CSN-505 Project Lab

Pool A: Sniffer program

Utkarsh Yadav
21535035 [Mtech Ist yr.]

INDEX

1. Problem statement

Write a sniffer program that has following basic functionality:

(i) Captures packets and stores them in a file

(ii) User can select packets by

  i. protocol or

  ii. by source address and protocol

and can display contents of the header fields of the packet.

2. Project description

To implement sniffer program there are below mentioned files.

1. **SnifferProgram.cpp**
   This file has the main driver function which will also ask the user if he/she wants to sniff packets based on protocol or source addres or both.

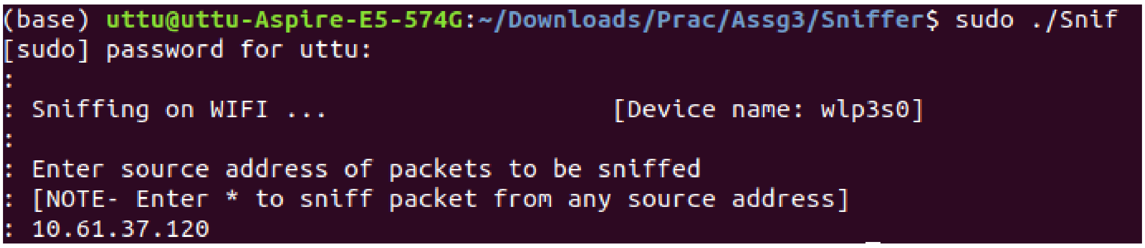
2. **SupportLibrary.cpp**
   This is the supporting library which consists of the functions to hander different types of headers like TCP, UDP, etc. This library is used by SnifferProgram.cpp

3. **Log files**
   There are different log files for each type of protocol that is for TCP we have "**logTCP.txt**", for UDP we have "**logUDP.txt**" and for ICMP we have "**logICMP.txt**".

3. Project functionalities

Below mention functions are provided to the user while sniffing packets:

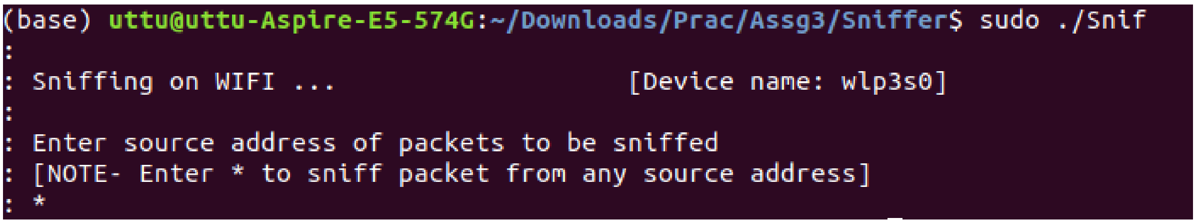
1. User is given a option to sniff packets from a specific source address only. [As shown in the fig-1]



```
(base) uttu@uttu-Aspire-E5-574G:~/Downloads/Prac/Assg3/Sniffer$ sudo ./Snif
[sudo] password for uttu:
:
: Sniffing on WIFI ...                    [Device name: wlp3s0]
:
: Enter source address of packets to be sniffed
: [NOTE- Enter * to sniff packet from any source address]
: 10.61.37.120
```

Fig-1: User want to sniff packets from specific source.

