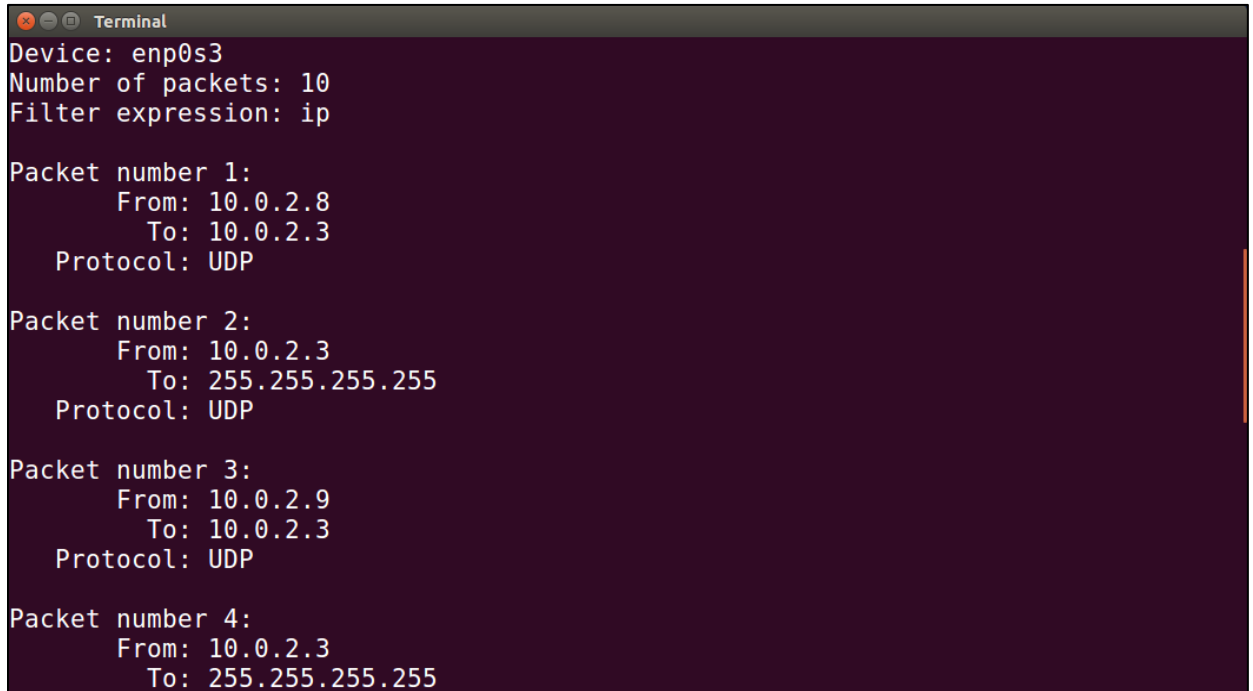


LAB PROJECT 1: Packet Sniffing and Spoofing Lab

Kushal Hebbar | C13031425 | khebbar@g.clemson.edu

❖ Task 1: Writing packet sniffing program

The figure below is a screenshot of the Sniffex program executing.

A screenshot of a terminal window with a dark purple background. The title bar at the top says "Terminal". The output of the Sniffex program is as follows:

```
Device: enp0s3
Number of packets: 10
Filter expression: ip

Packet number 1:
    From: 10.0.2.8
    To: 10.0.2.3
    Protocol: UDP

Packet number 2:
    From: 10.0.2.3
    To: 255.255.255.255
    Protocol: UDP

Packet number 3:
    From: 10.0.2.9
    To: 10.0.2.3
    Protocol: UDP

Packet number 4:
    From: 10.0.2.3
    To: 255.255.255.255
```

- **Problem 1:** Please use your own words to describe the sequence of the library calls that are essential for sniffer programs. This is meant to be a summary, not detailed explanation like the one in the tutorial.
1. **Device setup:** The first step in the sniffer program is to define the device and the interface from which capturing must begin. The device can either be defined in the sniffer program or the sniffer program itself looks up for active devices (pcap_lookupdev).
 2. **Initialize sniffing:** Once the device has been defined, sniffer initializes pcap and sends a signal to setup an environment to perform sniffing, this environment is defined as a session. Sessions are independent of each other, a single device can have multiple sessions, multiple sessions can sniff on one device as well.
 3. **Traffic filtering:** Sniffing requires a device to sniff on as well as certain rules for sniffing such as, what packets to sniff on and duration of sniffing. Such requirements need to be specified in the filter string and compiled to apply the rule. Filtering helps in sniffing only a certain type of packets from many packets in the network.

