LAB REPORT

Subject: Intrusion Detection and Prevention System

Session 03

Topic name: Write Snort rule

1. GENERAL INFORMATION:

Num	Full name	Student ID	Email
1	Nguyen Dinh Kha	20520562	20520562@gm.uit.edu.vn
2	Le Sy Cuong	20521149	20521149@gm.uit.edu.vn

2. IMPLEMENTATION CONTENT

Num	Work	Personal Responsible	Self-assessment result
1	Requirement 1.1	Nguyen Dinh Kha	100%
2	Requirement 1.2	Nguyen Dinh Kha	100%
3	Requirement 1.3	Nguyen Dinh Kha	100%
4	Requirement 1.4	Le Sy Cuong	100%
5	Requirement 1.5	Le Sy Cuong	100%

The section below of this report is the detailed documentation from the practical group.

DETAILED REPORT

Students undertake the practical exercise with the requirements below.

A. OVERVIEW

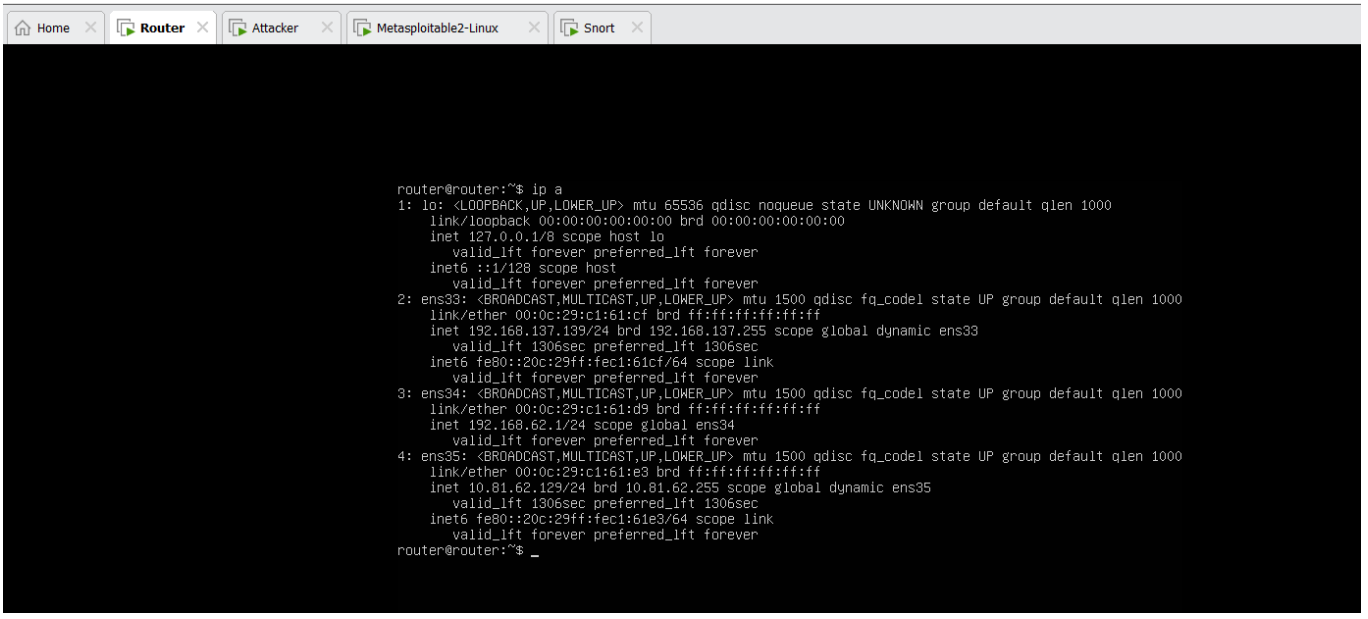
A.1 Objectives

- Learn and write rules for Snort (https://www.snort.org/documents#latest_rule_documents)
- Analyze traffic flows before and after rule deployment.

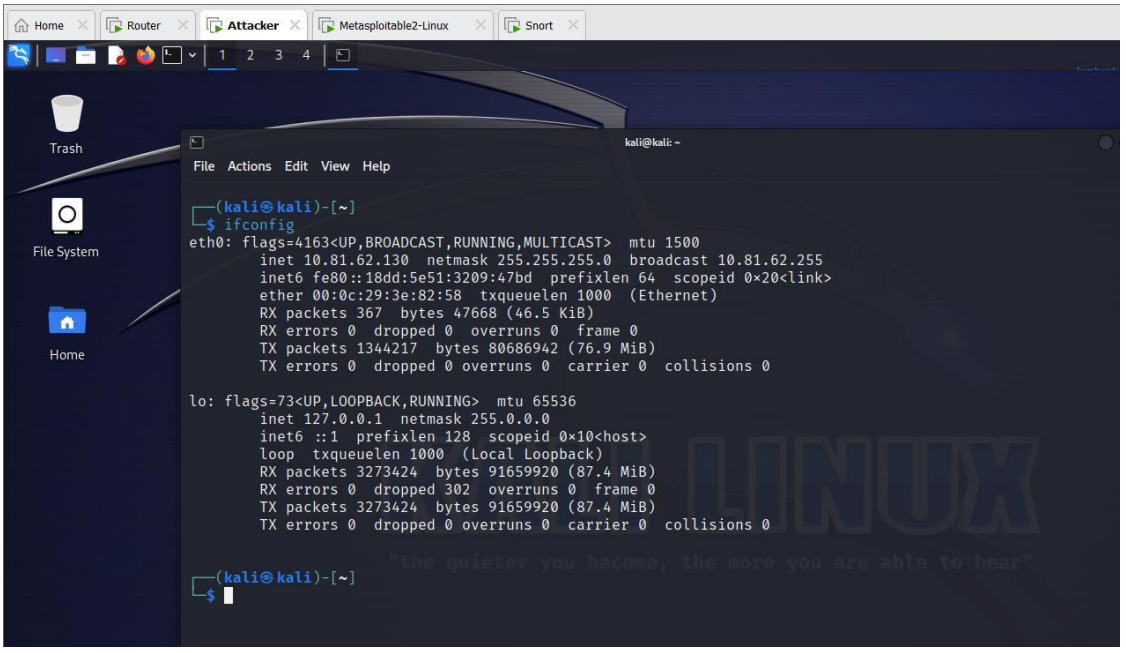
A.2 Setting up the environment

- Use the environment set up in practical exercise 02.

- Configure the IP of the Router:



- Configure the IP of the Attacker's machine:



- Configure the IP of the Victim's machine:

Home | Router | Attacker | Metasploitable2-Linux | Snort

```
msfadmin@metasploitable:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast qlen 1000
    link/ether 00:0c:29:c2:c9:0c brd ff:ff:ff:ff:ff:ff
    inet 192.168.62.200/24 brd 192.168.62.255 scope global eth0
    inet6 fe80::20c:29ff:fec2:c90c/64 scope link
       valid_lft forever preferred_lft forever
msfadmin@metasploitable:~$
```

- Configure the IP of the Snort machine:

Home | Router | Attacker | Metasploitable2-Linux | Snort

```
snort@snort:/var/log/snort$ 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
       valid_lft forever preferred_lft forever
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 00:0c:29:cf:48:69 brd ff:ff:ff:ff:ff:ff
    altname enp2s1
    inet 192.168.137.141/24 metric 100 brd 192.168.137.255 scope global dynamic ens33
       valid_lft 1498sec preferred_lft 1498sec
    inet6 fe80::20c:29ff:fecf:4869/64 scope link
       valid_lft forever preferred_lft forever
3: ens34: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 00:0c:29:cf:48:73 brd ff:ff:ff:ff:ff:ff
    altname enp2s2
    inet6 fe80::20c:29ff:fecf:4873/64 scope link
       valid_lft forever preferred_lft forever
4: ens35: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 00:0c:29:cf:48:7d brd ff:ff:ff:ff:ff:ff
    altname enp2s3
    inet6 fe80::20c:29ff:fecf:487d/64 scope link
       valid_lft forever preferred_lft forever
snort@snort:/var/log/snort$ _
```

- Run Snort Inline:

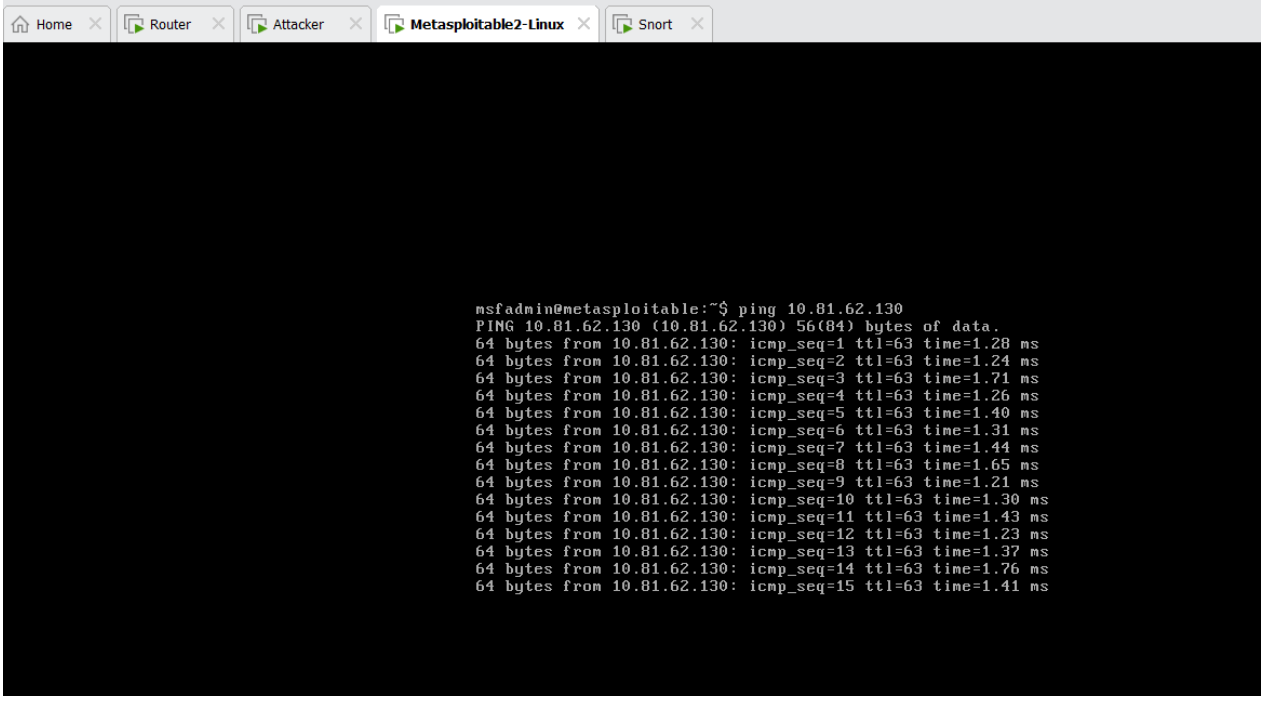
```
snort@snort:/var/log/snort$ sudo snort -c /etc/snort/nhom4-snort.conf -Q -i ens35:ens34_
```

```
Home | Router | Attacker | Metasploitable2-Linux | Snort

+-------------------[Rule Port Counts]-------------------------------------
|             tcp     udp    icmp      ip
|     src       0       0       0       0
|     dst       0       0       0       0
|     any       0       0       0       0
|      nc       0       0       0       0
|     s+d       0       0       0       0
+--------------------------------------------------------------------------

+----------------------[detection-filter-config]-----------------------------
| memory-cap : 1048576 bytes
+----------------------[detection-filter-rules]------------------------------
| none
--------------------------------------------------------------------------

+----------------------[rate-filter-config]----------------------------------
| memory-cap : 1048576 bytes
+----------------------[rate-filter-rules]-----------------------------------
| none
--------------------------------------------------------------------------

+----------------------[event-filter-config]---------------------------------
| memory-cap : 1048576 bytes
+----------------------[event-filter-global]---------------------------------
+----------------------[event-filter-local]----------------------------------
| none
+----------------------[suppression]-----------------------------------------
| none
--------------------------------------------------------------------------
Rule application order: pass->drop->sdrop->reject->alert->log
Verifying Preprocessor Configurations!
afpacket DAQ configured to inline.
Acquiring network traffic from "ens35:ens34".
Reload thread starting...
Reload thread started, thread 0x7fa8dde206c0 (2037)

        --== Initialization Complete ==--

           -*> Snort! <*-
  ,,_     Version 2.9.15.1 GRE (Build 15125)
  o" )~   By Martin Roesch & The Snort Team: http://www.snort.org/contact#team
   ''''   Copyright (C) 2014-2019 Cisco and/or its affiliates. All rights reserved.
          Copyright (C) 1998-2013 Sourcefire, Inc., et al.
          Using libpcap version 1.10.3 (with TPACKET_V3)
          Using PCRE version: 8.39 2016-06-14
          Using ZLIB version: 1.2.13

Commencing packet processing (pid=2028)
Decoding Ethernet
```

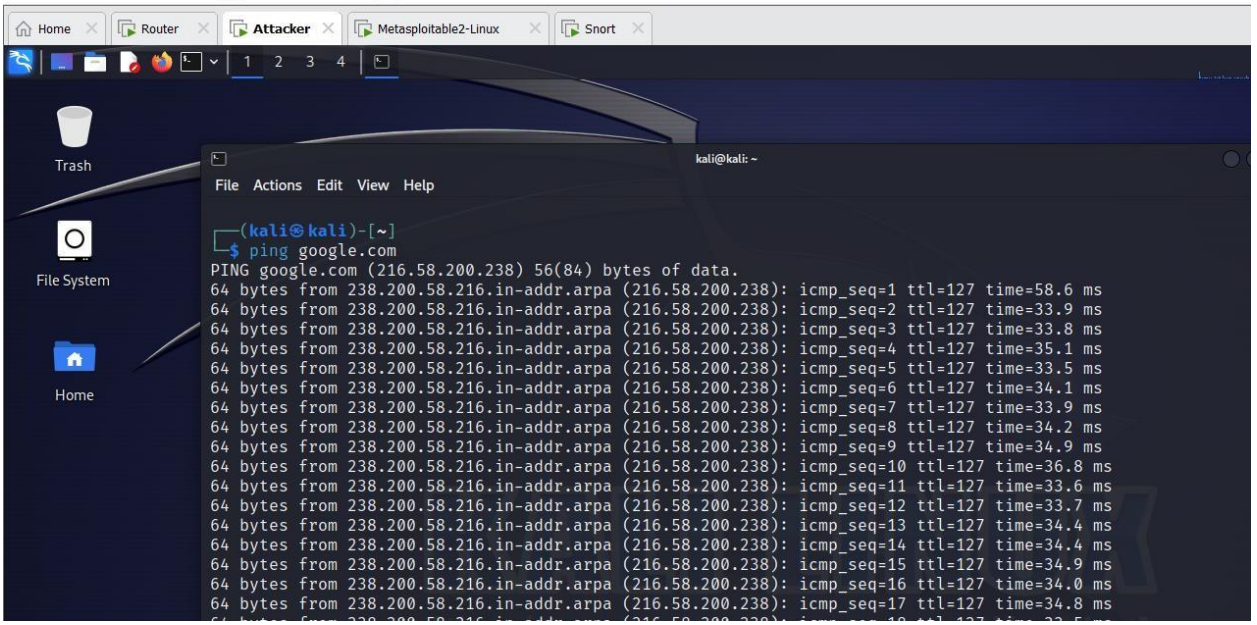
- Check if the Attacker's machine can ping the Victim's machine:

```
Home | Router | Attacker | Metasploitable2-Linux | Snort
1 2 3 4

Trash
File System
Home

kali@kali: ~
File  Actions  Edit  View  Help

┌──(kali㉿kali)-[~]
└─$ ping 192.168.62.200
PING 192.168.62.200 (192.168.62.200) 56(84) bytes of data.
64 bytes from 192.168.62.200: icmp_seq=1 ttl=63 time=1.72 ms
64 bytes from 192.168.62.200: icmp_seq=2 ttl=63 time=1.32 ms
64 bytes from 192.168.62.200: icmp_seq=3 ttl=63 time=1.47 ms
64 bytes from 192.168.62.200: icmp_seq=4 ttl=63 time=1.23 ms
64 bytes from 192.168.62.200: icmp_seq=5 ttl=63 time=1.35 ms
64 bytes from 192.168.62.200: icmp_seq=6 ttl=63 time=1.27 ms
```

