CS4150 : Computer Networks Lab Report

Lab-9 (IP Fragmentation and Checksum)

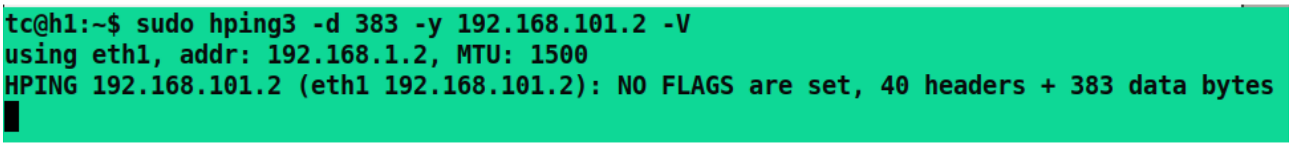
1. The command which I used to find the maximum payload of TCP on h2 is ***sudo hping3 -d (#Payload) -y 192.168.101.2 .*** First I putted 200 and I got "No Fragmetation Needed" for this but when I putted 500 then the message I got is "Fragmentations Needed" for this. And to get the maximum payload of TCP I used binary search algorithm and got **423 bytes (40 header bytes + 383 data bytes)** as maximum payload of TCP.

(-d – data size : used for data size, -y --dontfrag : used to prevent from fragmentation)

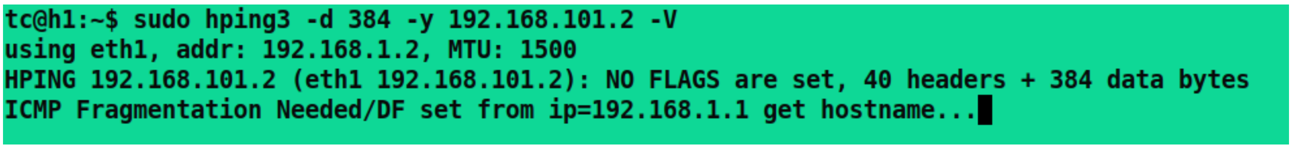
To find the ethernet frame size for the TCP packet size, the command I used on r2 is ***sudo tcpdump -i any -e host 192.168.1.2 .*** And I got x = **439 bytes (16 header bytes + 423 payload bytes).** I used **tcpdump** to capture the ethernet frame on h1 using -e option.

So, Password for r1 is user@(x-5), which is ***user@434***

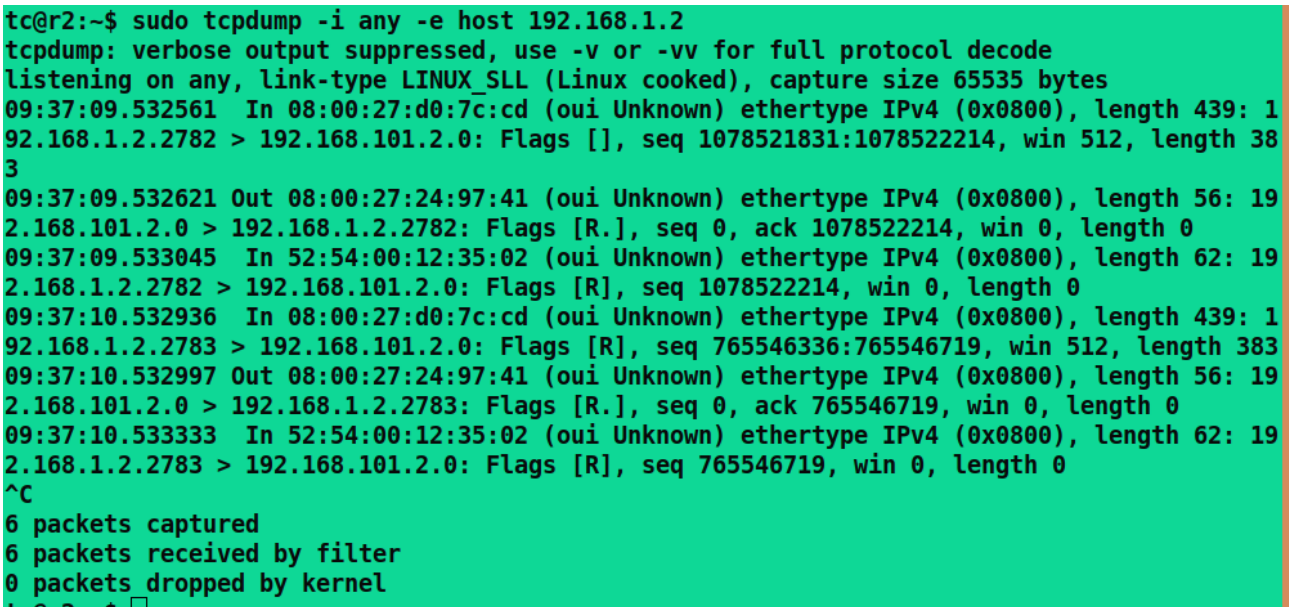
Snapshots:



```
tc@h1:~$ sudo hping3 -d 383 -y 192.168.101.2 -V
using eth1, addr: 192.168.1.2, MTU: 1500
HPING 192.168.101.2 (eth1 192.168.101.2): NO FLAGS are set, 40 headers + 383 data bytes
█
```



```
tc@h1:~$ sudo hping3 -d 384 -y 192.168.101.2 -V
using eth1, addr: 192.168.1.2, MTU: 1500
HPING 192.168.101.2 (eth1 192.168.101.2): NO FLAGS are set, 40 headers + 384 data bytes
ICMP Fragmentation Needed/DF set from ip=192.168.1.1 get hostname...█
```

