

21CY681– Internet Protocol lab

ASSIGNMENT -4

Name: B.Shebu

Register Number : CYS22005

Title: Analyzing Transport Layer Protocols using Wireshark.

Date of Assignment provided: 27/10/2022

Aim: To analyse transport layer protocols using wireshark.

TCP

1 a) What is the IP address and TCP port number used by the client computer (source) that is transferring the file to gaia.cs.umass.edu?

| Source | Src.port |
|---------------|----------|
| 192.168.1.102 | 1161 |

b) What is the IP address of gaia.cs.umass.edu? On what port number is it sending and receiving TCP segments for this connection?

| Destination | dpt port | Protocol |
|----------------|----------|----------|
| 128.119.245.12 | 80 | TCP |

c) What is the sequence number of the TCP SYN segment that is used to initiate the TCP connection between the client computer and gaia.cs.umass.edu? What is it in the segment that identifies the segment as a SYN segment?

| |
|--|
| Info |
| 1161 → 80 [SYN] Seq=0 Win=16384 Len=0 MSS=1460 SACK_PERM |

✓ **Flags: 0x002 (SYN)**

- 000. = Reserved: Not set
- ...0 = Accurate ECN: Not set
- 0... = Congestion Window Reduced: Not set
-0.. = ECN-Echo: Not set
-0. = Urgent: Not set
-0 = Acknowledgment: Not set
- 0... = Push: Not set
-0.. = Reset: Not set
- >1. = Syn: Set
-0 = Fin: Not set

d) What is the sequence number of the SYNACK segment sent by gaia.cs.umass.edu to the client computer in reply to the SYN? What is the value of the Acknowledgement field in the SYNACK segment? How did gaia.cs.umass.edu determine that value? What is it in the segment that identifies the segment as a SYNACK segment?

The sequence number of the SYNACK segment sent is 0.

```
✓ Transmission Control Protocol, Src Port: 80, Dst Port: 1161
  Source Port: 80
  Destination Port: 1161
  [Stream index: 0]
  [Conversation completeness: Incomplete, DATA (15)]
  [TCP Segment Len: 0]
  Sequence Number: 0 (relative sequence number)
  Sequence Number (raw): 883061785
```

The value of ACK in SYNACK segment is 1.

```
0111 ... = Header Length: 20 bytes (7)
✓ Flags: 0x012 (SYN, ACK)
  000. .... = Reserved: Not set
  ...0 .... = Accurate ECN: Not set
  .... 0... = Congestion Window Reduced: Not set
  .... .0.. = ECN-Echo: Not set
  .... ..0. = Urgent: Not set
  .... ...1 .... = Acknowledgment: Set
  .... .... 0... = Push: Not set
  .... .... .0.. = Reset: Not set
> .... .... ..1. = Syn: Set
```

If SYN & ACK packets have been sent then in the SYNACK segment both the SYN & ACK flags will be set to 1

