

Palestine
An-Najah National University
Faculty of Information Technology
Network and Information Security



دولة فلسطين
جامعة النجاح الوطنية
كلية تكنولوجيا المعلومات
قسم شبكات وامن المعلومات

Experiment#4 Packet Sniffing & Spoofing

Dyaa Al-dein Ashraf Tummazeh

11924899

Manar Eyad Harb

11924470

Yara Mohammad Sholi

11924207

Supervisor:

Dr. Amjad Hawash

Abstract

In this experiment, we're learning about the basics of network sniffing and spoofing, and how to implement it.

Introduction

Sniffing and spoofing both are types of network attacks each one works differently. Sniffing is a monitoring process for all the data packets that pass through the network, and usually, it's used by network administrators, and attackers also use it to steal information. Spoofing is seeming like men in the middle attack, the attacker uses the host IP address and represents himself that he is the owner of the address. In this way, the attacker achieves his goal 'take info. / give wrong info.'

Sniffers just snooping but spoofers can snoop and replay/sent messages also.

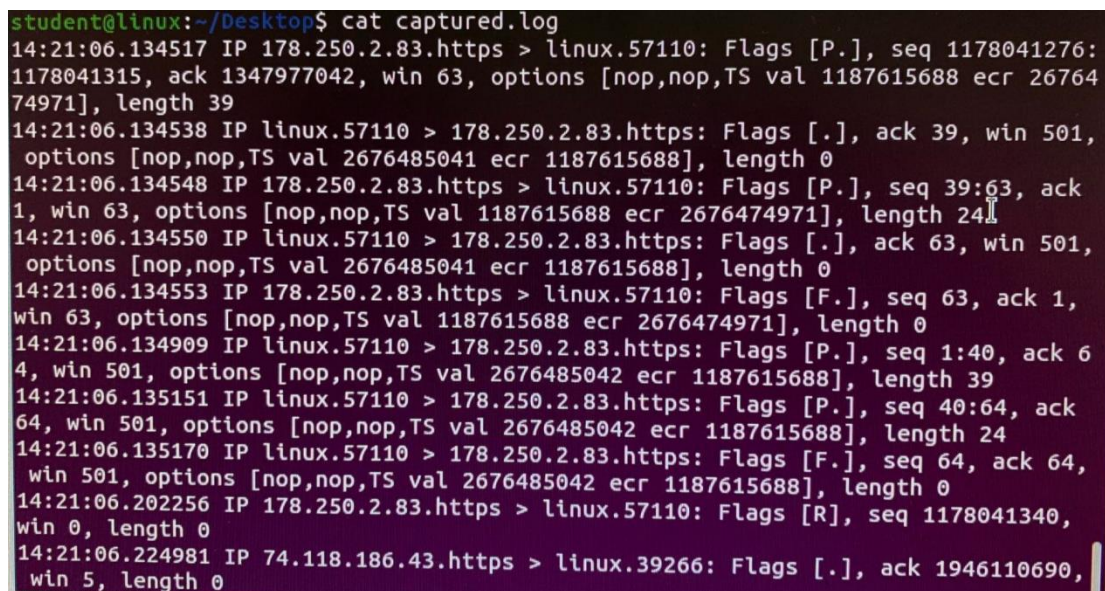
Procedures

3.1 Packet Sniffing Using tcpdump:

- a. Tcpdump tool used to analyze network traffic by displaying the packet that the device is running it.
- b. Run the tcpdump command for 3 seconds and log the results in an output file named captured.log. Check the content of the file and show a captured packet:

sudo timeout 3 tcpdump > captured.log

The content of the captured.log file is:



```
student@linux:~/Desktop$ cat captured.log
14:21:06.134517 IP 178.250.2.83.https > linux.57110: Flags [P.], seq 1178041276:
1178041315, ack 1347977042, win 63, options [nop,nop,TS val 1187615688 ecr 26764
74971], length 39
14:21:06.134538 IP linux.57110 > 178.250.2.83.https: Flags [.], ack 39, win 501,
options [nop,nop,TS val 2676485041 ecr 1187615688], length 0
14:21:06.134548 IP 178.250.2.83.https > linux.57110: Flags [P.], seq 39:63, ack
1, win 63, options [nop,nop,TS val 1187615688 ecr 2676474971], length 24
14:21:06.134550 IP linux.57110 > 178.250.2.83.https: Flags [.], ack 63, win 501,
options [nop,nop,TS val 2676485041 ecr 1187615688], length 0
14:21:06.134553 IP 178.250.2.83.https > linux.57110: Flags [F.], seq 63, ack 1,
win 63, options [nop,nop,TS val 1187615688 ecr 2676474971], length 0
14:21:06.134909 IP linux.57110 > 178.250.2.83.https: Flags [P.], seq 1:40, ack 6
4, win 501, options [nop,nop,TS val 2676485042 ecr 1187615688], length 39
14:21:06.135151 IP linux.57110 > 178.250.2.83.https: Flags [P.], seq 40:64, ack
64, win 501, options [nop,nop,TS val 2676485042 ecr 1187615688], length 24
14:21:06.135170 IP linux.57110 > 178.250.2.83.https: Flags [F.], seq 64, ack 64,
win 501, options [nop,nop,TS val 2676485042 ecr 1187615688], length 0
14:21:06.202256 IP 178.250.2.83.https > linux.57110: Flags [R], seq 1178041340,
win 0, length 0
14:21:06.224981 IP 74.118.186.43.https > linux.39266: Flags [.], ack 1946110690,
win 5, length 0
```

Figure1.1 shows the content in capture.log file

For the first packet, it has fields like the time it was sent, the destination IP address, destination port “HTTPS” here, the OS, unique sequence number of the packet “**1178041276**”, acknowledgment number which is unique also “**1347977042**”, and the length of the packet “**how many bytes**”.

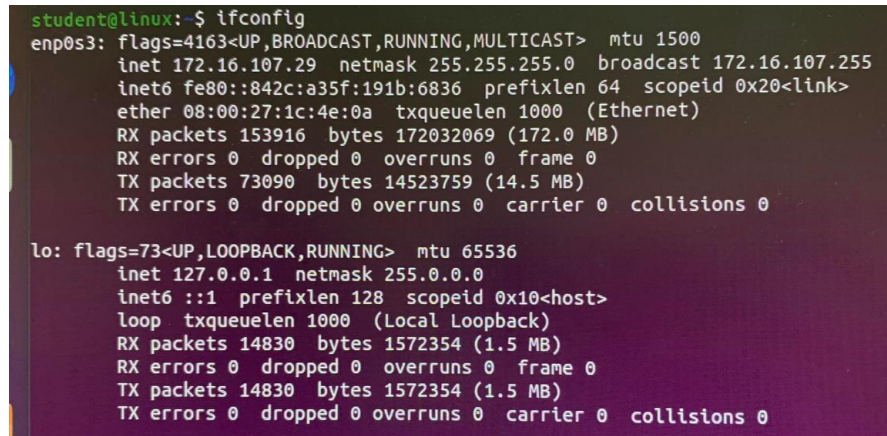
3.2 pcap Library:

Packet capture library (pcap) is a library that is used to monitor low-level networks.

a. Provide the name of the interface on which you are willing to do sniffing using pcap:

Interface name: **enp0s3**

To find the interface name using this command: `ifconfig`, the result will look like this:



```
student@linux:~$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.16.107.29 netmask 255.255.255.0 broadcast 172.16.107.255
    inet6 fe80::842c:a35f:191b:6836 prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:1c:4e:0a txqueuelen 1000 (Ethernet)
    RX packets 153916 bytes 172032069 (172.0 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 73090 bytes 14523759 (14.5 MB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 14830 bytes 1572354 (1.5 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 14830 bytes 1572354 (1.5 MB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Figure2.1 shows the information on the interface that was chosen.

b. Get the c file named `GetInterface.c` from internet. Then compile it using this command: (in this case we named the c file as `code.c`)

`sudo gcc code.c -o codefile -lcrypto`

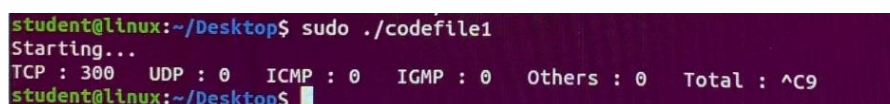
Then run the program and pass the name of the interface you wish to apply on as an argument, here we use interface **enp0s3**.

The promiscuous mode of a network interface is one of the computer networking operation modes that can access all the network data packets and monitor it by network adapters operating.

Traffic filtering in sniffing is filtering the data mean display specific data for example “if we write UDP in the filter box all UDP data will be displayed”.

c. Get the C file named `Sniffer.c` from internet. Then compile it and make sure that is error free.

d. Run the `Sniffer.c` code:



```
student@linux:~/Desktop$ sudo ./codefile1
Starting...
TCP : 300  UDP : 0  ICMP : 0  IGMP : 0  Others : 0  Total : ^C9
student@linux:~/Desktop$
```

Figure2.2 shows the output of running `Sniffer.c` code.

As it shows in figure2.2 the captured packet is TCP packets.

e. download the file sniffex.c from internet. Then compile it.

f. Now, run the code:

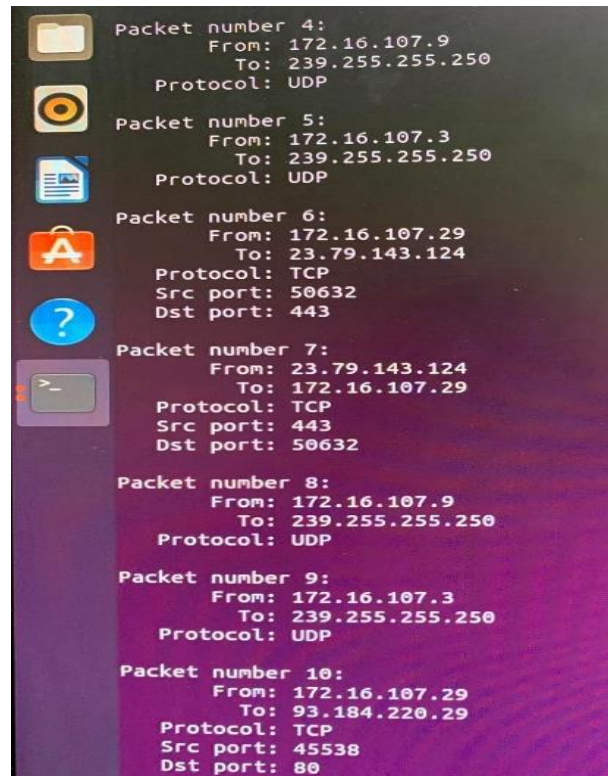


Figure2.3 shows some packets captured by the previous code.

As it shows in figure2.3, there is some packets that captured by sinffex.c code.

g. Modify the code sniffex.c so that it captures the password when some body is using telnet on the network you are monitoring.

To do that, first install telnet by using this command in two devices:

sudo apt-get install telnet

Then connect to telnet to one from another device, then run the code:


```
student@linux:~/Desktop$ telnet 172.16.107.18
Trying 172.16.107.18...
Connected to 172.16.107.18.
Escape character is '^]'.
Ubuntu 20.04.3 LTS
o1J2FP3RnMk4S login: student
Password:
Welcome to Ubuntu 20.04.3 LTS (GNU/Linux 5.13.0-28-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

115 updates can be installed immediately.
0 of these updates are security updates.
To see these additional updates run: apt list --upgradable

