

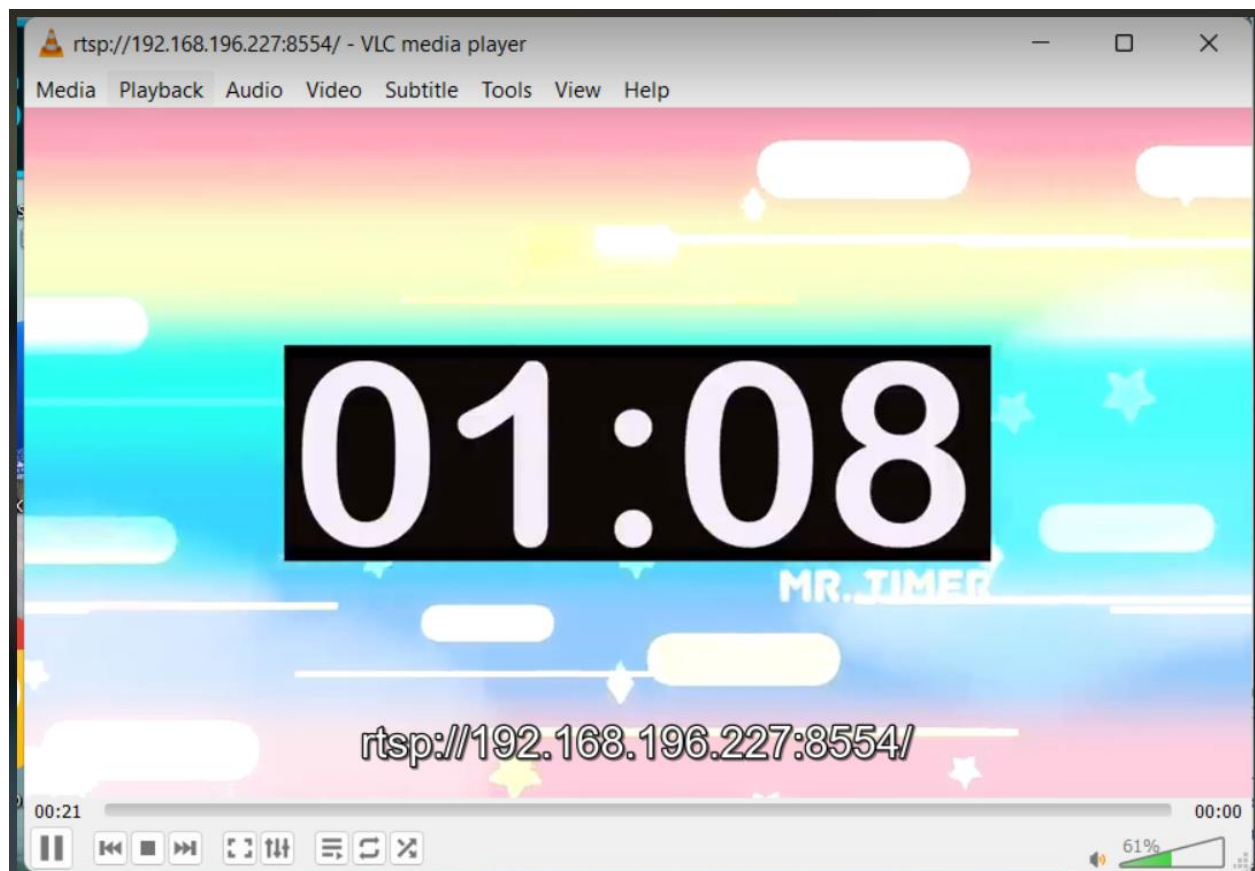
Họ Tên: Phan Trọng Tính

MSSV: 21522683

LỚP: IT005.N110.1

BÁO CÁO THỰC HÀNH LAB3 NHẬP MÔN MẠNG MÁY TÍNH

Video được stream từ máy có địa chỉ IP: 192.168.196.227



Câu 1: Chọn một gói tin UDP, xác định các trường (field) có trong UDP header và giải thích ý nghĩa của mỗi trường đó? Gợi ý: Xem tại phần User Datagram Protocol.

```
▼ User Datagram Protocol, Src Port: 56566, Dst Port: 60429
  Source Port: 56566
  Destination Port: 60429
  Length: 12
  Checksum: 0x6f5c [unverified]
  [Checksum Status: Unverified]
  [Stream index: 0]
  > [Timestamps]
    UDP payload (4 bytes)
```

- Source Port: Xác định cổng người gửi.
- Destination Port: Xác định cổng nhận thông tin.
- Length: 16 bit xác định chiều dài toàn bộ datagram: phần header và dữ liệu. Chiều dài 8 bit khi gói tin chỉ có header.
- Checksum: 16 bit dùng để kiểm tra lỗi của header và dữ liệu.