The SYN,ACK flag identifies it is a SYNACK segment

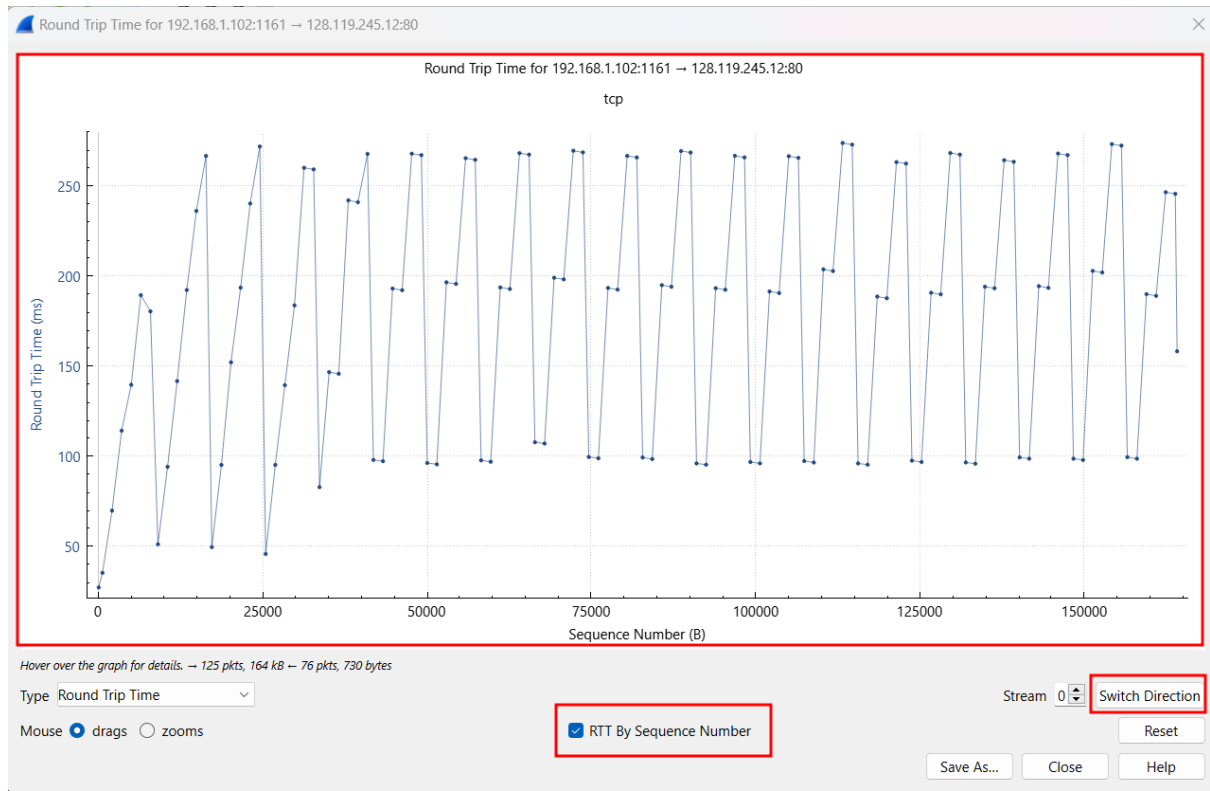
```
0111 ... = Header Length: 20 bytes (7)
✓ Flags: 0x012 (SYN, ACK)
  000. .... = Reserved: Not set
  ...0 .... = Accurate ECN: Not set
  .... 0... = Congestion Window Reduced: Not set
  .... .0.. = ECN-Echo: Not set
  .... ..0. = Urgent: Not set
  .... ...1 .... = Acknowledgment: Set
  .... .... 0... = Push: Not set
  .... .... .0.. = Reset: Not set
> .... .... ..1. = Syn: Set
```

e) What is the sequence number of the TCP segment containing the HTTP POST command? Note that in order to find the POST command, you'll need to dig into the packet content field at the bottom of the Wireshark window, looking for a segment with a "POST" within its DATA field.

In order to find the SEQ number of the POST command , we need to see the SEQ number of the first transferred packet which had the data .

| | | | | | | | | | |
|---|----------|-----------------|---------------|------|----------------|----|-----|------|--|
| 3 | 2004/234 | 13:44:20.593646 | 192.168.1.102 | 1161 | 128.119.245.12 | 80 | TCP | 54 | 1161 → 80 [ACK] Seq=1 Ack=1 Win=17520 Len=0 |
| 4 | 2004/234 | 13:44:20.596858 | 192.168.1.102 | 1161 | 128.119.245.12 | 80 | TCP | 619 | 1161 → 80 [PSH, ACK] Seq=1 Ack=1 Win=17520 Len=565 [TCP segment of a reassembled PDU] |
| 5 | 2004/234 | 13:44:20.600110 | 192.168.1.102 | 1161 | 128.119.245.12 | 80 | TCP | 1219 | 1161 → 80 [PSH, ACK] Seq=1 Ack=1 Win=17520 Len=1460 [TCP segment of a reassembled PDU] |

f) Plot the RTT graph using Wireshark.