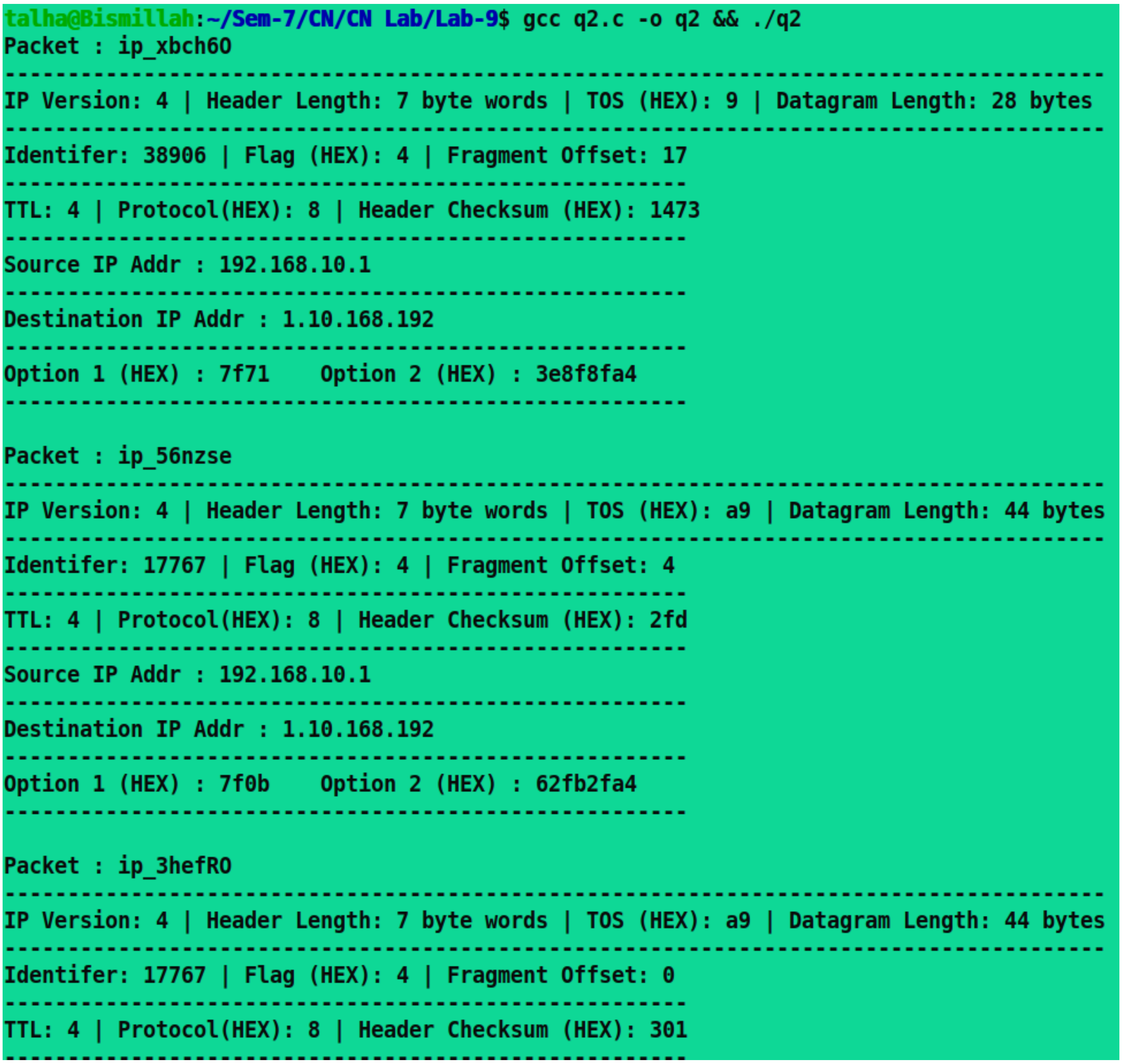


```
tc@r2:~$ sudo tcpdump -i any -e host 192.168.1.2
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on any, link-type LINUX_SLL (Linux cooked), capture size 65535 bytes
09:37:09.532561  In 08:00:27:d0:7c:cd (oui Unknown) ethertype IPv4 (0x0800), length 439: 1
92.168.1.2.2782 > 192.168.101.2.0: Flags [], seq 1078521831:1078522214, win 512, length 38
3
09:37:09.532621 Out 08:00:27:24:97:41 (oui Unknown) ethertype IPv4 (0x0800), length 56: 19
2.168.101.2.0 > 192.168.1.2.2782: Flags [R.], seq 0, ack 1078522214, win 0, length 0
09:37:09.533045  In 52:54:00:12:35:02 (oui Unknown) ethertype IPv4 (0x0800), length 62: 19
2.168.1.2.2782 > 192.168.101.2.0: Flags [R], seq 1078522214, win 0, length 0
09:37:10.532936  In 08:00:27:d0:7c:cd (oui Unknown) ethertype IPv4 (0x0800), length 439: 1
92.168.1.2.2783 > 192.168.101.2.0: Flags [R], seq 765546336:765546719, win 512, length 383
09:37:10.532997 Out 08:00:27:24:97:41 (oui Unknown) ethertype IPv4 (0x0800), length 56: 19
2.168.101.2.0 > 192.168.1.2.2783: Flags [R.], seq 0, ack 765546719, win 0, length 0
09:37:10.533333  In 52:54:00:12:35:02 (oui Unknown) ethertype IPv4 (0x0800), length 62: 19
2.168.1.2.2783 > 192.168.101.2.0: Flags [R], seq 765546719, win 0, length 0
^C
6 packets captured
6 packets received by filter
0 packets dropped by kernel
```