Câu 2: Qua thông tin hiển thị của Wireshark, xác định độ dài (tính theo byte) của mỗi trường trong UDP header?

Source Port: 2 byte

```
▼ User Datagram Protocol, Src Port: 56566, Dst Port: 60429
  Source Port: 56566
  Destination Port: 60429
  Length: 12
  Checksum: 0x6f5c [unverified]
  [Checksum Status: Unverified]
  [Stream index: 0]
  > [Timestamps]
    UDP payload (4 bytes)
  > Real-Time Transport Protocol
  > Data (4 bytes)
```

0010	00 20 03 42 00 00 80 11	2d 10 c0 a8 c4 46 c0 a8	- -B- - - - -F- -
0020	c4 e3 dc f6 ec 0d 00 0c	6f 5c ce fa ed fe	- - - - - o\ - - -

Source Port (udp.srcport), 2 bytes

Packets: 1763 · Displayed: 1522 (86.3%) | Profile: Default

Destination Port: 2 byte

```
▼ User Datagram Protocol, Src Port: 56566, Dst Port: 60429
  Source Port: 56566
  Destination Port: 60429
  Length: 12
  Checksum: 0x6f5c [unverified]
  [Checksum Status: Unverified]
  [Stream index: 0]
  > [Timestamps]
    UDP payload (4 bytes)
  > Real-Time Transport Protocol
  > Data (4 bytes)
```

0010	00 20 03 42 00 00 80 11	2d 10 c0 a8 c4 46 c0 a8	- -B- - - - -F- -
0020	c4 e3 dc f6 ec 0d 00 0c	6f 5c ce fa ed fe	- - - - - o\ - - -

Destination Port (udp.dstport), 2 bytes

Packets: 1763 · Displayed: 1522 (86.3%) | Profile: Default

Checksum: 2 byte

```
▼ User Datagram Protocol, Src Port: 56566, Dst Port: 60429
  Source Port: 56566
  Destination Port: 60429
  Length: 12
  Checksum: 0x6f5c [unverified]
  [Checksum Status: Unverified]
  [Stream index: 0]
  > [Timestamps]
    UDP payload (4 bytes)
  > Real-Time Transport Protocol
  > Data (4 bytes)
```

0010	00 20 03 42 00 00 80 11	2d 10 c0 a8 c4 46 c0 a8	- -B- - - - -F- -
0020	c4 e3 dc f6 ec 0d 00 0c	6f 5c ce fa ed fe	- - - - - o\ - - -

Details at: <https://www.wireshark.org/doc...AdvChecksums.html> (udp.checksum), 2 bytes | Packets: 1763 · Displayed: 1522 (86.3%) | Profile: Default

Câu 3: Giá trị của trường Length trong UDP header là độ dài của gì? Chứng minh nhận định này?

Trường Length trong UDP header là độ dài của tổng datagram tức là tổng độ dài header và độ dài data.

VD: Trong ví dụ trên là 8 byte và với 4 byte của Data nên Length là 12.

```
▼ User Datagram Protocol, Src Port: 56566, Dst Port: 60429
  Source Port: 56566
  Destination Port: 60429
  Length: 12
  Checksum: 0x6f5c [unverified]
  [Checksum Status: Unverified]
  [Stream index: 0]
  > [Timestamps]
    UDP payload (4 bytes)
```

Câu 4: Số bytes lớn nhất mà payload (phần chứa dữ liệu gốc, không tính UDP header và IP header) của UDP có thể chứa?

- Số bytes lớn nhất mà payload (phần chứa dữ liệu gốc, không tính UDP header và IP header) của UDP có thể chứa là:

+ Kích thước lớn nhất theo lý thuyết là: $2^{16} - 1 = 65535$ bytes.

+ Kích thước cho phép: $65535 - 8$ (8 bytes header) = 65527 bytes.

Câu 5: Giá trị lớn nhất có thể có của port nguồn (Source port)?

Giá trị lớn nhất là: $2^{16} - 1 = 65535$ bytes.

Câu 6: Tìm và kiểm tra một cặp gói tin sử dụng giao thức UDP gồm: gói tin do máy mình gửi và gói tin phản hồi của gói tin đó. Miêu tả mối quan hệ về port number của 2 gói tin này.

Trong quá trình gửi yêu cầu, IP nguồn gửi request packet sẽ trở thành destination port, source port sẽ trở thành destination port, còn IP của người gửi response sẽ trở thành IP source. Source port và destination port của 2 gói tin này ngược nhau.

