TCP COMMUNICATION ANALYSIS

Part 1:

1. The messages:

tcp_server.py > ...

```python
import socket
server = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
server.bind(('', 12345))
server.listen(5)

while True:
    client_socket, client_address = server.accept()
    print('Connection from: ', client_address)
    data = client_socket.recv(100)
    print('Received: ', data)
    client_socket.send(data.upper())
    data = client_socket.recv(100)
    print('Received: ', data)
    client_socket.send(data.upper())
    client_socket.close()
    print('Client disconnected')
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\net2>  c:; cd 'c:\net2'; & 'c:\Users\User\AppData\Local\Python\pythoncore-3.14-64\python.exe' 'c:\Users\User\.vscode\extensions\ms-python.debugpy-2025.18.0-win32-x64\bundled\libs\debugpy\launcher' '59957' '--' 'c:\net2\tcp_server.py'
Connection from:  ('192.168.1.224', 54962)
Received:  b'Noya Shindler and Oscar Stilogalo'
Received:  b'2152162849 327613030'
Client disconnected
```

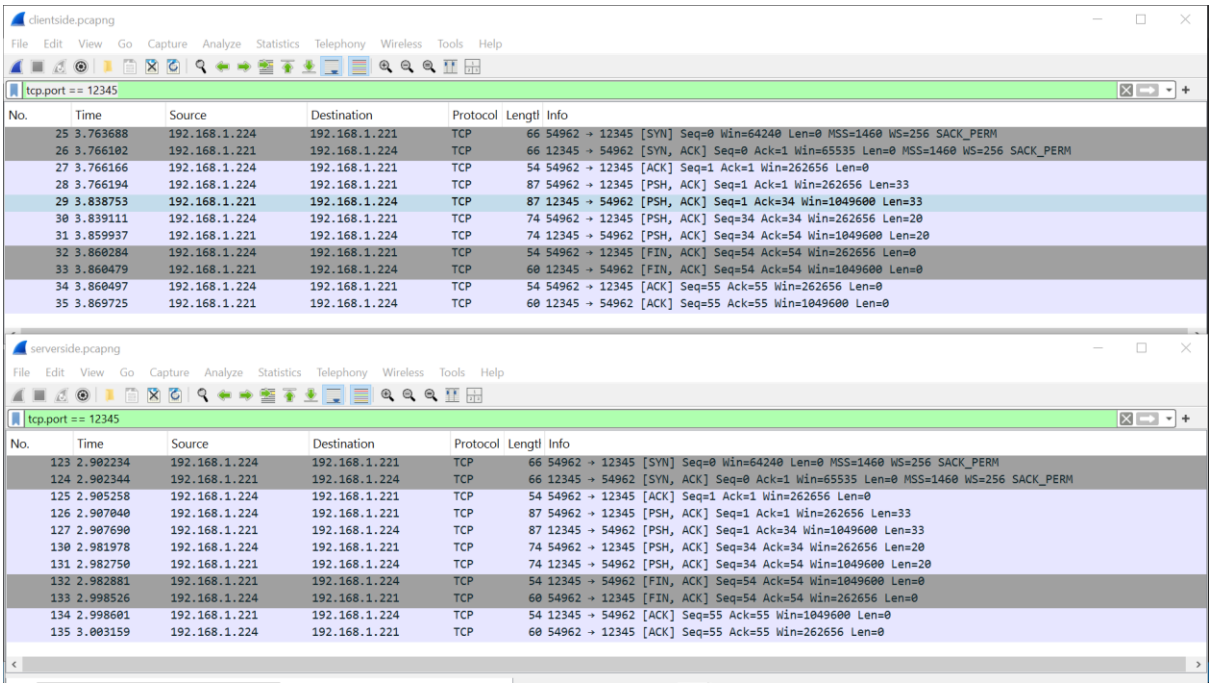
tcp_client.py > ...

```python
import socket
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect(('192.168.1.221', 12345))
s.send(b'Noya Shindler and Oscar Stilogalo')
data = s.recv(100)
print("Server sent: ", data)
s.send(b'2152162849 327613030')
data = s.recv(100)
print("Server sent: ", data)
s.close()
```

PROBLEMS OUTPUT TERMINAL ... Python Debug Console

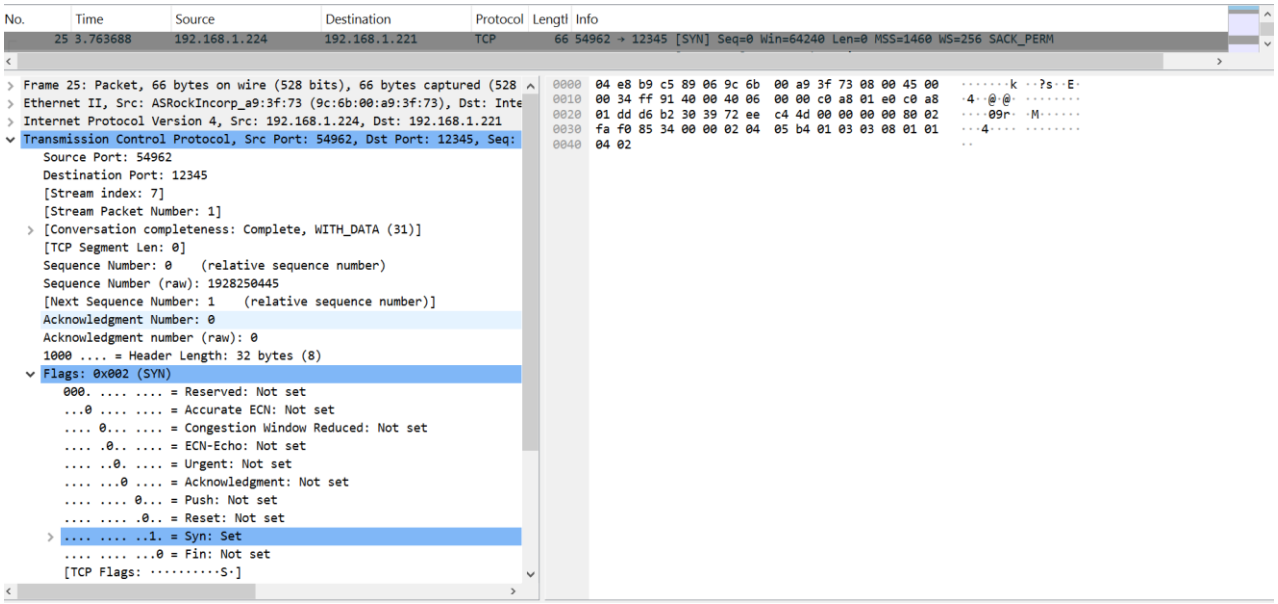
```
PS O:\net2>  o:; cd 'o:\net2'; & 'c:\Users\oscar\AppData\Local\Programs\Python\Python313\python.exe' 'c:\Users\oscar\.vscode\extensions\ms-python.debugpy-2025.18.0-win32-x64\bundled\libs\debugpy\launcher' '54958' '--' 'o:\net2\tcp_client.py'
Server sent:  b'NOYA SHINDLER AND OSCAR STILOGALO'1ff1-d1ce-4149-afe8-149b3dbc09a
Server sent:  b'2152162849 327613030'
PS O:\net2>
```