4. **Sniffing:** Packets can be sniffed in 2 ways: first method involves each packet being sniffed and analyzed individually, whereas the second method involves sniffing n packets by running the sniffer in a loop and then analyzing the entire group of packets. The sniffed packets can either be displayed to the user or stored in a file for further analysis.
5. **Session termination:** This is the final step of the sniffing process. Once the sniffing is completed and enough data about the network is collected the sniffer terminates and either displays relevant information or stores the information in a file.
 - **Problem 2:** Why do you need the root privilege to run Sniffex? Where does the program fail if executed without the root privilege?

The screenshot below shows the error that occurs when the Sniffex program is executed without root permission.

```
[09/16/18]seed@VM:~/Desktop$ ./sniffex
sniffex - Sniffer example using libpcap
Copyright (c) 2005 The Tcpdump Group
THERE IS ABSOLUTELY NO WARRANTY FOR THIS PROGRAM.

Device: enp0s3
Number of packets: 10
Filter expression: ip
Couldn't open device enp0s3: enp0s3: You don't have permission to capture on tha
t device (socket: Operation not permitted)
```

Pcap_lookupdev() function requires root permission as it requires access to network interfaces and it is impossible without root access in Linux. Sniffer program requires access to raw sockets which is not possible without root permission.

- **Problem 3:** Please turn on and turn off the promiscuous mode in the sniffer program. Can you demonstrate the difference when this mode is on and off? Please describe how you demonstrate this.

Promiscuous mode is used to allow the network sniffer to pass all the traffic in the network and not just the packets that are intended to be received by the network controller. This mode can be changed in pcap_open_live function and can vary from 1 -> 0 or vice versa.

```
// promiscuous mode on
```

```
handle = pcap_open_live(dev, SNAP_LEN, 1, 1000, errbuf);
```

The following is the demonstration of promiscuous mode, VM2 with IP: 10.0.2.9 I decided to ping the IP 8.8.8.8 to produce packets.

```

Terminal
[09/19/18]seed@VM:~$ ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=119 time=48.0 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=119 time=48.4 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=119 time=52.3 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=119 time=48.0 ms
64 bytes from 8.8.8.8: icmp_seq=5 ttl=119 time=48.5 ms
64 bytes from 8.8.8.8: icmp_seq=6 ttl=119 time=48.5 ms
64 bytes from 8.8.8.8: icmp_seq=7 ttl=119 time=47.9 ms
64 bytes from 8.8.8.8: icmp_seq=8 ttl=119 time=48.0 ms
64 bytes from 8.8.8.8: icmp_seq=9 ttl=119 time=49.3 ms
64 bytes from 8.8.8.8: icmp_seq=10 ttl=119 time=48.0 ms
64 bytes from 8.8.8.8: icmp_seq=11 ttl=119 time=46.2 ms
64 bytes from 8.8.8.8: icmp_seq=12 ttl=119 time=50.5 ms
64 bytes from 8.8.8.8: icmp_seq=13 ttl=119 time=50.7 ms
64 bytes from 8.8.8.8: icmp_seq=14 ttl=119 time=49.0 ms
64 bytes from 8.8.8.8: icmp_seq=15 ttl=119 time=63.8 ms
64 bytes from 8.8.8.8: icmp_seq=16 ttl=119 time=48.5 ms
64 bytes from 8.8.8.8: icmp_seq=17 ttl=119 time=48.1 ms
64 bytes from 8.8.8.8: icmp_seq=18 ttl=119 time=49.6 ms
64 bytes from 8.8.8.8: icmp_seq=19 ttl=119 time=48.9 ms
64 bytes from 8.8.8.8: icmp_seq=20 ttl=119 time=48.1 ms