Lấy ví dụ gói tin 26 và 36:

Request packet:

The image shows a Wireshark packet capture window titled "21522683_RTSP.pcapng". The packet list shows several RTP packets. Packet 26 is selected, showing a source IP of 192.168.196.70 and a destination IP of 192.168.196.227. The details pane shows the following information:

- > Frame 26: 46 bytes on wire (368 bits), 46 bytes captured (368 bits) on interface \Device\NPF_{FA944E43-...}
- > Ethernet II, Src: Chongqin_52:6f:df (40:23:43:52:6f:df), Dst: LiteonTe_3d:18:9f (94:08:53:3d:18:9f)
- > Internet Protocol Version 4, Src: 192.168.196.70, Dst: 192.168.196.227
- > User Datagram Protocol, Src Port: 56566, Dst Port: 60429
 - Source Port: 56566
 - Destination Port: 60429
 - Length: 12
 - Checksum: 0x6f5c [unverified]
 - [Checksum Status: Unverified]
 - [Stream index: 0]
 - > [Timestamps]
 - UDP payload (4 bytes)
- > Real-Time Transport Protocol
- > Data (4 bytes)

Response packet:

21522683_RTSP.pcapng

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

rtsp

No.	Time	Source	Destination	Protocol	Length	Info
36	0.693168	192.168.196.227	192.168.196.70	RTP	475	PT=MPEG-I/II Audio, S
37	0.693168	192.168.196.227	192.168.196.70	RTP	1442	PT=DynamicRTP-Type-96
38	0.693332	192.168.196.227	192.168.196.70	RTP	579	PT=DynamicRTP-Type-96
40	0.703667	192.168.196.227	192.168.196.70	RTP	785	PT=DynamicRTP-Type-96
41	0.724307	192.168.196.227	192.168.196.70	RTP	638	PT=DynamicRTP-Type-96
42	0.728256	192.168.196.227	192.168.196.70	RTP	475	PT=MPEG-I/II Audio, S
43	0.746072	192.168.196.227	192.168.196.70	RTP	475	PT=MPEG-I/II Audio, S
45	0.770118	192.168.196.227	192.168.196.70	RTP	475	PT=MPEG-I/II Audio, S

> Frame 36: 475 bytes on wire (3800 bits), 475 bytes captured (3800 bits) on interface \Device\NPF_{FA944E43-...}

> Ethernet II, Src: LiteonTe_3d:18:9f (94:08:53:3d:18:9f), Dst: Chongqin_52:6f:df (40:23:43:52:6f:df)

> Internet Protocol Version 4, Src: 192.168.196.227, Dst: 192.168.196.70

> User Datagram Protocol, Src Port: 60429, Dst Port: 56566

Source Port: 60429

Destination Port: 56566

Length: 441

Checksum: 0x7a52 [unverified]

[Checksum Status: Unverified]

[Stream index: 0]

> [Timestamps]

UDP payload (433 bytes)

> Real-Time Transport Protocol

Câu 7: Tìm địa chỉ IP và TCP port của máy Client?

21522683_TCP.pcapng

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

tcp

No.	Time	Source	Destination	Protocol	Length	Info
44	6.546935	192.168.196.227	192.168.196.70	TCP	66	[TCP Port numbers reused]
45	6.547092	192.168.196.70	192.168.196.227	TCP	54	54181 → 8080 [ACK] Seq=1
46	6.547411	192.168.196.70	192.168.196.227	HTTP	159	GET / HTTP/1.0
47	6.575045	192.168.196.227	192.168.196.70	TCP	157	8080 → 54181 [PSH, ACK]
48	6.617756	192.168.196.70	192.168.196.227	TCP	54	54181 → 8080 [ACK] Seq=1
49	6.676749	192.168.196.70	31.13.75.17	TLSv1.2	83	Application Data
50	6.723933	31.13.75.17	192.168.196.70	TCP	56	443 → 54131 [ACK] Seq=1
51	6.724059	27.71.113.32	192.168.196.70	TLSv1.2	78	Application Data

> Frame 47: 157 bytes on wire (1256 bits), 157 bytes captured (1256 bits) on interface \Device\NPF_{FA944E43-...}

> Ethernet II, Src: LiteonTe_3d:18:9f (94:08:53:3d:18:9f), Dst: Chongqin_52:6f:df (40:23:43:52:6f:df)