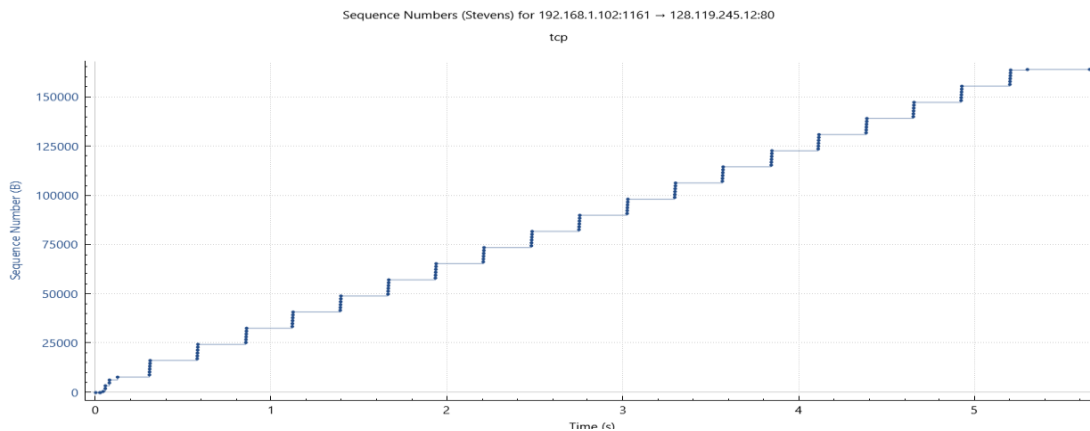


g) What is the length of each of the first six TCP segments (HTTP POST)?

The length of first 6 TCP packet segments are 565, 1460, 1460,1460, 1460, 1460

| |
|---|
| > Internet Protocol Version 4, Src: 192.168.1.102, Dst: 128.119.245.12 |
| > Transmission Control Protocol, Src Port: 1161, Dst Port: 80, Seq: 164041, Ack: 1, Len: 50 |
| > [122 Reassembled TCP Segments (164090 bytes)] #4(565), #5(1460), #7(1460), #8(1460), #10(1460), #11(1460), #13(1460), #18(1460), #19(1460), #20(1460), #21(1460), |
| > Hypertext Transfer Protocol |
| > MIME Multipart Media Encapsulation, Type: multipart/form-data, Boundary: "-----265001916915724" |

h) Are there any retransmitted segments in the trace file? What did you check for (in the trace) in order to answer this question?



No there are no retransmitted segments in the file since there is no drop in the graph

l) What is the throughput (bytes transferred per unit time) for the TCP connection? Explain how you calculated this value.

Throughput = Total amount of data transferred / Total amount of time

The final value of ACK packet is 164091 , so the total amount of data transferred is 164091.

```
60 [80 → 1161 [ACK] Seq=1 Ack=164091 Win=62780 Len=0
784 HTTP/1.1 200 OK (text/html)
```

Time of first packet since reference is 0.026477000

```
[Time delta from previous displayed frame: 0.003212000 seconds]
[Time since reference or first frame: 0.026477000 seconds]
```

Time of last packet since reference is 5.455830000

```
[Time delta from previous displayed frame: 0.007943000 seconds]
[Time since reference or first frame: 5.455830000 seconds]
```

Throughput = 164090 / (5.455830000 – 0.026477000)

= 302222 bytes => **30 kilobytes per second**

UDP

j) Select one UDP packet from your trace. From this packet, determine how many fields there are in the UDP header. Name these fields

There are 4 fields in UDP header.