Your Hardware Enablement Stack (HWE) is supported until April 2025.
Last login: Tue Feb 15 15:15:25 EET 2022 from 172.16.107.29 on pts/2
student@o1J2FP3RnMk4S:~$
```

Figure2.4 shows telnet connection.

Using wireshark to show the captured password:

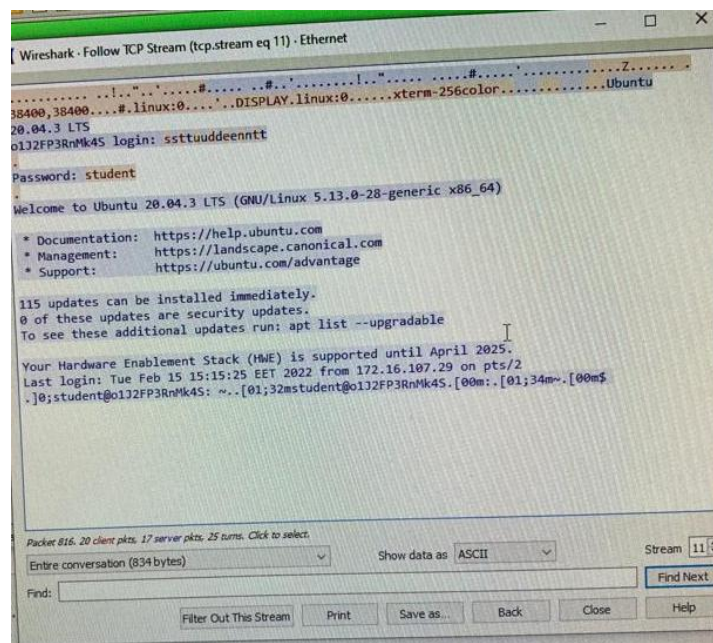


Figure2.5 shows the captured password.

3.3 Packet Spoofing:

Packet spoofing and called IP spoofing, is when someone impersonates a computer system in the network using its IP and sends a packet from that's IP. There is some attacks using it such as DDoS attack, DoS attack, and Man in the Middle attack.

- a. Download the file spoof.c from internet. Then compile it.

b. Use the compiled code to create an ICMP echo request packet and send it from the IP address 128.10.130.190 to the destination IP address 128.10.130.191 using this command:

```
[+] Spoofed IP packet sent successfully!
student@linux:~/Desktop$ sudo ./codefile5 --type=ping --payload='hello' --src-ip=128.10.130.190 --dst-ip=128.10.130.191
-----+
| PURDUE Univ. CS528 - Network Security |
| Lab 1: Packet Sniffing and Spoofing   |
| Task 2: Spoof Ping packets and Ethernet frames |
+-----+
[+] Spoofed IP packet sent successfully!
student@linux:~/Desktop$
```

Figure3.1 shows ICMP echo request.

c. Run tcpdump from another terminal and re-run the spoof program, then capture the sent packet.

The captured sent packet content is:

```
student@linux:~/Desktop$ sudo tcpdump | grep "128.10.130.191"
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on enp0s3, link-type EN10MB (Ethernet), capture size 262144 bytes
15:54:49.632163 IP 128.10.130.190 > 128.10.130.191: ICMP echo request, id 0, seq 0, length 13
```

Figure3.2 shows the captured packet information.

d. Construct a spoofed Ethernet frame with the destination MAC address 99:99:99:99:99:99 from the source IP address 01:02:03:04:05:06 with a payload containing the date of today.

And that's done by using this command after running the code:

```
sudo ./codefile6 --type=ethernet --payload='22-2-2022' --src-mac=fe80:dbf6:19b5:c640:8057 --dst-mac=01:02:03:04:05:06
```

Then run tcpdump and capture the spoofed Ethernet frame by using the same commands that shows in Figure3.2, but changing ip address 128.10.130.191 to the mac address 01:02:03:04:05:06.

Conclusion

In the end, this lab shows what is the difference between packet spoofing and packet sniffing, and how can be implemented using the pcap library.

References

**<https://seedsecuritylabs.org/>
Wikipedia
openssl.org**