> Internet Protocol Version 4, Src: 192.168.196.227, Dst: 192.168.196.70

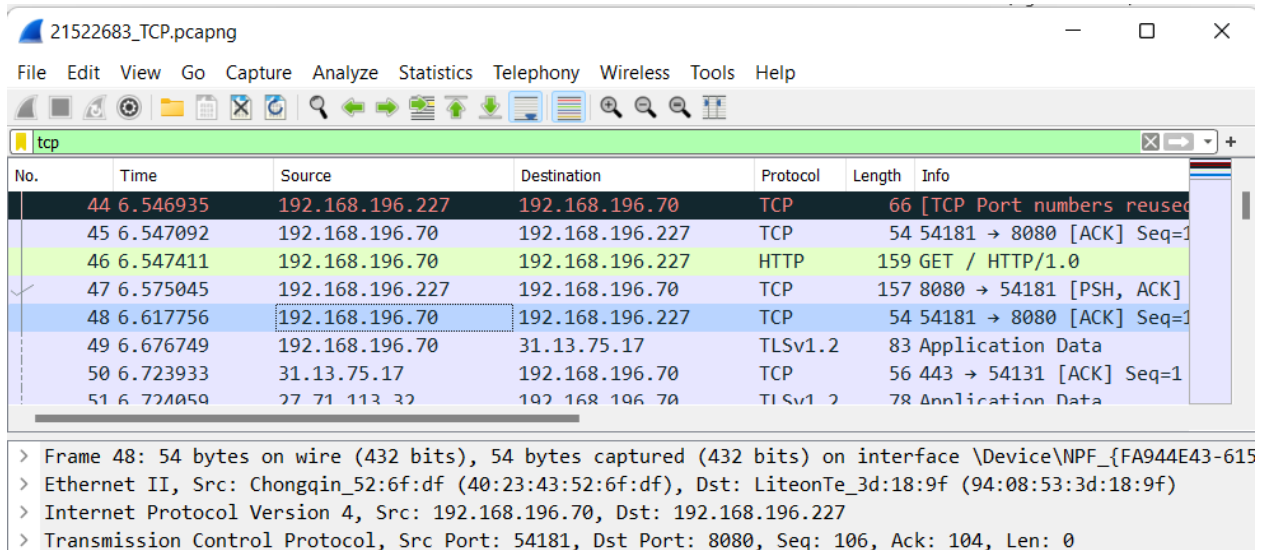
> Transmission Control Protocol, Src Port: 8080, Dst Port: 54181, Seq: 1, Ack: 106, Len: 103

Gói tin số 47 ta thấy được:

IP của Client : 192.168.196.227

TCP port của Client : 54181

Câu 8: Tìm địa chỉ IP của Server? Kết nối TCP dùng để gửi và nhận các segments sử dụng port nào?



No.	Time	Source	Destination	Protocol	Length	Info
44	6.546935	192.168.196.227	192.168.196.70	TCP	66	[TCP Port numbers reused]
45	6.547092	192.168.196.70	192.168.196.227	TCP	54	54181 → 8080 [ACK] Seq=1
46	6.547411	192.168.196.70	192.168.196.227	HTTP	159	GET / HTTP/1.0
47	6.575045	192.168.196.227	192.168.196.70	TCP	157	8080 → 54181 [PSH, ACK]
48	6.617756	192.168.196.70	192.168.196.227	TCP	54	54181 → 8080 [ACK] Seq=1
49	6.676749	192.168.196.70	31.13.75.17	TLSv1.2	83	Application Data
50	6.723933	31.13.75.17	192.168.196.70	TCP	56	443 → 54131 [ACK] Seq=1
51	6.724050	27.71.113.32	192.168.196.70	TLSv1.2	78	Application Data

> Frame 48: 54 bytes on wire (432 bits), 54 bytes captured (432 bits) on interface \Device\NPF_{FA944E43-615...}

> Ethernet II, Src: Chongqin_52:6f:df (40:23:43:52:6f:df), Dst: LiteonTe_3d:18:9f (94:08:53:3d:18:9f)