2. Wireshark



First message- client to server (224->221) only syn flag asks to make a connection

Sequence Number (raw): 1928250445 – where the stack starts

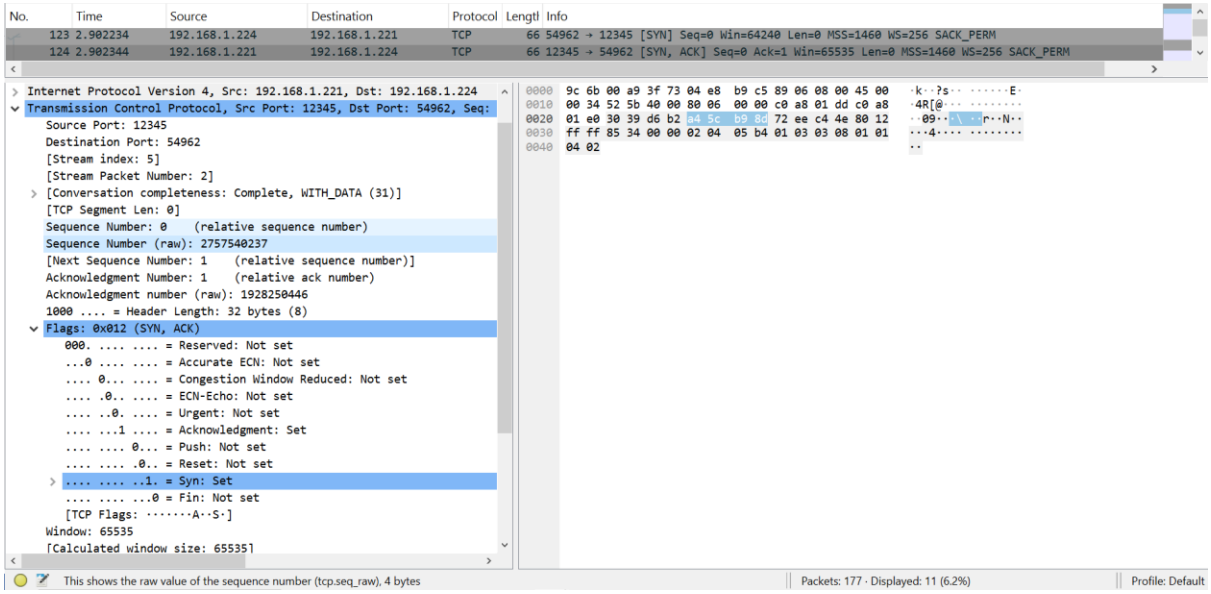


Second message- server to client (221->224), 2rd part of 3 part handshake,

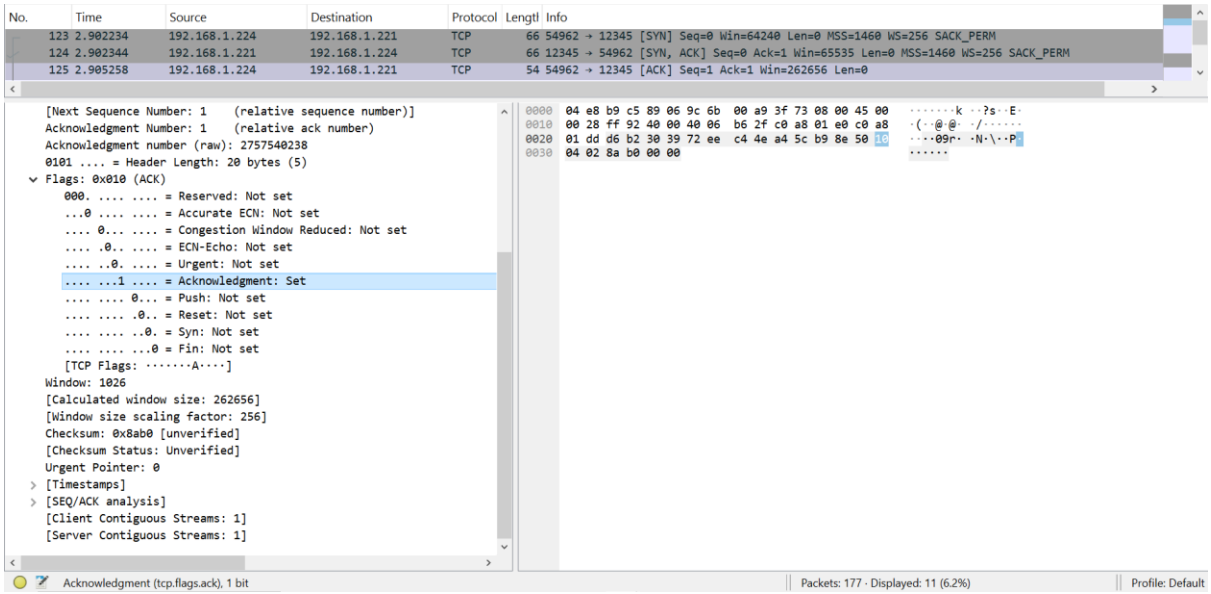
Acknowledgment number (raw): 1928250446 for one bit after the seq in the client

for syn the client sent.

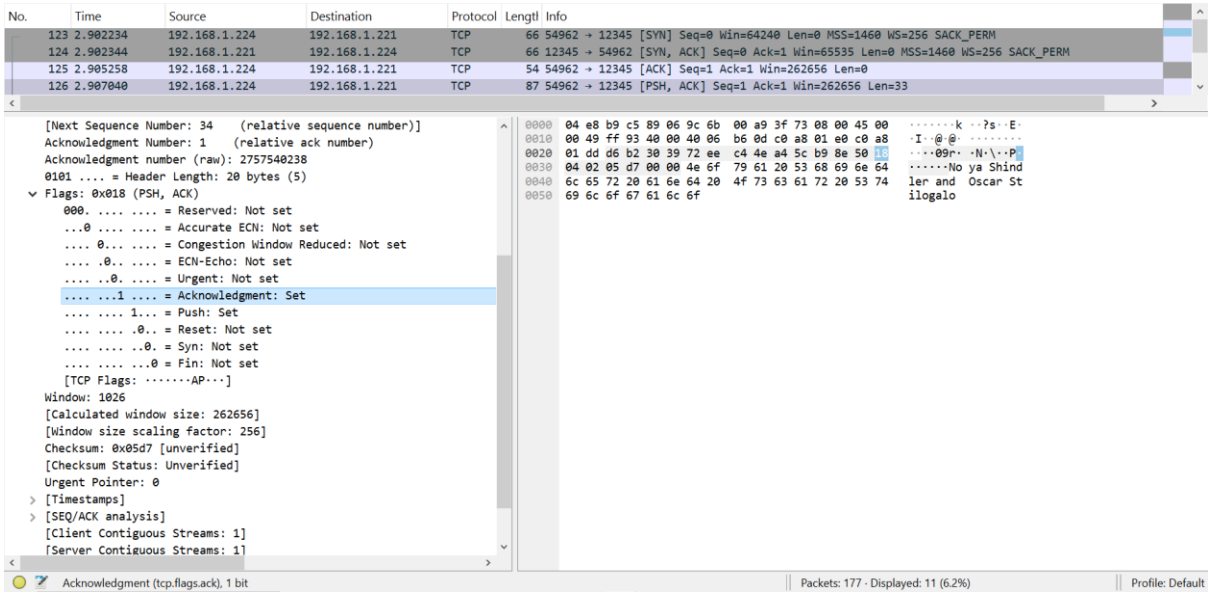
Sequence Number (raw): 2757540237: which is where the stack starts for the server



