.123.129
Host is up (0.0000070s latency).
Not shown: 998 closed tcp ports (reset)
PORT   STATE SERVICE
22/tcp open  ssh
80/tcp open  http

# Nmap done at Tue Jul 29 19:13:06 2025 -- 1 IP address (1 host up) scanned in 13.11 seconds
```

🧰 Tools & Skills Used

Primary Tool: Nmap (Network Mapper)
Operating System: Linux Environment (using the terminal/command line)
Core Skills:

- **Host Discovery:** Identifying live hosts on a network segment using ping scans (`-sn`).
- **Port Scanning:** Detecting open TCP/UDP ports using various techniques, including default scans, specific port scans (`-p`), and full port scans (`-p-`).
- **Service & Version Detection:** Enumerating the specific applications and their versions running on open ports (`-sV`).
- **OS Detection:** Fingerprinting the remote operating system using TCP/IP stack analysis (`-O`).
- **Stealth Scanning:** Performing SYN scans (`-sS`) to identify open ports without completing the full TCP three-way handshake, making the scan less detectable.
- **Aggressive Scanning:** Combining multiple advanced techniques (including OS detection, version detection, script scanning, and traceroute) into a single, comprehensive scan (`-A`).
- **Target Selection:** Scanning single hosts, multiple specified hosts, and entire network subnets (CIDR notation).
- **Output Management:** Saving scan results to a text file for documentation and analysis (`-oN`).
- **Basic Networking:** Using the `ip a` command to identify the local machine's IP address and network configuration.

🧠 Reflection & Takeaways

In this lab, I gained hands-on experience with Nmap, a powerful and essential tool for network reconnaissance. I initially ran into a small issue by targeting the wrong IP address. This was a valuable mistake because it forced me to use the ip a command to verify my own machine's network details and correctly identify the target's subnet. This experience underscored the importance of proper reconnaissance and target validation before launching any scan.

Lab#12: Exiftool

In this lab, you will use ExifTool to extract GPS coordinates from a photo taken with your phone. You will then convert the coordinates into decimal format and use Google Maps to locate where the photo was taken. This lab simulates how forensic investigators can track locations using image metadata.

🎯 Objective

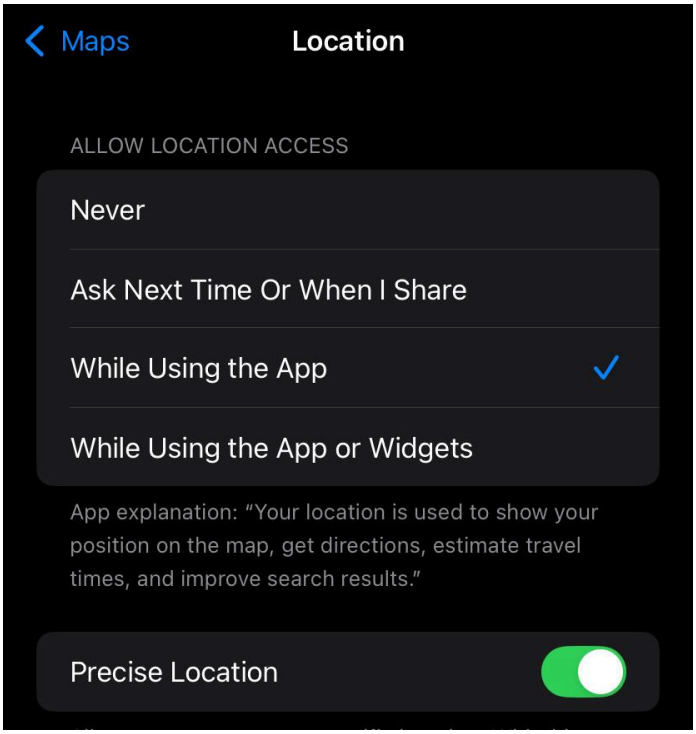
Learning Outcome
By completing this lab, students will:
• Understand how digital photos store forensic evidence.
• Learn how to extract, convert, and map GPS data.
• Be aware of privacy risks associated with photo metadata.

Step-by-Step Instructions / Summary

- Step 1 - Enable Location Services on Your Phone
- Step 2 - Transfer the Photo to Your Computer
- Step 3 - Download and Set Up ExifTool
- Step 4 - Extract Metadata from the Photo
- Step 5 - Convert GPS Coordinates to Decimal Degrees
- Step 6 - Search Location on Google Maps
- Step 7 - (Optional) Remove EXIF Metadata for Privacy Awareness

Step 1 - Enable Location Services on Your Phone

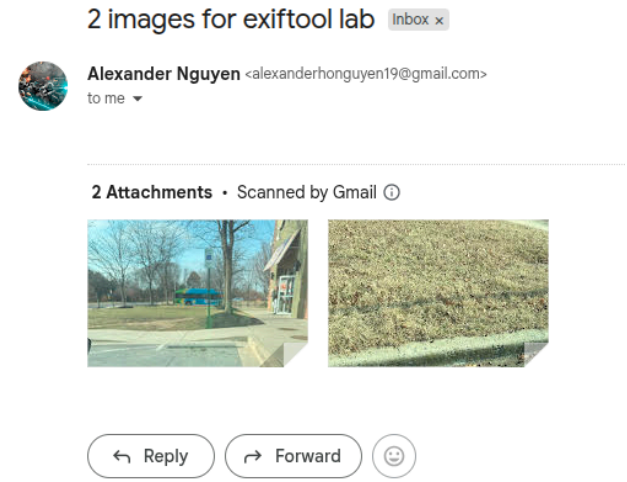
Going to settings to enable location services for the camera.



After photos are taken, pictures are saved

Step 2 - Transfer the Photo to Your Computer

Received photos through email



Saved in a folder


IMG_1438.jpeg


IMG_1440.jpeg

Step 3 - Download and Set Up ExifTool

Setting up and installing from linux terminal
sudo apt-get install libimage-exiftool-perl

```
alexmayretire@AlexPC447:~/Downloads$ sudo apt-get install libimage-exiftool-perl
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
libimage-exiftool-perl is already the newest version (12.76+dfsg-1).
0 upgraded, 0 newly installed, 0 to remove and 8 not upgraded.
```

Step 4 - Extract Metadata from the Photo

Extracting metadata for the first photo
exiftool IMG_1438.jpeg

```
alexmayretire@AlexPC447:~/Pictures$ exiftool IMG_1438.jpeg
ExifTool Version Number         : 12.76
File Name                       : IMG_1438.jpeg
Directory                       : .
File Size                       : 4.8 MB
File Modification Date/Time     : 2025:07:29 20:28:38-04:00
File Access Date/Time           : 2025:07:29 20:53:00-04:00
File Inode Change Date/Time     : 2025:07:29 20:28:38-04:00
File Permissions                : -rw-rw-r--
File Type                       : JPEG
File Type Extension             : jpg
MIME Type                       : image/jpeg
JFIF Version                    : 1.01
Exif Byte Order                 : Big-endian (Motorola, MM)
Make                            : Apple
Camera Model Name               : iPhone 16 Pro Max
Orientation                     : Rotate 90 CW
X Resolution                    : 72
Y Resolution                    : 72
Resolution Unit                 : inches
Software                        : 18.3.1
```

The GPS Latitude and GPS Longitude in the output for this file

```
GPS Altitude                    : 149.8 m Above Sea Level
GPS Date/Time                   : 2025:03:13 20:15:20Z
GPS Latitude                    : 39 deg 11' 2.67" N
GPS Longitude                   : 77 deg 15' 41.47" W
```

Extracting metadata for the second photo
exiftool IMG_1440.jpeg

```
alexmayretire@AlexPC447:~/Pictures$ exiftool IMG_1440.jpeg
ExifTool Version Number         : 12.76
File Name                       : IMG_1440.jpeg
Directory                       : .