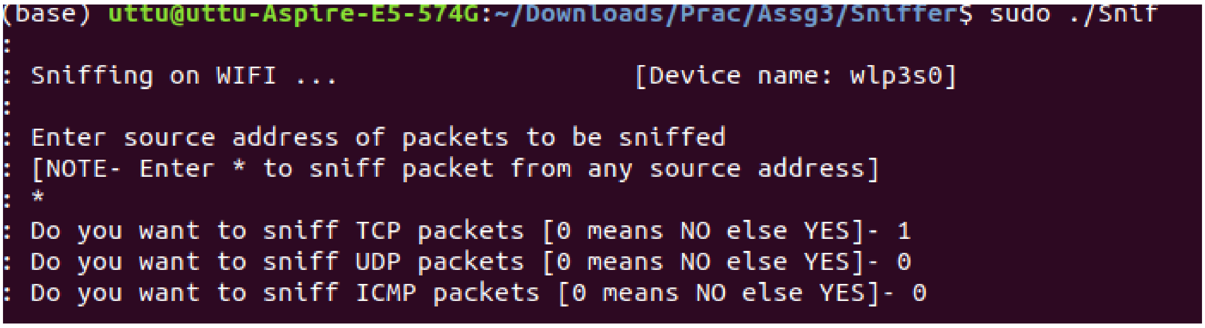
2. User is given a option to sniff all the packets irrespective of the source address. [As shown in the fig-2]



```
(base) uttu@uttu-Aspire-E5-574G:~/Downloads/Prac/Assg3/Sniffer$ sudo ./Snif
:
: Sniffing on WIFI ...                    [Device name: wlp3s0]
:
: Enter source address of packets to be sniffed
: [NOTE- Enter * to sniff packet from any source address]
: *
```

Fig-2: User want to sniff all the packets [any source address].

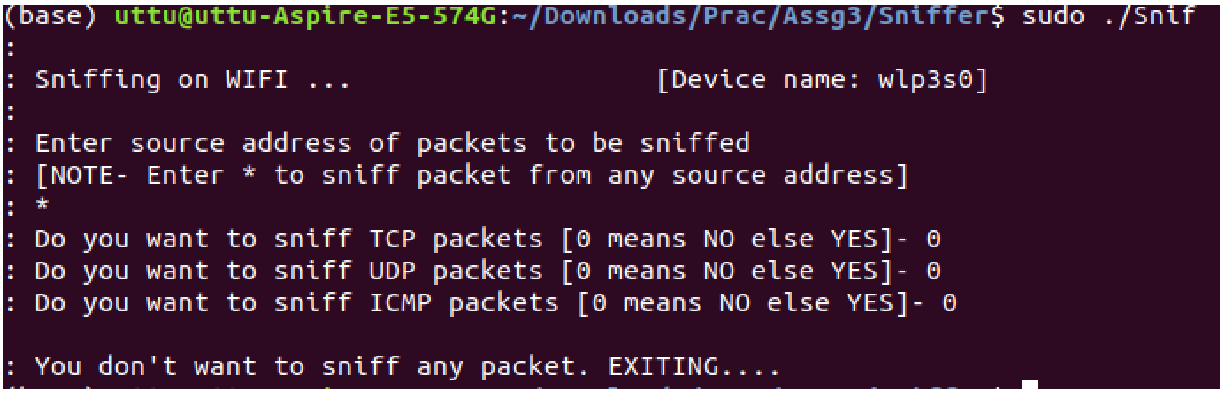
3. User is given a choice to allow sniffing based on the protocol of received packet. User input ‘0’ is considered as No, while other values are considered as Yes. [As shown in fig-3]



```
(base) uttu@uttu-Aspire-E5-574G:~/Downloads/Prac/Assg3/Sniffer$ sudo ./Snif
:
: Sniffing on WIFI ...                    [Device name: wlp3s0]
:
: Enter source address of packets to be sniffed
: [NOTE- Enter * to sniff packet from any source address]
: *
: Do you want to sniff TCP packets [0 means NO else YES]- 1
: Do you want to sniff UDP packets [0 means NO else YES]- 0
: Do you want to sniff ICMP packets [0 means NO else YES]- 0
```

Fig-3: User only wants to sniff TCP packets.

