Experiment 1

Aim: Basic Commands of Network Security

Ping command

Theory: The ping command checks if a host is reachable over the network. It sends ICMP Echo Request packets and waits for replies to measure connection status and delay (latency).

Output:

```
C:\Users\IPG 3>ping 10.104.14.117

Pinging 10.104.14.117 with 32 bytes of data:
Reply from 10.104.17.195: Destination host unreachable.
Reply from 10.104.17.195: Destination host unreachable.
Reply from 10.104.17.195: Destination host unreachable.
Reply from 10.104.17.195: Destination host unreachable.

Ping statistics for 10.104.14.117:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
```

Arp Command

Theory: The Arp command shows the ARP cache, which maps IP addresses to MAC (hardware) addresses. It's used to diagnose LAN-related communication problems

Output:

```
C:\Users\IPG 3>arp -a

Interface: 10.104.17.195 --- 0x11
  Internet Address      Physical Address      Type
  10.104.18.54          bc-03-58-b5-01-c6     dynamic
  10.104.31.254         c8-82-34-70-53-98     dynamic
  10.104.31.255         ff-ff-ff-ff-ff-ff     static
  224.0.0.2             01-00-5e-00-00-02     static
  224.0.0.22            01-00-5e-00-00-16     static
  224.0.0.251           01-00-5e-00-00-fb     static
  224.0.0.252           01-00-5e-00-00-fc     static
  239.255.255.250       01-00-5e-7f-ff-fa     static
  255.255.255.255       ff-ff-ff-ff-ff-ff     static

Interface: 192.168.56.1 --- 0x16
  Internet Address      Physical Address      Type
  192.168.56.255        ff-ff-ff-ff-ff-ff     static
  224.0.0.2             01-00-5e-00-00-02     static
  224.0.0.22            01-00-5e-00-00-16     static
  224.0.0.251           01-00-5e-00-00-fb     static
  224.0.0.252           01-00-5e-00-00-fc     static
  239.255.255.250       01-00-5e-7f-ff-fa     static
```

route command

Theory: the route command displays the IP routing table used by the operating system to determine the path for sending network packets. It helps in diagnosing routing issues.

Output:

```
C:\Users\IPG 3>route print
===========================================================================
Interface List
 22...0a 00 27 00 00 16 ......VirtualBox Host-Only Ethernet Adapter
  4...c4 75 ab 3b 45 0d ......Microsoft Wi-Fi Direct Virtual Adapter #3
 14...c6 75 ab 3b 45 0c ......Microsoft Wi-Fi Direct Virtual Adapter #4
 17...c4 75 ab 3b 45 0c ......Intel(R) Wi-Fi 6 AX201 160MHz
 19...c4 75 ab 3b 45 10 ......Bluetooth Device (Personal Area Network)
 21...e4 a8 df c7 ba 69 ......Realtek PCIe GbE Family Controller
  1...........................Software Loopback Interface 1
===========================================================================

IPv4 Route Table
===========================================================================
Active Routes:
Network Destination        Netmask          Gateway       Interface  Metric
          0.0.0.0          0.0.0.0    10.104.31.254    10.104.17.195     30
       10.104.0.0    255.255.224.0         On-link     10.104.17.195    286
    10.104.17.195  255.255.255.255         On-link     10.104.17.195    286
    10.104.31.255  255.255.255.255         On-link     10.104.17.195    286
        127.0.0.0        255.0.0.0         On-link         127.0.0.1    331
        127.0.0.1  255.255.255.255         On-link         127.0.0.1    331
  127.255.255.255  255.255.255.255         On-link         127.0.0.1    331
     192.168.56.0    255.255.255.0         On-link      192.168.56.1    281
     192.168.56.1  255.255.255.255         On-link      192.168.56.1    281
   192.168.56.255  255.255.255.255         On-link      192.168.56.1    281
        224.0.0.0        240.0.0.0         On-link         127.0.0.1    331
        224.0.0.0        240.0.0.0         On-link      192.168.56.1    281
        224.0.0.0        240.0.0.0         On-link     10.104.17.195    286
  255.255.255.255  255.255.255.255         On-link         127.0.0.1    331
  255.255.255.255  255.255.255.255         On-link      192.168.56.1    281
  255.255.255.255  255.255.255.255         On-link     10.104.17.195    286
===========================================================================
Persistent Routes:
  None

IPv6 Route Table
===========================================================================
Active Routes:
 If Metric Network Destination      Gateway
  1    331 ::1/128                  On-link
  1    331 ff00::/8                 On-link
===========================================================================
Persistent Routes:
  None
```

Netstat Command

Theory: netstat (Network Statistics) shows active network connections, listening ports, and protocol statistics. It's useful for checking which applications are using the network.

Output:

```
C:\Users\IPG 3>netstat -an

Active Connections

  Proto  Local Address          Foreign Address        State
  TCP    0.0.0.0:135            0.0.0.0:0              LISTENING
  TCP    0.0.0.0:445            0.0.0.0:0              LISTENING
  TCP    0.0.0.0:3306           0.0.0.0:0              LISTENING
  TCP    0.0.0.0:5040           0.0.0.0:0              LISTENING
  TCP    0.0.0.0:7680           0.0.0.0:0              LISTENING
  TCP    0.0.0.0:20080          0.0.0.0:0              LISTENING
  TCP    0.0.0.0:33060          0.0.0.0:0              LISTENING
  TCP    0.0.0.0:49664          0.0.0.0:0              LISTENING
  TCP    0.0.0.0:49665          0.0.0.0:0              LISTENING
  TCP    0.0.0.0:49666          0.0.0.0:0              LISTENING
  TCP    0.0.0.0:49667          0.0.0.0:0              LISTENING
  TCP    0.0.0.0:49668          0.0.0.0:0              LISTENING
  TCP    0.0.0.0:49670          0.0.0.0:0              LISTENING
  TCP    0.0.0.0:49671          0.0.0.0:0              LISTENING
  TCP    0.0.0.0:49672          0.0.0.0:0              LISTENING
  TCP    0.0.0.0:49673          0.0.0.0:0              LISTENING
  TCP    0.0.0.0:49674          0.0.0.0:0              LISTENING
  TCP    0.0.0.0:49684          0.0.0.0:0              LISTENING
  TCP    10.104.17.195:139      0.0.0.0:0              LISTENING
  TCP    10.104.17.195:12456    3.233.158.24:443       ESTABLISHED
  TCP    10.104.17.195:12463    3.233.158.24:443       ESTABLISHED
  TCP    10.104.17.195:12474    52.86.122.36:443       CLOSE_WAIT
  TCP    10.104.17.195:13348    192.178.134.94:443     CLOSE_WAIT
  TCP    10.104.17.195:13629    104.101.17.253:443     ESTABLISHED
  TCP    10.104.17.195:15475    52.206.28.63:443       CLOSE_WAIT
  TCP    10.104.17.195:15477    52.71.234.170:443      CLOSE_WAIT
  TCP    10.104.17.195:15479    52.71.234.170:443      CLOSE_WAIT
  TCP    10.104.17.195:15495    142.250.193.2:443      CLOSE_WAIT
  TCP    10.104.17.195:15503    52.206.28.63:443       CLOSE_WAIT
  TCP    10.104.17.195:15505    52.71.234.170:443      CLOSE_WAIT
  TCP    10.104.17.195:15508    52.71.234.170:443      CLOSE_WAIT
  TCP    10.104.17.195:15511    142.250.182.174:443    CLOSE_WAIT
  TCP    10.104.17.195:15550    18.142.103.110:443     CLOSE_WAIT
  TCP    10.104.17.195:15942    54.151.190.11:443      FIN_WAIT_2
  TCP    10.104.17.195:15943    54.151.190.11:443      ESTABLISHED
  TCP    10.104.17.195:15979    172.64.151.195:443     ESTABLISHED
  TCP    10.104.17.195:15981    104.18.42.202:443      ESTABLISHED
  TCP    10.104.17.195:16044    104.18.42.202:443      ESTABLISHED
  TCP    10.104.17.195:16045    172.64.151.195:443     ESTABLISHED
  TCP    10.104.17.195:16114    4.213.25.240:443       ESTABLISHED
  TCP    10.104.17.195:16296    172.217.194.188:5228   ESTABLISHED
  TCP    10.104.17.195:16297    172.217.194.188:5228   ESTABLISHED
  TCP    10.104.17.195:16334    52.98.88.242:443       ESTABLISHED
  TCP    10.104.17.195:16335    52.98.88.242:443       ESTABLISHED
  TCP    10.104.17.195:17971    52.86.122.36:443       ESTABLISHED
  TCP    10.104.17.195:17972    52.86.122.36:443       ESTABLISHED
  TCP    10.104.17.195:18031    142.250.183.234:443    ESTABLISHED
```

Tracert Command

Theory: The tracert command traces the route packets take to reach a destination. It lists each hop (router) and how long it takes to reach there, helping diagnose path delays.

Output:

```
C:\Users\IPG 3>tracert google.com

Tracing route to google.com [172.217.27.174]
over a maximum of 30 hops:

  1     5 ms     6 ms     1 ms  10.104.31.254
  2     5 ms    12 ms     1 ms  10.0.253.1
  3     4 ms     4 ms     9 ms  202.12.103.126
  4    11 ms    11 ms     6 ms  74.125.50.56
  5    15 ms     9 ms     3 ms  172.253.68.91
  6     8 ms     3 ms    11 ms  172.253.67.95
  7     3 ms     4 ms     4 ms  kix05s07-in-f14.1e100.net [172.217.27.174]

Trace complete.
```

Ipconfig Command

Theory: The ipconfig command displays the current network configuration, including IP address, subnet mask, default gateway, and DNS settings of all network interfaces.

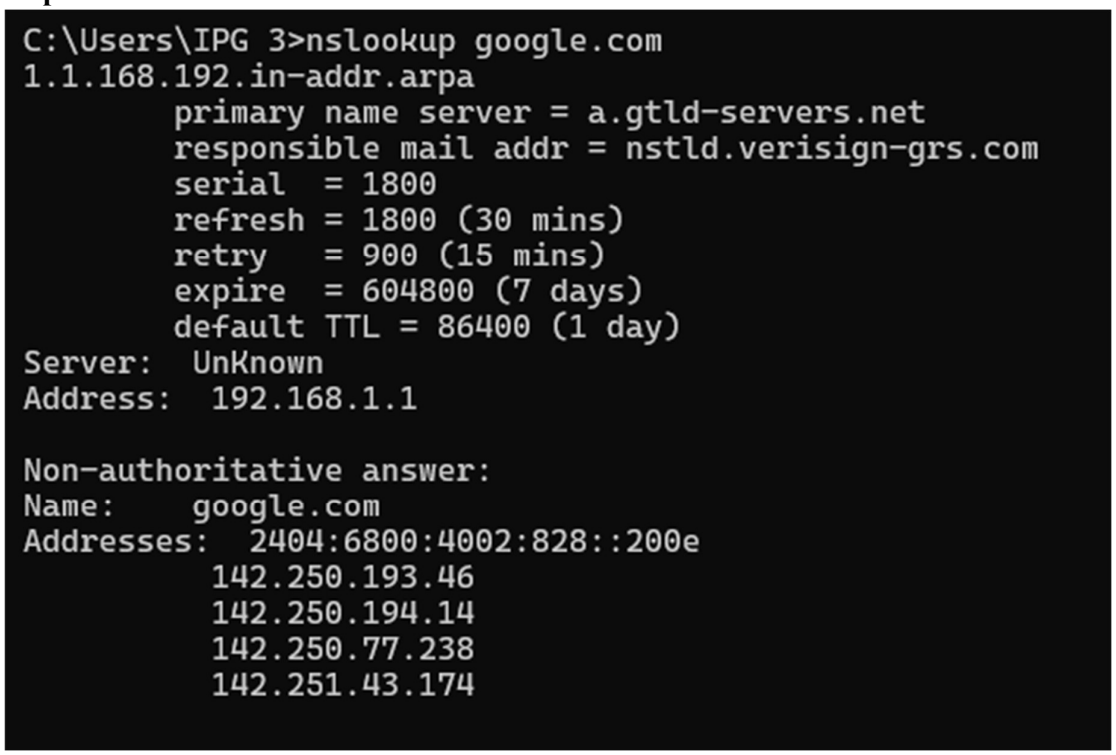
Output:

```
C:\Users\IPG 3>ipconfig /all

Windows IP Configuration

   Host Name . . . . . . . . . . . . : LAPTOP-T8U5BGH4
   Primary Dns Suffix  . . . . . . . :
   Node Type . . . . . . . . . . . . : Hybrid
   IP Routing Enabled. . . . . . . . : No
   WINS Proxy Enabled. . . . . . . . : No

Ethernet adapter Ethernet 3:

   Connection-specific DNS Suffix  . :
   Description . . . . . . . . . . . : VirtualBox Host-Only Ethernet Adapter
   Physical Address. . . . . . . . . : 0A-00-27-00-00-16
   DHCP Enabled. . . . . . . . . . . : No
   Autoconfiguration Enabled . . . . : Yes
   IPv4 Address. . . . . . . . . . . : 192.168.56.1(Preferred)
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . :
   NetBIOS over Tcpip. . . . . . . . : Enabled

Wireless LAN adapter Local Area Connection* 3:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :
   Description . . . . . . . . . . . : Microsoft Wi-Fi Direct Virtual Adapter #3
   Physical Address. . . . . . . . . : C4-75-AB-3B-45-0D
   DHCP Enabled. . . . . . . . . . . : Yes
   Autoconfiguration Enabled . . . . : Yes

Wireless LAN adapter Local Area Connection* 4:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :
   Description . . . . . . . . . . . : Microsoft Wi-Fi Direct Virtual Adapter #4
   Physical Address. . . . . . . . . : C6-75-AB-3B-45-0C
   DHCP Enabled. . . . . . . . . . . : No
   Autoconfiguration Enabled . . . . : Yes
```