File Size                       : 3.8 MB
File Modification Date/Time     : 2025:07:29 20:28:24-04:00
File Access Date/Time           : 2025:07:29 20:52:41-04:00
File Inode Change Date/Time     : 2025:07:29 20:28:24-04:00
File Permissions                : -rw-rw-r--
File Type                       : JPEG
File Type Extension             : jpg
MIME Type                       : image/jpeg
JFIF Version                    : 1.01
Exif Byte Order                 : Big-endian (Motorola, MM)
Make                            : Apple
Camera Model Name               : iPhone 16 Pro Max
Orientation                     : Rotate 90 CW
X Resolution                    : 72
Y Resolution                    : 72
Resolution Unit                 : inches
```

The GPS Latitude and GPS Longitude in the output for this file

```
GPS Altitude                    : 149.2 m Above Sea Level
GPS Date/Time                   : 2025:03:13 20:17:20Z
GPS Latitude                    : 39 deg 10' 58.60" N
GPS Longitude                   : 77 deg 15' 43.67" W
```

Step 5 - Convert GPS Coordinates to Decimal Degrees

Converting the coordinates
Decimal Degrees = Degrees + (Minutes / 60) + (Seconds / 3600)

First image coordinates:

- **Latitude**: **39° 11' 2.67" N**
  Decimal: 39 + (11 / 60) + (2.67 / 3600) = 39 + 0.183333 + 0.0007417 ≈ **39.184075° N**

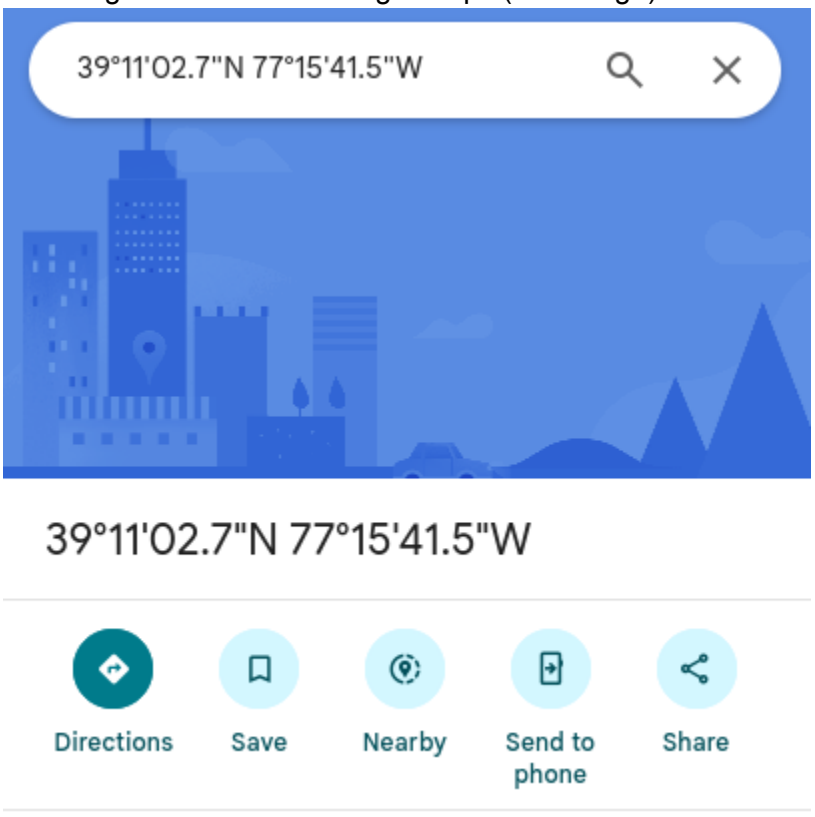
- **Longitude**: 77° 15' 41.47" W
  Decimal: 77 + (15 / 60) + (41.47 / 3600) = 77 + 0.25 + 0.011519 ≈ **77.261519° W**
  (So, **-77.261519°**)

Second image coordinates:

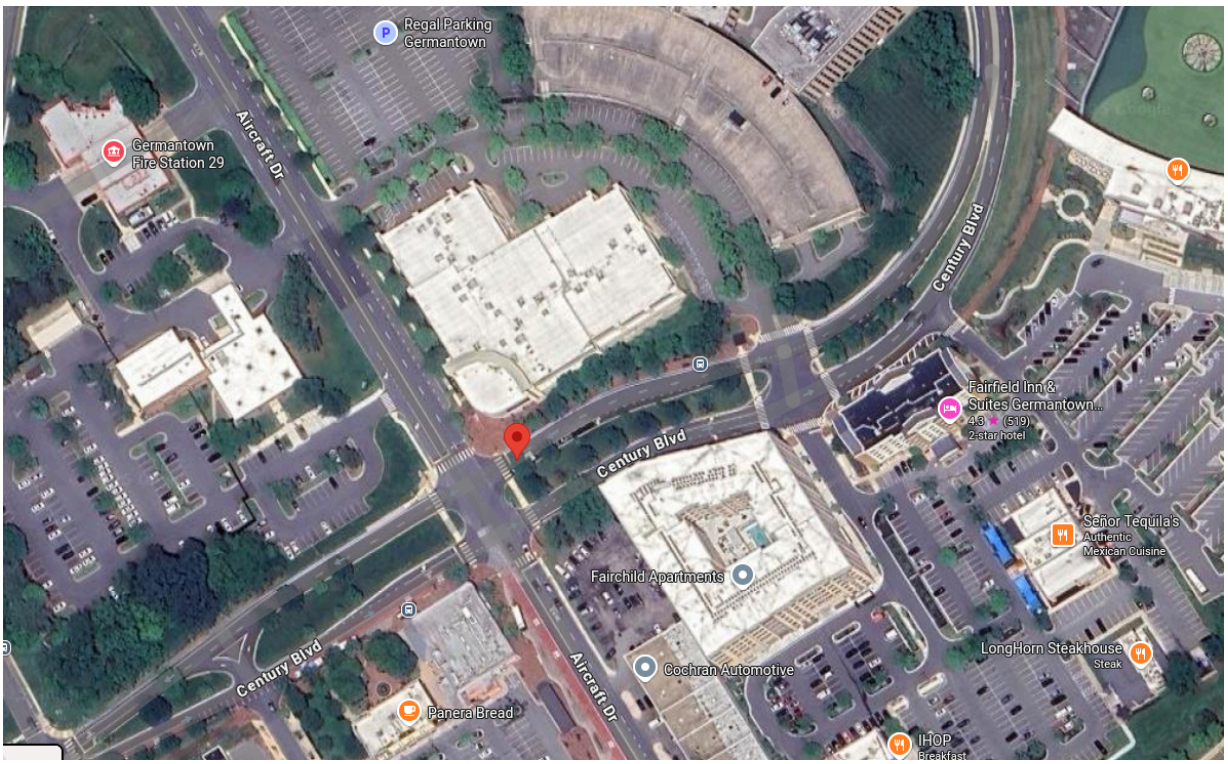
- **Latitude**: **39° 10' 58.60" N**
  Decimal: 39 + (10 / 60) + (58.60 / 3600) = 39 + 0.1666667 + 0.0162778 ≈ **39.182944° N**
- **Longitude**: **77° 15' 43.67" W**
  Decimal: 77 + (15 / 60) + (43.67 / 3600) = 77 + 0.25 + 0.0121306 ≈ **77.262131° W**
- **(So, -77.262131°)**

Step 6 - Search Location on Google Maps

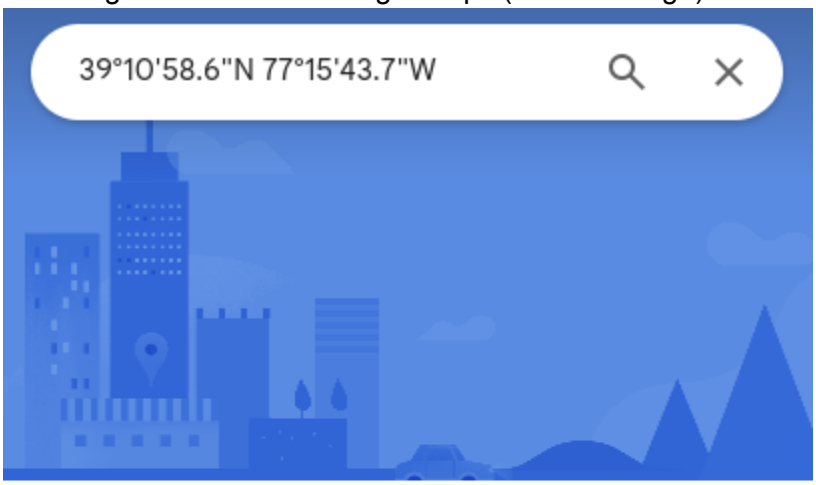
Checking coordinates in Google Maps (first image)



Screenshot for showing the pin where the first image is