4. If User input for all these three type of protocol is No, then program will simply close as user do not want to sniff any supported protocols. [As shown in fig-4]

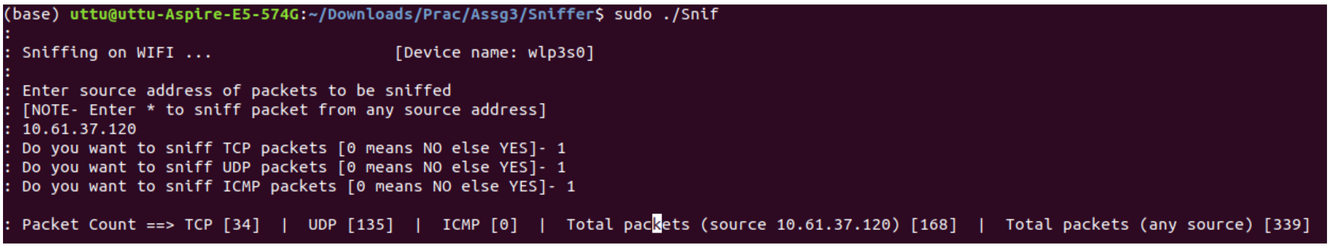


```
(base) uttu@uttu-Aspire-E5-574G:~/Downloads/Prac/Assg3/Sniffer$ sudo ./Snif
:
: Sniffing on WIFI ...                    [Device name: wlp3s0]
:
: Enter source address of packets to be sniffed
: [NOTE- Enter * to sniff packet from any source address]
: *
: Do you want to sniff TCP packets [0 means NO else YES]- 0
: Do you want to sniff UDP packets [0 means NO else YES]- 0
: Do you want to sniff ICMP packets [0 means NO else YES]- 0

: You don't want to sniff any packet. EXITING....
```

Fig-4: User do not want to sniff any type of packet, hence exiting.

5. Depending on the user input, once sniffing starts the number of packets sniffed is continously updated on the terminal. On terminal count of below mentioned packets will be shown:
    - ○ TCP packets
    - ○ UDP packets
    - ○ ICMP packets
    - ○ Total number of packets [from specified source]
    - ○ Total number of packets [from any source]

    [As shown in fig-5]

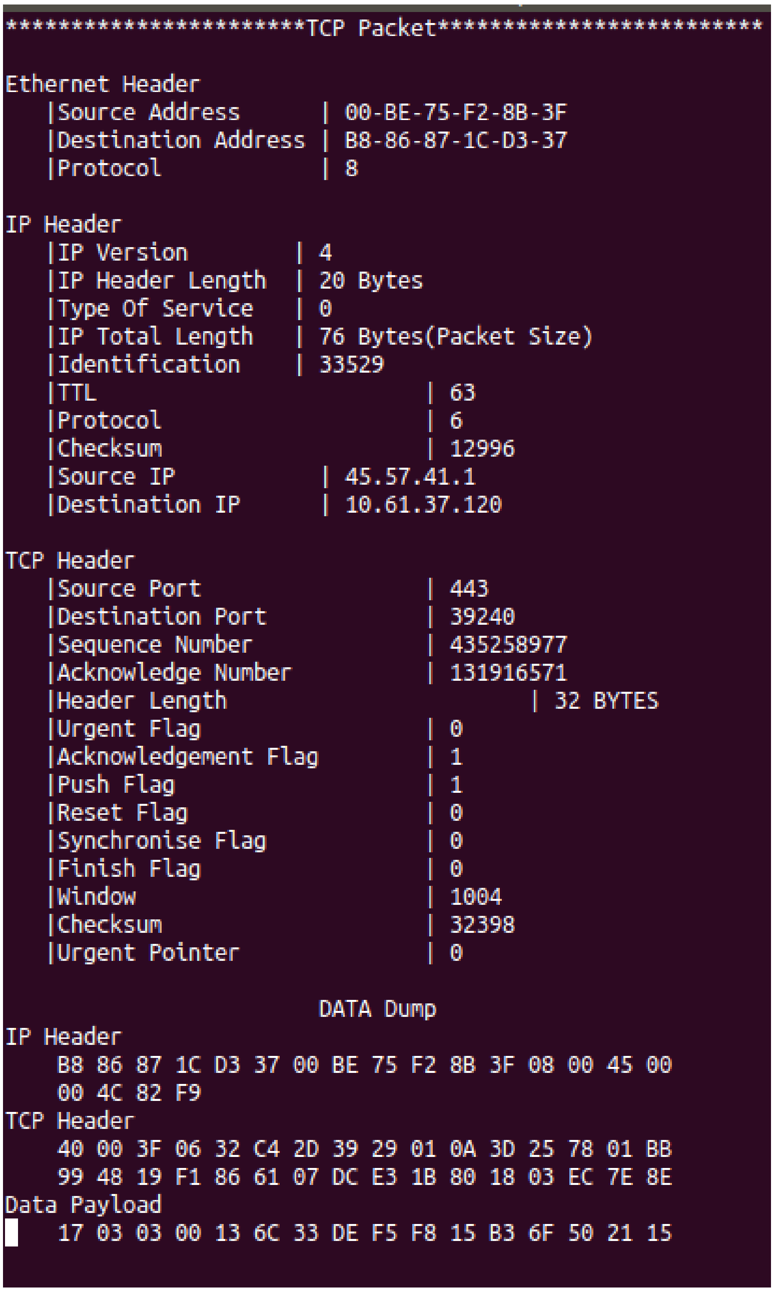


```
(base) uttu@uttu-Aspire-E5-574G:~/Downloads/Prac/Assg3/Sniffer$ sudo ./Snif
:
: Sniffing on WIFI ...                    [Device name: wlp3s0]
:
: Enter source address of packets to be sniffed
: [NOTE- Enter * to sniff packet from any source address]
: 10.61.37.120
: Do you want to sniff TCP packets [0 means NO else YES]- 1
: Do you want to sniff UDP packets [0 means NO else YES]- 1
: Do you want to sniff ICMP packets [0 means NO else YES]- 1

: Packet Count ==> TCP [34]  |  UDP [135]  |  ICMP [0]  |  Total packets (source 10.61.37.120) [168]  |  Total packets (any source) [339]
```