2. Code Overview (q2.cpp)

- **int isChecksumValid(char *byte)** function is checking for validation of checksum. Returning 1 if valid, otherwise 0 (If checksum = 0xffff then it is valid).
- **void header(IPPacket *packet)** function is used to print all header details of legitimate packet.
- **void packetRead(char* filename)** function reads filename from ipfrags directory into IPPacket structure. It calls isChecksumValid function to verify the checksum anf if it finds checksum valid then calls header function to print the header details of the fragments.
- In **main()** function, I'm handling directory (reading "ipfrags" directory and the files inside it).

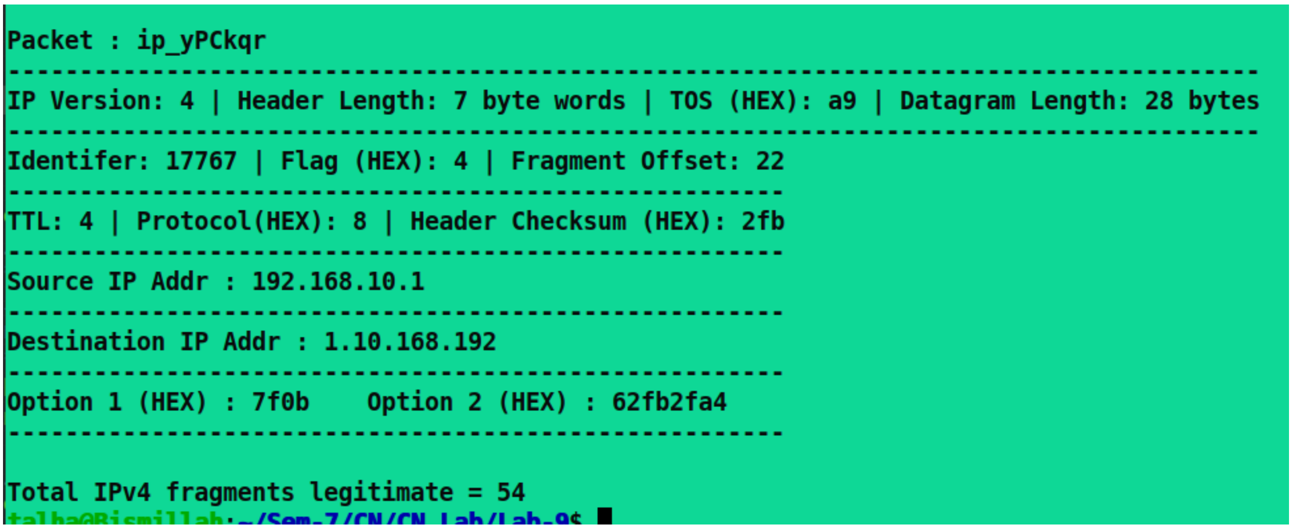
Snapshots:



```
talha@Bismillah:~/Sem-7/CN/CN Lab/Lab-9$ gcc q2.c -o q2 && ./q2
Packet : ip_xbch60
----------------------------------------------------------------------------------------
IP Version: 4 | Header Length: 7 byte words | TOS (HEX): 9 | Datagram Length: 28 bytes
----------------------------------------------------------------------------------------
Identifer: 38906 | Flag (HEX): 4 | Fragment Offset: 17
-------------------------------------------------------
TTL: 4 | Protocol(HEX): 8 | Header Checksum (HEX): 1473
-------------------------------------------------------
Source IP Addr : 192.168.10.1
-------------------------------------------------------
Destination IP Addr : 1.10.168.192
-------------------------------------------------------
Option 1 (HEX) : 7f71    Option 2 (HEX) : 3e8f8fa4
-------------------------------------------------------

Packet : ip_56nzse
----------------------------------------------------------------------------------------
IP Version: 4 | Header Length: 7 byte words | TOS (HEX): a9 | Datagram Length: 44 bytes
----------------------------------------------------------------------------------------
Identifer: 17767 | Flag (HEX): 4 | Fragment Offset: 4
-------------------------------------------------------
TTL: 4 | Protocol(HEX): 8 | Header Checksum (HEX): 2fd
-------------------------------------------------------
Source IP Addr : 192.168.10.1
-------------------------------------------------------
Destination IP Addr : 1.10.168.192
-------------------------------------------------------
Option 1 (HEX) : 7f0b    Option 2 (HEX) : 62fb2fa4
-------------------------------------------------------

Packet : ip_3hefRO
----------------------------------------------------------------------------------------
IP Version: 4 | Header Length: 7 byte words | TOS (HEX): a9 | Datagram Length: 44 bytes
----------------------------------------------------------------------------------------
Identifer: 17767 | Flag (HEX): 4 | Fragment Offset: 0
-------------------------------------------------------
TTL: 4 | Protocol(HEX): 8 | Header Checksum (HEX): 301
-------------------------------------------------------
```