Checking coordinates in Google Maps (second image)

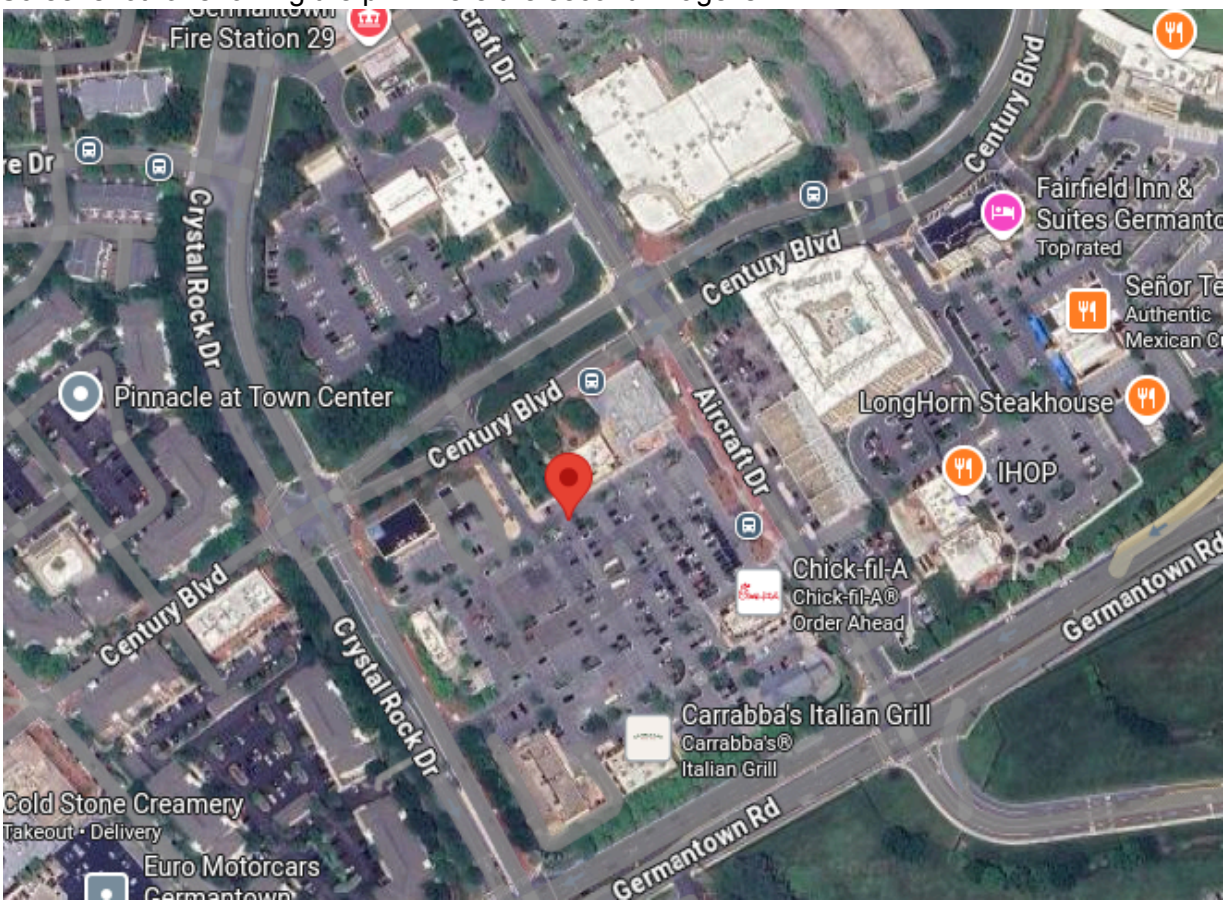


39°10'58.6"N 77°15'43.7"W

39.182944, -77.262131

Directions Save Nearby Send to phone Share

Screenshot for showing the pin where the second image is



Step 7 - (Optional) Remove EXIF Metadata for Privacy Awareness

Remove all the metadata with the exiftool
exiftool -all= IMG_1438.jpeg

Checking if the first image the data is removed
exiftool IMG_1438.jpeg

Remove all the metadata with the exiftool
exiftool -all= IMG_1438.jpeg

Checking if the first image the data is removed
exiftool IMG_1438.jpeg

```
alexmayretire@AlexPC447:~/Pictures$ exiftool -all= IMG_1438.jpeg
Warning: ICC_Profile deleted. Image colors may be affected - IMG_1438.jpeg
    1 image files updated
alexmayretire@AlexPC447:~/Pictures$ exiftool IMG_1438.jpeg
ExifTool Version Number         : 12.76
File Name                       : IMG_1438.jpeg
Directory                       : .
File Size                       : 4.2 MB
File Modification Date/Time     : 2025:07:29 21:34:45-04:00
File Access Date/Time           : 2025:07:29 21:34:47-04:00
File Inode Change Date/Time     : 2025:07:29 21:34:45-04:00
File Permissions                : -rw-rw-r--
File Type                       : JPEG
File Type Extension             : jpg
MIME Type                       : image/jpeg
Image Width                     : 4032
Image Height                    : 3024
Encoding Process                : Baseline DCT, Huffman coding
Bits Per Sample                 : 8
Color Components                : 3
Y Cb Cr Sub Sampling            : YCbCr4:2:0 (2 2)
Image Size                      : 4032x3024
Megapixels                      : 12.2
alexmayretire@AlexPC447:~/Pictures$ exiftool -all= IMG_1440.jpeg
Warning: ICC_Profile deleted. Image colors may be affected - IMG_1440.jpeg
    1 image files updated
alexmayretire@AlexPC447:~/Pictures$ exiftool IMG_1440.jpeg
ExifTool Version Number         : 12.76
File Name                       : IMG_1440.jpeg
Directory                       : .