Third message- client to server (224->221),3rd part of 3part handshake only ack flag sets both server and Acknowledgment number (raw): 2757540238 for the 1 bit of syn

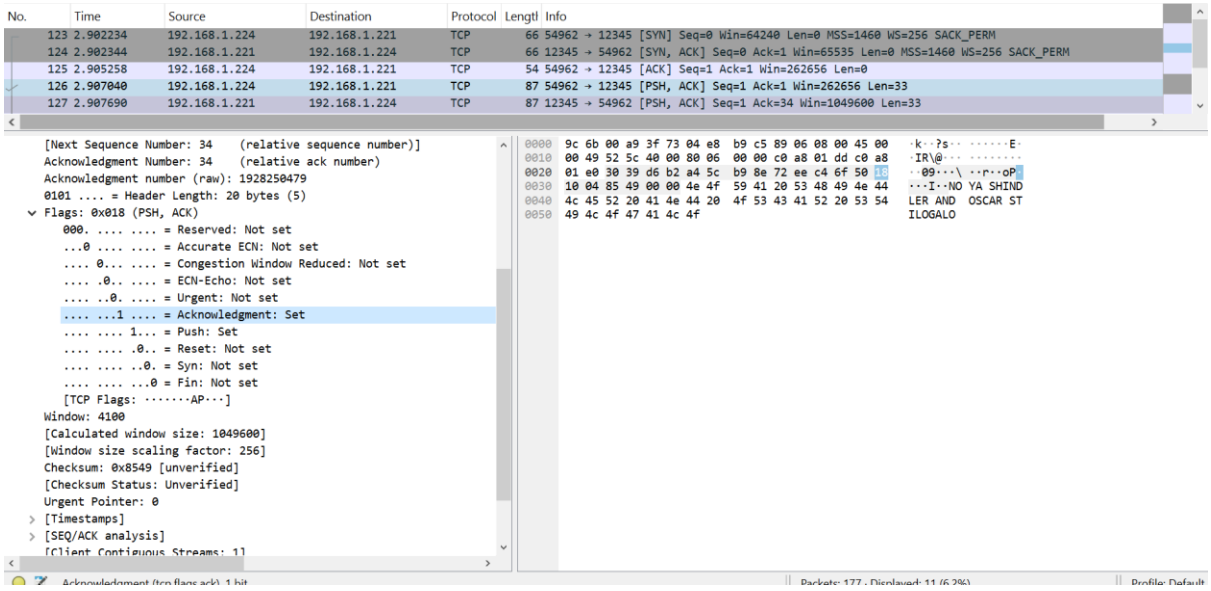


Forth message- client to server (224->221), sent the names with ack and psh flag Ack is at 1(dynamically) because that's where we start the ack list, note that msg len is 33



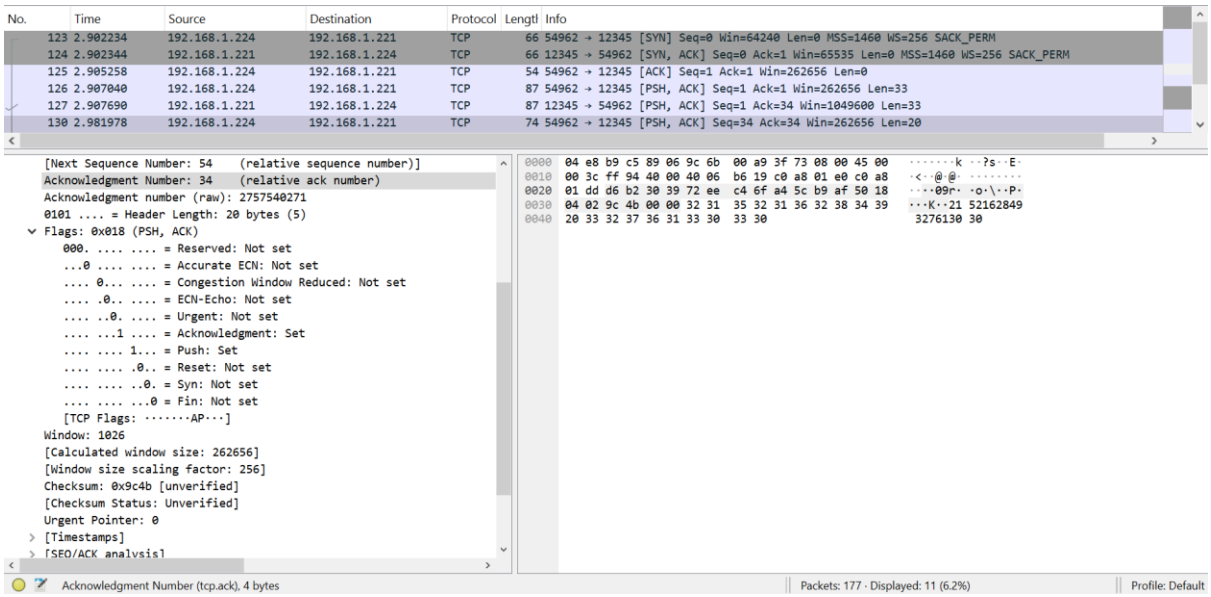
Fifth message- server to client (221->224), sent Acknowledgment number (raw): 1928250479 (meaning took all the 79-46 = 33 len of the msg) and sent together its response which as well has len 33