> Internet Protocol Version 4, Src: 192.168.196.70, Dst: 192.168.196.227

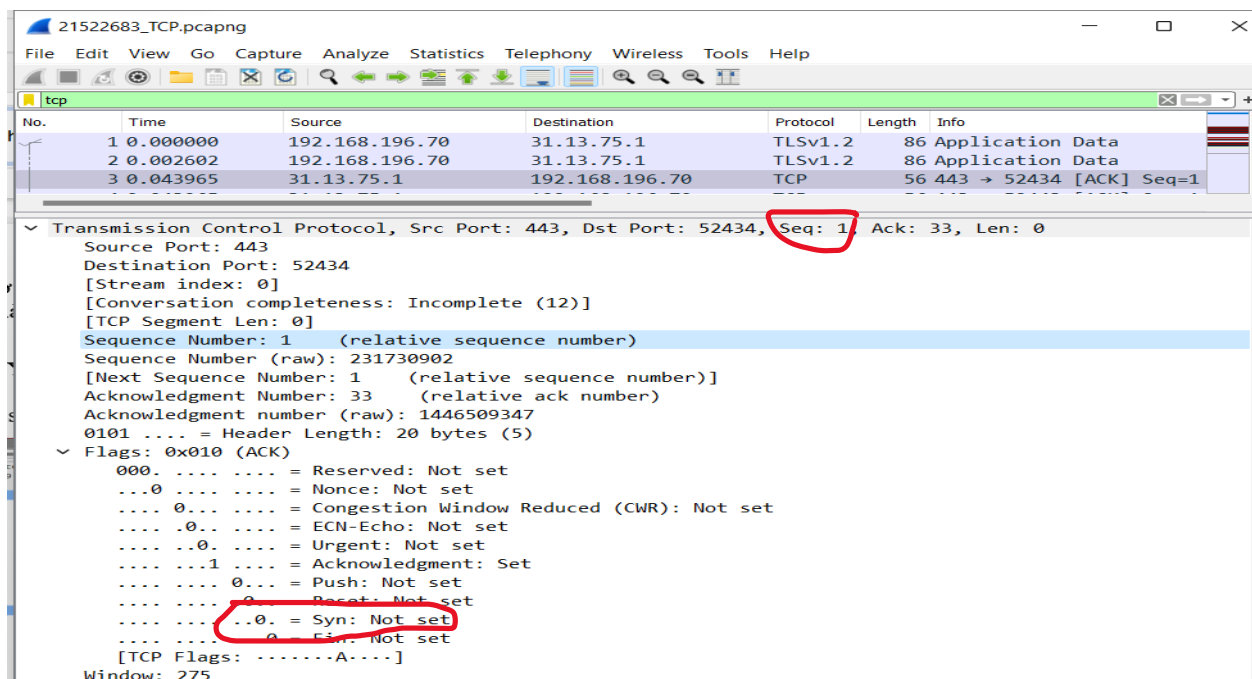
> Transmission Control Protocol, Src Port: 54181, Dst Port: 8080, Seq: 106, Ack: 104, Len: 0

Nhìn vào gói tin số 48:

IP của server là 192.168.196.70

Kết nối TCP dùng để gửi và nhận các segments sử dụng port : 8080

Câu 9: TCP SYN segment (gói tin TCP có cờ SYN) sử dụng sequence number nào để khởi tạo kết nối TCP giữa client và server? Thành phần nào trong segment cho ta biết segment đó là TCP SYN segment?



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.196.70	31.13.75.1	TLSv1.2	86	Application Data
2	0.002602	192.168.196.70	31.13.75.1	TLSv1.2	86	Application Data
3	0.043965	31.13.75.1	192.168.196.70	TCP	56	443 → 52434 [ACK] Seq=1

Transmission Control Protocol, Src Port: 443, Dst Port: 52434, Seq: 1, Ack: 33, Len: 0

Source Port: 443

Destination Port: 52434

[Stream index: 0]

[Conversation completeness: Incomplete (12)]

[TCP Segment Len: 0]

Sequence Number: 1 (relative sequence number)

Sequence Number (raw): 231730902

[Next Sequence Number: 1 (relative sequence number)]

Acknowledgment Number: 33 (relative ack number)

Acknowledgment number (raw): 1446509347

0101 = Header Length: 20 bytes (5)

Flags: 0x010 (ACK)

000. = Reserved: Not set

...0 = Nonce: Not set

...0 = Congestion Window Reduced (CWR): Not set

....0 = ECN-Echo: Not set

....0 = Urgent: Not set

....1 = Acknowledgment: Set

....0 = Push: Not set

....0 = Reset: Not set

....0 = Syn: Not set

....0 = Fin: Not set

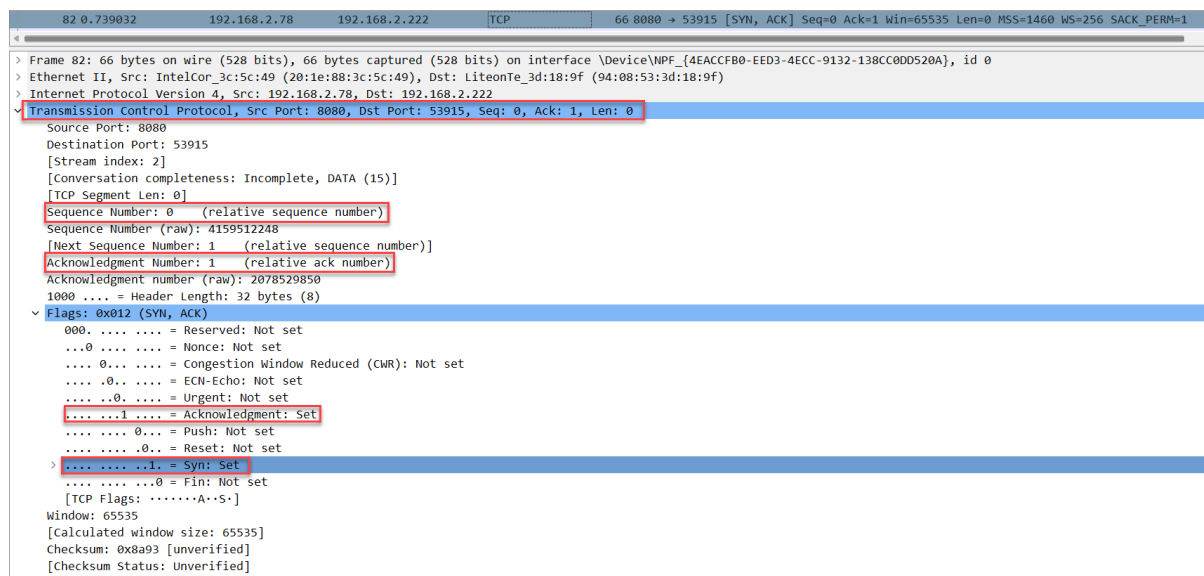
[TCP Flags:A....]