File Size                       : 3.1 MB
File Modification Date/Time     : 2025:07:29 21:35:21-04:00
File Access Date/Time           : 2025:07:29 21:35:23-04:00
File Inode Change Date/Time     : 2025:07:29 21:35:21-04:00
File Permissions                : -rw-rw-r--
File Type                       : JPEG
File Type Extension             : jpg
MIME Type                       : image/jpeg
Image Width                     : 5712
Image Height                    : 4284
Encoding Process                : Baseline DCT, Huffman coding
Bits Per Sample                 : 8
Color Components                : 3
Y Cb Cr Sub Sampling            : YCbCr4:2:0 (2 2)
Image Size                      : 5712x4284
Megapixels                      : 24.5
```

🧰 Tools & Skills Used

Primary Tool: ExifTool
Supporting Platforms:

- Linux Command Line/Terminal
- Smartphone Camera (with location services)
- Google Maps

Core Skills:

- **Metadata Extraction:** Using exiftool to read and display EXIF (Exchangeable Image File Format) data embedded within a JPEG image.
- **Digital Forensics:** Analyzing metadata to uncover potentially sensitive information, such as GPS coordinates, timestamps, and camera details.
- **Coordinate System Conversion:** Manually converting GPS coordinates from the **DMS (Degrees, Minutes, Seconds)** format to **DD (Decimal Degrees)** using the formula: DD=D+(M/60)+(S/3600).
- **Geospatial Analysis:** Plotting converted GPS coordinates on a mapping service to visualize the precise physical location where a photo was taken.
- **Data Sanitization:** Using exiftool -all= to permanently strip all metadata from a file, a crucial skill for protecting personal privacy.
- **Linux Package Management:** Installing software from the terminal using sudo apt-get install.

🧠 Reflection & Takeaways

This lab helped me use the exiftool to manually extract GPS coordinates and find the exact location being popped up in Google maps. When using this tool, I thought it would help me during my time with digital forensics and the precision of this tool could be unsettling. This leads to the next topic with privacy. I also learned to use extiftool -all= to remove all the metadata from images.

SecGuru National Security Defense Plan

Aldo Leveroni Marion Johnson

Afrika Nyasuma Maxime Gangbe

Alexander Nguyen Derelys Peterson

Cybersecurity Practical Applications

July 12, 2025

Contact/ Team Lead: Aldo Leveroni

Executive Summary

Purpose

Address urgent cyber threats due to rising risks to national infrastructure

Deploy full defense strategy:

1.) Cyber simulations

2.) Incident response plans

3.) Recommendations / Roadmap

The SECGURUS team executed a comprehensive, cross-functional assessment to fortify CyberDome's infrastructure.

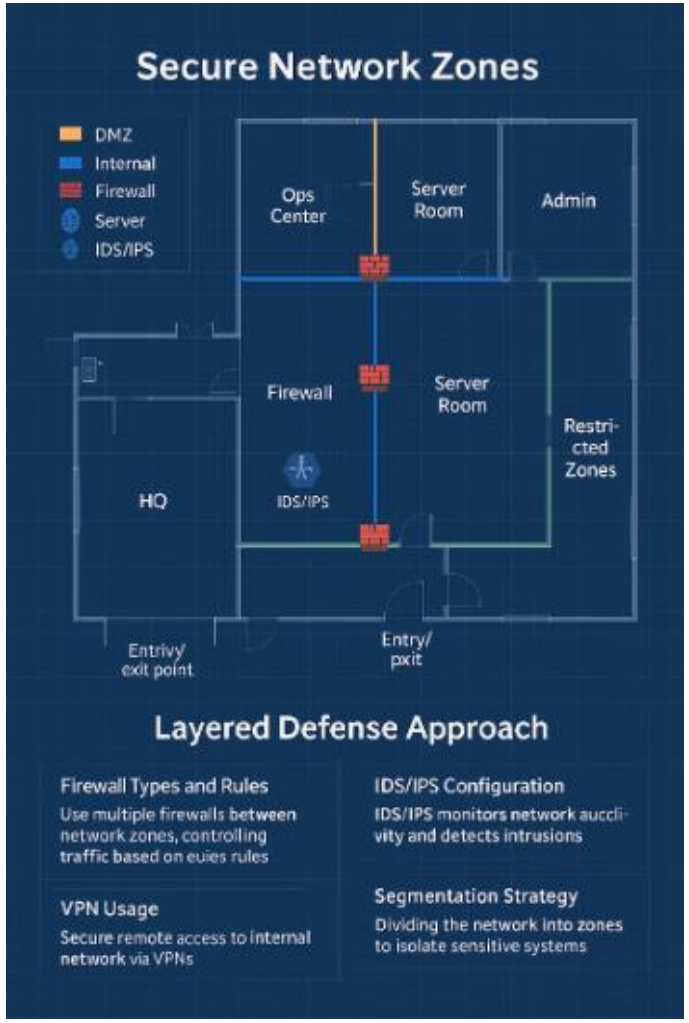
Their efforts included designing a hybrid network topology alongside a layered defense strategy, implementing role-based access control and multi-factor authentication (MFA), and developing Security Information and Event Management (SIEM) monitoring tools and incident response playbooks.

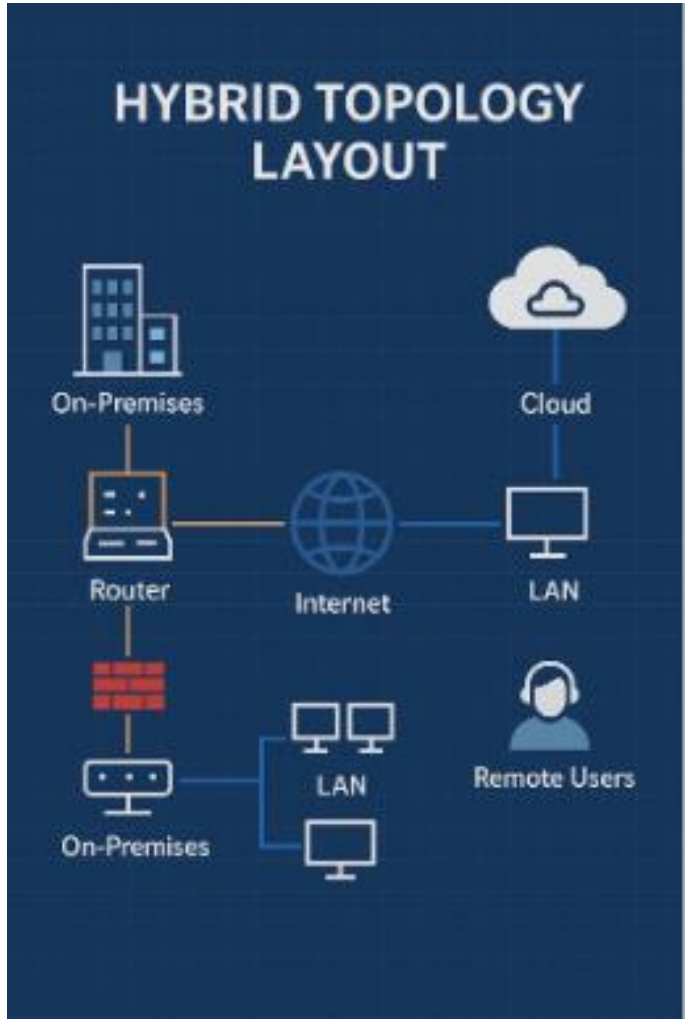
They also established business continuity and disaster recovery plans, deployed physical access controls with insider threat detection capabilities, and conducted Red vs. Blue Team simulations using MITRE techniques.