Sequence Number (raw): 2757540238 is still at the start

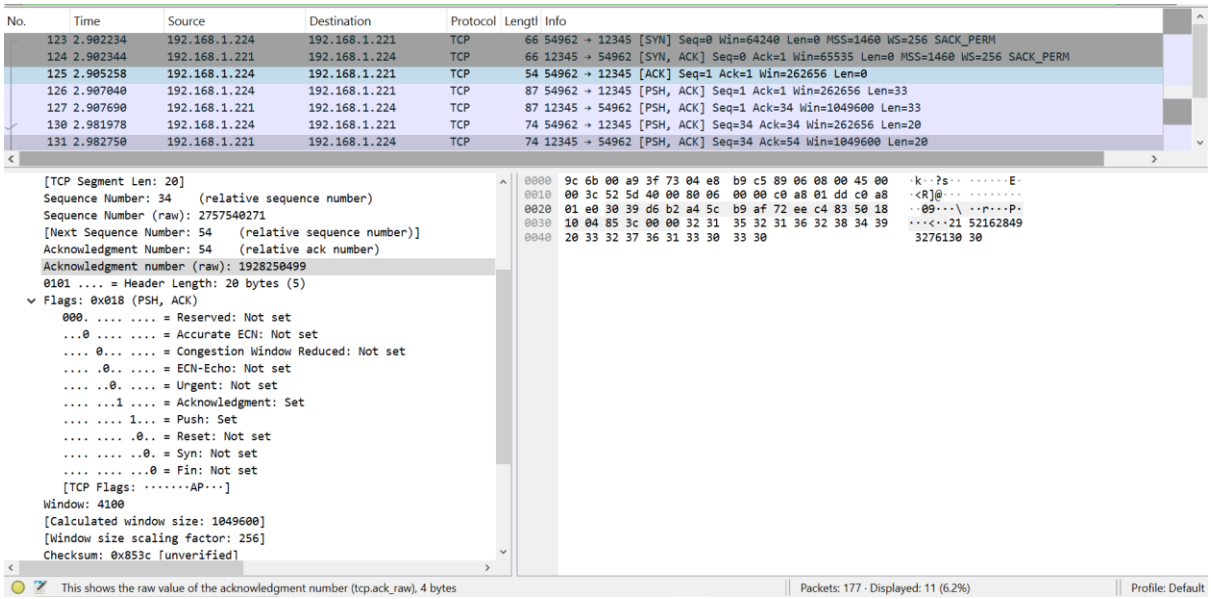


Sixth message- client to server (224->221), sent Acknowledgment number (raw): 2757540271 (meaning took all the 71 -38 = 33 len of the msg) and sent the ids which have len 20

Acknowledgment number (raw): 2757540271 still after the 33 + 1 bits

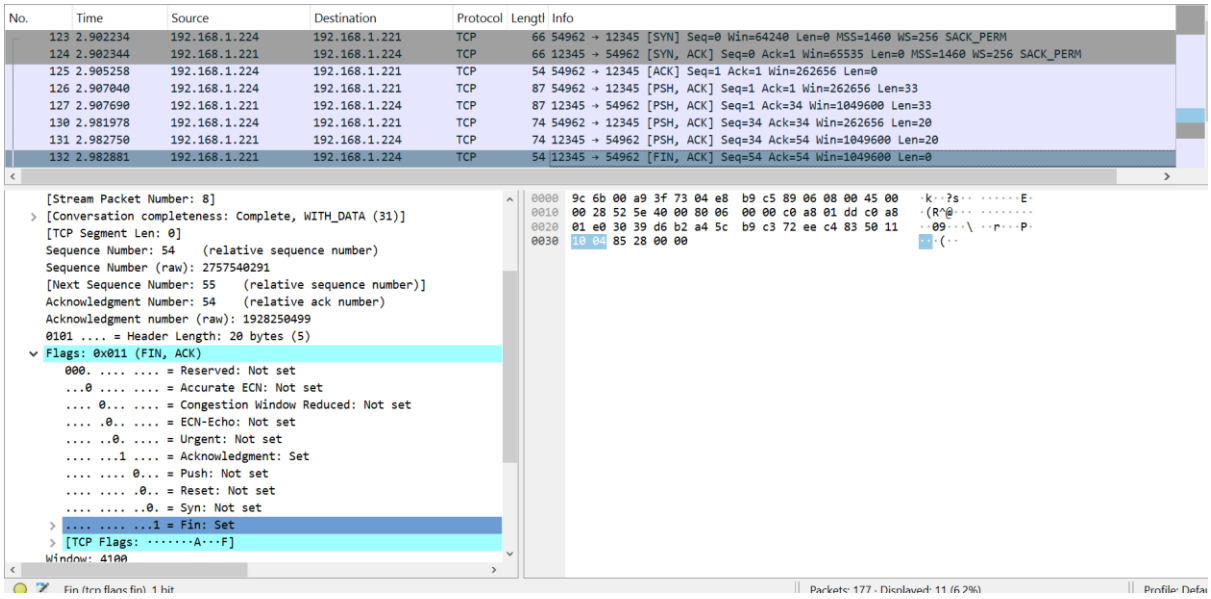


Seventh message- server to client (221->224), sent Acknowledgment number (raw): 1928250499 (1928250445+1+33+20) and sent the message back

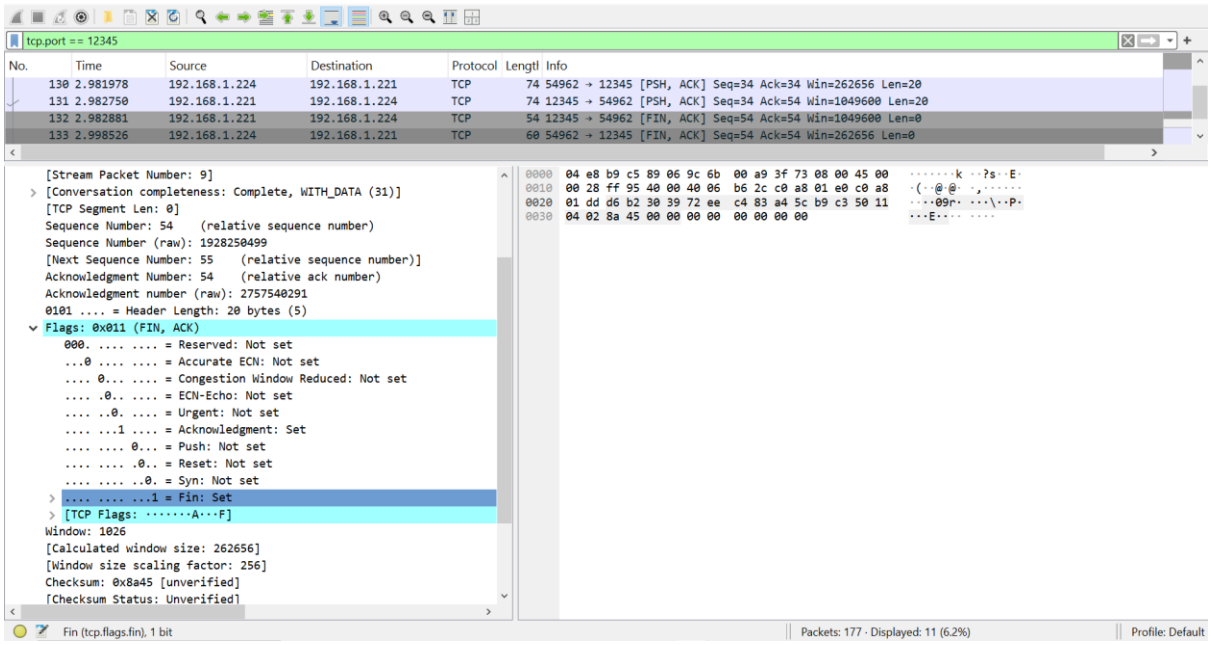


Eight message- server to client (221->224), asking to close connection Fin flag up part 1 of 4 part handshake, this happened before client could give ack for the server msg