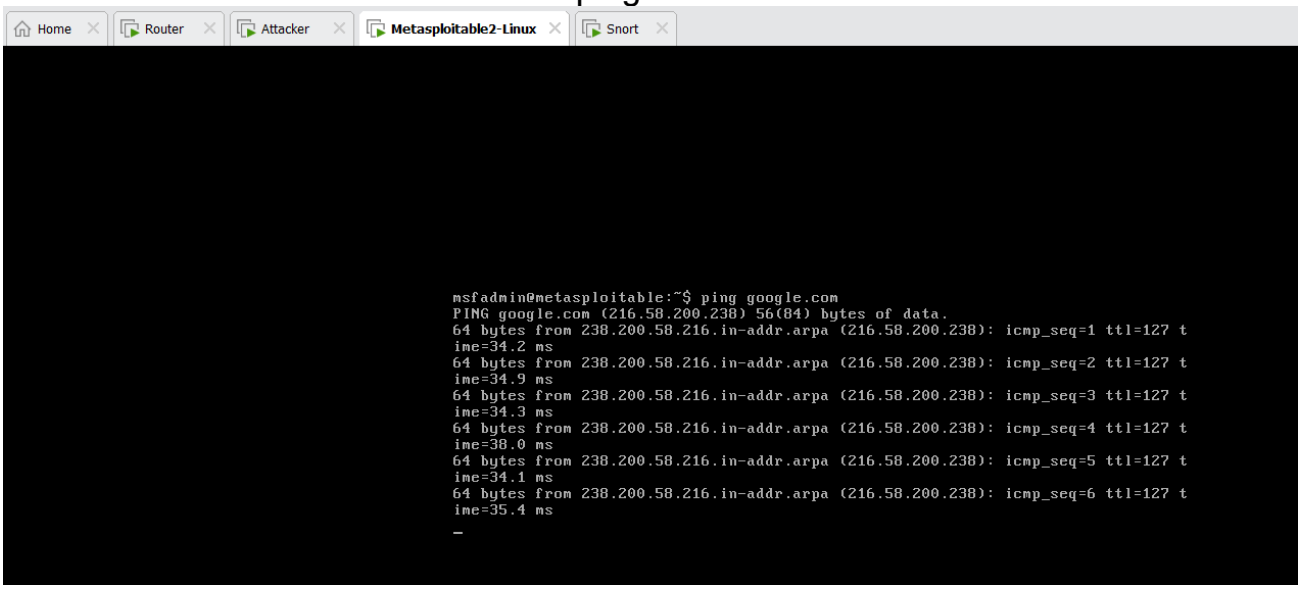
- Check if the Victim's machine can ping the Attacker's machine:



- Kiểm tra máy Attacker ping ra internet:



- Check if the Attacker's machine can ping the internet:



B. PRACTICE

Students will perform practical exercises with the following requirements.

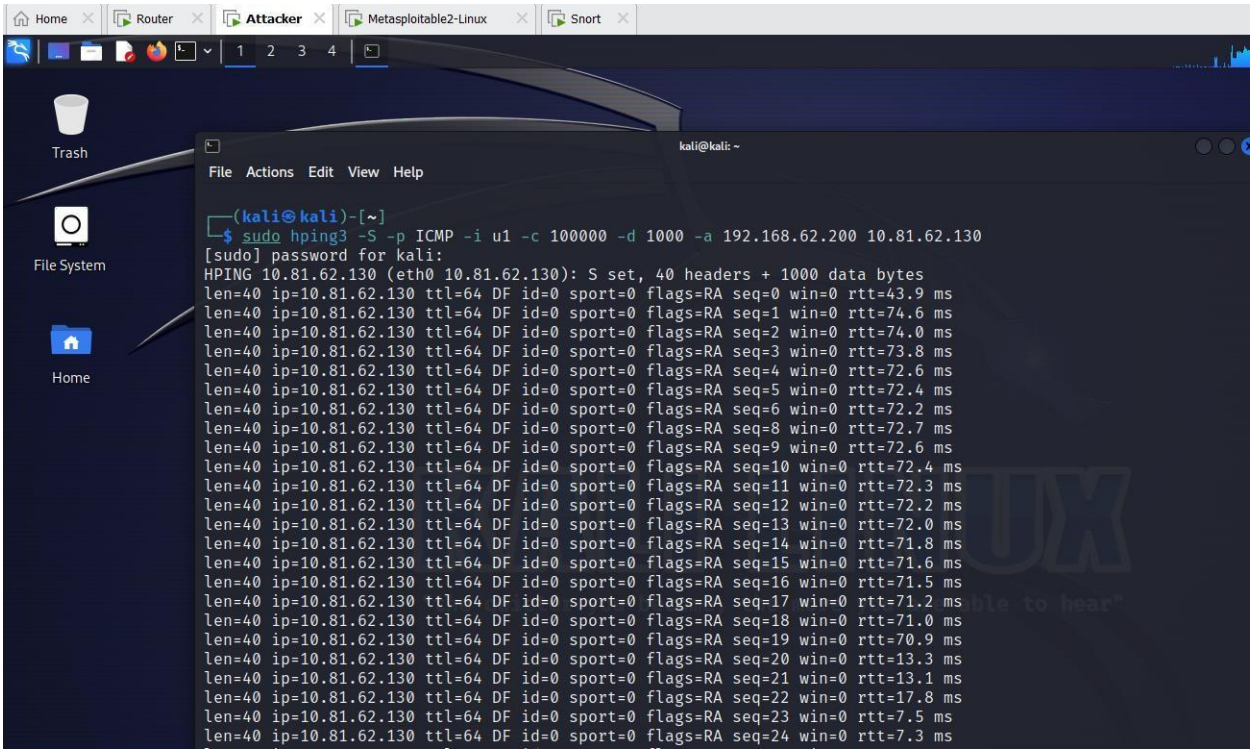
B.1 Write a rule for Snort

Requirement 1.1 Prevenet ICMP Flood attacks

- Write a Snort rule to limit ICMP packets to the Victim's machine. The threshold is not more than 23 packets/5s.
- Use the hping3 tool on the Attacker's machine to carry out the attack.
- Check the results before and after installing the rule.

- Before installing the rule:

- The Attacker's machine carries out an ICMP Flood attack:



- Victim's machine: Use tcpdump to check if the Victim's machine has been attacked by ICMP Flood:

Home | Router | Attacker | Metasploitable2-Linux | Snort

```
02:58:59.580366 IP 10.81.62.130.0 > 192.168.62.200.9937: R 0:0(0) ack 1271452230
 win 0
02:58:59.580454 IP 10.81.62.130.0 > 192.168.62.200.9938: R 0:0(0) ack 1352815421
 win 0
02:58:59.580556 IP 10.81.62.130.0 > 192.168.62.200.9939: R 0:0(0) ack 1263648525
 win 0
02:58:59.580663 IP 10.81.62.130.0 > 192.168.62.200.9940: R 0:0(0) ack 976794618
win 0
02:58:59.580781 IP 10.81.62.130.0 > 192.168.62.200.9941: R 0:0(0) ack 2021910026
 win 0
02:58:59.580852 IP 10.81.62.130.0 > 192.168.62.200.9942: R 0:0(0) ack 13970830 w
in 0
02:58:59.582903 IP 10.81.62.130.0 > 192.168.62.200.9960: R 0:0(0) ack 1121557075
 win 0
02:58:59.588014 IP 10.81.62.130.0 > 192.168.62.200.10013: R 0:0(0) ack 193465547
6 win 0
02:58:59.592707 IP 10.81.62.130.0 > 192.168.62.200.10062: R 0:0(0) ack 235795573
 win 0
02:58:59.599124 IP 10.81.62.130.0 > 192.168.62.200.10123: R 0:0(0) ack 44824735
win 0
02:58:59.601404 IP 10.81.62.130.0 > 192.168.62.200.10124: R 0:0(0) ack 139468436
 win 0
02:58:59.604462 IP 10.81.62.130.0 > 192.168.62.200.10125: R 0:0(0) ack 60913987
win 0
02:58:59.605970 IP 10.81.62.130.0 > 192.168.62.200.10126: R 0:0(0) ack 157488336
```

Home | Router | Attacker | Metasploitable2-Linux | Snort

```
02:59:08.943865 IP 10.81.62.130.0 > 192.168.62.200.27962: R 0:0(0) ack 143405319
2 win 0
02:59:08.945379 IP 10.81.62.130.0 > 192.168.62.200.27963: R 0:0(0) ack 176849522
5 win 0
02:59:08.946879 IP 10.81.62.130.0 > 192.168.62.200.27964: R 0:0(0) ack 142953669
2 win 0
02:59:08.948460 IP 10.81.62.130.0 > 192.168.62.200.27965: R 0:0(0) ack 656500861
 win 0
02:59:08.949956 IP 10.81.62.130.0 > 192.168.62.200.27966: R 0:0(0) ack 311628228
 win 0
02:59:08.951540 IP 10.81.62.130.0 > 192.168.62.200.27967: R 0:0(0) ack 536389711
 win 0
02:59:08.952959 IP 10.81.62.130.0 > 192.168.62.200.27968: R 0:0(0) ack 148007702
1 win 0
02:59:08.954297 IP 10.81.62.130.0 > 192.168.62.200.27969: R 0:0(0) ack 822012934
 win 0
02:59:08.955880 IP 10.81.62.130.0 > 192.168.62.200.27970: R 0:0(0) ack 224322355
 win 0
02:59:08.957309 IP 10.81.62.130.0 > 192.168.62.200.27971: R 0:0(0) ack 195403632
6 win 0

7906 packets captured
60739 packets received by filter
52470 packets dropped by kernel
msfadmin@metasploitable:~$ _
```

- ⇨ Using tcpdump, we observe that the Victim's machine has been attacked by ICMP Flood.