To enhance situational awareness, they integrated real-time threat intelligence and behavioral analytics into CyberDome's cybersecurity framework.

Topology & Layered Defense Strategy

- SecGuru is a hybrid network architecture combining on-premises systems, cloud infrastructure, and secure remote access. Routers, firewalls, and LAN components support connectivity and enforce perimeter security. Cloud platforms provide scalability, while VPNs ensure secure remote user access.
- The layered defense model segments the network into:
  - **DMZ** for public-facing services
  - **Internal Zone** for business operations
  - **Admin Zone** with restricted access
- Next-gen firewalls, SIEM, and IDS/IPS tools are strategically placed to monitor, detect, and respond to threats. Physical HQ security includes surveillance, entry controls, and biometric authentication. Together, these layers provide strong cyber-physical protection and limit breach impact.






SecGuru's Cybersecurity Plan

Asset Inventory

List of critical systems

- **Email server (Microsoft Exchange)**
- **Web servers, databases**
- **IoT devices**
- **VPN gateway (GlobalProtect, Fortinet)**
- **SIEM system (Splunk/Security Onion/elk slack)**
- **Database server (SQL)**
- **Secure network zones (DMZ)**
- **Entry/exit points for physical layout (HQ blueprint style)**
- **Labeled departments (Ops Center, Server Room, Restricted Zones, LAN, WAN)**
- **Switches and routers**
- **Mobile devices**
- **Generator, UPS**

Network Defense Strategy

SecGurus' defense model applies a layered, proactive approach to protect critical infrastructure. Key components include:

- **Firewalls** – Deployed at all network boundaries (DMZ, internal, admin), enforcing strict rules that block unauthorized traffic and log denied attempts.
- **IDS/IPS** – Intrusion systems detect and block suspicious behavior and threats in real time, helping stop attacks before they spread.
- **VPN Access** – Secure, encrypted tunnels for remote users with 2FA, centralized monitoring, and no split tunneling.
- **Segmentation** – The network is divided into zones (DMZ, Internal, Admin) with firewalls and RBAC to contain breaches and minimize lateral movement.

Together, these measures provide visibility, control, and rapid response to evolving threats.

Monitoring & Logging (SIEM)

splunk>

Logs that are collected:

- **System logs (OS information)**
- **Network logs (Routers, Switches, and other network devices)**
- **Application logs (Apache/Ngnix)**
- **Security logs (Firewall, IDS, IPS, EDR/Anti-virus alerts)**
- **Firewall logs**
- **Email gateway logs**

Anomalies that are monitored:

- **Multiple failed login attempts:** Possible brute-force attack
- **Unusual outbound connections:** Data-exfiltration or command and control (c2) activity
- **Unauthorized file changes:** Potential malware or Insider threat
- **Phishing or malware attachment detected:** Email gateway alert
- **Access to sensitive files outside business hours:** Insider threat or compromised credentials

Analyzed the failed login attempts with Splunk

```
1 source="*failedlogins64.csv"
2 | stats count as "Failed Login Attempts" values(IP) as "IP Addresses" by Username
3 | sort - Attempts
4 | head 5
```

Username	Failed Login Attempts	IP Addresses
ABurke	2	192.168.1.10
ACase	2	192.168.1.2
AMays	2	192.168.1.5
EChan	1	192.168.1.8
EFisher	1	192.168.1.12

Finding the malicious upload with the IP address

```
1 source="*uploadedhashes.csv" IP="192.168.1.10"
2 | table Timestamp IP Filename "File Hash" "User Agent"
```

Timestamp	IP	Filename	File Hash	User Agent
6/4/2023 17:59	192.168.1.10	EvilScript.exe	3AADBF7E527FC1A050E1C97FEA1CBA4D	Opera/75.0.3969.218

Authentication & Access Control

Multi Factor Authentication & Password Policy

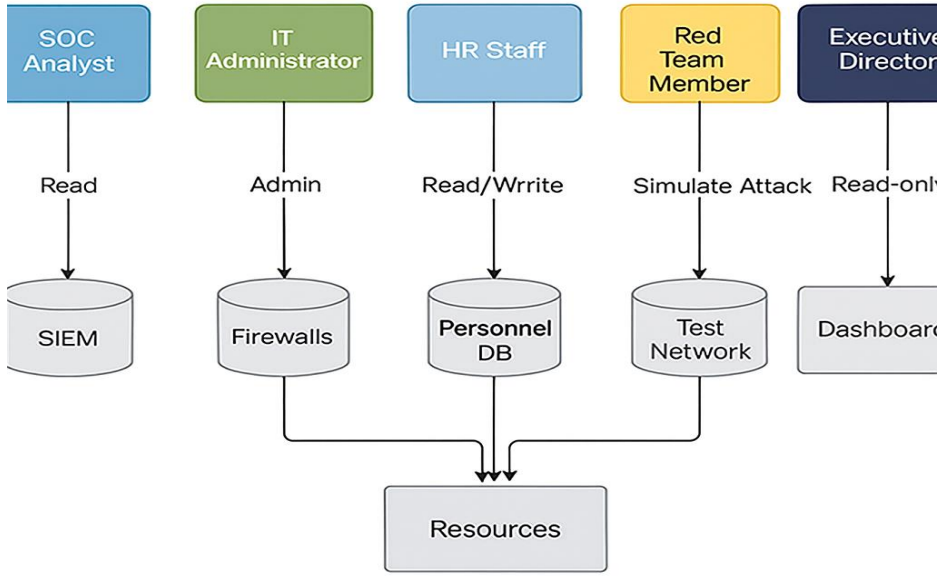
To protect SecGuru's mission-critical assets, a Zero Trust Architecture is enforced, centered around strong identity verification, least privilege, and continuous monitoring.