Nslookup Command

Theory: The nslookup command queries DNS servers to resolve domain names into IP addresses and vice versa. It helps troubleshoot DNS-related issues.

Output:



Experiment 2

Aim: Basic Commands of Network Security in LINUX

Iptables

Iptables is a Linux firewall tool that controls network traffic by checking packets against rules. It decides whether to allow, block, or redirect traffic based on source/destination IPs, ports, and protocols. Rules are organized in chains: INPUT (incoming), OUTPUT (outgoing), and FORWARD (traffic passing through).

View Current Rules

```
ubuntu@ubuntu:~$ sudo iptables -L -v -n
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destinati
on

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destinati
on

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destinati
on
```

Shows all active firewall rules with packet counts. The -L gives List rules, -v gives details, -n shows raw IPs instead of hostnames.

Block an IP Address

```
ubuntu@ubuntu:~$ sudo iptables -A INPUT -s 192.168.1.100 -j DROP

ubuntu@ubuntu:~$ sudo iptables -A OUTPUT -p tcp --dport 443 -j ACCEPT
```

iptables -A INPUT -s 192.168.1.100 -j DROP

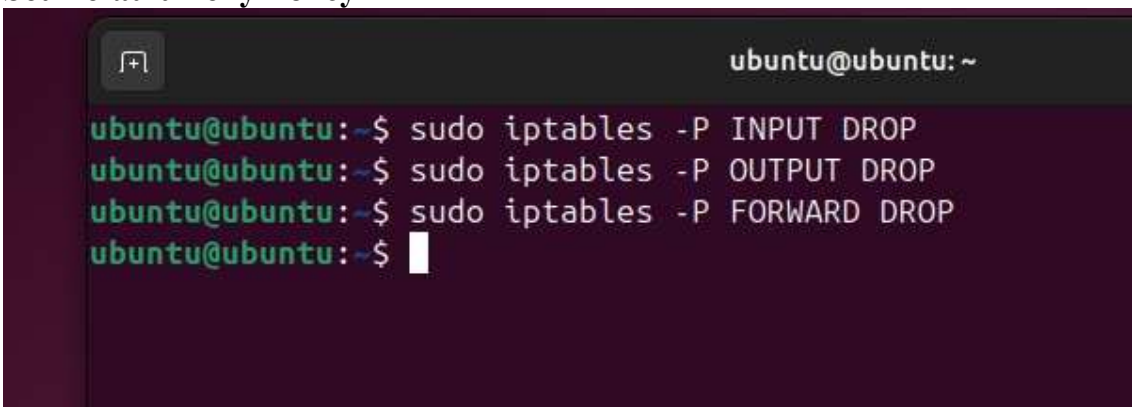
Blocks all traffic from IP 192.168.1.100. The packet gets silently dropped without any response.

Allow HTTPS Traffic Out

iptables -A OUTPUT -p tcp --dport 443 -j ACCEPT

Permits outbound connections to port 443 (HTTPS). Useful when you have restrictive default policies.

Set Default Deny Policy

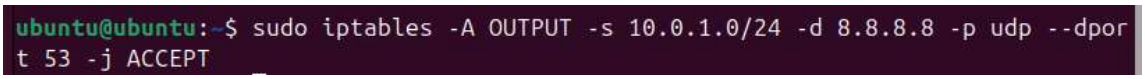


Makes the firewall block everything by default. You then add specific ACCEPT rules for what you need.

Allow Network to HTTPS

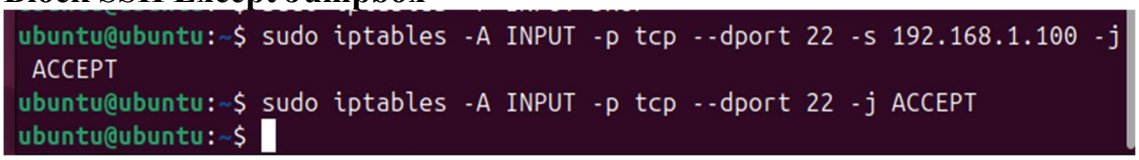
Let's the entire 10.0.0.0/24 subnet access HTTPS anywhere. Good for office networks.

Allow DNS from Subnet



Permits DNS queries from 10.0.1.0/24 to Google's DNS server. DNS uses UDP port 53.

Block SSH Except Jumpbox



Only allows SSH from the jumpbox (192.168.1.10), blocks it from everywhere else. Order matters here.

Netstat Command

Netstat shows network connections and listening services on your system.
Common Usage

netstat -tuln

- **-t**: TCP connections
- **-u**: UDP connections
- **-l**: Only listening ports
- **-n**: Show numbers, not names

Show Active Connections

netstat -tulnp

The -p adds process names, so you see which program is using each port.

Network Statistics

netstat -s

Displays packet counts and error statistics for each network protocol.

Experiment 3

Aim: To learn and perform network reconnaissance using Nmap.

Theory

Nmap is a powerful network discovery and security auditing tool. It uses raw IP packets to determine what hosts are available on the network, what services those hosts are offering, what operating systems they are running, and what type of packet filters/firewalls are in use. Nmap is commonly used for:

- Network inventory and asset discovery
- Security auditing and penetration testing
- Network troubleshooting
- Monitoring host or service uptime

Step 1: Install Nmap

Get Nmap installed on your system using the package manager.
For Debian/Ubuntu-based systems

sudo apt install nmap

Hit Y when it asks for confirmation. This downloads and installs the Nmap scanner along with its dependencies.

```
ubuntu@ubuntu:~$ sudo apt install nmap
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libblas3 liblinear4 libssh2-1t64 nmap-common
Suggested packages:
  liblinear-tools liblinear-dev ncat ndiff zenmap
The following NEW packages will be installed:
  libblas3 liblinear4 libssh2-1t64 nmap nmap-common
0 upgraded, 5 newly installed, 0 to remove and 0 not upgraded.
Need to get 6,286 kB of archives.
After this operation, 27.4 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://archive.ubuntu.com/ubuntu noble-updates/main amd64 libblas3 amd64 3.12.0-3build1.1 [238 kB]
Get:2 http://archive.ubuntu.com/ubuntu noble/universe amd64 liblinear4 amd64 2.3.0+dfsg-5build1 [42.3 kB]
Get:3 http://archive.ubuntu.com/ubuntu noble/main amd64 libssh2-1t64 amd64 1.11.0-4.1build2 [120 kB]
Get:4 http://archive.ubuntu.com/ubuntu noble/universe amd64 nmap-common all 7.94+git20230807.3be01efb1+dfsg-3build2 [4,1
92 kB]
Get:5 http://archive.ubuntu.com/ubuntu noble/universe amd64 nmap amd64 7.94+git20230807.3be01efb1+dfsg-3build2 [1,694 kB
]
Fetched 6,286 kB in 3s (1,830 kB/s)
Selecting previously unselected package libblas3:amd64.
(Reading database ... 212032 files and directories currently installed.)
Preparing to unpack .../libblas3_3.12.0-3build1.1_amd64.deb ...
Unpacking libblas3:amd64 (3.12.0-3build1.1) ...
Selecting previously unselected package liblinear4:amd64.
Preparing to unpack .../liblinear4_2.3.0+dfsg-5build1_amd64.deb ...
Unpacking liblinear4:amd64 (2.3.0+dfsg-5build1) ...
Selecting previously unselected package libssh2-1t64:amd64.
Preparing to unpack .../libssh2-1t64_1.11.0-4.1build2_amd64.deb ...
Unpacking libssh2-1t64:amd64 (1.11.0-4.1build2) ...
```

```
Selecting previously unselected package nmap-common.
Preparing to unpack .../nmap-common_7.94+git20230807.3be01efb1+dfsg-3build2_all.
deb ...
Unpacking nmap-common (7.94+git20230807.3be01efb1+dfsg-3build2) ...
Selecting previously unselected package nmap.
Preparing to unpack .../nmap_7.94+git20230807.3be01efb1+dfsg-3build2_amd64.deb .
..
Unpacking nmap (7.94+git20230807.3be01efb1+dfsg-3build2) ...
Setting up libblas3:amd64 (3.12.0-3build1.1) ...
update-alternatives: using /usr/lib/x86_64-linux-gnu/blas/libblas.so.3 to provid
e /usr/lib/x86_64-linux-gnu/libblas.so.3 (libblas.so.3-x86_64-linux-gnu) in auto
 mode
Setting up nmap-common (7.94+git20230807.3be01efb1+dfsg-3build2) ...
Setting up libssh2-1t64:amd64 (1.11.0-4.1build2) ...
Setting up liblinear4:amd64 (2.3.0+dfsg-5build1) ...
Setting up nmap (7.94+git20230807.3be01efb1+dfsg-3build2) ...
Processing triggers for man-db (2.12.0-4build2) ...
Processing triggers for libc-bin (2.39-0ubuntu8.4) ...
```

Step 2: Verify Installation

Make sure Nmap installed properly by checking its version.

nmap –version

```
ubuntu@ubuntu:~$ nmap --version
Nmap version 7.94SVN ( https://nmap.org )
Platform: x86_64-pc-linux-gnu
Compiled with: liblua-5.4.6 openssl-3.0.13 libssh2-1.11.0 libz-1.3 libpcre2-10.42 libpcap-1.10.4 nmap-libdnet-1.12 ipv6
Compiled without:
Available nsock engines: epoll poll select
```

You'll see version info and compile details. If this works, Nmap is ready to scan networks.

Step 3: Basic Host Discovery (Ping Scan)

Start with a simple ping scan to see if the target host is alive and responding.

nmap -sn scanme.nmap.org

```
ubuntu@ubuntu:~$ nmap -sn scanme.nmap.org
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-08-11 18:47 UTC
Nmap scan report for scanme.nmap.org (45.33.32.156)
Host is up (0.28s latency).
Other addresses for scanme.nmap.org (not scanned): 2600:3c01::f03c:91ff:fe18:bb2f
Nmap done: 1 IP address (1 host up) scanned in 0.59 seconds
```

This sends ping packets to check if the host is up. It's like knocking on the door to see if anyone's home before trying to look through windows.

What happens: Nmap sends ICMP echo requests and checks for responses. No port scanning occurs here.