⇨ However, Snort did not generate an alert because no rule has been set for Snort yet:

```
Would       icmp4::echo: 0
                ip6::ttl: 0
Would           ip6::ttl: 0
               ip6::opts: 10
Would          ip6::opts: 0
              icmp6::echo: 0
Would       icmp6::echo: 0
            tcp::syn_opt: 0
Would       tcp::syn_opt: 0
                tcp::opt: 0
Would           tcp::opt: 0
                tcp::pad: 0
Would           tcp::pad: 0
                tcp::rsv: 0
Would           tcp::rsv: 0
                 tcp::ns: 0
Would            tcp::ns: 0
                tcp::urp: 0
Would           tcp::urp: 0
            tcp::ecn_pkt: 0
Would       tcp::ecn_pkt: 0
             tcp::ts_ecr: 0
Would        tcp::ts_ecr: 0
            tcp::req_urg: 0
Would       tcp::req_urg: 0
            tcp::req_pay: 0
Would       tcp::req_pay: 0
            tcp::req_urp: 0
Would       tcp::req_urp: 0
            tcp::ecn_ssn: 0
Would       tcp::ecn_ssn: 0
             tcp::ts_nop: 0
Would        tcp::ts_nop: 0
           tcp::ips_data: 0
Would      tcp::ips_data: 0
              tcp::block: 0
Would         tcp::block: 0
           tcp::trim_syn: 0
Would      tcp::trim_syn: 0
           tcp::trim_rst: 0
Would      tcp::trim_rst: 0
           tcp::trim_win: 0
Would      tcp::trim_win: 0
           tcp::trim_mss: 0
Would      tcp::trim_mss: 0
===============================================================================
Snort exiting
snort@snort:/var/log/snort$
snort@snort:/var/log/snort$ cat alert
snort@snort:/var/log/snort$
```

⇨ Checking the alert file of Snort, we see that it's empty, meaning Snort has not detected anything because no rule has been set.

- After setting the rule:

- We proceed to set the rule for Snort:

```
snort@snort:/var/log/snort$
snort@snort:/var/log/snort$ cat /etc/snort/rules/nhom4.rules
alert icmp any any -> 192.168.62.200 any (msg: "ICMP Flood to 192.168.62.200"; itype:8; threshold: type both ,track by_src, count 23, seconds 5; sid:1000001; re
v:1;)
snort@snort:/var/log/snort$ _
```

(alert icmp any any -> 192.168.62.200 any (msg: “ICMP Flood to 192.16862.200”; itype:8; threshold: type both ,track by_src, count 23, seconds 5; sid: 1000001; rev:1;)

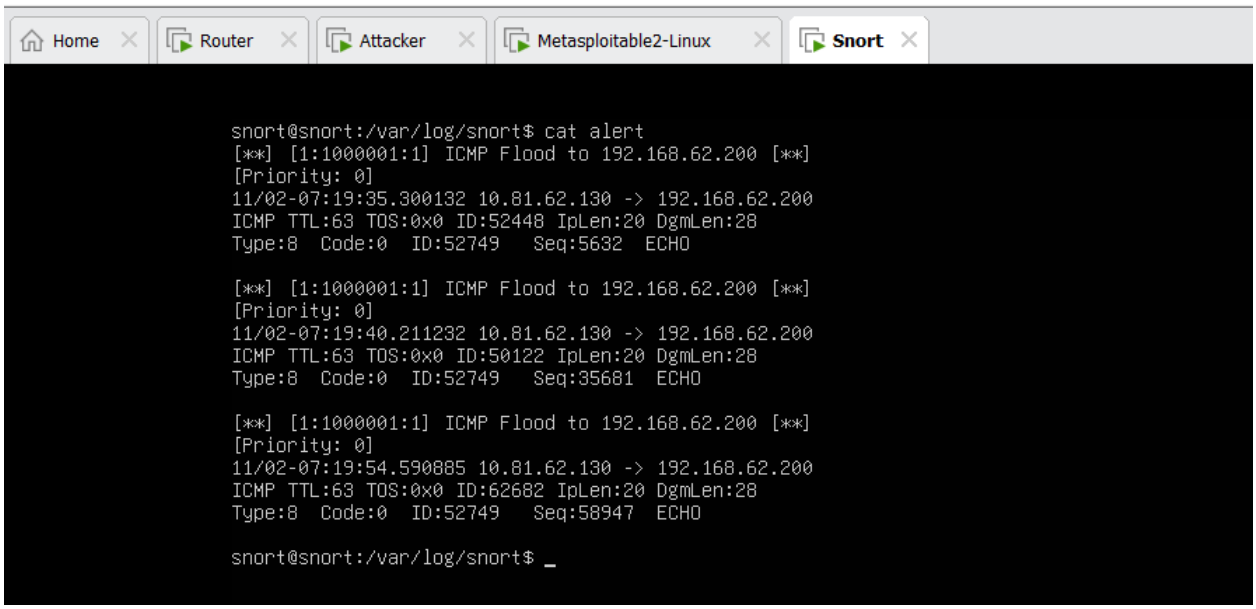
- After setting the rule, we carry out the ICMP Flood attack again on the Attacker's machine:

```
(kali㉿kali)-[~]
└─$ sudo hping3 -S -p ICMP -i u1 -c 100000 -d 1000 -a 192.168.62.200 10.81.62.130
[sudo] password for kali:
HPING 10.81.62.130 (eth0 10.81.62.130): S set, 40 headers + 1000 data bytes
len=40 ip=10.81.62.130 ttl=64 DF id=0 sport=0 flags=RA seq=0 win=0 rtt=43.9 ms
len=40 ip=10.81.62.130 ttl=64 DF id=0 sport=0 flags=RA seq=1 win=0 rtt=74.6 ms
len=40 ip=10.81.62.130 ttl=64 DF id=0 sport=0 flags=RA seq=2 win=0 rtt=74.0 ms
len=40 ip=10.81.62.130 ttl=64 DF id=0 sport=0 flags=RA seq=3 win=0 rtt=73.8 ms
len=40 ip=10.81.62.130 ttl=64 DF id=0 sport=0 flags=RA seq=4 win=0 rtt=72.6 ms
len=40 ip=10.81.62.130 ttl=64 DF id=0 sport=0 flags=RA seq=5 win=0 rtt=72.4 ms
len=40 ip=10.81.62.130 ttl=64 DF id=0 sport=0 flags=RA seq=6 win=0 rtt=72.2 ms
len=40 ip=10.81.62.130 ttl=64 DF id=0 sport=0 flags=RA seq=8 win=0 rtt=72.7 ms
len=40 ip=10.81.62.130 ttl=64 DF id=0 sport=0 flags=RA seq=9 win=0 rtt=72.6 ms
len=40 ip=10.81.62.130 ttl=64 DF id=0 sport=0 flags=RA seq=10 win=0 rtt=72.4 ms
len=40 ip=10.81.62.130 ttl=64 DF id=0 sport=0 flags=RA seq=11 win=0 rtt=72.3 ms
len=40 ip=10.81.62.130 ttl=64 DF id=0 sport=0 flags=RA seq=12 win=0 rtt=72.2 ms
len=40 ip=10.81.62.130 ttl=64 DF id=0 sport=0 flags=RA seq=13 win=0 rtt=72.0 ms
len=40 ip=10.81.62.130 ttl=64 DF id=0 sport=0 flags=RA seq=14 win=0 rtt=71.8 ms
len=40 ip=10.81.62.130 ttl=64 DF id=0 sport=0 flags=RA seq=15 win=0 rtt=71.6 ms
len=40 ip=10.81.62.130 ttl=64 DF id=0 sport=0 flags=RA seq=16 win=0 rtt=71.5 ms
len=40 ip=10.81.62.130 ttl=64 DF id=0 sport=0 flags=RA seq=17 win=0 rtt=71.2 ms
len=40 ip=10.81.62.130 ttl=64 DF id=0 sport=0 flags=RA seq=18 win=0 rtt=71.0 ms
len=40 ip=10.81.62.130 ttl=64 DF id=0 sport=0 flags=RA seq=19 win=0 rtt=70.9 ms
len=40 ip=10.81.62.130 ttl=64 DF id=0 sport=0 flags=RA seq=20 win=0 rtt=13.3 ms
len=40 ip=10.81.62.130 ttl=64 DF id=0 sport=0 flags=RA seq=21 win=0 rtt=13.1 ms
len=40 ip=10.81.62.130 ttl=64 DF id=0 sport=0 flags=RA seq=22 win=0 rtt=17.8 ms
len=40 ip=10.81.62.130 ttl=64 DF id=0 sport=0 flags=RA seq=23 win=0 rtt=7.5 ms
len=40 ip=10.81.62.130 ttl=64 DF id=0 sport=0 flags=RA seq=24 win=0 rtt=7.3 ms
```