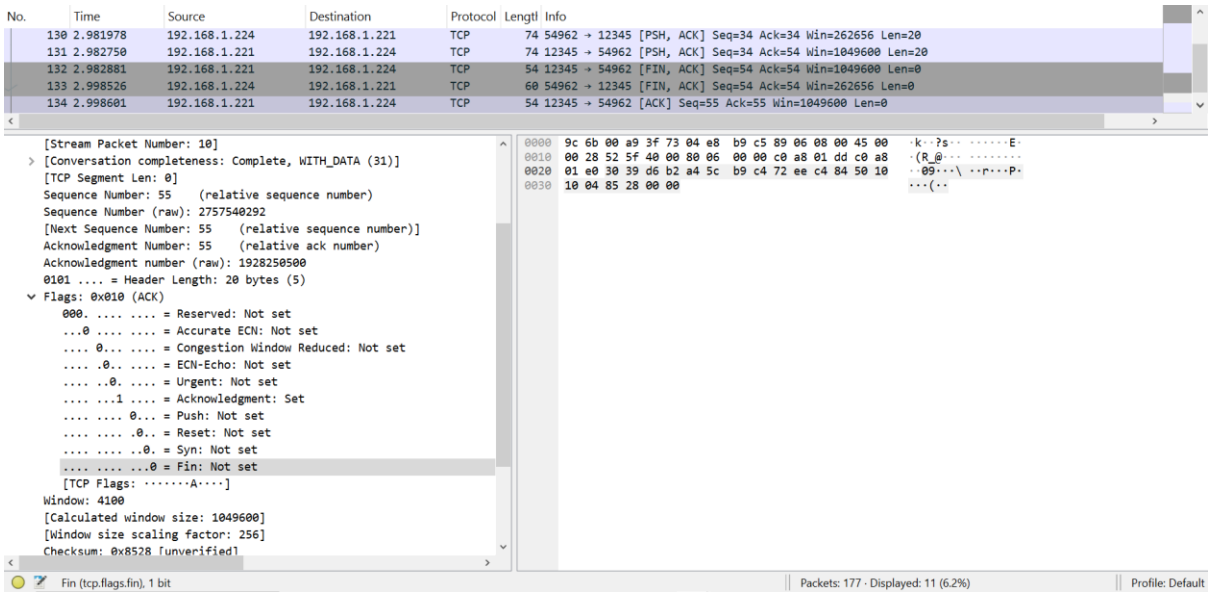
Sequence Number (raw): 2757540291 – even though we didn't get ack till here because client didn't send ack yet



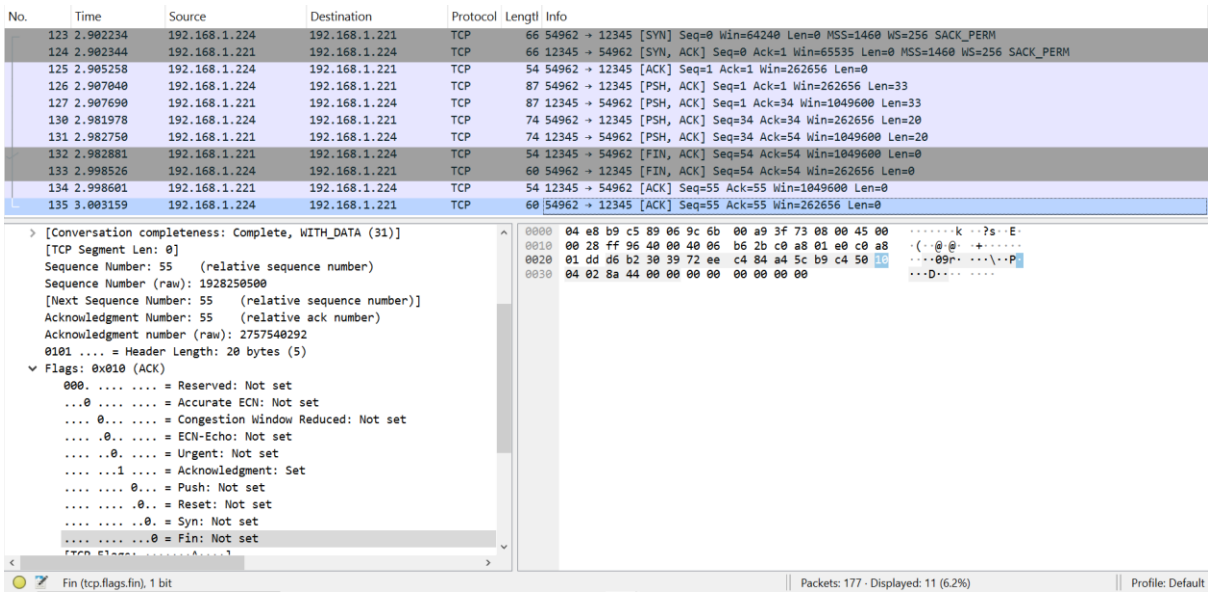
Ninth message- client to server (224->221), part 3 of 4 part handshake client didn't receive fin yet but it started to close the connection itself with Acknowledgment number (raw): 2757540291 (meaning it got the last 20)



Tenth message- server to client (221->224), part 4 of 4 part handshake client received server ack to close connection so it can close the connection now



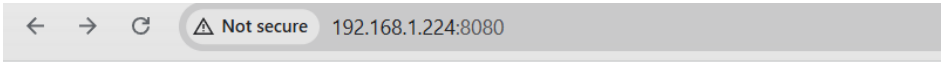
Final message- client to server (224->221),part 2 of 4 part handshake server received ack for fin and now can close connection



Part 2:

1. We will demonstrate the **keep alive** functionality using single_fin and double_fin files

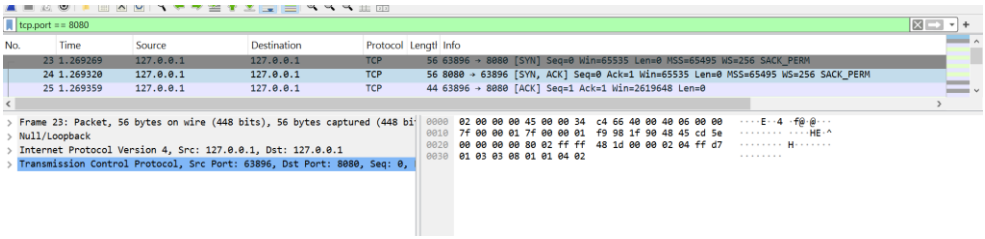
Single_fin- here I opened the default linkage of index.html and quickly refreshed as to demonstrate the keep alive (in practice chrome opens 2 ports for every request so I actually had to make python client to do this with the same port)