Window: 275

TCP SYN segment (gói tin TCP có cờ SYN) được sử dụng sequence number 1 để khởi tạo kết nối TCP giữa client và server .

Như hình thấy được Flags cờ SYN được set bằng 0=> là TCP segment

Câu 10: Tìm sequence number của gói tin SYN/ACK segment được gửi bởi server đến client để trả lời cho SYN segment? Tìm giá trị của Acknowledgement trong SYN/ACK segment? Làm sao server có thể xác định giá trị đó? Thành phần nào trong segment cho ta biết segment đó là SYN/ACK segment?



-Giá trị của Acknowledgement trong SYN/ACK segment:

Sequence number: 0

Acknowledgement: 1

- Giá trị của Acknowledgement trong gói SYN/ACK được xác định bởi Server :

Server khởi tạo sequence number đầu tiên SYN segment từ client là 0 => giá trị của Acknowledgement trong gói SYN/ACK là 1. Một segment sẽ

là một SYN/ACK segment nếu có cả cờ SYN và cờ ACK đều set là 1.

-Thành phần Flags sẽ cho ta biết đó là SYN/ACK segment.

Câu 11: Chỉ ra 6 segment đầu tiên mà server gửi cho Client (dựa vào Số thứ tự gói – No)

- Tìm sequence number của 6 segments đầu tiên đó?
- Xác định thời gian mà mỗi segment được gửi, thời gian ACK cho mỗi segment được nhận?
- Đưa ra sự khác nhau giữa thời gian mà mỗi segment được gửi và thời gian ACK cho mỗi segment được nhận bằng cách tính RTT (Round Trip Time) cho 6 segments này?

STT	Thời gian gửi	Thời gian nhận ACK	RTT (Round trip time)
47	6.575045	6.617756	0.042711
72	7.167450	7.167526	0.000076
75	7.194596	7.194680	0.000084
77	7.207795	7.253728	0.045933
80	7.273347	7.273430	0.000083
84	7.321617	7.321697	0.00008

47	6.575045	192.168.196.227	192.168.196.70	TCP	1578080 → 54181 [PSH, ACK] Seq=1 Ack=106 Win=1049600 Len=103 [TCP segment of a reassembled PDU]
48	6.617756	192.168.196.70	192.168.196.227	TCP	54 54181 → 8080 [ACK] Seq=106 Ack=104 Win=131072 Len=0
72	7.167450	192.168.196.227	192.168.196.70	TCP	414 8080 → 54181 [PSH, ACK] Seq=13448 Ack=106 Win=1049600 Len=360 [TCP segment of a reassembled PDU]
73	7.167526	192.168.196.70	192.168.196.227	TCP	54 54181 → 8080 [ACK] Seq=106 Ack=13808 Win=131328 Len=0
75	7.194596	192.168.196.227	192.168.196.70	TCP	11228080 → 54181 [PSH, ACK] Seq=15268 Ack=106 Win=1049600 Len=1068 [TCP segment of a reassembled PDU]
76	7.194680	192.168.196.70	192.168.196.227	TCP	54 54181 → 8080 [ACK] Seq=106 Ack=16336 Win=131328 Len=0
77	7.207795	192.168.196.227	192.168.196.70	TCP	12078080 → 54181 [PSH, ACK] Seq=16336 Ack=106 Win=1049600 Len=1153 [TCP segment of a reassembled PDU]
78	7.253728	192.168.196.70	192.168.196.227	TCP	54 54181 → 8080 [ACK] Seq=106 Ack=17489 Win=130048 Len=0
80	7.273347	192.168.196.227	192.168.196.70	TCP	618080 → 54181 [PSH, ACK] Seq=18949 Ack=106 Win=1049600 Len=7 [TCP segment of a reassembled PDU]
81	7.273430	192.168.196.70	192.168.196.227	TCP	54 54181 → 8080 [ACK] Seq=106 Ack=18956 Win=131328 Len=0
84	7.321617	192.168.196.227	192.168.196.70	TCP	10348080 → 54181 [PSH, ACK] Seq=21876 Ack=106 Win=1049600 Len=980 [TCP segment of a reassembled PDU]
85	7.321697	192.168.196.70	192.168.196.227	TCP	54 54181 → 8080 [ACK] Seq=106 Ack=22856 Win=131328 Len=0

Câu 12: Có segment nào được gửi lại hay không? Thông tin nào trong quá trình truyền tin cho chúng ta biết điều đó?