```

I then executed the Sniffex program in VM1 with IP: 10.0.2.8. The figure below depicts the sniffer picking up packets generated in VM2, hence demonstrating promiscuous mode.

```

Terminal
Device: enp0s3
Number of packets: 10
Filter expression: ip

Packet number 1:
    From: 10.0.2.9
    To: 8.8.8.8
    Protocol: ICMP

Packet number 2:
    From: 8.8.8.8
    To: 10.0.2.9
    Protocol: ICMP

Packet number 3:
    From: 10.0.2.9
    To: 8.8.8.8
    Protocol: ICMP

```

// promiscuous mode off

```
handle = pcap_open_live(dev, SNAP_LEN, 0, 1000, errbuf);
```

The following image depicts non-promiscuous mode of sniffing, where the sniffer picks up packets only sent to and from VM1.

```
Terminal
Device: enp0s3
Number of packets: 10
Filter expression: ip

Packet number 1:
    From: 10.0.2.8
    To: 192.168.1.254
    Protocol: UDP

Packet number 2:
    From: 10.0.2.8
    To: 192.168.1.254
    Protocol: UDP

Packet number 3:
    From: 10.0.2.8
    To: 192.168.1.254
    Protocol: UDP
```

- **Problem 4:** Please write filter expressions to capture each of the followings. In your lab reports, you need to include screenshots to show the results of applying each of these filters.
 - Capture the ICMP packets between two specific hosts.
 - Capture the TCP packets that have a destination port range from to port 10 - 100.
- Capturing ICMP packets between two specific hosts:
In order to perform this task, the filter_exp[] variable in the Sniffer program needs to be altered to,

```
char filter_exp[] = "icmp and (src host 10.0.2.8 and dst host 8.8.8.8) or (src host 8.8.8.8 and dst host 10.0.2.8)";
```

Next, execute the Sniffex in VM1 with IP: 10.0.2.8 and ping 8.8.8.8 to see the following result,

```
Device: enp0s3
Number of packets: 10
Filter expression: icmp and (src host 10.0.2.8 and dst host 8.8.8.8) or (src host 8.8.8.8 and dst host 10.0.2.8)

Packet number 1:
    From: 10.0.2.8
    To: 8.8.8.8
    Protocol: ICMP

Packet number 2:
    From: 8.8.8.8
    To: 10.0.2.8
    Protocol: ICMP

Packet number 3:
    From: 10.0.2.8
    To: 8.8.8.8
    Protocol: ICMP

Packet number 4:
    From: 8.8.8.8
    To: 10.0.2.8
    Protocol: ICMP
```

- Capturing the TCP packets that have a destination port range from to port 10 – 100:
In order to perform this task, the filter_exp[] variable in the Sniffer program needs to be altered to,

```
char filter_exp[] = "tcp dst portrange 10-100";
```

Now execute the Sniffex program in VM1 and simultaneously open any http (Port 80) website on a web browser (here, <http://www.google.com>) to produce the following results,

```
Device: enp0s3
Number of packets: 10
Filter expression: tcp dst portrange 10-100
```

```
Packet number 1:
    From: 10.0.2.8
    To: 128.230.208.76
    Protocol: TCP
    Src port: 35992
    Dst port: 80
```

```
Packet number 2:
    From: 10.0.2.8
    To: 128.230.208.76
    Protocol: TCP
    Src port: 35994
    Dst port: 80
```

```
Packet number 3:
    From: 10.0.2.8
    To: 128.230.208.76
    Protocol: TCP
```

```
    Src port: 35996
    Dst port: 80
```

```
Packet number 4:
    From: 10.0.2.8
    To: 128.230.208.76
    Protocol: TCP
    Src port: 35998
    Dst port: 80
```

```
Packet number 5:
    From: 10.0.2.8
    To: 128.230.208.76
    Protocol: TCP
    Src port: 36000
    Dst port: 80
```

```
Packet number 6:
    From: 10.0.2.8
    To: 128.230.208.76
    Protocol: TCP
    Src port: 35992
```