Step 4: Port Scanning (Finding Open Services)

Once you know the host is alive, scan for open ports to see what services are running.

nmap --top-ports 100 scanme.nmap.org

```
ubuntu@ubuntu:~$ nmap --top-ports 100 scanme.nmap.org
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-08-11 18:47 UTC
Nmap scan report for scanme.nmap.org (45.33.32.156)
Host is up (0.28s latency).
Other addresses for scanme.nmap.org (not scanned): 2600:3c01::f03c:91ff:fe18:bb2f
Not shown: 98 filtered tcp ports (no-response)
PORT   STATE SERVICE
22/tcp open  ssh
80/tcp open  http

Nmap done: 1 IP address (1 host up) scanned in 16.78 seconds
```

This checks the 100 most commonly used ports for any that are open and accepting connections.
What you'll see: A list showing which ports are open (like 22 for SSH, 80 for HTTP) and which are closed or filtered.

Step 5: Service and Version Detection
Now probe the open ports to identify exactly what software is running on them.

nmap -sV scanme.nmap.org

```
ubuntu@ubuntu:~$ nmap -sV scanme.nmap.org
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-08-11 18:48 UTC
Nmap scan report for scanme.nmap.org (45.33.32.156)
Host is up (0.28s latency).
Other addresses for scanme.nmap.org (not scanned): 2600:3c01::f03c:91ff:fe18:bb2f
Not shown: 997 filtered tcp ports (no-response)
PORT      STATE SERVICE    VERSION
22/tcp    open  ssh        OpenSSH 6.6.1p1 Ubuntu 2ubuntu2.13 (Ubuntu Linux; protocol 2.0)
80/tcp    open  http       Apache httpd 2.4.7 ((Ubuntu))
31337/tcp open  tcpwrapped
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 43.61 seconds
```

This connects to open ports and tries to determine what service and version is running. Instead of just knowing port 80 is open, you'll learn it's running Apache 2.4.41.
What it reveals: Specific software names, version numbers, and sometimes OS details. This helps identify potential security vulnerabilities in outdated software.

Additional Nmap Techniques

OS Detection

nmap -O scanme.nmap.org

Attempts to identify the target's operating system based on network stack fingerprinting.

Aggressive Scan

nmap -A scanme.nmap.org

Combines OS detection, version detection, script scanning, and traceroute in one command.

Scan Specific Ports

nmap -p 22,80,443 scanme.nmap.org

Scans only the specified ports (SSH, HTTP, HTTPS in this example).

Scan Range of IPs

nmap -sn 192.168.1.1-254

Ping scans an entire subnet to find active hosts.

Experiment 4

Aim: To Installing & Using Suricata for Network Intrusion Detection.

Step1: Install Suricata.

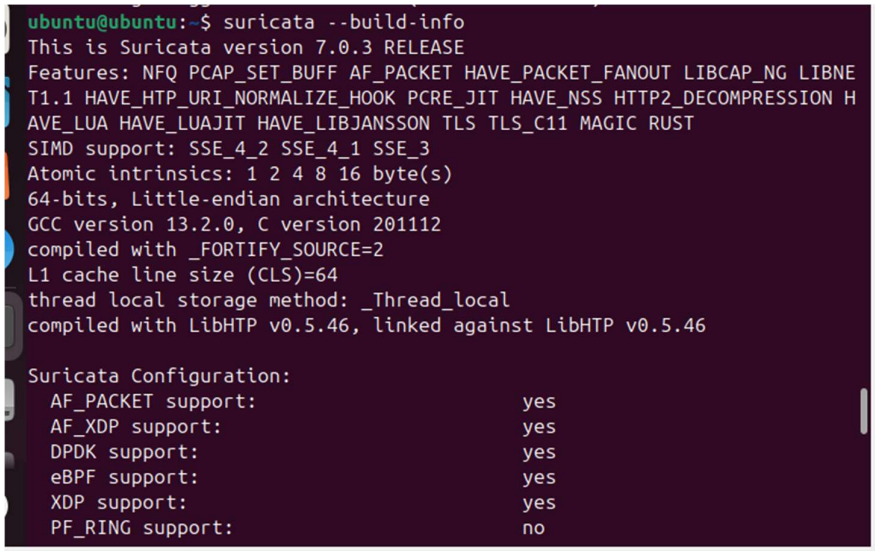
In this step, Suricata is installed on the system using the package manager. Since Suricata itself is only the detection engine, we also install rules using suricata-update. These rule files contain predefined signatures and patterns that help Suricata recognize malicious activities, such as port scans, malware traffic, or suspicious behavior. Without rules, Suricata would not know what to detect.

```
ubuntu@ubuntu:~$ sudo apt update
Ign:1 cdrom://Ubuntu 24.04.2 LTS _Noble Numbat_ - Release amd64 (2025021
5) noble InRelease
Hit:2 cdrom://Ubuntu 24.04.2 LTS _Noble Numbat_ - Release amd64 (2025021
5) noble Release
Hit:4 http://archive.ubuntu.com/ubuntu noble InRelease
Get:5 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB
]
Get:6 http://archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:7 http://archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB
]
Get:8 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packag
es [1054 kB]
Get:9 http://archive.ubuntu.com/ubuntu noble-updates/main amd64 Packages
 [1313 kB]
Get:10 http://archive.ubuntu.com/ubuntu noble-updates/main i386 Packages
 [506 kB]
Get:11 http://archive.ubuntu.com/ubuntu noble-updates/main Translation-e
n [264 kB]
Get:12 http://archive.ubuntu.com/ubuntu noble-updates/main amd64 Compone
```

```
ubuntu@ubuntu:~$ sudo apt install suricata -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  isa-support libevent-core-2.1-7t64 libevent-pthreads-2.1-7t64
  libfdt1 libhiredis1.1.0 libhtp2 libhyperscan5 libluajit-5.1-2
  libluajit-5.1-common libnet1 libnetfilter-log1 libnetfilter-queue1
  librte-bus-pci24 librte-bus-vdev24 librte-eal24 librte-ethdev24
  librte-hash24 librte-ip-frag24 librte-kvargs24 librte-log24
  librte-mbuf24 librte-mempool24 librte-meter24 librte-net-bond24
  librte-net24 librte-pci24 librte-rcu24 librte-ring24 librte-sched24
  librte-telemetry24 libxdp1 oinkmaster snort-rules-default
  sse3-support suricata-update
Suggested packages:
  snort | snort-pgsql | snort-mysql libtcmalloc-minimal4
The following NEW packages will be installed:
  isa-support libevent-core-2.1-7t64 libevent-pthreads-2.1-7t64
  libfdt1 libhiredis1.1.0 libhtp2 libhyperscan5 libluajit-5.1-2
  libluajit-5.1-common libnet1 libnetfilter-log1 libnetfilter-queue1
  librte-bus-pci24 librte-bus-vdev24 librte-eal24 librte-ethdev24
```

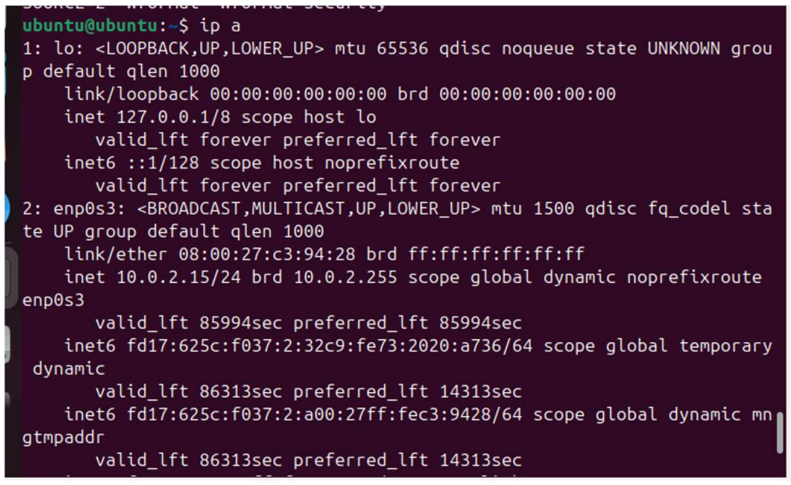
Step 2: Verify the Installation.

After installation, the Suricata build information is checked using a verification command (e.g., suricata --build-info). This ensures that Suricata is correctly installed, compatible with the system, and ready to run with its dependencies. It also confirms that rule updates were successfully applied.



Step 3: Find Network Interface

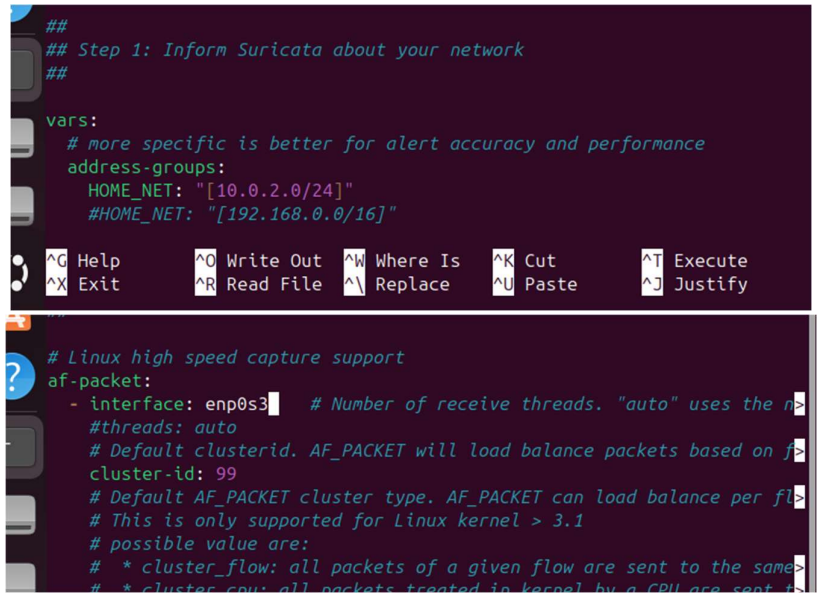
The next step involves identifying the correct network interface (like eth0, ens33, or wlan0) using commands such as ip a or ifconfig. Suricata must be pointed to the right interface because that's where it will capture and inspect real-time network traffic. Choosing the wrong interface would mean Suricata monitors no data or the wrong data.



Step 4: Edit Configuration

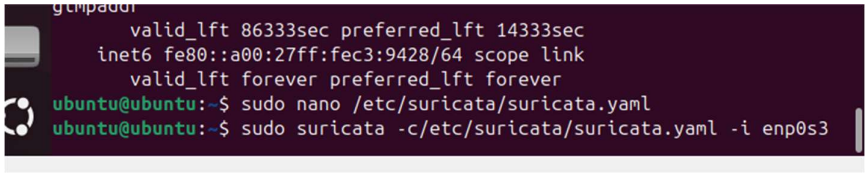
In this step, Suricata's main configuration file (/etc/suricata/suricata.yaml) is edited. Here we specify which network interface to monitor, directories for logs, and which rule sets are

active. Proper configuration ensures Suricata captures traffic from the correct source and saves alerts in accessible locations for later analysis.



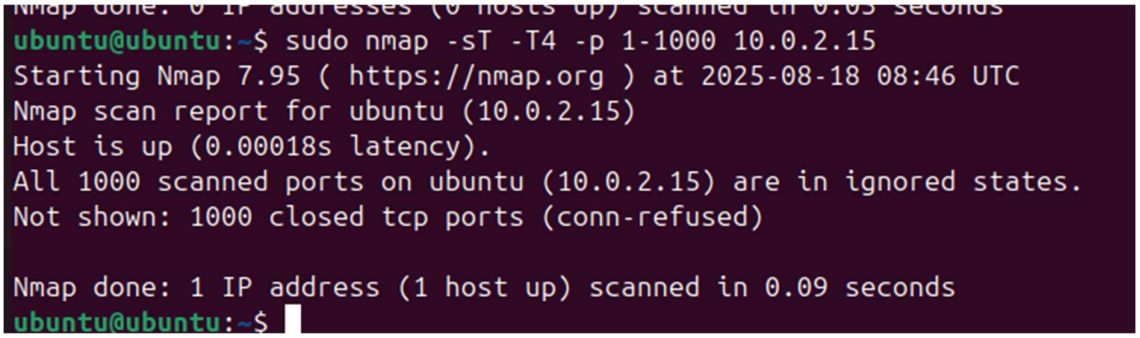
Step 5: Run Suricata in IDS Mode

Suricata is started in Intrusion Detection System (IDS) mode using the configured interface. In IDS mode, it passively observes traffic without blocking it, analyzing packets against rule signatures. Any suspicious activity is logged for review, making this mode ideal for monitoring and learning without disrupting network operations.