- The Victim's machine uses tcpdump to check:

```
03:03:10.553050 IP 10.81.62.130.0 > 192.168.62.200.5247: R 0:0(0) ack 1181366097
 win 0
03:03:10.553299 IP 10.81.62.130.0 > 192.168.62.200.5248: R 0:0(0) ack 1002419361
 win 0
03:03:10.553548 IP 10.81.62.130.0 > 192.168.62.200.5249: R 0:0(0) ack 248493722
win 0
03:03:10.553684 IP 10.81.62.130.0 > 192.168.62.200.5250: R 0:0(0) ack 171777785
win 0
03:03:10.553974 IP 10.81.62.130.0 > 192.168.62.200.5251: R 0:0(0) ack 384308921
win 0
03:03:10.554121 IP 10.81.62.130.0 > 192.168.62.200.5252: R 0:0(0) ack 1835783624
 win 0
03:03:10.554267 IP 10.81.62.130.0 > 192.168.62.200.5253: R 0:0(0) ack 755448768
win 0
03:03:10.554429 IP 10.81.62.130.0 > 192.168.62.200.5254: R 0:0(0) ack 83891704 w
in 0
03:03:10.554575 IP 10.81.62.130.0 > 192.168.62.200.5255: R 0:0(0) ack 444959123
win 0
03:03:10.554682 IP 10.81.62.130.0 > 192.168.62.200.5256: R 0:0(0) ack 2056903365
 win 0
03:03:10.554778 IP 10.81.62.130.0 > 192.168.62.200.5257: R 0:0(0) ack 291854584
win 0
03:03:10.554985 IP 10.81.62.130.0 > 192.168.62.200.5258: R 0:0(0) ack 1570126016
 win 0
03:03:10.555094 IP 10.81.62.130.0 > 192.168.62.200.5259: R 0:0(0) ack 1455608327
```

- On the Snort side, ICMP Flood has been detected:



⇨ Snort has detected and alerted ICMP Flood to the Victim's machine (192.168.62.200)

Requirement 1.2 Allow access only to services running on the Victim

- Use nmap to scan open ports on the Victim's machine.
- Write a Snort rule to only allow access to the open ports of the Victim's machine. Block all other ports.
- Use telnet or nmap tools on the Attacker's machine to carry out the attack.
- Check the results before and after installing the rule.

- Before setting the rule:

- The Attacker's machine conducts an nmap attack:

```
┌──(kali㉿kali)-[~]
└─$ sudo nmap  192.168.62.200
Starting Nmap 7.94 ( https://nmap.org ) at 2023-11-02 05:55 EDT
Nmap scan report for 192.168.62.200 (192.168.62.200)
Host is up (0.027s latency).
Not shown: 977 closed tcp ports (reset)
PORT     STATE SERVICE
21/tcp   open  ftp
22/tcp   open  ssh
23/tcp   open  telnet
25/tcp   open  smtp
53/tcp   open  domain
80/tcp   open  http
111/tcp  open  rpcbind
139/tcp  open  netbios-ssn
445/tcp  open  microsoft-ds
512/tcp  open  exec
513/tcp  open  login
514/tcp  open  shell
1099/tcp open  rmiregistry
1524/tcp open  ingreslock
2049/tcp open  nfs
2121/tcp open  ccproxy-ftp
3306/tcp open  mysql
5432/tcp open  postgresql
5900/tcp open  vnc
6000/tcp open  X11
6667/tcp open  irc
8009/tcp open  ajp13
8180/tcp open  unknown

Nmap done: 1 IP address (1 host up) scanned in 1.77 seconds
```

- After setting the rule:

- We proceed to set the rule for Snort:

```
snort@snort:/var/log/snort$ sudo cat /etc/snort/rules/nhom4.rules
#alert icmp any any -> 192.168.62.200 any (msg: "ICMP Flood to 192.168.62.200"; threshold: type both, track by_src, count 23, seconds 5; sid:1000001; rev:1;)

# Rule to allow access to all ports on Victim
alert tcp any any -> 192.168.62.200 any (msg:"Allowed TCP Connection"; sid: 1000001; rev:1;)
alert udp any any -> 192.168.62.200 any (msg:"Allowed UDP Connection"; sid: 1000002; rev:1;)

# Rule to block all other ports
alert tcp any any -> 192.168.62.200 !80,!443,!22,!53 (msg:"Blocked TCP Connection"; sid: 100003; rev:1;)
alert udp any any -> 192.168.62.200 !80,!443,!22,!53 (msg:"Blocked UDP Connection"; sid: 100004; rev:1;)

snort@snort:/var/log/snort$
```

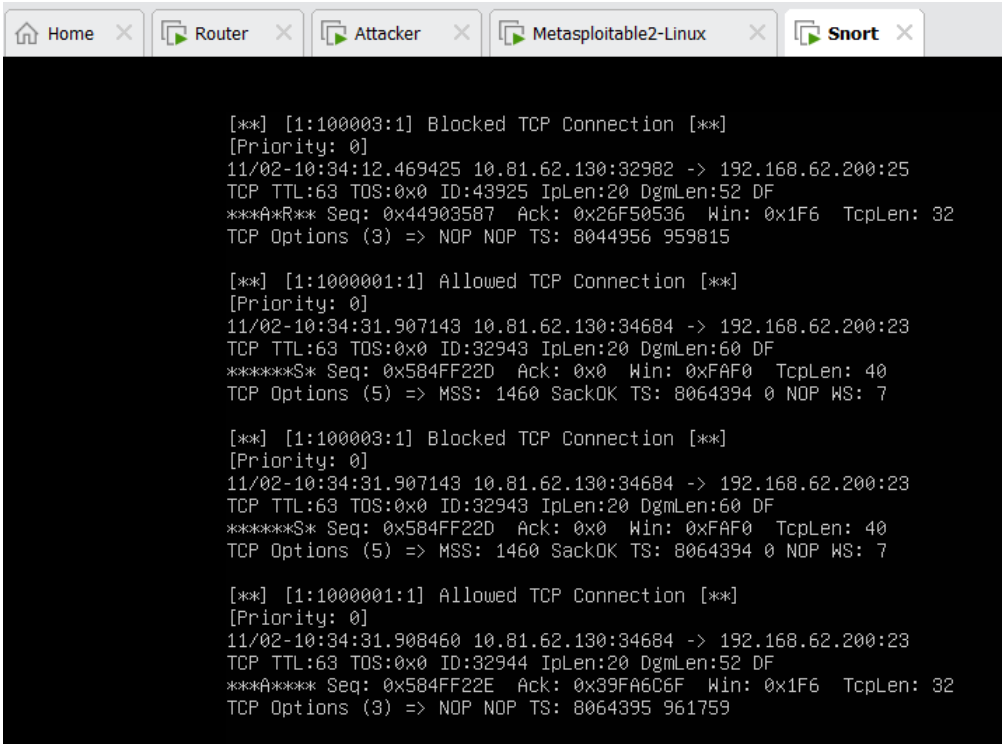
⇨ The # Rule to allow access to ports on the victim should be changed from alert to pass because alert is used to see if the rule works or not.