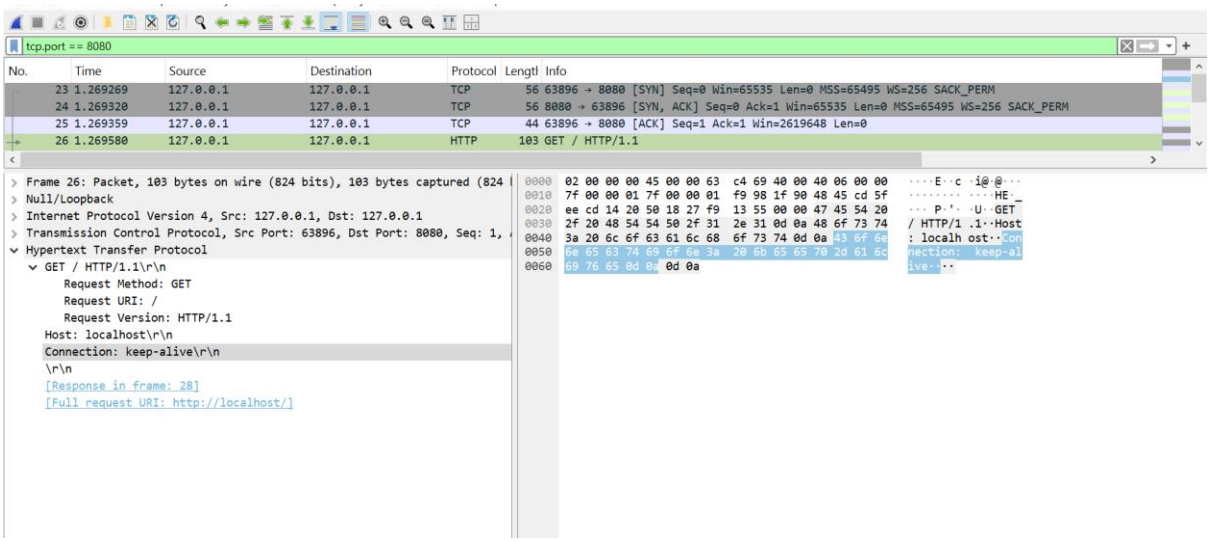
Hello World

lets analyze single_fin.pcapng:

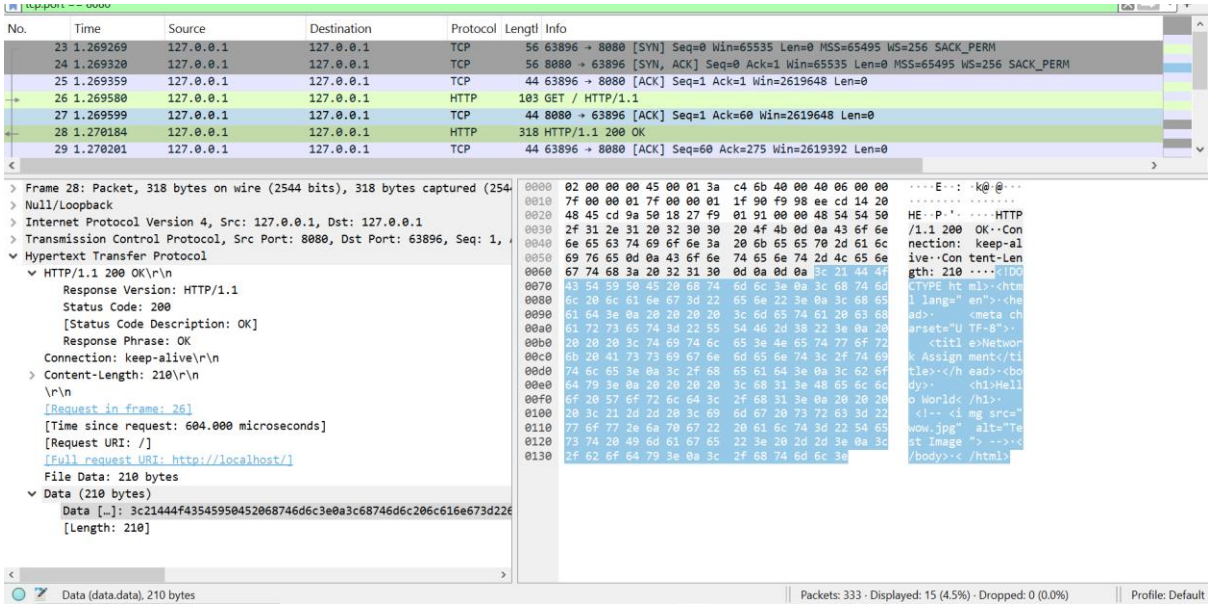
First 3 packets are syn handshake as explained in part 1



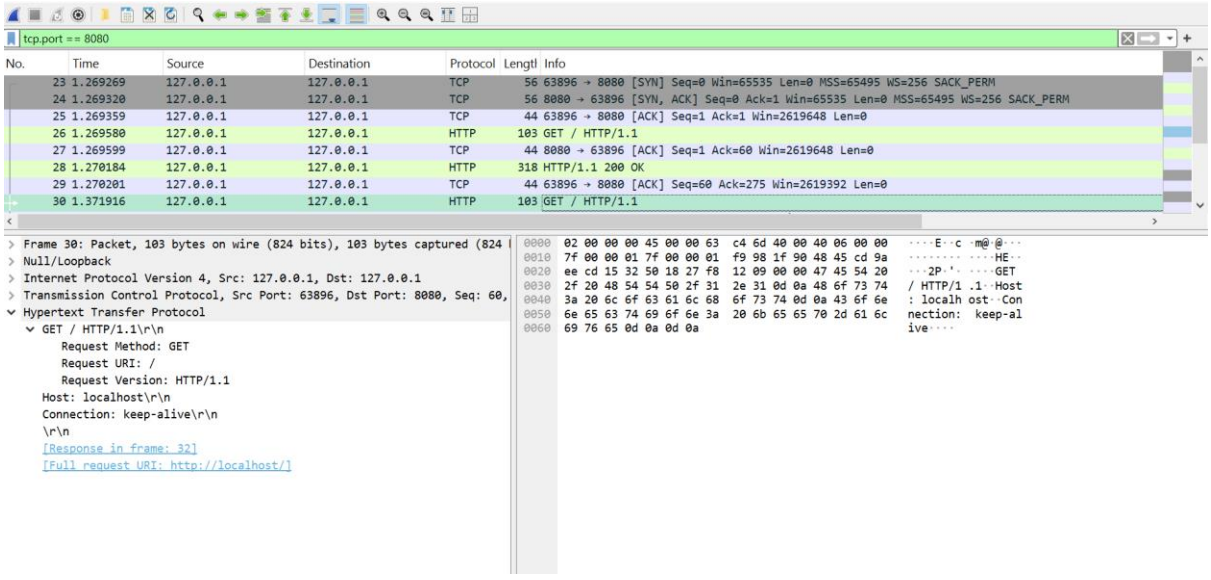
Next message is the get / HTTP/1.1 request for the page- do notice that the connection type in the http protocol is keep-alive