Fig-5: Showing count of different type of packets received.

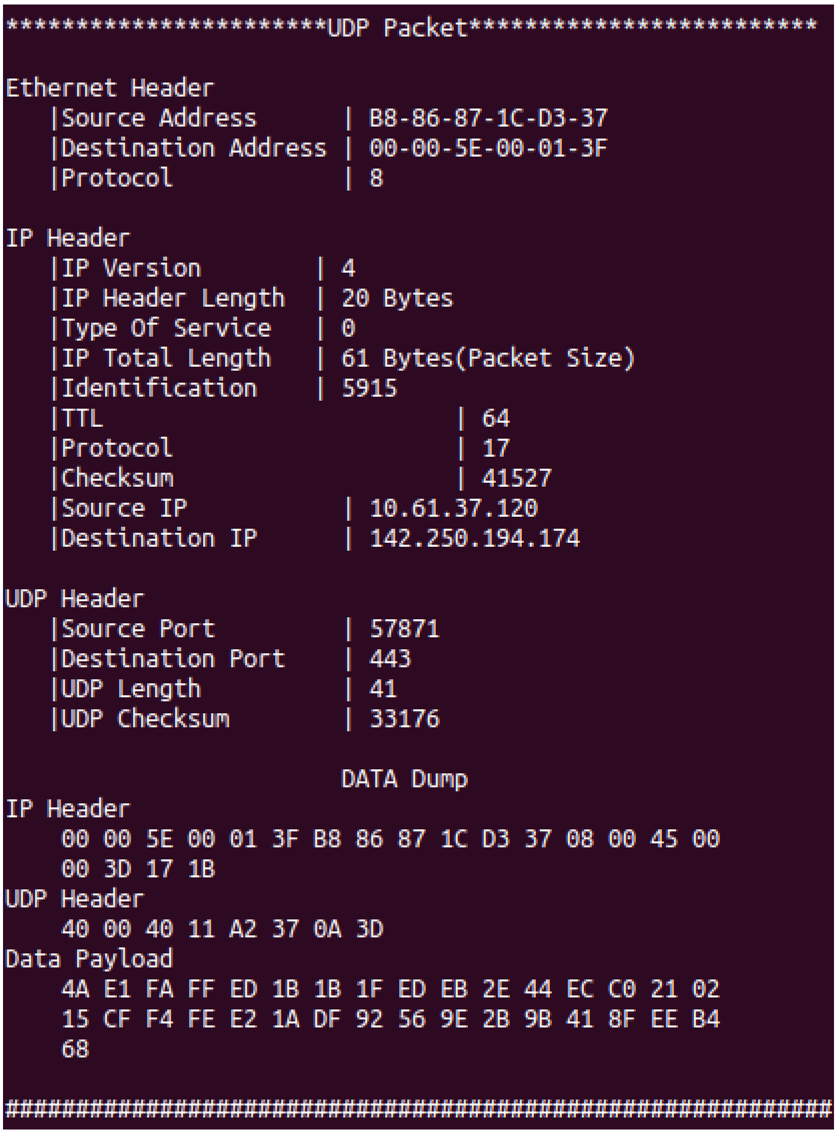
6. Content of packet header and fields of IP header, fields of TCP/UDP/ICMP header and fields of ethernet header are printed in different log files depending on the protocol of packet. Below mentioned log files are maintained:
    - ○ logTCP.txt
    - ○ logUDP.txt
    - ○ logICMP.txt