⇨ The part to Block ports should set a rule using alert except for ports (80, 443, 22, 53)

The Attacker attacks nmap on some specified ports:

```
┌──(kali㉿kali)-[~]
└─$ sudo nmap -sT -p 80 -Pn 192.168.62.200

Starting Nmap 7.94 ( https://nmap.org ) at 2023-11-02 06:32 EDT
Nmap scan report for 192.168.62.200 (192.168.62.200)
Host is up (0.0018s latency).

PORT   STATE SERVICE
80/tcp open  http

Nmap done: 1 IP address (1 host up) scanned in 0.03 seconds

┌──(kali㉿kali)-[~]
└─$ sudo nmap -sT -p 443 -Pn 192.168.62.200

Starting Nmap 7.94 ( https://nmap.org ) at 2023-11-02 06:32 EDT
Nmap scan report for 192.168.62.200 (192.168.62.200)
Host is up (0.016s latency).

PORT    STATE  SERVICE
443/tcp closed https

Nmap done: 1 IP address (1 host up) scanned in 0.04 seconds

┌──(kali㉿kali)-[~]
└─$ sudo nmap -sT -p 22 -Pn 192.168.62.200

Starting Nmap 7.94 ( https://nmap.org ) at 2023-11-02 06:33 EDT
Nmap scan report for 192.168.62.200 (192.168.62.200)
Host is up (0.0032s latency).

PORT   STATE SERVICE
22/tcp open  ssh

Nmap done: 1 IP address (1 host up) scanned in 0.03 seconds

┌──(kali㉿kali)-[~]
└─$ sudo nmap -sT -p 53 -Pn 192.168.62.200

Starting Nmap 7.94 ( https://nmap.org ) at 2023-11-02 06:33 EDT
Nmap scan report for 192.168.62.200 (192.168.62.200)
Host is up (0.0018s latency).

PORT   STATE SERVICE
53/tcp open  domain

Nmap done: 1 IP address (1 host up) scanned in 0.04 seconds
```

- On the Snort machine, alerts have appeared:



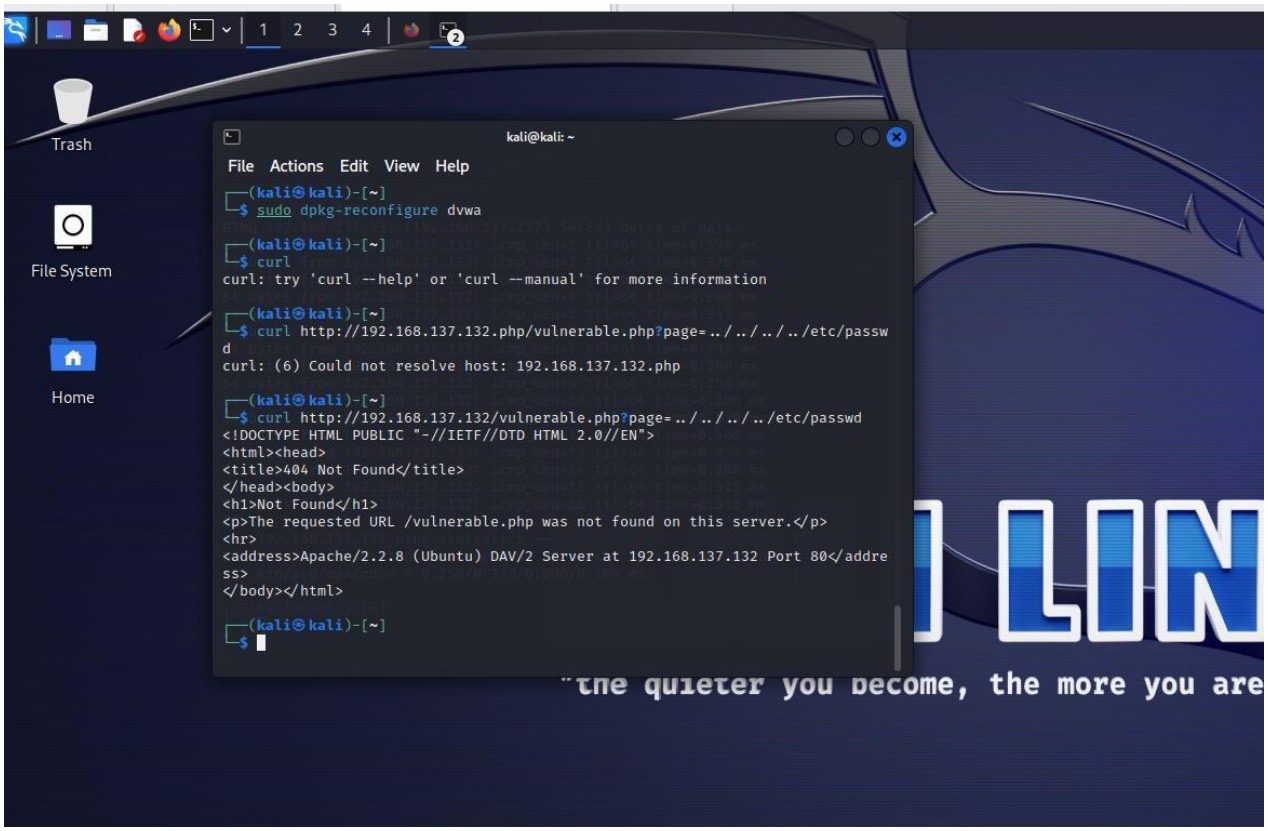
Requirement 1.3 Prevent password scanning attacks on the Web application

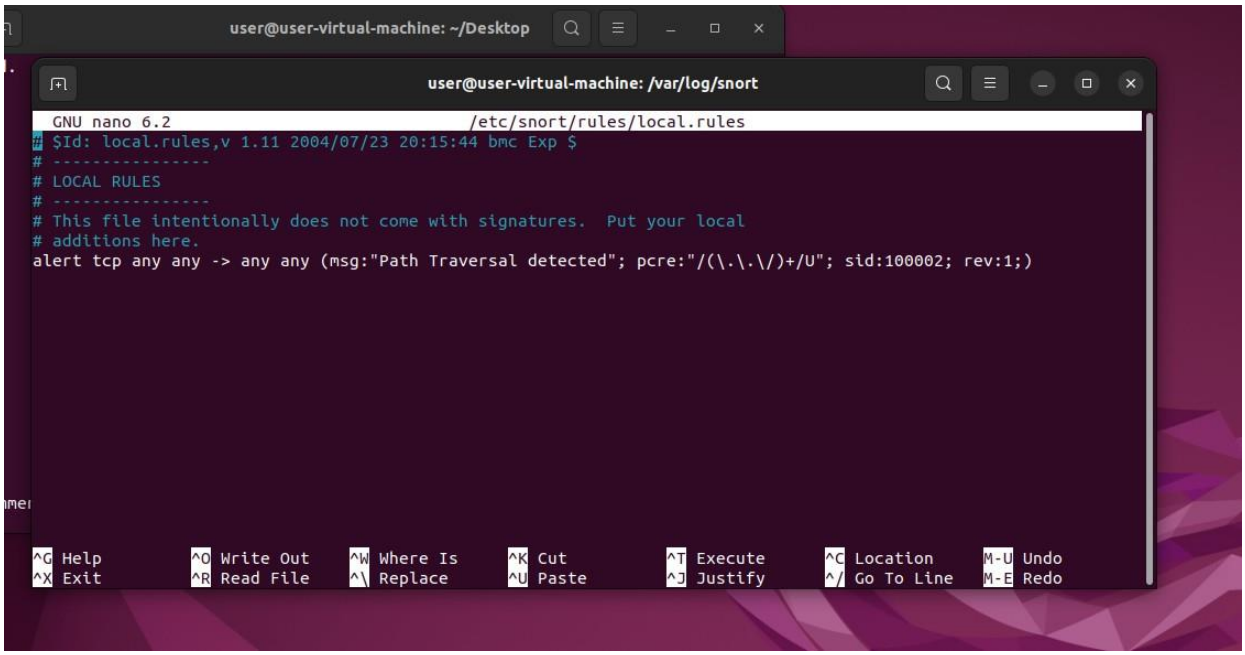
- Access the Mutillidae web application (/mutillidae/index.php?page=login.php) on the Victim's machine. Write a Snort rule to prevent password login scanning attacks on this web application. Note: only block password scanning, the web application should still be accessible normally.
- Use the hydra tool on the Attacker's machine to carry out the attack.
- Check the results before and after installing the rule.