Password Policy

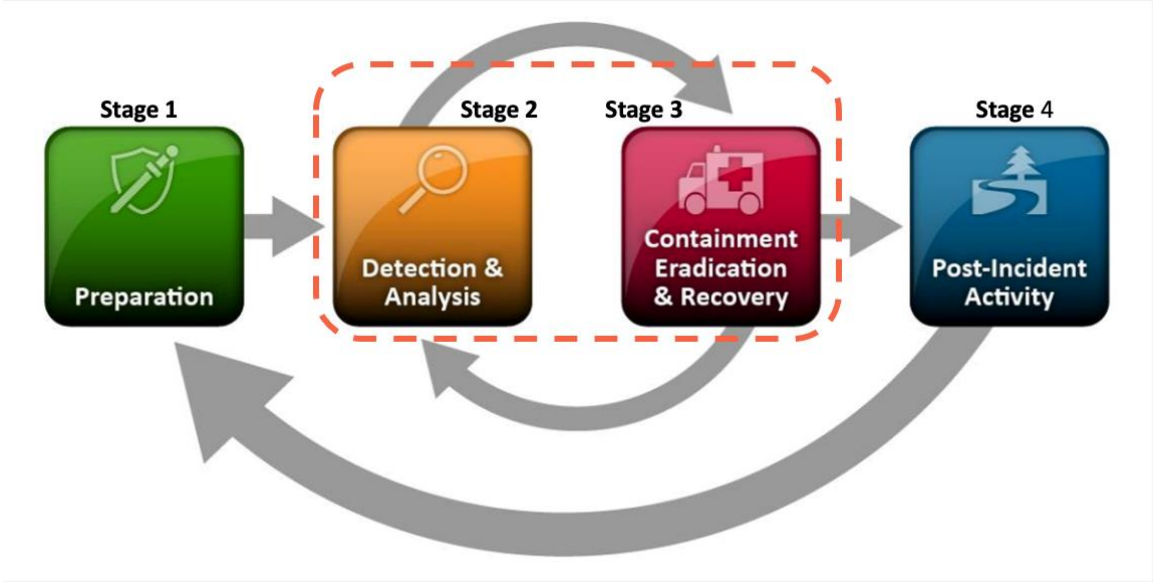
To reduce brute force and credential stuffing attacks, a **strict password policy** is enforced across all domains:

Policy Element	Value
Length	Minimum 16 characters
Complexity	Must include uppercase, lowercase, number, and special character
Expiration	Every 90 days
Reuse Restrictions	Cannot reuse any of the last 10 passwords
Storage	Passwords stored using salted SHA-512 hashes
Account Lockout	Lock account after 5 failed attempts (auto-unlock after 30 min or admin reset)
Passwordless Support	Biometrics + token-based authentication for top-clearance

Role Based Access Control (RBAC)

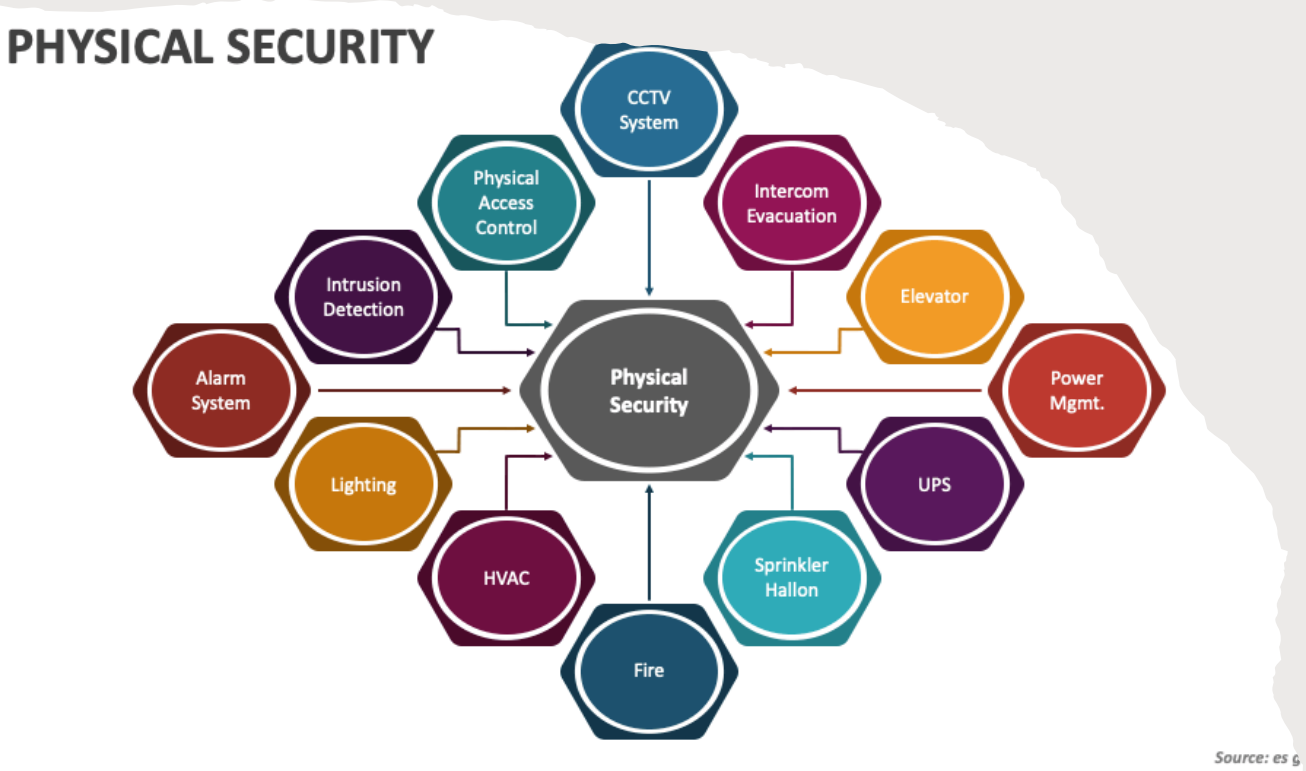


Incident Response Playbook (NIST SP 800-61)



Phrase	Objective	Key Actions (Examples)
Preparation	Build readiness	Policy team, training, tools, communication
Detection & Analysis	Identify & confirm incidents	Monitoring, alerting, analysis, reporting
Containment	Limit impact	Isolation, segmentation, stakeholder communication
Eradication	Remove cause	Malware removal, patching, vulnerability mitigation
Recovery	Restore operation	Backups, validation, monitoring, gradual restoration
Lessons Learned	Continuous improvement	Review, documentation, plan updates, awareness

Physical Security Blueprint



Source: es g

SecGuru's physical security strategy protects people, assets, and infrastructure from threats like intrusion, theft, natural disasters, and sabotage. Security controls are integrated with cybersecurity to ensure operational continuity.

Core measures to access to SecGuru's HQ include:

- **Access Control** – Biometric scans, keycards, and restricted zones
- **Perimeter Defense** – Fencing, barriers, and surveillance coverage
- **Intrusion Detection** – Alarms and systems that trigger alerts in real time
- **Emergency Preparedness** – Fire, disaster, and security breach response planning
- **CPS Protection** – Securing systems that link digital controls to physical processes
- **Insider Threat Monitoring** – Behavioral analysis, role-based access, and activity logging

This convergence of cyber and physical controls ensures defense-in-depth across all facilities.

Insider Threat Mitigation Plan

SecGuru's Insider Threat Analyst plays a crucial role in safeguarding organization, including those involved in National Cyber Defense and Facility Security.

- Threat Detection and Analysis: analyze data, identify threats, and suspicious behavior.
- Investigation and Response: determine root cause and implement appropriate response actions.
- Program Development and Enhancement: develop and recommend improvements based on metrics and reporting.
- Collaboration: here, teams played crucial role among themself to ensure the insider threat program aligns with organizational goals and compliance requirements.
- Reporting: prepare and present analysis and findings to stakeholders, including government leads and managers.

- SecGuru's Insider Threat in these fields combines technical expertise with investigative skills to protect sensitive information, systems, and facilities from threats originating from within the organization. SecGuru's used the following tools: Coralogix, Wazuh, OSSEC, Anodot for real time detection which analyze data in real-time and flag anomalies, which could indicate errors, fraud, or other critical situations. SecGuru's performed regular training to all users.

Threat Intelligence Integration

Sources of Threat Intelligence

- SecGuru leverages open-source threat intelligence sources, tailored for critical infrastructure protection:

Open-Source Threat Intelligence (OSINT) & Community Platforms:

- AlienVault Open Threat Exchange (OTX): A community-powered threat intelligence platform where security professionals share threat data, enabling collaborative research and rapid dissemination of IoCs (IPs, domains, file hashes, URLs). We integrate OTX pulses directly into our SIEM and other security tools.