Step 6: Simulate Attack Using Nmap

To test Suricata's detection, we simulate an attack by scanning the target system using Nmap. This generates network activity that resembles a real attack (like port scans or OS fingerprinting). Suricata should detect this suspicious behavior and log it as an alert, proving the IDS is working.



```
Not shown: 1000 closed tcp ports (conn-refused)

Nmap done: 1 IP address (1 host up) scanned in 0.09 seconds
ubuntu@ubuntu:~$ nmap A 10.0.2.15/24
Starting Nmap 7.95 ( https://nmap.org ) at 2025-08-18 08:56 UTC
Failed to resolve "A".
Nmap scan report for ubuntu (10.0.2.15)
Host is up (0.00013s latency).
All 1000 scanned ports on ubuntu (10.0.2.15) are in ignored states.
Not shown: 1000 closed tcp ports (conn-refused)

Nmap done: 256 IP addresses (1 host up) scanned in 4.31 seconds
ubuntu@ubuntu:~$
```

Step 7: View Alerts and View Detailed Logs

Finally, we examine Suricata's logs, typically stored in /var/log/suricata/. The fast.log provides a quick overview of triggered alerts, while eve.json contains detailed JSON-formatted logs for in-depth analysis. By reviewing these, we can confirm Suricata successfully detected the simulated attack and gain insights into the type of threat.

```
Nmap done: 256 IP addresses (1 host up) scanned in 4.31 seconds
ubuntu@ubuntu:~$ sudo cat /var/log/suricata/eve.json | jq '.'
```

```
ubuntu@ubuntu: ~                    ubuntu@ubuntu: ~
        "internal": 0
      }
    },
    "expectations": 0
  },
  "memcap_pressure": 5,
  "memcap_pressure_max": 5,
  "http": {
    "memuse": 0,
    "memcap": 0
  },
  "ftp": {
    "memuse": 0,
    "memcap": 0
  },
  "file_store": {
    "open_files": 0
  }
}
}
```

Experiment 5 Foundations of Computer Forensics Investigation

Aim: To understand the structured computer forensics investigation process and gain hands-on experience with the specialized tools required for a forensic analysis. This lab focuses on mastering fundamental tasks, including recovering deleted files from evidence, verifying data integrity using hash calculations, viewing files of various formats, analyzing evidence data, and creating a forensic disk image of a hard disk partition.

Software Required

- EaseUS Data Recovery Wizard
- HashCalc
- MD5 Calculator
- File Viewer 9.5
- AccessData FTK Imager
- R-Drive Image

Procedure

This experiment is divided into six core tasks that cover the foundational workflow of a digital forensic investigation.

Task 1: Recovering Deleted Data with EaseUS Data Recovery Wizard

The first step in many investigations is to recover data that has been intentionally deleted. In this task, we simulate a scenario where a user permanently deletes files to hide their tracks.

1. Simulate Data Deletion: Copy two files, Financial Statement Sample.pdf and Profit and Loss Statement Sample.xlsx, to a designated drive (e.g., Forensic Disk (F:)). Permanently delete these files using the

Shift+Del command.

2. Install and Launch Tool: Install and run the EaseUS Data Recovery Wizard.
3. Scan for Lost Data: From the main window, select the drive where the files were deleted (Forensic Disk (F:)) and initiate a scan for lost data.
4. Recover Files: Once the scan is complete, the tool will list the recoverable files. Select the deleted PDF and XLSX files.
5. Save Recovered Data: Choose a new, secure location (e.g., a "Recovered Files" folder in Documents) and export the recovered files to that destination.

Task 2: Calculating Hashes and Verifying with VirusTotal

Hashing is critical for verifying the integrity of a file and for identifying known malicious files by comparing their hash values against a database.

1. Install HashCalc: Install and launch the HashCalc application.
2. Calculate File Hash:
    - Set the

Data Format to File and select an evidence file, such as Kitty.jpg.

- Ensure the

HMAC box is unchecked, select the desired hash algorithms (e.g., MD5, SHA1), and click Calculate to generate the hash values.

3. Identify a Malicious File:
    - Using HashCalc, calculate the MD5 hash for the

Infected.pdf file.

- Copy the generated MD5 hash value.

4. Check Hash on VirusTotal:
    - Open a web browser and navigate to

https://www.virustotal.com.

- Paste the copied MD5 hash into the search bar and execute the search.
- Review the results to see how many security vendors have flagged the file as malicious, confirming its status.

Task 3: Comparing Hash Values to Ensure Integrity

If an original hash value for a file is known, it can be compared against a newly generated hash to determine if the file has been altered.

1. Install MD5 Calculator: Install and run the MD5 Calculator tool.
2. Generate a Hash: Add the file Friends2.jpg to the tool and click Calculate to generate its current MD5 hash value.
3. Retrieve Original Hash: Open the Hashes.txt file, which contains the pre-calculated, original hash values for the evidence files. Copy the hash corresponding to

Friends2.jpg.

4. Compare Hashes: Paste the original hash from the text file into the Verify MD5 Value field in MD5 Calculator and click Compare. The tool will confirm if the values match, thereby verifying the file's integrity.
5. Identify a Mismatch: Repeat the process for the file Model.png. The tool will report that the hashes do not match, indicating the file has been modified since the original hash was calculated.

Task 4: Viewing Files of Various Formats

Forensic investigators must be able to view a wide range of file types, including those that may have incorrect extensions or are corrupted.

1. Install File Viewer: Install and launch the File Viewer 9.5 application.
2. View an Image File:
    - Use the

File -> Open menu to locate and open cartoon-article.jpg. The image will be displayed.

- Navigate to

File -> File Properties to inspect metadata such as image height, width, and color depth.

3. Analyze a Suspicious File:

    - Attempt to open the file

520px-Biohazard_symbol_(blue).mp4.

- Observe that the tool fails to render the video, possibly displaying an error. This indicates the file may be corrupt or have a deliberately misleading file extension, flagging it for further investigation.

Task 5: Handling Evidence with FTK Imager

Forensic suites like FTK Imager allow investigators to safely examine a bit-for-bit copy of a storage device (a forensic image) without altering the original evidence.

1. Install FTK Imager: Install and run AccessData FTK Imager.
2. Add Evidence:

    - Go to

File -> Add Evidence Item... and select Image File as the source type.

- Browse to and select the forensic image file

Windows_Evidence_001.dd.

3. Explore the Image:

    - In the

Evidence Tree pane, expand the file system nodes to browse the folders and files contained within the image.

- Select a file (e.g., an image) to view its contents in the viewer pane and its properties and hexadecimal data in the bottom panes.

4. Identify and Export Deleted Files:

    - Browse the file list and locate files marked with a red 'X', which indicates they are deleted.
    - Right-click on a deleted file and select

Export Files... to save a copy of it to a new location for further analysis.

Task 6: Creating a Forensic Disk Image

To preserve the original evidence, the first step in a forensic process is often to create a disk image—a bit-by-bit copy of the source drive.

1. Install R-Drive Image: Install and launch the R-Drive Image tool on the Windows 11 machine.
2. Select Source Partition: From the main menu, choose Create Image. In the

Partition Selection window, select the C: drive as the source for the image and proceed.

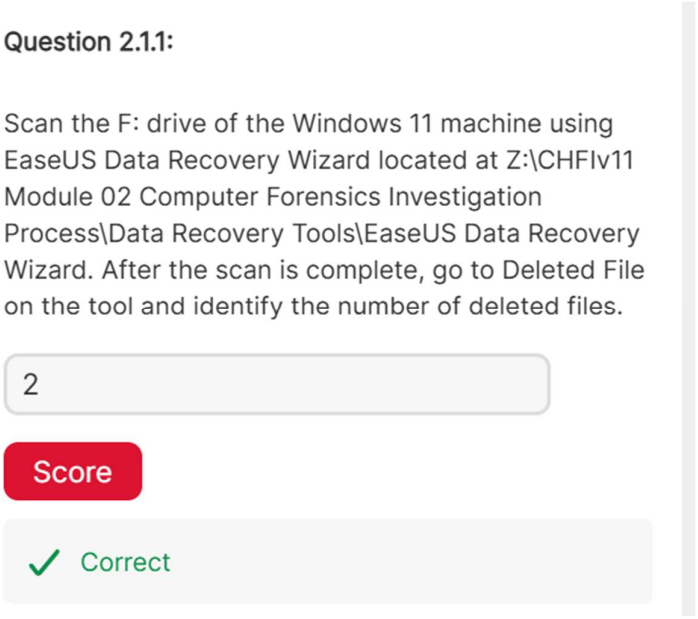
3. Set Destination: Choose a destination location to save the image file (e.g., a network drive or external disk).
4. Begin Imaging Process: Review the summary of operations and click Start to begin the imaging process. The tool will create a

.rdr file containing a complete image of the source partition.

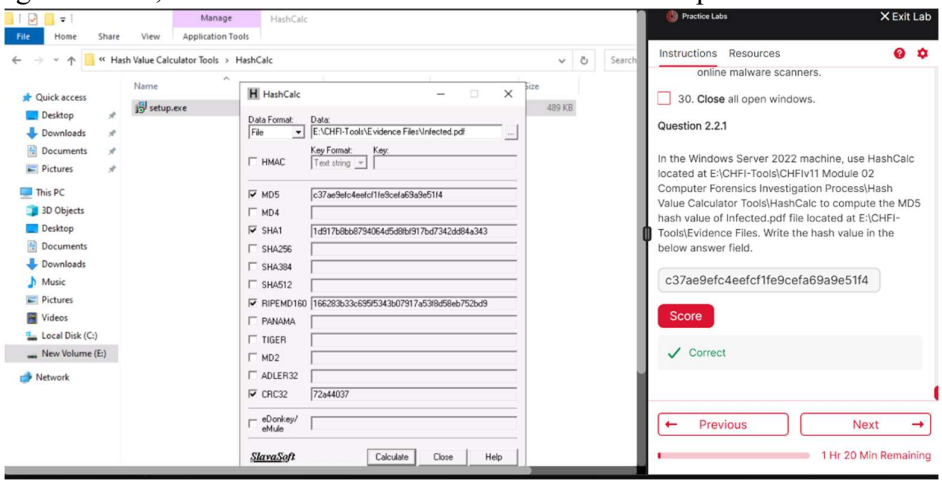
5. Verify Image Creation: After the process completes successfully, navigate to the destination folder to confirm that the new disk image file has been created.

Questions

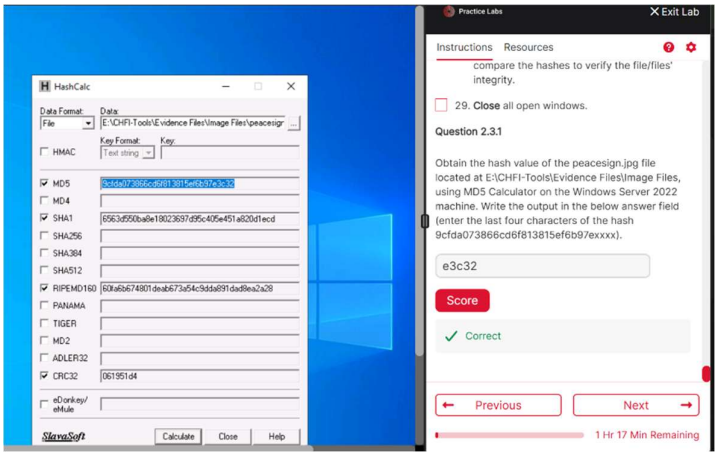
5.1: After completing the scan of the F: drive with EaseUS Data Recovery Wizard, how many deleted files were identified by the tool?