▼ User Datagram Protocol, Src Port: 4334, Dst Port: 161

Source Port: 4334
Destination Port: 161
Length: 58
Checksum: 0x65f8 [unverified]
[Checksum Status: Unverified]
[Stream index: 1]
> [Timestamps]
UDP payload (50 bytes)

k) By consulting the displayed information in Wireshark's packet content field for this packet, determine the length (in bytes) of each of the UDP header fields.

▼ User Datagram Protocol, Src Port: 4334, Dst Port: 161

Source Port: 4334
Destination Port: 161
Length: 58
Checksum: 0x65f8 [unverified]
[Checksum Status: Unverified]
[Stream index: 1]
> [Timestamps]
UDP payload (50 bytes)

$58 - 50 \Rightarrow 8$

Destination Port: 161

Length: 58

Checksum: 0x65f8 [unverified]
[Checksum Status: Unverified]
[Stream index: 1]
> [Timestamps]
UDP payload (50 bytes)

▼ Simple Network Management Protocol

version: version-1 (0)
community: public
> data: get-request (0)
[\[Response In: 2\]](#)

| | | |
|------|---|------------|
| 0000 | 00 30 c1 61 eb ed 00 08 74 4f 36 23 08 00 45 00 | -0-a.... t |
| 0010 | 00 4e 02 fd 00 00 80 11 00 00 c0 a8 01 66 c0 a8 | -N..... - |
| 0020 | 01 68 10 ee 00 a1 00 3a 65 f8 30 30 02 01 00 04 | -h....: e |
| 0030 | 06 70 75 62 6c 69 63 a0 23 02 02 18 fb 02 01 00 | -public- # |
| 0040 | 02 01 00 20 17 20 15 05 11 2b 05 01 01 01 0b 02 | -a a |

From the above, we can see that each field has length 2 bytes. So total length is 8 bytes.

l) The value in the Length field is the length of what? Verify your claim with your captured UDP packet.

Size of the UDP payload is 62 . Size of each field is 2 . So total length of fields is 8 .The total length of the TCP packet is 70(62+8).

```

User Datagram Protocol, Src Port: 137, Dst Port: 137
  Source Port: 137
  Destination Port: 137
  Length: 70
  Checksum: 0x3eea [unverified]
  [Checksum Status: Unverified]
  [Stream index: 11]
  > [Timestamps]
  UDP payload (62 bytes)
```

m) What is the protocol number for UDP? Give your answer in both hexadecimal and decimal notation.

The protocol number for UDP is 17. In hexadecimal it is 0x11.

```

Internet Protocol Version 4, Src: 192.168.1.102, Dst: 192.168.1.100
  0100 .... = Version: 4
  .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
  Total Length: 90
  Identification: 0x030c (780)
  > 000. .... = Flags: 0x0
  ...0 0000 0000 0000 = Fragment Offset: 0
  Time to Live: 128
  Protocol: UDP (17)
  Header Checksum: 0x0000 [validation disabled]
```

Protocol: UDP (17)

Header Checksum: 0x0000 [validation disabled]

[Header checksum status: Unverified]

```

00 80 ad 73 8d ce 00 08 74 4f 36 20
00 5a 03 0c 00 00 80 11 00 00 c0 a8
01 64 00 89 00 89 00 46 3e ea 97 f1
00 01 00 00 00 00 20 45 4f 45 50 41
41 43 41 43 41 43 41 43 41 43 41 43
41 43 41 43 41 43 41 43 41 43 41 43
```

n) Examine a pair of UDP packets in which your host sends the first UDP packet and the second UDP packet is a reply to this first UDP packet. (Hint: for a second packet to be sent in response to a first packet, the sender of the first packet should be the destination of the second packet). Describe the relationship between the port numbers in the two packets.

When 4334 is the source address the destination is 161 in request. In response it is the exact opposite.

| Src.port | Destination | dpt port |
|----------|---------------|----------|
| 4334 | 192.168.1.104 | 161 |
| 161 | 192.168.1.102 | 4334 |

RESULT –

Thus , we have successfully analyzed TCP and UDP using wiresharkl.