Mỗi chấm trong biểu đồ tượng trưng cho một TCP segment có sequence number tương ứng với thời gian segment đó được gửi đi.

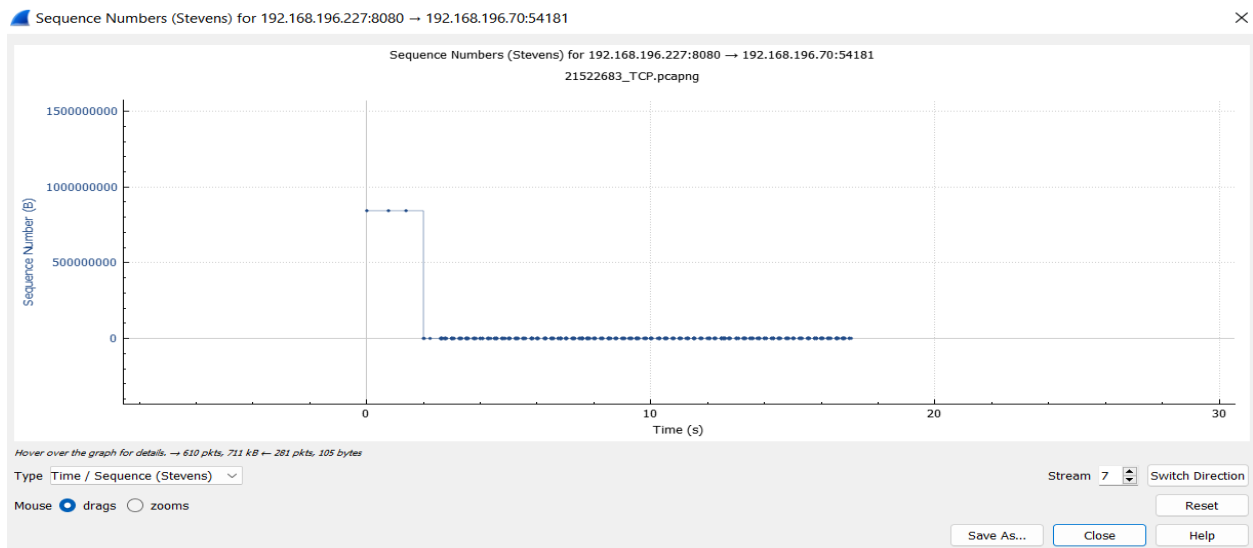
Lưu ý là một chồng các dấu chấm tương ứng với một chuỗi các gói tin được gửi liên tiếp nhau. Nếu có 2 chấm có cùng sequence number được gửi ở 2 thời điểm khác nhau (có 2 điểm nằm ngang) => Có gói tin được gửi lại và gói tin có Source Port là 54181 được gửi lại nhiều lần và biểu đồ bên dưới cũng diễn tả rõ là có rất nhiều chấm nằm ngang nhau.

21522683_TCP.pcapng

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Apply a display filter ... <Ctrl-F>