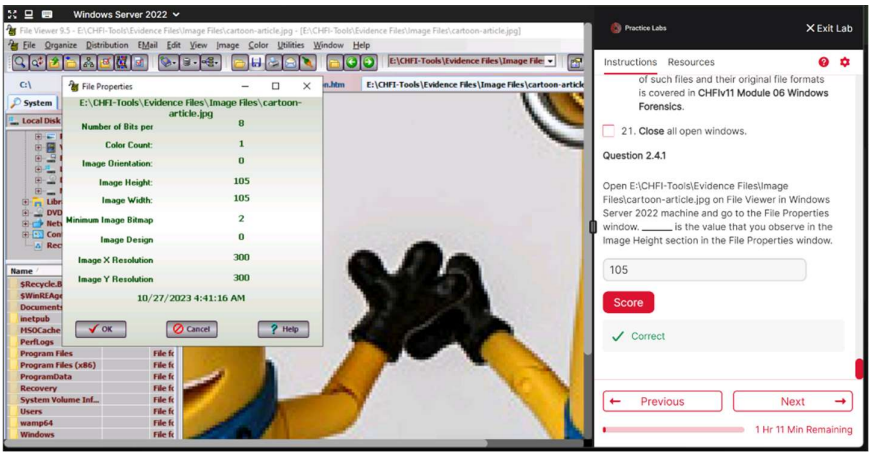
5.2: Using HashCalc, what is the MD5 hash value of the Infected.pdf file?



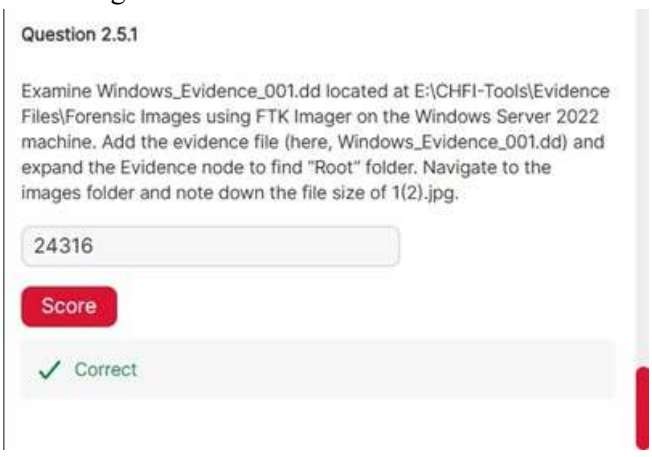
5.3: Using MD5 Calculator, what are the last four characters of the MD5 hash value for the peacesign.jpg file?



5.4: When viewing the file properties of cartoon-article.jpg in File Viewer, what is the value listed for the Image Height?



5.5: In FTK Imager, after loading the Windows_Evidence_001.dd image, what is the file size of 1(2).jpg located in the images folder?



5.6: After creating a disk image of the C: drive using R-Drive Image, what is the full name of the image file that was created?

Conclusion

This experiment provided a hands-on overview of the foundational tasks in a computer forensics investigation. Key skills were developed in recovering deleted data , verifying file integrity through the generation and comparison of hash values , and creating bit-by-bit disk images to preserve original evidence. The use of specialized tools like FTK Imager demonstrated how to analyze a forensic image to find and export evidence, including deleted files. Overall, this lab reinforced the importance of following a structured and repeatable process to ensure that digital evidence is handled correctly and maintains its integrity for use in legal proceedings.

Experiment 6 Advanced File System and Data Analysis

Aim: To perform an in-depth analysis of hard disk images and file systems. This experiment focuses on using advanced forensic tools to analyze Linux and Windows file systems , recover deleted data through file carving , create and analyze a timeline of file system activity , and examine the underlying structure of various file formats using a hex editor.

Software Required

- Autopsy
- WinHex
- The Sleuth Kit (TSK)
- Strawberry Perl
- Hex Editor Neo

Procedure

This experiment is divided into five tasks that cover the analysis of file systems, data recovery, timeline creation, and file format inspection.

Task 1: File System Analysis of a Linux Image with Autopsy

Autopsy provides a graphical interface for analyzing disk images and is a common starting point in an investigation.

1. **Create a New Case**: Launch Autopsy and create a new case, providing a name (e.g., Linux_Analysis) and a base directory to store the case files. Add optional examiner information.
2. **Add Data Source**: Add the Linux_Evidence_001.img forensic image as a new data source. Configure and run the ingest modules to analyze the image content automatically.
3. **Explore the File System**: Once the image is processed, use the **Data Sources** tree in the left pane to navigate the Linux file system structure.
4. **Analyze a File**: Navigate to the /etc directory and select the crontab file. Use the viewer tabs at the bottom to inspect its text content, hexadecimal representation, and file metadata, which includes timestamps and the MD5 hash value.

Task 2: Command-Line Analysis of a Windows Image with The Sleuth Kit (TSK)

The Sleuth Kit (TSK) is a powerful collection of command-line tools that allows for granular analysis of disk images and file systems.

1. **Open Command Prompt**: Navigate to the bin directory of The Sleuth Kit and open a command prompt window in that location.
2. **Analyze Volume System**: Use the mmls command to display the partition layout of the Windows_Evidence_002.dd image.

3. **Analyze File System**: Use the fsstat command on Windows_Evidence_001.dd to display general file system details, such as the file system type (NTFS) and the original operating system.
4. **Inspect MFT Entries**: Use the istat command to view detailed metadata for specific Master File Table (MFT) entries, such as $MFT (entry 0), $MFTMirr (entry 1), and $Boot (entry 7).
5. **List and Recover Files**: Use the fls command to list all files and directories within the image. Then, use the

tsk_recover command to extract all allocated and unallocated files from the image into a specified output folder.

Task 3: Recovering Deleted Files with WinHex

WinHex is an advanced hex editor and disk analysis tool capable of recovering files by searching for their unique headers, a process known as file carving.

1. **Open Image File**: Launch WinHex and open the Linux_Evidence_001.img forensic image.
2. **Initiate File Recovery**: Navigate to **Tools -> Disk Tools -> File Recovery by Type...**.
3. **Select File Types and Destination**: In the recovery window, select the types of files you wish to recover (e.g., expand **Documents** and select various formats). Specify a destination folder (e.g., "Retrieved Files") to save the recovered data.
4. **Execute Recovery**: Start the recovery process. WinHex will scan the entire disk image for the selected file signatures and extract any matching data into the destination folder. Once complete, you can view the recovered files in the output folder.

Task 4: Creating a File System Timeline with TSK

Creating a timeline of file system activity is crucial for understanding the sequence of events on a suspect machine.

1. **Generate a Body File**: Using the TSK command prompt, run the fls command with the -m flag on the Windows_Evidence_001.dd image. This gathers MAC (Modified, Accessed, Changed) time data for every file and directory. Redirect the output to create a file named

body.txt.

2. **Install Perl**: To process the body file, install Strawberry Perl, which is required to run the mactime script.
3. **Create the Timeline**: Use the mactime.pl Perl script to process the body.txt file. This command sorts all the file system events chronologically. Redirect the output to a new file named

timeline.txt.

4. **Analyze Timeline**: Open timeline.txt to view a detailed, time-stamped log of all file activities, which can be used to reconstruct the user's actions.

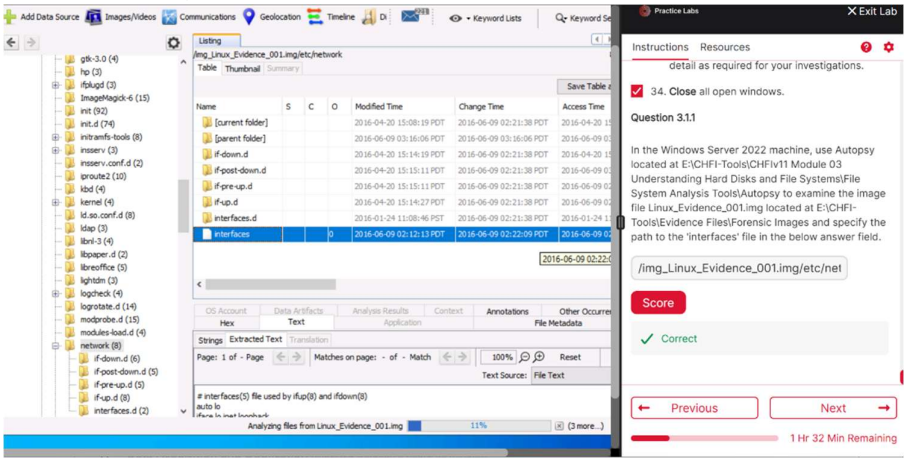
Task 5: Analyzing File Headers with Hex Editor Neo

Analyzing a file's raw hexadecimal data allows an investigator to identify its true type based on its "file signature," regardless of the file extension.

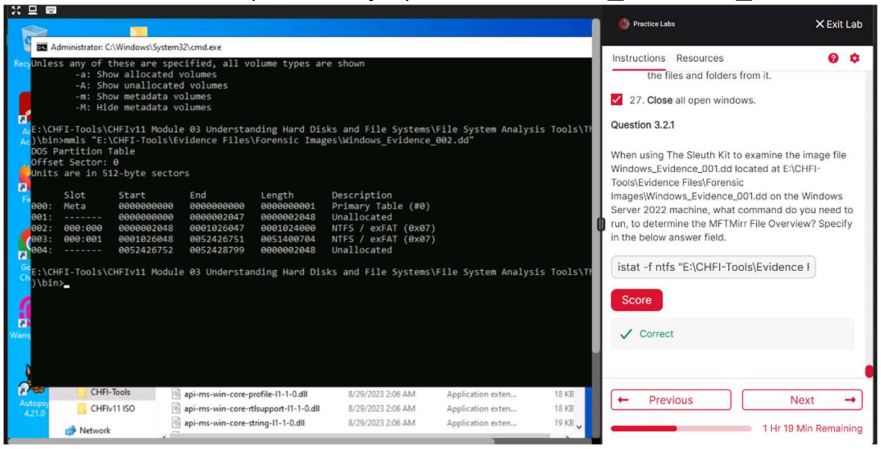
1. **Install and Launch Hex Editor**: Install and open the Hex Editor Neo application.
2. **Open a Sample File**: Use the editor to open various sample files, such as Sample.docx, Sample.jpg, and Sample.pdf.
3. **Identify File Signatures**: For each file, observe the first few bytes in the hex view. These unique sequences, known as file signatures or magic numbers, identify the file type. Note the distinct signatures for each format:
    - **JPEG (.jpg)**: Starts with ff d8 ff.
    - **PDF (.pdf)**: Starts with 25 50 44 46 (ASCII for %PDF).
    - **PNG (.png)**: Starts with 89 50 4e 47.
    - **DOCX (.docx)**: Starts with 50 4b 03 04 (signature for a ZIP archive).

Questions

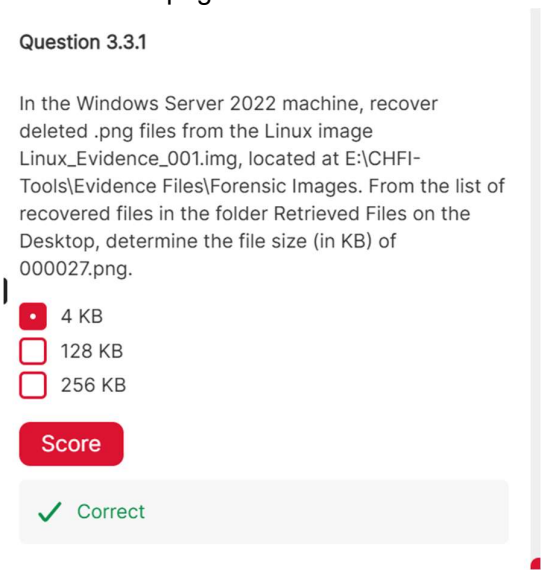
6.1: In Autopsy, what is the full path to the interfaces file within the Linux_Evidence_001.img?