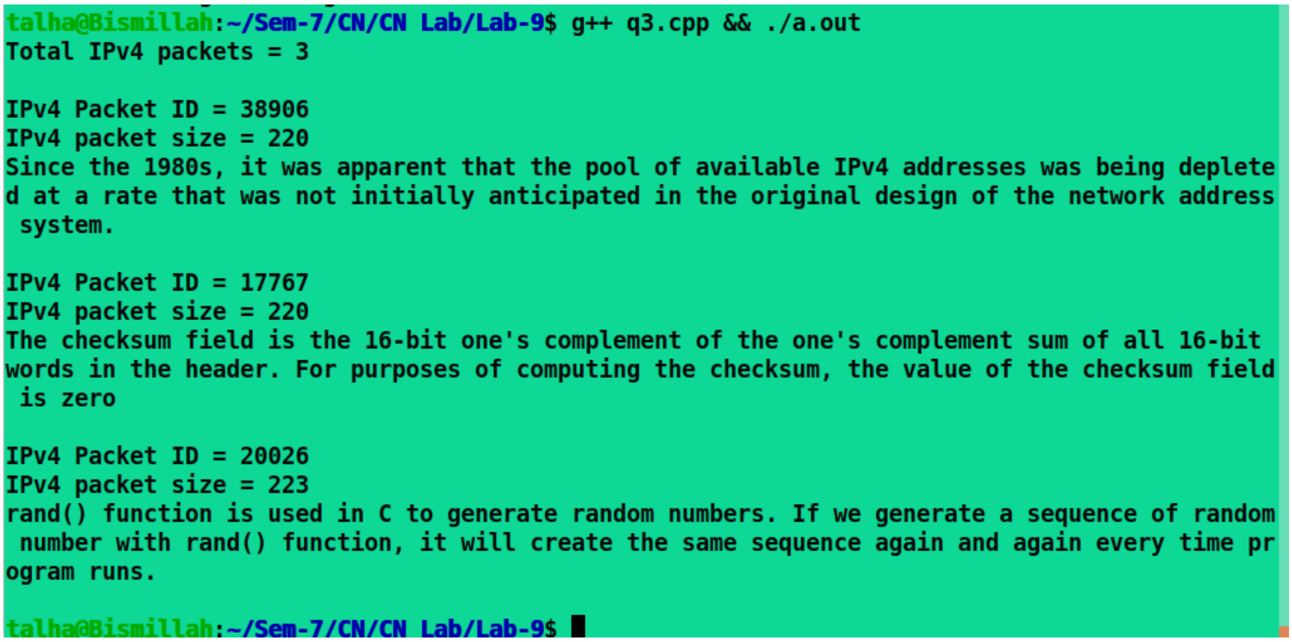
```
Packet : ip_yPCkqr
--------------------------------------------------------------------------------
IP Version: 4 | Header Length: 7 byte words | TOS (HEX): a9 | Datagram Length: 28 bytes
--------------------------------------------------------------------------------
Identifer: 17767 | Flag (HEX): 4 | Fragment Offset: 22
-------------------------------------------------------
TTL: 4 | Protocol(HEX): 8 | Header Checksum (HEX): 2fb
-------------------------------------------------------
Source IP Addr : 192.168.10.1
-------------------------------------------------------
Destination IP Addr : 1.10.168.192
-------------------------------------------------------
Option 1 (HEX) : 7f0b    Option 2 (HEX) : 62fb2fa4
-------------------------------------------------------

Total IPv4 fragments legitimate = 54
talha@Bismillah:~/Sem-7/CN/CN Lab/Lab-9$ 
```

3. Code Overview (q3.cpp)

- **int compare(int a, int b)** function is used to compare two packet number on the basis of offset value. And returning 1, if offset valuse of packet number b is greater than offset valuse of packet number a. Else 0.
- **int isChecksumValid(char *byte)** function is same as previous.
- **void packetRead(char* filename)** function is used to reads filename from ipfrags directory and assigning packet number in respective id.
- In **main()** function, I'm handling directory (reading "ipfrags" directory and the files inside it) and sorting packet on the basis of offset value and then assembling packets and finding data.

Snapshots:



```
talha@Bismillah:~/Sem-7/CN/CN Lab/Lab-9$ g++ q3.cpp && ./a.out
Total IPv4 packets = 3

IPv4 Packet ID = 38906
IPv4 packet size = 220
Since the 1980s, it was apparent that the pool of available IPv4 addresses was being deplete
d at a rate that was not initially anticipated in the original design of the network address
 system.

IPv4 Packet ID = 17767
IPv4 packet size = 220
The checksum field is the 16-bit one's complement of the one's complement sum of all 16-bit
words in the header. For purposes of computing the checksum, the value of the checksum field
 is zero

IPv4 Packet ID = 20026
IPv4 packet size = 223
rand() function is used in C to generate random numbers. If we generate a sequence of random
 number with rand() function, it will create the same sequence again and again every time pr
ogram runs.

talha@Bismillah:~/Sem-7/CN/CN Lab/Lab-9$ 
```