- MISP (Malware Information Sharing Platform): Used for sharing, storing, and correlating threat information, including IoCs, malware samples, and attack patterns. MISP facilitates structured information exchange within our organization and with trusted partners.
- VirusTotal: For analyzing suspicious files and URLs, providing insights into known malware signatures and detection rates across various antivirus engines.
- SANS Internet Storm Center (ISC): Provides daily insights into internet threats and vulnerabilities.
- Threat Intelligence Blogs & Research Papers: Monitoring reputable cybersecurity blogs (e.g., from major security vendors, independent researchers) and academic research for emerging threats and vulnerabilities.

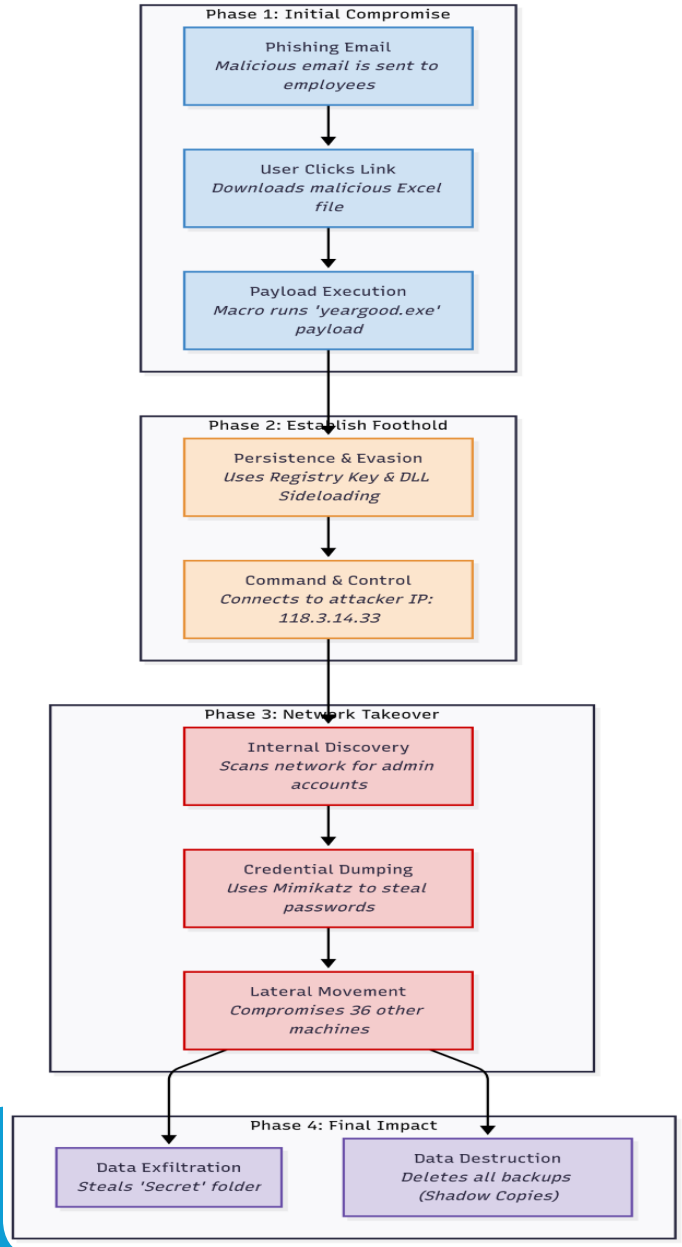
Examples of Threat Alerts

ICS/SCADA Vulnerability (e.g., Log4Shell)

Source: CISA + Commercial Feeds (Recorded Future)

Action Taken:

- Asset Identification → Segmentation → Patching
- IoC Hunting via SIEM/EDR
- Firewall/IPS Updates
- Internal & Partner Dissemination (AIS, TLP protocols)



Red Team / Blue Team Exercise Report

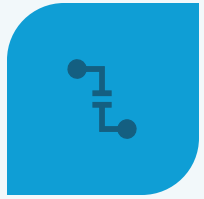
- Red Team / Blue Team Exercise
- A simulated multi-stage attack tested SecGuru's defenses against realistic threats like credential theft, data exfiltration, and persistence. The scenario used phishing, macro payloads, DLL sideloading, and command-and-control (C2) communications to emulate an advanced intrusion.
- **Red Team Tactics (MITRE-aligned):**
  - **Phishing via malicious Excel attachments**
  - **Macro execution and persistence via registry keys**
  - **DLL sideloading for evasion**
  - **Credential dumping with Mimikatz**
  - **Lateral movement through RDP**
  - **Data destruction and outbound C2 traffic**
- **Blue Team Response:**
  - **Detected key indicators via SIEM, endpoint monitoring, and network analysis**
  - **Flagged macro activity, DLL hashes, and unauthorized memory access**
  - **Isolated affected hosts and triggered containment playbooks**
- **Key Gaps Identified:**
  - **Phishing bypassed email filters**
  - **No MFA on privileged accounts**
  - **Excessive user privileges enabled compromise**
  - **C2 traffic remained active for hours**
- **Lessons Learned:**
  - **Enforce MFA and least privilege**
  - **Improve email filtering and segmentation**
  - **Enhance behavioral detection and outbound traffic controls**
  - **Conduct regular simulation training**

Business Continuity Plan

To ensure SecGuru's mission-critical operations remain functional during cyberattacks, outages, or disasters, we developed a robust Business Continuity Plan. This plan minimizes downtime, protects data integrity, and ensures rapid recovery by combining technical safeguards, structured response protocols, and continuous testing.

CRITICAL SYSTEMS IDENTIFIED: BIA OUTLINES RTO/RPO FOR ASSETS LIKE SIEM, VPN, AND SQL DATABASES.

REDUNDANT INFRASTRUCTURE: ENCRYPTED BACKUPS (CLOUD + ON-SITE), GENERATOR POWER, AND INTERNET FAILOVER.



CONTINUITY OF OPERATIONS (COOP): VPN/VDI ACCESS AND A FULLY EQUIPPED ALTERNATE HQ ENSURE RESILIENCE.

SECURE COMMUNICATION PROTOCOLS: CONTACT TREE, ENCRYPTED MESSAGING, AND PRE-APPROVED RESPONSE TEMPLATES.

ONGOING TESTING & UPDATES: QUARTERLY DRILLS AND POST-INCIDENT REVIEWS IMPROVE READINESS.

VENDOR COMPLIANCE: PARTNERS MUST MEET ISO 27001 AND SOC 2 STANDARDS WITH PROVEN BCPS.

Disaster Recovery Plan (DRP)

- **Objective:** To ensure the rapid and orderly restoration of critical IT systems and data in the event of a significant disruption or disaster. This plan is built on a foundation of risk analysis and is designed to meet predefined Recovery Time Objectives (RTO) and Recovery Point Objectives (RPO) for all essential business functions.

Component	Strategy	Key Actions
1. **Roles & Responsibilities**	Establish a clear chain of command.	A dedicated **Disaster Recovery Team** is formed with defined roles (Coordinator, Technical Leads, Communications Manager) to manage the crisis response.
2. **Backup & Recovery**	Guarantee data resilience using the **3-2-1 backup rule**.	**Three Copies** of data are maintained on **Two Media Types** (e.g., on-prem disk, cloud storage), with **One Off-Site Copy** in a secure Azure cloud region.