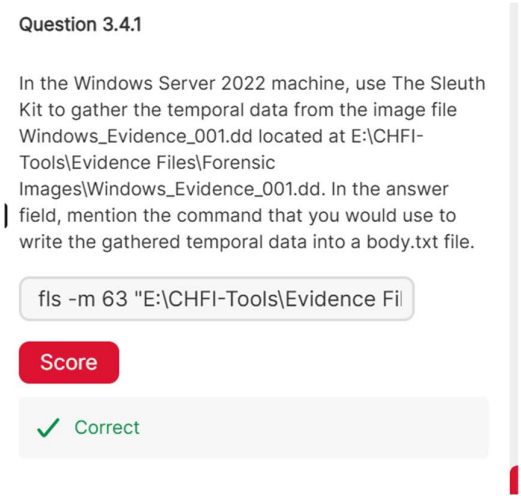
6.2: When using The Sleuth Kit, what is the exact command used to display the metadata overview for the $MFTMirr file (MFT entry 1) in the Windows_Evidence_001.dd image?



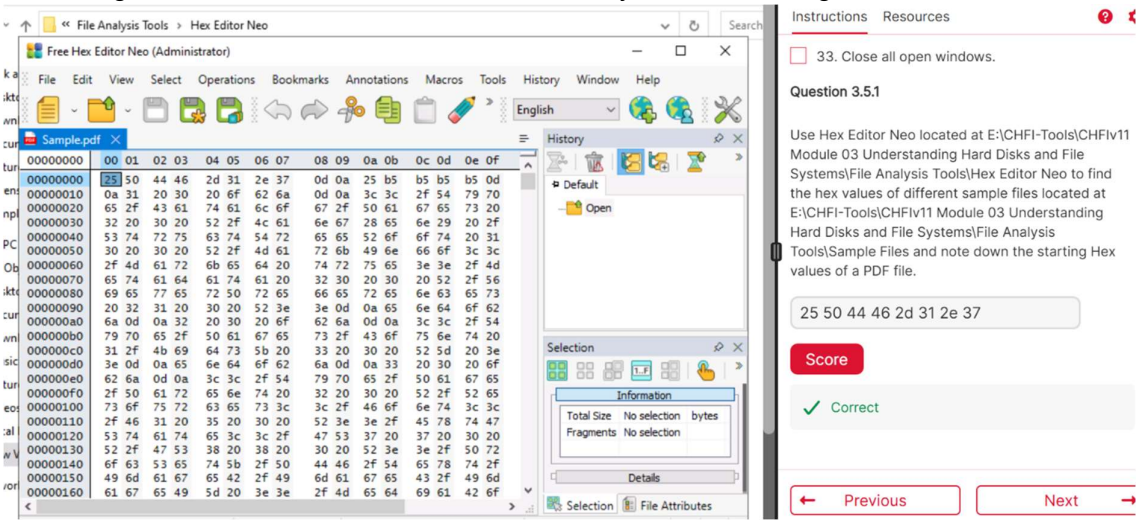
6.3: After recovering .png files from Linux_Evidence_001.img using WinHex, what is the file size (in KB) of the file named 000027.png?



6.4: Using The Sleuth Kit, what is the full command required to gather temporal data from the Windows_Evidence_001.dd image and write it directly to a file named body.txt?



6.5: Using Hex Editor Neo, what are the first four bytes of the file signature for a PDF file?



Conclusion

This experiment provided in-depth, practical experience with the core components of file system forensics. By utilizing both graphical (Autopsy, WinHex) and command-line (The Sleuth Kit) tools, we successfully analyzed Linux and Windows disk images, recovered data, and reconstructed a timeline of events. The analysis of file formats at the hexadecimal level further reinforced the fundamental skills required to verify file types and identify anomalies. Mastering these techniques is essential for any investigator needing to conduct a thorough and accurate examination of digital evidence.

Experiment 7 Mastering Cross-Site Scripting (XSS) Vulnerabilities

Aim: To understand and exploit the three primary types of Cross-Site Scripting (XSS) vulnerabilities—Reflected, Stored, and DOM-based—by successfully completing the six challenges in the Google XSS Game. This experiment focuses on analyzing vulnerable code snippets and crafting malicious payloads to demonstrate each vulnerability.

Prerequisites

- A modern web browser (e.g., Google Chrome, Mozilla Firefox).
- An active internet connection.
- The Google XSS Game website: https://xss-game.appspot.com/

Procedure

This experiment is divided into six tasks, one for each level of the Google XSS Game.

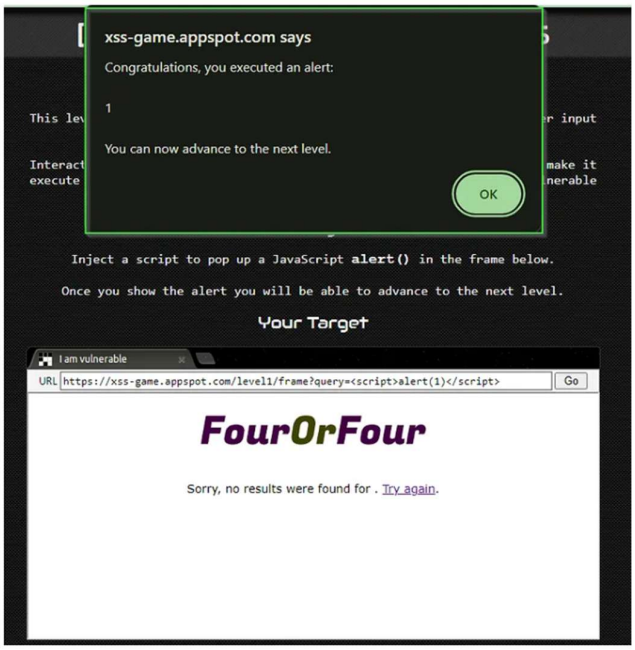
Task 1: Level 1 - Hello, world of XSS (Reflected XSS)

This level introduces a basic **Reflected XSS** vulnerability, where user input is immediately returned (reflected) by the web server in an HTTP response without proper validation.

- **Objective**: Execute a JavaScript alert() in the application's window.
- **Vulnerability Analysis**: The application takes user input from a search query (?query=...) and directly embeds it into the HTML of the results page. There is no sanitization, meaning the browser will interpret any HTML tags included in the query.
- **Payload**: By inputting a script directly into the search box, the browser will execute it when the page loads.

HTML

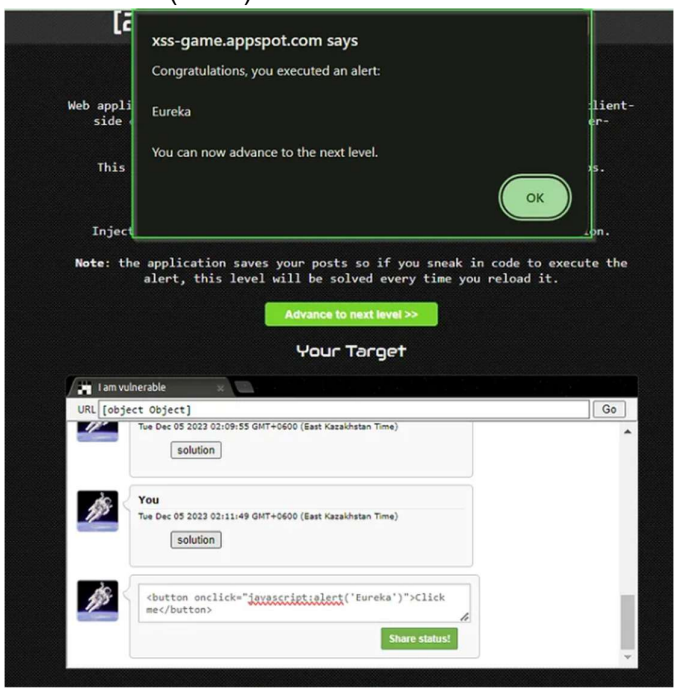
```
<script>alert('XSS');</script>
```



Task 2: Level 2 - Persistence is Key (Stored XSS)
This level demonstrates a **Stored XSS** vulnerability. The malicious script is permanently stored on the target server (e.g., in a database) and is served to any user who views the page, making it a more widespread threat.

- **Objective**: Inject a script that executes when other users view your post.
- **Vulnerability Analysis**: The application allows users to submit text for a status update. While it filters out <script> tags, it does not prevent the injection of other HTML elements like images.
- **Payload**: You can inject an <img> tag with an invalid src attribute. The magic happens in the onerror attribute, which executes JavaScript if the image fails to load. The " is used to close the preceding HTML attribute, allowing the injection.

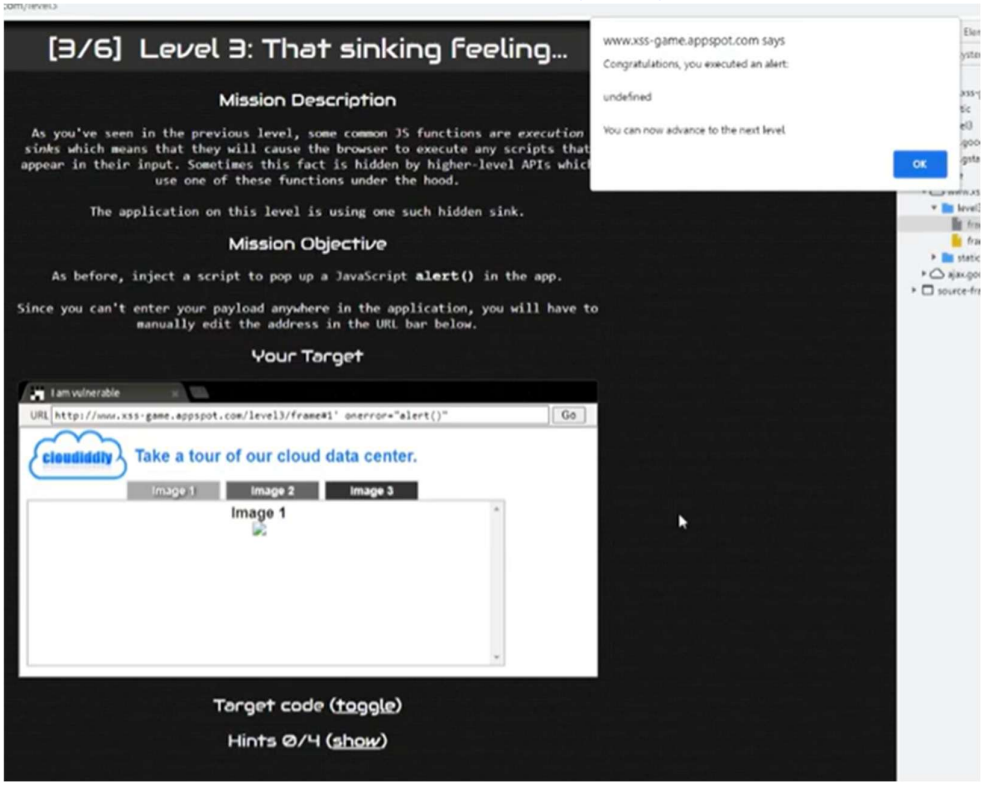
HTML
<img src="invalid" onerror="alert('XSS')">



Task 3: Level 3 - That's a Deep Rabbit Hole (DOM-based XSS)
This level introduces **DOM-based XSS**, where the vulnerability exists in the client-side code rather than the server-side code. The attack payload is executed as a result of modifying the DOM "environment" in the victim's browser.