Requirement 1.4 Prevent Path Traversal attacks

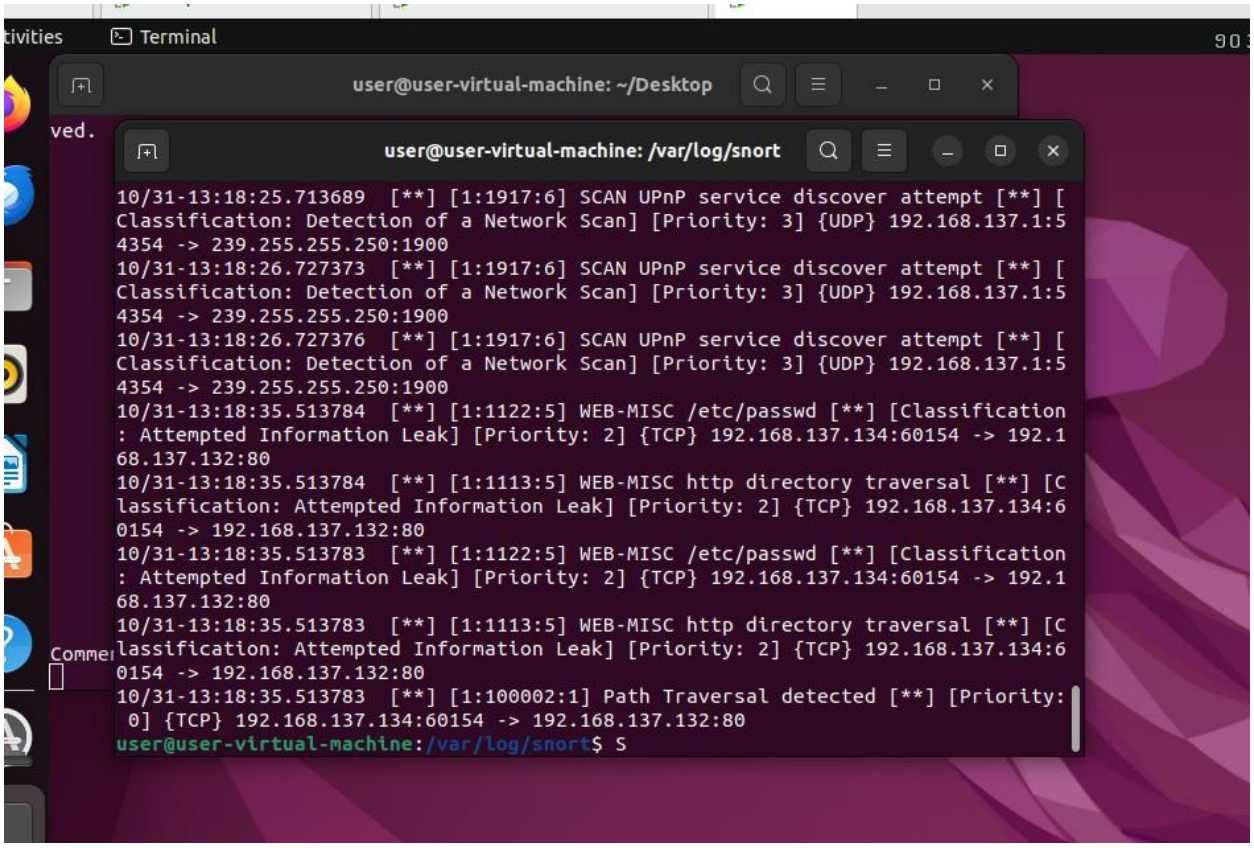
- Write a Snort rule to block Path Traversal attacks.
- On the Attacker's machine, access the path http://192.168.x.200/dvwa/ to carry out the attack.
- Check the results before and after carrying out the attack.

● We proceed to carry out the Path Traversal attack with the Curl command, the Victim's machine IP is 192.168.137.132





⇨ Set Snort rule to detect Path Traversal with content that will detect strings containing one or more repetitions of "../", which is also a sign of Path Traversal



⇨ On the Snort machine, Path Traversal attack has been detected

> Requirement 1.5 Students are required to develop 2 additional attack scenarios and write Snort rules to prevent attacks.

- Students independently develop 2 unrelated attack scenarios not related to DoS and web attacks, then write Snort rules to prevent the attacks.
- Perform writing Snort rules, check the results before and after the attacks as above requirements.
- Evaluation depends on the complexity of the scenarios.

⇨ .

<u>Network scan attack:</u>

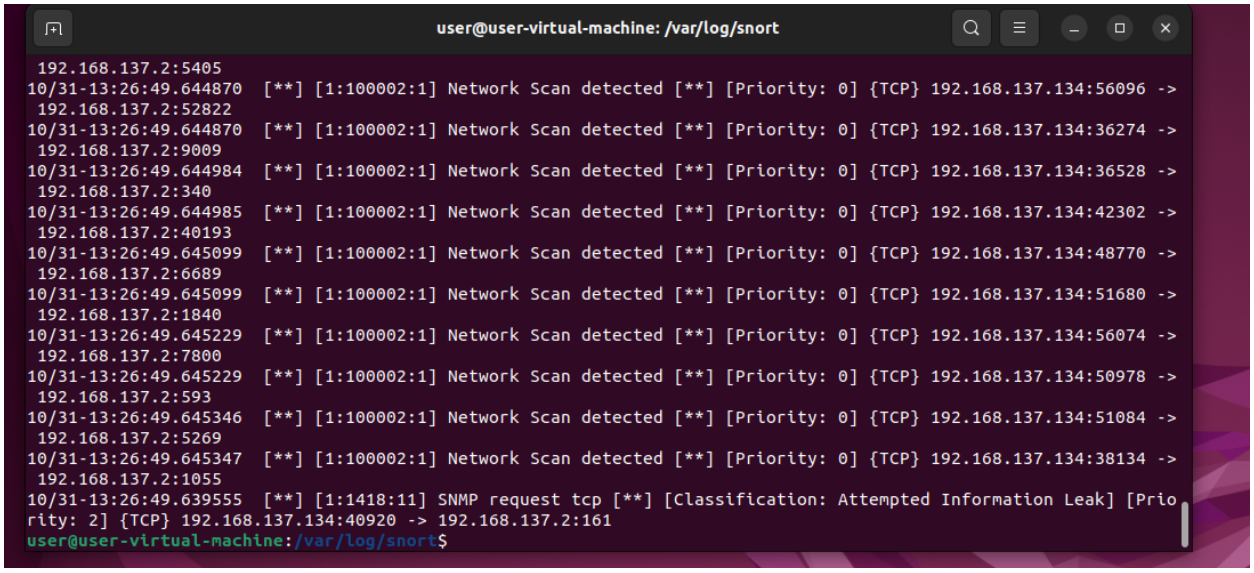
```
kali@kali: ~
File  Actions  Edit  View  Help
</body></html>

┌──(kali㉿kali)-[~]
└─$ nmap 192.168.137.0/24
Starting Nmap 7.94 ( https://nmap.org ) at 2023-10-31 02:26 EDT
Nmap scan report for 192.168.137.2
Host is up (0.00044s latency).
Not shown: 999 closed tcp ports (conn-refused)
PORT   STATE SERVICE
53/tcp open  domain

Nmap scan report for 192.168.137.129
Host is up (0.0014s latency).
Not shown: 999 closed tcp ports (conn-refused)
PORT   STATE SERVICE
80/tcp open  http

Nmap scan report for 192.168.137.132
Host is up (0.0028s latency).
Not shown: 977 closed tcp ports (conn-refused)
PORT     STATE SERVICE
21/tcp   open  ftp
22/tcp   open  ssh
23/tcp   open  telnet
25/tcp   open  smtp
53/tcp   open  domain
80/tcp   open  http
```

On the Attacker's machine, we perform a scan using nmap.