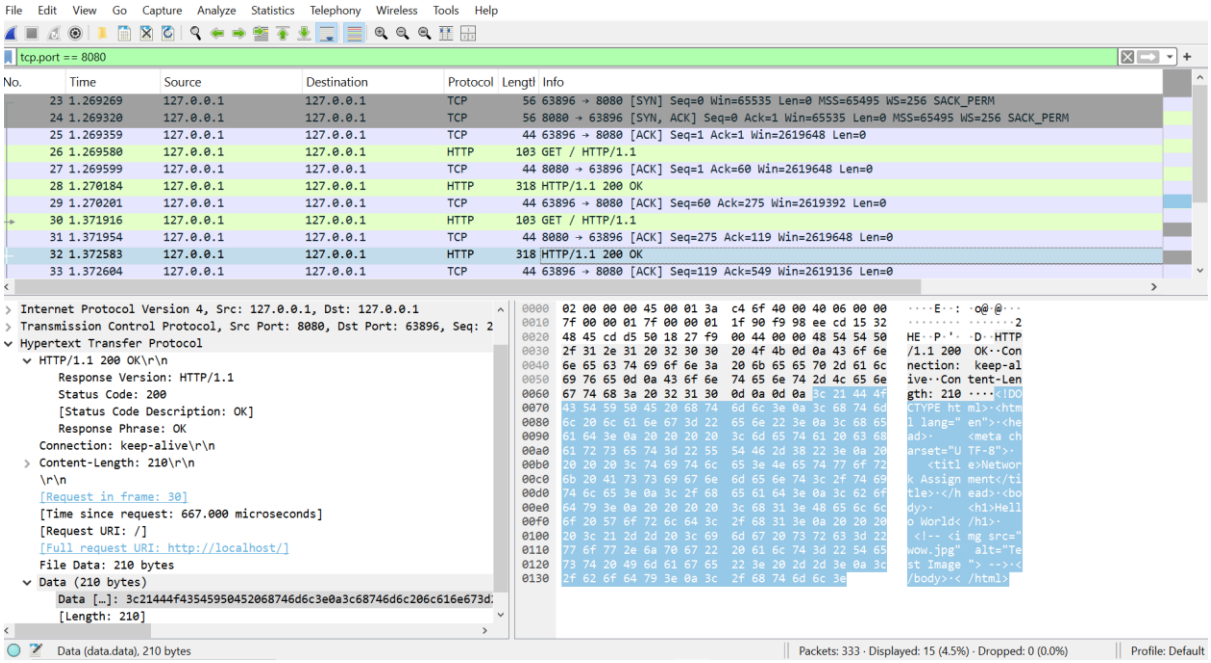
The next 3 messages go as follows: 1: server send ack on the get / request 2:server sent HTTP/1.1 message with code 200 meaning success and sent the data which contains the header hello word for example , notice connection type is still keep-alive 3: client sent ack on all the bits (274 presumably)



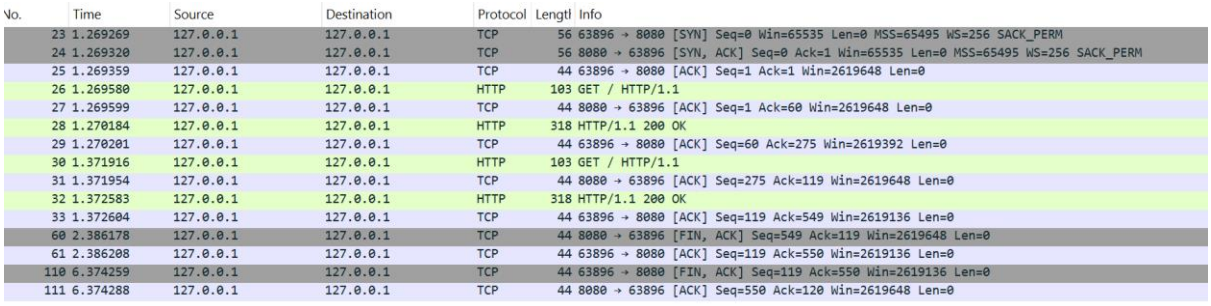
The second message: immediately after we receive and sent ack (this at 1.27) at time 1.37 we sent another http get/ request



As the connection is kept alive we don't need to connect to the server again (no syn and we didn't event get fin yet) we get immediately the ack and response as it was in the first message:

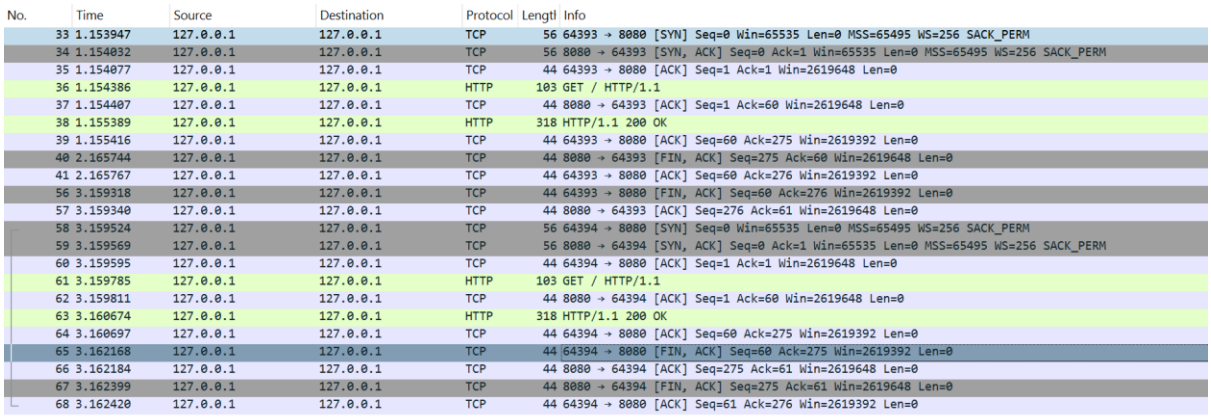


The last ack was sent at 1.37 and after a second at 2.38 we received the first fin to sever the connection and do the 4 stage handshake