- **Objective**: Execute an alert by manipulating the URL fragment (#).
- **Vulnerability Analysis**: The client-side JavaScript reads the value from the URL fragment (the part after the #) and uses it to select which image tab to display. The code directly writes this value into the HTML using .innerHTML, creating an opportunity for injection.
- **Payload**: You must provide a valid image number for the script to proceed but can inject an HTML tag afterward. The apostrophe closes the src attribute of the <img> tag, and the onerror attribute executes the payload.
- 1' onerror='alert("XSS")'

The full URL would look like this: https://xss-game.appspot.com/level3/frame#1'%20onerror='alert("XSS")'

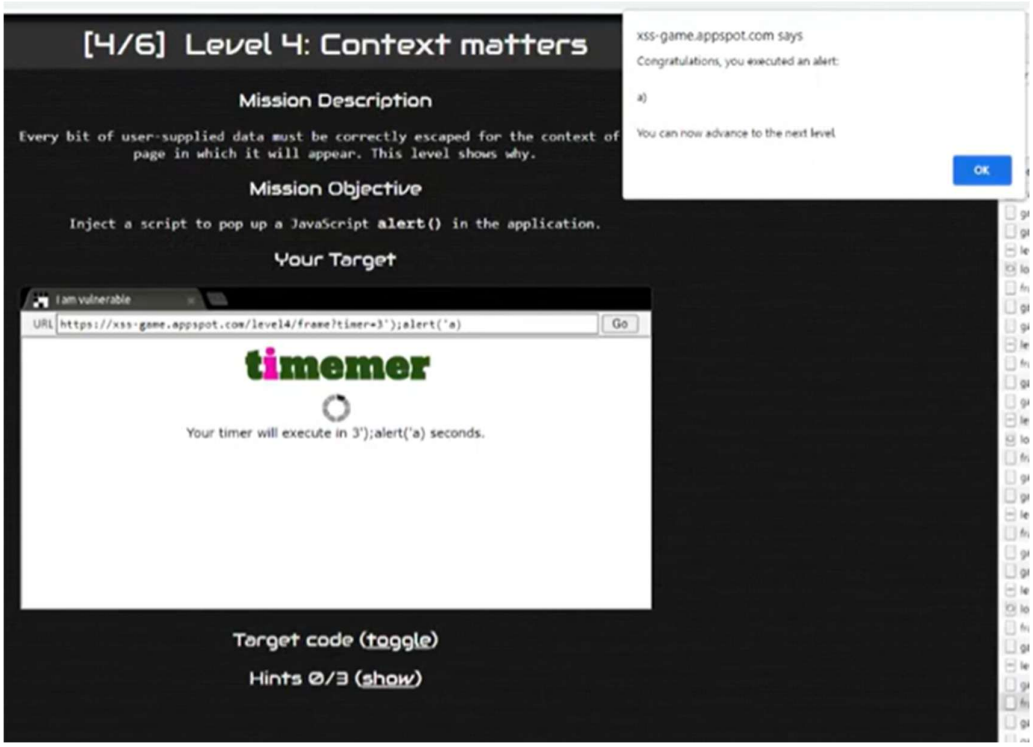


Task 4: Level 4 - The Case of the DOM-based XSS

This challenge presents a slightly more complex **DOM-based XSS** where the payload is passed into a JavaScript timer function.

- **Objective**: Execute an alert after the timer loads.
- **Vulnerability Analysis**: The timer parameter from the URL is directly passed into a startTimer function. This function then uses the input inside a setTimeout call, which evaluates the string as JavaScript.
- **Payload**: The goal is to break out of the existing setTimeout function syntax. The payload ');alert('XSS closes the alert() call within the setTimeout and then appends the malicious alert.

JavaScript
');alert('XSS

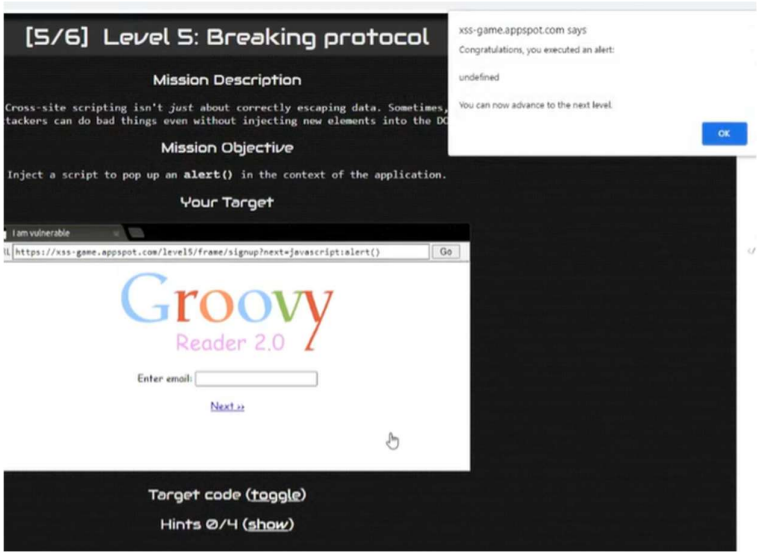


Task 5: Level 5 - Breaking News

This level demonstrates another **DOM-based XSS** that is triggered by client-side navigation.

- **Objective**: Execute an alert by manipulating the URL to control the "Next" button's behavior.
- **Vulnerability Analysis**: The page includes a "Next" link that points to the URL specified in the next query parameter. When a user clicks a link to sign up, the JavaScript changes the location.href of the "Next" button to javascript:signup(). The vulnerability lies in the fact that the next parameter is not validated.
- **Payload**: You can set the next parameter to a javascript: URL. When the user is redirected, this payload will execute.

JavaScript
javascript:alert('XSS')
The full URL would be: https://xss-game.appspot.com/level5/frame/signup?next=javascript:alert('XSS')



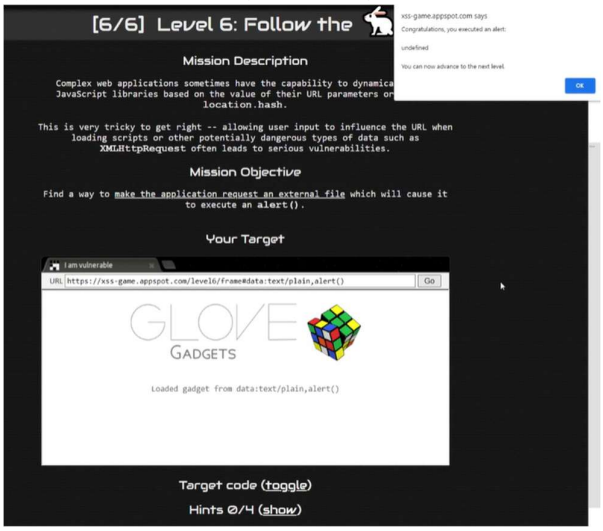
Task 6: Level 6 - Follow the (Advanced DOM-based XSS)

The final level involves bypassing a filter to load a remote script in a **DOM-based XSS** scenario.

- **Objective**: Load and execute a script from an external URL.
- **Vulnerability Analysis**: The application loads a gadget from a URL specified in the URL fragment. However, it includes a filter that only allows URLs starting with https://.... The vulnerability is that this check is not robust enough.
- **Payload**: The filter can be bypassed by providing a URL that appears to be external but is actually a locally executed script. A **data URI** is perfect for this. The browser interprets the data URI and executes the Base64-encoded JavaScript.
- data:text/javascript;base64,YWxlcnQoJ1hTUycp

(Note: YWxlcnQoJ1hTUycp is the Base64 encoding for alert('XSS'))

An alternative payload that also works is using a URL that starts with // after the https://, which can sometimes bypass simple string matching.



Conclusion

This experiment successfully demonstrated the practical exploitation of the three main categories of Cross-Site Scripting. We observed that **Reflected XSS** occurs when unsanitized input is immediately returned to the user, **Stored XSS** occurs when malicious input is saved on the server, and **DOM-based XSS** occurs when client-side scripts unsafely handle user-controlled data. Completing these challenges highlights the critical need for developers to always **validate and sanitize user input** and use **context-aware output encoding** to build secure web applications.

Experiment 8 Configuring and Verifying Standard ACLs

Aim: To understand the function of standard Access Control Lists (ACLs) and to learn the step-by-step process of configuring, applying, and verifying a standard numbered ACL on a Cisco router to filter IP traffic based on the source address.

Prerequisites

- Cisco Packet Tracer simulation software or physical Cisco routers and switches.
- Basic knowledge of Cisco IOS commands and IP addressing.

Network Topology

This lab uses a simple network with two routers, a switch, and two PCs. The goal is to configure an ACL on **Router0** to block traffic from **PC0** from reaching the network connected to **Router1**.

- **Router0**:
  - FastEthernet0/0 (Fa0/0): 10.0.0.1 /8
  - Serial2/0 (S2/0): 192.168.1.1 /24
- **Router1**:
  - Serial2/0 (S2/0): 192.168.1.2 /24
  - FastEthernet0/0 (Fa0/0): 172.16.0.1 /16
- **PC0**: 10.0.0.2 /8 (Gateway: 10.0.0.1)
- **PC1**: 172.16.0.2 /16 (Gateway: 172.16.0.1)

Procedure

This experiment is divided into five main tasks, from initial setup to final verification.

Task 1: Build and Configure the Network

First, build the network topology in Cisco Packet Tracer and configure the IP addresses and routing.

1. **Configure PC0 and PC1**: Assign the static IP addresses, subnet masks, and default gateways as listed in the topology.

Configure Router0:

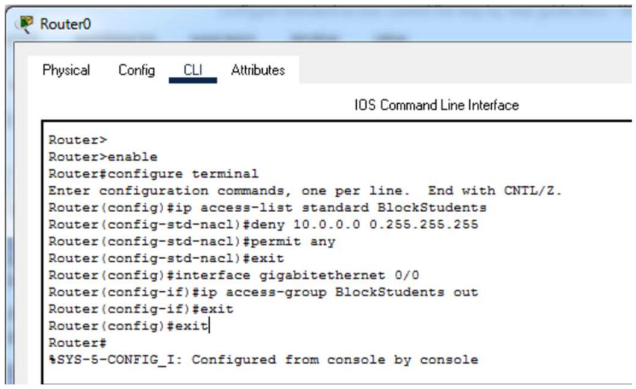
```
enable
configure terminal
interface FastEthernet0/0
ip address 10.0.0.1 255.0.0.0
no shutdown
exit
interface Serial2/0
ip address 192.168.1.1 255.255.255.0
no shutdown
exit
```

Configure Router1:

enable
configure terminal
interface FastEthernet0/0
ip address 172.16.0.1 255.255.0.0
no shutdown
exit
interface Serial2/0
ip address 192.168.1.2 255.255.255.0
no shutdown
exit

Configure Routing: Add a static route on **Router0** so it can reach PC1's network.

! On Router0
ip route 172.16.0.0 255.255.0.0 192.168.1.2

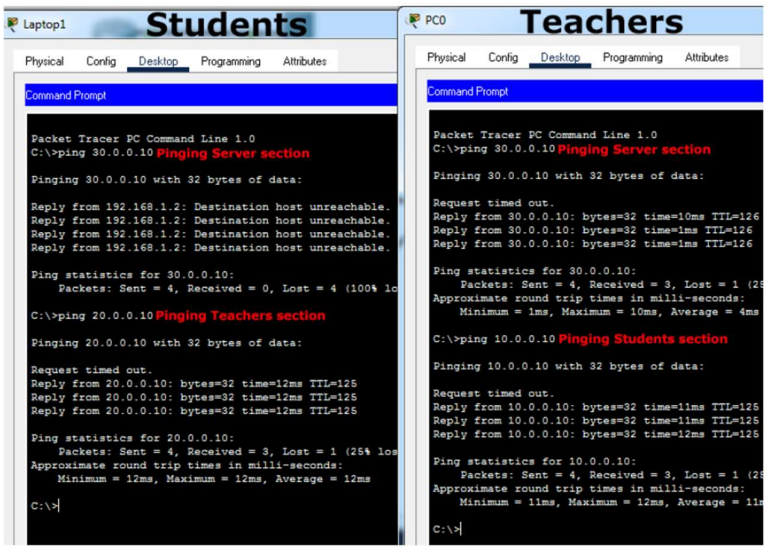


Task 2: Verify Initial Connectivity

Before applying the ACL, confirm that all devices can communicate with each other.