On the Snort machine, we set rules to detect Network Scan:

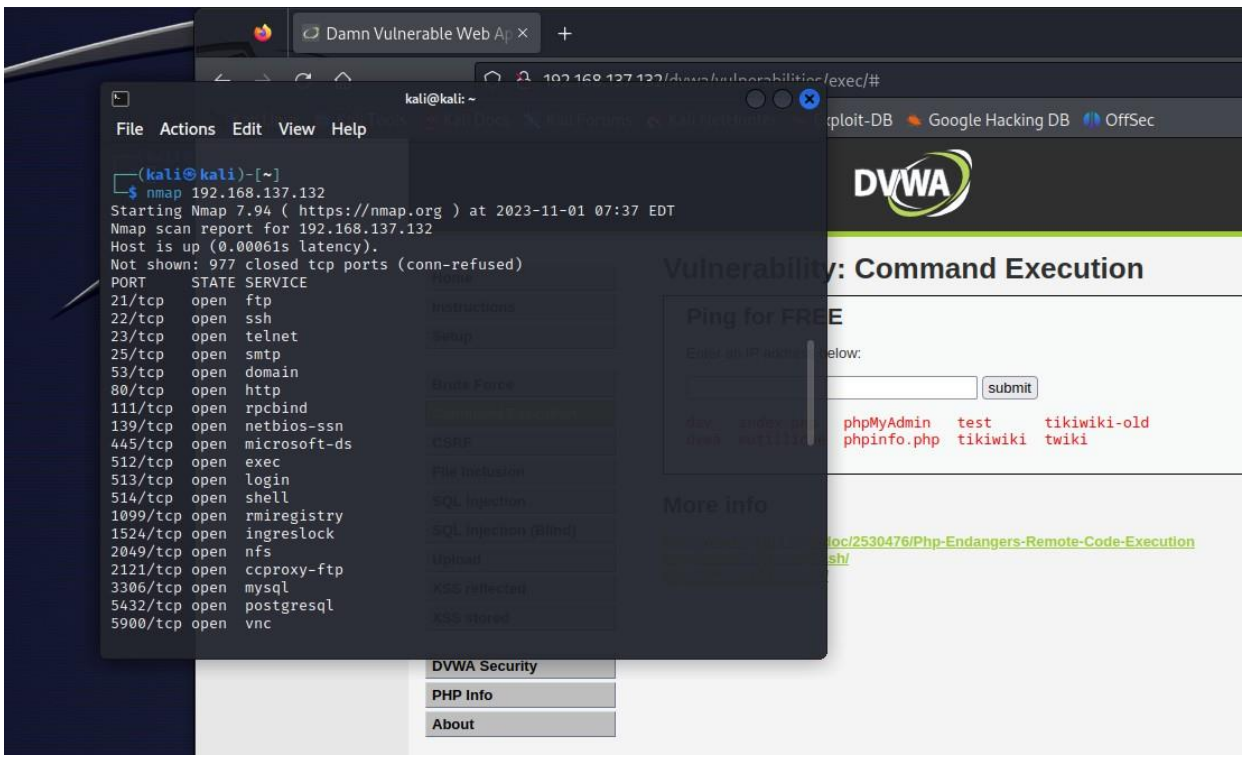
```
user@user-virtual-machine: /var/log/snort
  GNU nano 6.2                    /etc/snort/rules/local.rules
# $Id: local.rules,v 1.11 2004/07/23 20:15:44 bmc Exp $
# ----------------
# LOCAL RULES
# ----------------
# This file intentionally does not come with signatures.  Put your local
# additions here.
alert tcp any any -> any any (msg:"Network Scan detected"; flags:S,12;; sid:100002; rev:1;)

                    [ File '/etc/snort/rules/local.rules' is unwritable ]
^G Help      ^O Write Out   ^W Where Is   ^K Cut       ^T Execute     ^C Location    M-U Undo
^X Exit      ^R Read File   ^\ Replace    ^U Paste     ^J Justify     ^/ Go To Line  M-E Redo
```

⇨ **Alert tcp any any -> any any (msg:"Network Scan detected"; flags:S,12;; sid:100002; rev:1;)**



⇨ The Snort result detects Network Scan

SYN Flood attack:



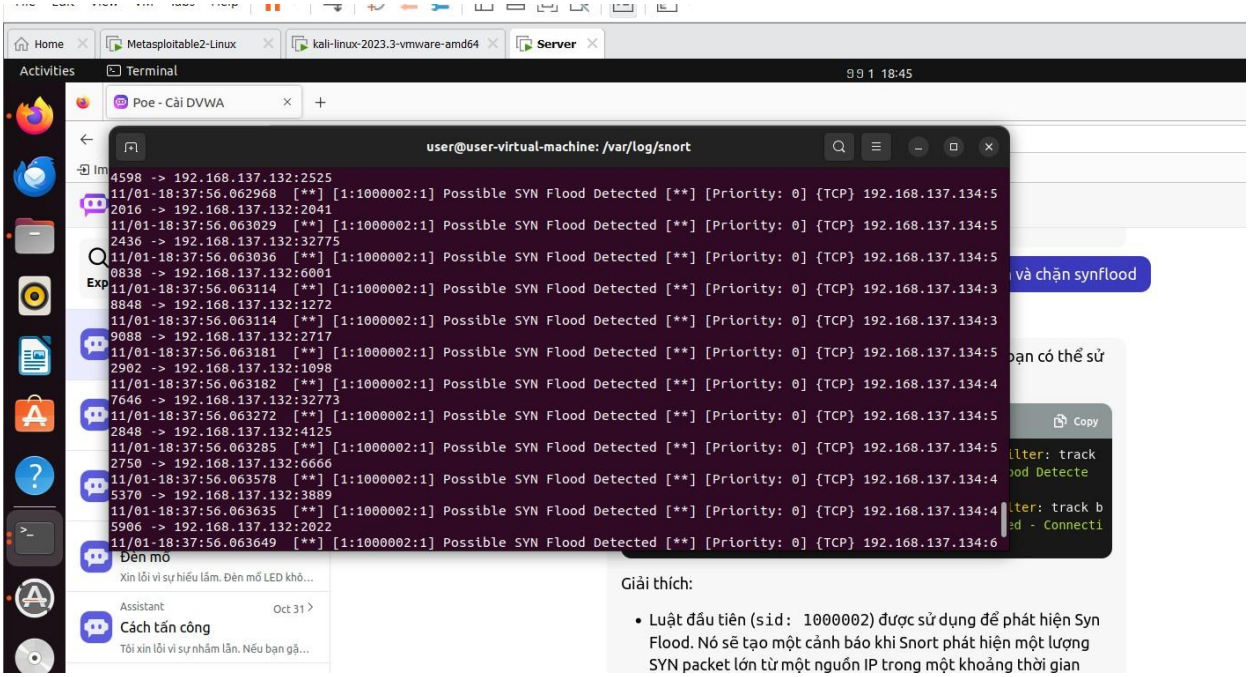
⇨ Perform nmap to find open ports on the Victim's machine

```
kali@kali: ~
File  Actions  Edit  View  Help
       =[ metasploit v6.3.27-dev                          ]
+ -- --=[ 2335 exploits - 1220 auxiliary - 413 post       ]
+ -- --=[ 1385 payloads - 46 encoders - 11 nops           ]
+ -- --=[ 9 evasion                                       ]

Metasploit tip: Open an interactive Ruby terminal with
irb
Metasploit Documentation: https://docs.metasploit.com/

msf6 > use auxiliary/dos/tcp/synflood
msf6 auxiliary(dos/tcp/synflood) > set RHOST 192.168.137.132
RHOST => 192.168.137.132
msf6 auxiliary(dos/tcp/synflood) > set RPORT 512
RPORT => 512
msf6 auxiliary(dos/tcp/synflood) > set NUM 0
NUM => 0
msf6 auxiliary(dos/tcp/synflood) > run
[*] Running module against 192.168.137.132
```

⇨ Proceed to set RHOST, RPORT, NUM values and carry out the SYNFlood attack to the Victim's machine

- We set Snort rule to detect and drop SYN Flood packets:

```
user@user-virtual-machine: /var/log/snort
GNU nano 6.2                    /etc/snort/rules/local.rules
# $Id: local.rules,v 1.11 2004/07/23 20:15:44 bmc Exp $
# ----------------
# LOCAL RULES
# ----------------
# This file intentionally does not come with signatures.  Put your local
# additions here.
alert tcp any any -> any any (flags: S; detection_filter: track by_src, count 15, seconds 10; msg: "Possible SYN Flood Detected"; sid: 1000002; rev: 1;)
drop tcp any any -> any any (flags: S; detection_filter: track by_src, count 30, seconds 10; msg: "SYN Flood Detected - Connection Dropped"; sid: 1000003; rev: 1;)

^G Help      ^O Write Out  ^W Where Is   ^K Cut       ^T Execute    ^C Location   M-U Undo      M-A Set Mark  M-] To Bracket  M-Q Previous
^X Exit      ^R Read File  ^\ Replace    ^U Paste     ^J Justify    ^/ Go To Line M-E Redo      M-6 Copy      ^Q Where Was    M-W Next
```



⇨ After setting the rule, the result shows that Snort has detected suspicious attacks