3. **Disaster Recovery Site**	Utilize a **"warm site" hybrid cloud model** for rapid infrastructure failover.	Pre-configured and updated VM templates are maintained in Azure. In a disaster, these VMs are rapidly deployed, and data is restored from cloud backups.
4. **Communication Plan**	Implement a **multi-channel communication plan** to keep all stakeholders informed.	Use a mass notification system for alerts, maintain an out-of-band channel (e.g., Signal) for the DR Team, and use pre-approved templates for status updates.
5. **Testing & Maintenance**	Treat the DRP as a **living document** through continuous testing and updates.	Conduct **quarterly tabletop exercises** to validate procedures and perform **annual failover tests** to ensure technical recovery processes function as expected.

Recommendations

Enforce Multi-Factor Authentication (MFA): Apply MFA across all privileged and sensitive user accounts to reduce the impact of credential theft.

Implement Least Privilege: Remove unnecessary administrative rights and ensure role-based access control is enforced.

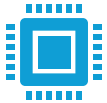
Strengthen Email Filtering & User Awareness: Enhance filtering rules and run regular phishing simulation training to reduce human risk.

Improve Network Segmentation: Isolate sensitive systems to reduce lateral movement opportunities.

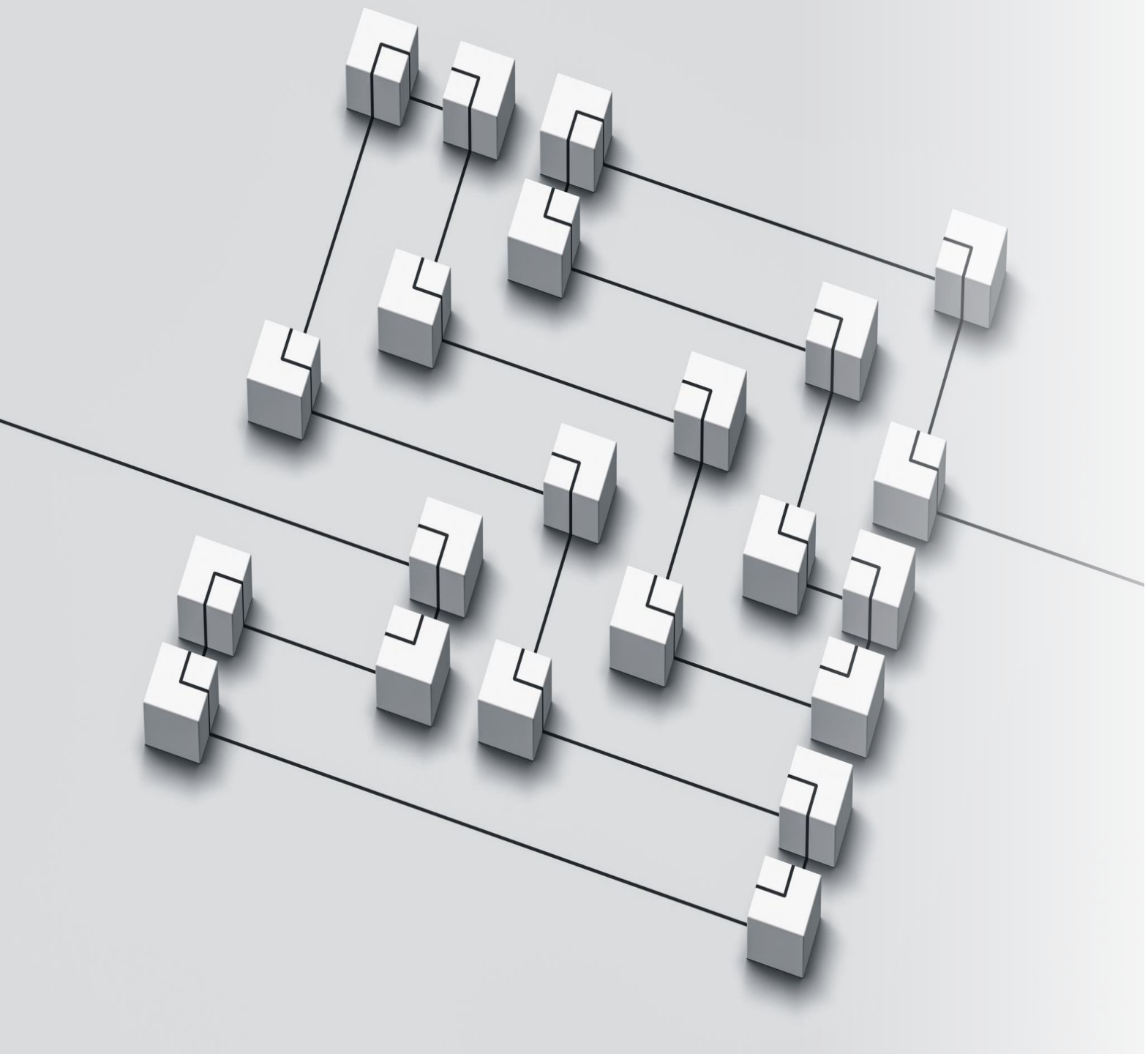
Integrate Physical and Cybersecurity: Strengthen building access control, surveillance, and emergency readiness in parallel with cyber defenses.

Enhance Endpoint & Memory Monitoring: Deploy advanced detection tools (e.g., Sysmon, EDR) to catch in-memory attacks.

Expand Threat Intelligence Integration: Leverage platforms like CISA alerts and AlienVault OTX to stay ahead of emerging threats.



Audit and Test Incident Response Plans: Simulate attacks routinely to refine detection and response playbooks.



Conclusion and Final Thoughts

This SecGuru presentation has successfully demonstrated a comprehensive, layered security architecture that integrates both technical and physical safeguards to defend critical infrastructure and data assets. Through the collaborative efforts of the SecGuru team, we have identified strengths in detection, containment, and response across network, endpoint, and physical domains.

The Red Team/Blue Team exercise revealed valuable insights. While the Blue Team showed strong detection capabilities against lateral movement, credential theft, and data destruction, the Red Team's initial phishing campaign exposed vulnerabilities in email filtering, privilege access, and outbound traffic monitoring. These lessons reinforce the need for continuous hardening of your security posture.

This SecGuru presentation reflects a real-world approach to securing hybrid infrastructures. Continued testing, simulation, and policy enforcement will ensure that SecGuru remains resilient in the face of evolving threats.

Appendices

Roles

Aldo – Executive Team Lead
Marion – Security Architect
Afrika – SIEM Lead
Max – Physical Security Analyst
Alex – Red/Blue Simulation Lead
Derelys – Threat Intelligence Lead

Citations:

https://attack.mitre.org/

https://www.upguard.com/blog/disaster-recovery-plan

https://www.derekseaman.com/2025/01/3-2-1-go-a-step-by-step-guide-to-implementing-foolproof-backups.html

https://www.splunk.com/en_us/blog/learn/splunk-tutorials.html

https://armorpoint.com/2024/05/08/a-step-by-step-guide-to-incident-response-practical-guidance-from-nist-sp-800-61/

Montgomery College. TechMAP Cybersecurity. Instructed by Prof. Reza Mirabrishami, Cyber Department, Spring 2025. Montgomery College, Germantown Campus. Lecture.

THANK YOU!

ANY QUESTIONS?