1. From **PC0**, open the command prompt and ping **PC1**.
2. ping 30.0.0.10

The ping should be successful. If not, troubleshoot the IP configurations and routes.

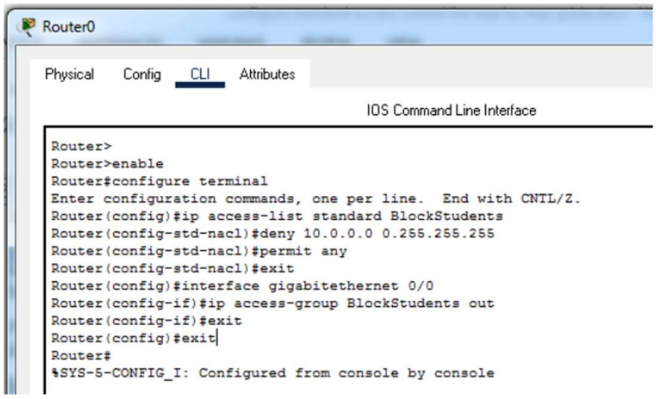


Task 3: Configure the Standard ACL

Now, create the ACL on **Router0**. The goal is to deny traffic from **PC0** (10.0.0.2) but permit all other traffic. Standard numbered ACLs range from 1 to 99.

1. Enter global configuration mode on **Router0**.
2. Create an ACL with the number 10. The first line will deny the specific host, and the second line will permit all other traffic. An implicit "deny all" is at the end of every ACL, so the permit any statement is crucial.

```
! On Router0
configure terminal
access-list 10 deny host 10.0.0.2
access-list 10 permit any
```



Task 4: Apply the ACL to an Interface

An ACL does nothing until it is applied to an interface in a specific direction (either in for inbound traffic or out for outbound traffic).

1. **Identify the Interface and Direction**: To stop traffic from **PC0** from reaching other networks, we should apply the ACL on the interface closest to the source. In this case, that is the **Fa0/0** interface on **Router0**. We will apply it in the **inbound** direction to filter traffic as it enters the router.
2. Apply the ACL to the interface:

```
! On Router0
interface FastEthernet0/0
ip access-group 10 in
exit
```

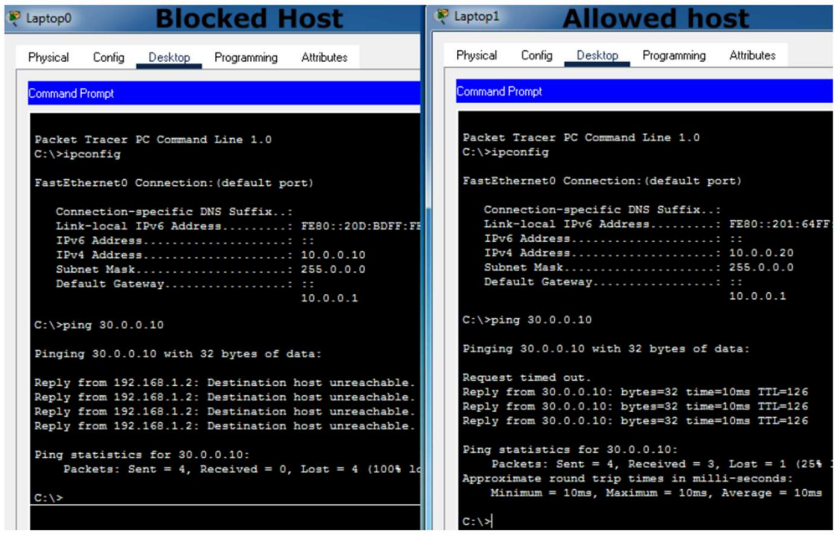
Task 5: Verify the ACL

Finally, test the network again to confirm the ACL is working as expected.

1. **Test the Deny Rule**: From **PC0**, try to ping **PC1** again.
2. ping 172.16.0.2

The ping should now fail, and you should receive a "Destination host unreachable" message from the gateway (10.0.0.1).

3. **Verify the Permit Rule**: If you had another device on the 10.0.0.0/8 network (e.g., a PC at 10.0.0.3), a ping from that device to **PC1** would be successful because of the access-list 10 permit any rule.
4. **Check ACL Statistics**: Use the show access-lists command on **Router0** to see the ACL rules and how many packets have matched each rule.
5. ! On Router0
6. show access-lists



Conclusion

This experiment demonstrated the configuration and application of a standard numbered ACL. We learned that standard ACLs filter traffic based only on the **source IP address** and should be placed as **close to the destination** as possible to avoid blocking traffic unnecessarily. The process involved defining the access rules, applying them to an interface in the correct direction, and verifying the configuration to ensure it met the security objective. This is a foundational skill for network security and traffic management.

Experiment 11

Aim: Configuring Multi-Factor Authentication (MFA) in Azure Active Directory
To enable and configure Multi-Factor Authentication (MFA) in Microsoft Azure Active Directory for enhanced account security.

Theory:
Multi-Factor Authentication (MFA) adds an extra layer of protection to user sign-ins by requiring multiple forms of verification before granting access. Instead of relying solely on a username and password (something you know), MFA introduces additional factors such as mobile app notifications, text messages, or verification codes (something you have).
Azure Active Directory (Azure AD) supports MFA as part of its security features. Administrators can enable MFA for all or selected users to mitigate credential theft and unauthorized access. Azure AD Premium P1/P2 licenses are required to use the full MFA functionality, including conditional access and advanced identity protection.

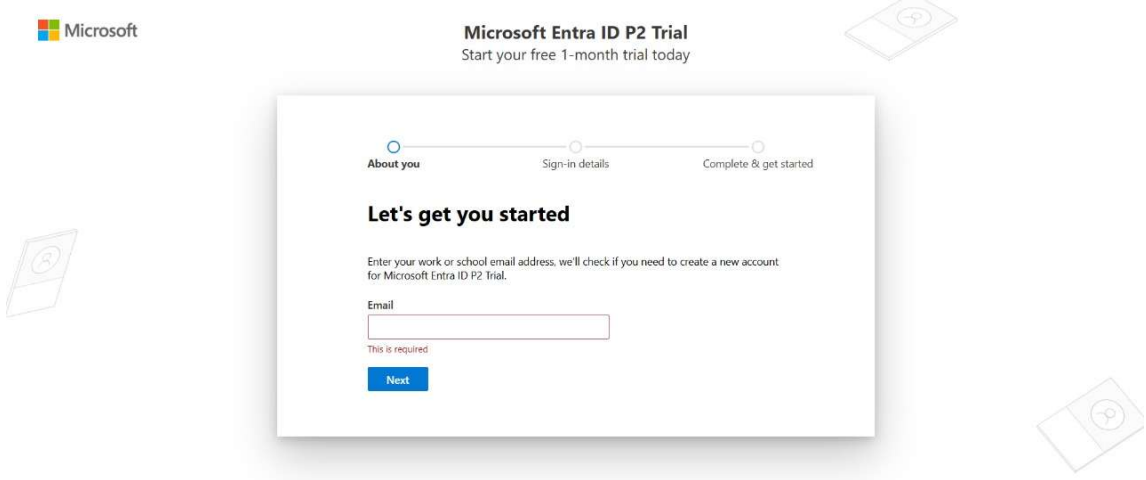
Tool Used: Azure Active Directory (Azure Portal)

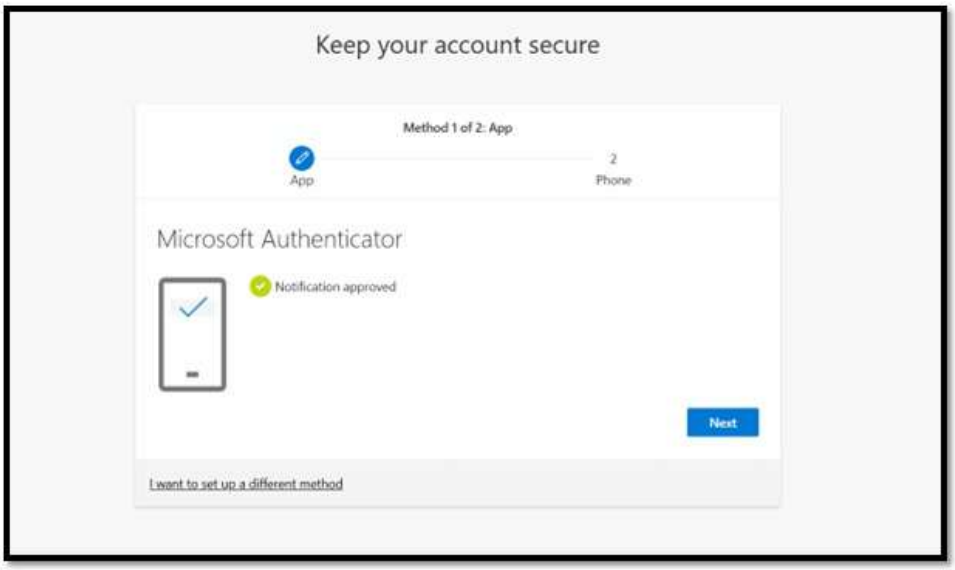
- **Type:** Cloud-based identity and access management service
- **Functions:**
  • Manages users, groups, and authentication policies.
  • Supports MFA, conditional access, and identity protection.
  • Integrates with Microsoft 365, Azure, and third-party apps.
- **Security Importance:**
  o Strengthens authentication by combining multiple verification factors.
  o Protects organizational data from phishing, password leaks, and identity attacks.
  o Ensures compliance with modern security standards (Zero Trust).

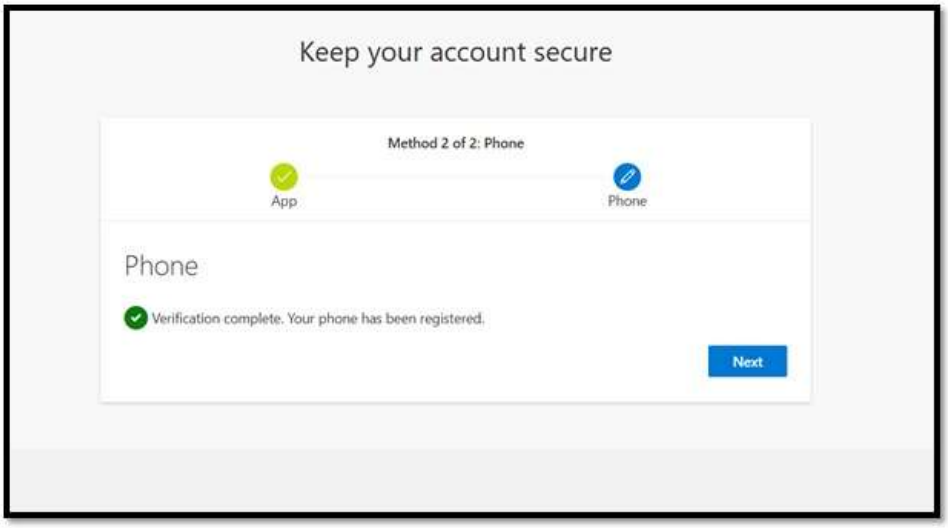
Steps:

Step 1: Enable Azure AD Premium Trial

1. Open **Azure Portal**: https://portal.azure.com
2. Navigate to **Azure Active Directory → Overview**.Click **Upgrade** to start the **Azure AD Premium P1/P2 trial**, required to use MFA features.







Step 2: Configure MFA Settings

1. In **Azure AD**, go to **Security → Multifactor Authentication**.
2. Click **Service settings**.
3. Under **Verification options**, select:
   Mobile app notification
   Mobile app verification code
   Text message to phone
4. Click **Save** to apply settings.

Step 3: Enable MFA for Users

1. In the **MFA portal**, select **Users**.
2. Check the **test user account** for which MFA will be enabled.
3. Click **Enable → Enable multi-factor auth**.
4. On the next login, the selected user will be prompted to configure MFA (via phone or app).