15 Packet capturing and filtering application

15.1 Aim

Develop a packet capturing and filtering application using raw sockets.

15.2 Theory

- **Raw Sockets** A raw socket is a network socket which allows sending and recieving of packets without any protocol specific transport layer formatting.

15.3 Algorithm

Algorithm 1 Packet Capturing Algorithm

Open a raw socket
Capture a packet using the raw socket
Decode the header of the packet
Print the header details of the packet
Depending on the protocol used by the packet,decode the data contained in the packet
Print the data contained in the packet
Close the raw socket

15.4 Code

Server

```
import socket,sys
from struct import *


s_tcp = socket.socket(socket.AF_INET, socket.SOCK_RAW, socket.
    IPPROTO_TCP)
s_udp = socket.socket(socket.AF_INET,socket.SOCK_RAW,socket.
    IPPROTO_UDP)

filt = sys.argv[1]
filt = filt.lower()

while True:
  packet = s_tcp.recvfrom(65565) if filt.lower()=="tcp" else s_udp.
    recvfrom(65565)
  packet = packet[0]
  ip_header = packet[0:20]
  iph = unpack('!BBHHHBBH4s4s',ip_header)
```

```
16  version_ihl = iph[0]
17  version =     version_ihl>>4
18  ihl = version_ihl&0xF
19
20  iph_length = ihl*4
21  ttl = iph[5]
22  protocol = iph[6]
23  s_addr = socket.inet_ntoa(iph[8])
24  d_addr = socket.inet_ntoa(iph[9])
25  print("\n----------------------PACKET BEGIN
      ----------------------------")
26  print("Version :",str(version),"\nIP Header Length :",str(ihl),"\
      nTTL",str(ttl),"\nProtocol :",str(protocol),"\nSource Address :
      ",str(s_addr),"\nDestination Address :",str(d_addr))
27  if filt == "tcp":
28      tcp_header = packet[iph_length:iph_length+20]
29      tcph = unpack('!HHLLBBHHH' , tcp_header)
30      source_port = tcph[0]
31      dest_port = tcph[1]
32      sequence = tcph[2]
33      acknowledgement = tcph[3]
34      doff_reserved = tcph[4]
35      tcph_length = doff_reserved>>4
36      print("\n-----------------------------------TCP HEADER
      -----------------------------------")
37      print("Source Port :",str(source_port),"\nDestination Port :"
      ,str(dest_port),"\nSequence Number :",str(sequence),"\
      nAcknowledgement :",str(acknowledgement),"\nHeader Length :",
      str(tcph_length))
38      print("\n----------------------PACKET END
      -----------------------------------")
39  else:
40      udp_header = packet[iph_length:iph_length+8]
41      udph = unpack('!HHHH',udp_header)
42      source_port = udph[0]
43      dest_port = udph[1]
44      length = udph[2]
45      checksum = udph[3]
46      print("\n-------------------------------UDP Header
      -------------------------------")
47      print("Source Port :",str(source_port),"\nDestination Port :"
      ,str(dest_port),"\nLength :",str(length),"\nChecksum :",str(
      checksum))
48      print("\n----------------------PACKET END
      -------------------------------")
```

15.5 Output

```
----------------------PACKET BEGIN----------------------------
Version : 4
IP Header Length : 5
TTL 123
```

```
Protocol : 6
Source Address : 172.217.194.189
Destination Address : 192.168.0.101

-------------------------TCP PACKET DATA-----------------------------
Source Port : 443
Destination Port : 39674
Sequence Number : 572864528
Acknowledgement : 724192025
TCP Header Length : 8

---------------------PACKET END--------  ----------------------
```

15.6 Result

Packet Capturing was implemented using python and the captured packet was analysed.