No.	Time	Source	Destination	Protocol	Length	Info
74	7.194596	192.168.196.227	192.168.196.70	TCP	1514	8080 → 54181 [ACK] Seq=13808 Ack=106 Win=1049600 Len=1460 [TCP segment of a reassembled PDU]
75	7.194596	192.168.196.227	192.168.196.70	TCP	1122	8080 → 54181 [PSH, ACK] Seq=15268 Ack=106 Win=1049600 Len=1068 [TCP segment of a reassembled PDU]
76	7.194680	192.168.196.70	192.168.196.227	TCP	54	54181 → 8080 [ACK] Seq=106 Ack=16336 Win=131328 Len=0
77	7.287795	192.168.196.227	192.168.196.70	TCP	1287	8080 → 54181 [PSH, ACK] Seq=16336 Ack=106 Win=1049600 Len=1153 [TCP segment of a reassembled PDU]
78	7.253728	192.168.196.70	192.168.196.227	TCP	54	54181 → 8080 [ACK] Seq=106 Ack=17489 Win=130048 Len=0
79	7.273347	192.168.196.227	192.168.196.70	TCP	1514	8080 → 54181 [ACK] Seq=17489 Ack=106 Win=1049600 Len=1460 [TCP segment of a reassembled PDU]
80	7.273347	192.168.196.227	192.168.196.70	TCP	61	8080 → 54181 [PSH, ACK] Seq=18949 Ack=106 Win=1049600 Len=7 [TCP segment of a reassembled PDU]
81	7.273430	192.168.196.70	192.168.196.227	TCP	54	54181 → 8080 [ACK] Seq=106 Ack=18956 Win=131328 Len=0
82	7.321617	192.168.196.227	192.168.196.70	TCP	1514	8080 → 54181 [ACK] Seq=18956 Ack=106 Win=1049600 Len=1460 [TCP segment of a reassembled PDU]
83	7.321617	192.168.196.227	192.168.196.70	TCP	1514	8080 → 54181 [ACK] Seq=20416 Ack=106 Win=1049600 Len=1460 [TCP segment of a reassembled PDU]
84	7.321617	192.168.196.227	192.168.196.70	TCP	1034	8080 → 54181 [PSH, ACK] Seq=21876 Ack=106 Win=1049600 Len=980 [TCP segment of a reassembled PDU]
85	7.321697	192.168.196.70	192.168.196.227	TCP	54	54181 → 8080 [ACK] Seq=106 Ack=22856 Win=131328 Len=0
86	7.321971	192.168.196.227	192.168.196.70	TCP	1514	8080 → 54181 [ACK] Seq=22856 Ack=106 Win=1049600 Len=1460 [TCP segment of a reassembled PDU]
87	7.321971	192.168.196.227	192.168.196.70	TCP	1514	8080 → 54181 [ACK] Seq=24316 Ack=106 Win=1049600 Len=1460 [TCP segment of a reassembled PDU]
88	7.322812	192.168.196.70	192.168.196.227	TCP	54	54181 → 8080 [ACK] Seq=106 Ack=25776 Win=131328 Len=0
89	7.322138	192.168.196.227	192.168.196.70	TCP	862	8080 → 54181 [PSH, ACK] Seq=25776 Ack=106 Win=1049600 Len=808 [TCP segment of a reassembled PDU]



21522683_TCP.pcapng

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

tcp

	Time	Source	Destination	Protocol	Length	Info
35	4.547685	192.168.196.70	192.168.196.227	TCP	66	54181 → 8080 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
36	4.554457	192.168.196.227	192.168.196.70	TCP	54	8080 → 54181 [RST, ACK] Seq=843295303 Ack=1 Win=0 Len=0
37	5.068581	192.168.196.70	192.168.196.227	TCP	66	[TCP Retransmission] [TCP Port numbers reused] 54181 → 8080 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
38	5.228134	192.168.196.70	64.233.189.188	TCP	55	54130 → 5228 [ACK] Seq=1 Ack=1 Win=513 Len=1
39	5.278143	64.233.189.188	192.168.196.70	TCP	66	5228 → 54130 [ACK] Seq=1 Ack=2 Win=265 Len=0 SLE=1 SRE=2
40	5.317065	192.168.196.227	192.168.196.70	TCP	54	8080 → 54181 [RST, ACK] Seq=843295303 Ack=1 Win=0 Len=0
41	5.824350	192.168.196.70	192.168.196.227	TCP	66	[TCP Retransmission] [TCP Port numbers reused] 54181 → 8080 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
42	5.934412	192.168.196.227	192.168.196.70	TCP	54	8080 → 54181 [RST, ACK] Seq=843295303 Ack=1 Win=0 Len=0
43	6.448333	192.168.196.70	192.168.196.227	TCP	66	[TCP Retransmission] [TCP Port numbers reused] 54181 → 8080 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
44	6.546935	192.168.196.227	192.168.196.70	TCP	66	[TCP Port numbers reused] 8080 → 54181 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM=1
45	6.547092	192.168.196.70	192.168.196.227	TCP	54	54181 → 8080 [ACK] Seq=1 Ack=1 Win=131328 Len=0
46	6.547411	192.168.196.70	192.168.196.227	HTTP	159	GET / HTTP/1.0
47	6.575805	192.168.196.227	192.168.196.70	TCP	157	8080 → 54181 [PSH, ACK] Seq=1 Ack=106 Win=1049600 Len=103 [TCP segment of a reassembled PDU]
48	6.617756	192.168.196.70	192.168.196.227	TCP	54	54181 → 8080 [ACK] Seq=106 Ack=104 Win=131072 Len=0
49	6.676749	192.168.196.70	31.13.75.17	TLv1.2	83	Application Data

Ngoài ra ta có thể thấy gói tin 43 bị lỗi và sẽ được gửi lại.