    [As shown in fig-6 and fig-7]



```
***********************TCP Packet*************************

Ethernet Header
   |Source Address      | 00-BE-75-F2-8B-3F
   |Destination Address | B8-86-87-1C-D3-37
   |Protocol            | 8

IP Header
   |IP Version        | 4
   |IP Header Length  | 20 Bytes
   |Type Of Service   | 0
   |IP Total Length   | 76 Bytes(Packet Size)
   |Identification    | 33529
   |TTL                         | 63
   |Protocol                    | 6
   |Checksum                    | 12996
   |Source IP           | 45.57.41.1
   |Destination IP      | 10.61.37.120

TCP Header
   |Source Port                 | 443
   |Destination Port            | 39240
   |Sequence Number             | 435258977
   |Acknowledge Number          | 131916571
   |Header Length                       | 32 BYTES
   |Urgent Flag                 | 0
   |Acknowledgement Flag        | 1
   |Push Flag                   | 1
   |Reset Flag                  | 0
   |Synchronise Flag            | 0
   |Finish Flag                 | 0
   |Window                      | 1004
   |Checksum                    | 32398
   |Urgent Pointer              | 0

                        DATA Dump
IP Header
    B8 86 87 1C D3 37 00 BE 75 F2 8B 3F 08 00 45 00
    00 4C 82 F9
TCP Header
    40 00 3F 06 32 C4 2D 39 29 01 0A 3D 25 78 01 BB
    99 48 19 F1 86 61 07 DC E3 1B 80 18 03 EC 7E 8E
Data Payload
    17 03 03 00 13 6C 33 DE F5 F8 15 B3 6F 50 21 15
```

Fig-6: Log file content for TCP packets.



```
***********************UDP Packet*************************

Ethernet Header
   |Source Address      | B8-86-87-1C-D3-37
   |Destination Address | 00-00-5E-00-01-3F
   |Protocol            | 8

IP Header
   |IP Version        | 4
   |IP Header Length  | 20 Bytes
   |Type Of Service   | 0
   |IP Total Length   | 61 Bytes(Packet Size)
   |Identification    | 5915
   |TTL                         | 64
   |Protocol                    | 17
   |Checksum                    | 41527
   |Source IP           | 10.61.37.120
   |Destination IP      | 142.250.194.174

UDP Header
   |Source Port         | 57871
   |Destination Port    | 443
   |UDP Length          | 41
   |UDP Checksum        | 33176

                        DATA Dump
IP Header
    00 00 5E 00 01 3F B8 86 87 1C D3 37 08 00 45 00
    00 3D 17 1B
UDP Header
    40 00 40 11 A2 37 0A 3D
Data Payload
    4A E1 FA FF ED 1B 1B 1F ED EB 2E 44 EC C0 21 02
    15 CF F4 FE E2 1A DF 92 56 9E 2B 9B 41 8F EE B4
    68

###########################################################
```

Fig-7: Log file content for UDP packets.

4. Application and technologies used

1. C++ language [used pcap library]
2. Linux operating system used