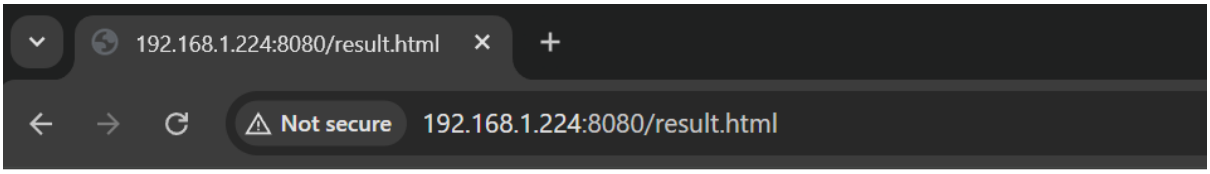


All this shows that the server sent fin only after waiting a while for as the client sent the connection type as keep-alive

Now in file **double_fin** you can see where we waited to send the second message it sent the fin before we could send the second message and so we got syn again and 2 disconnects meaning 4 fins



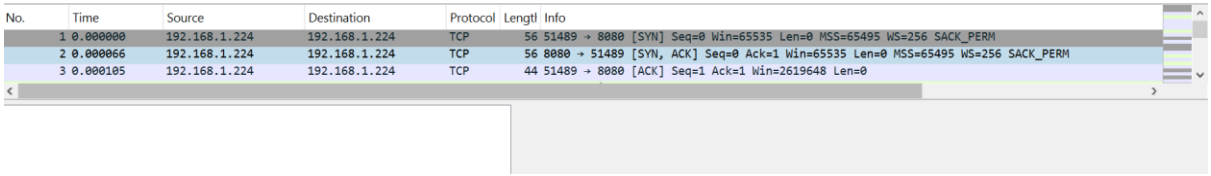
2. **Let analyze redirect.pcapng:**



Success!

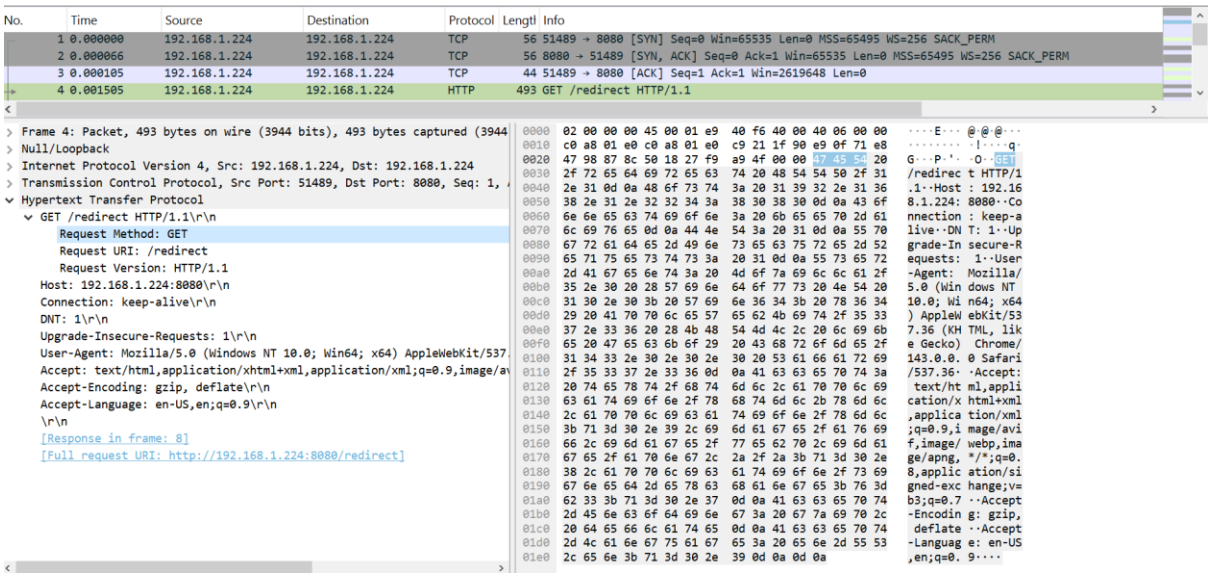
You have been redirected to result.html

We as usual get the 3 part handshake for connection:



No.	Time	Source	Destination	Protocol	Lengtl	Info
1	0.000000	192.168.1.224	192.168.1.224	TCP	56	51489 → 8080 [SYN] Seq=0 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM
2	0.000066	192.168.1.224	192.168.1.224	TCP	56	8080 → 51489 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM
3	0.000105	192.168.1.224	192.168.1.224	TCP	44	51489 → 8080 [ACK] Seq=1 Ack=1 Win=2619648 Len=0

Now we send a http message get/redirect

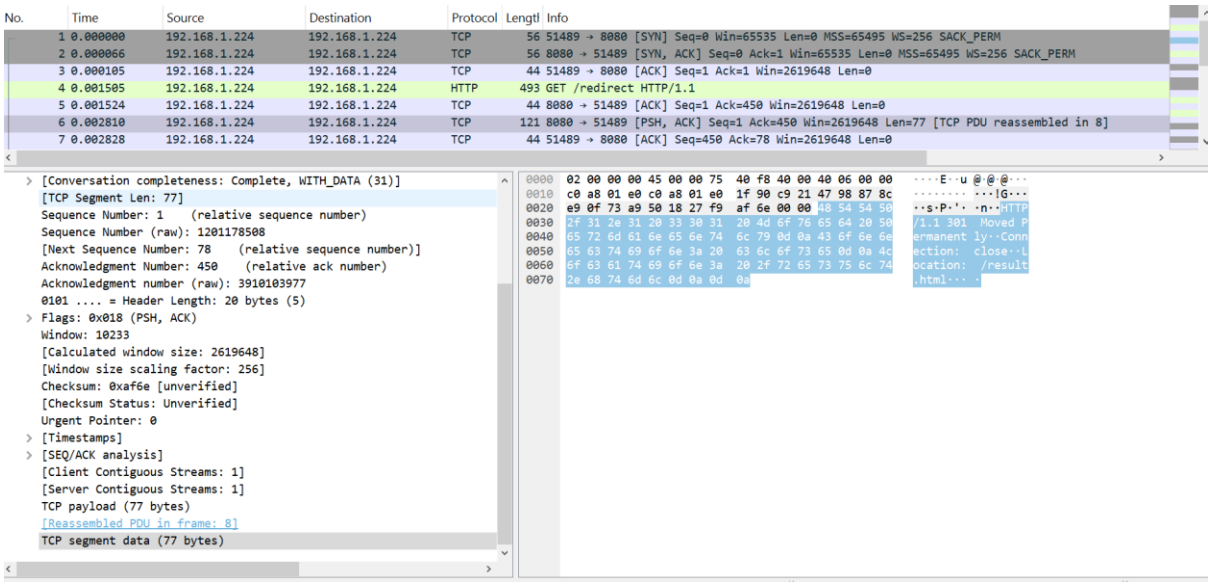


No.	Time	Source	Destination	Protocol	Lengtl	Info
1	0.000000	192.168.1.224	192.168.1.224	TCP	56	51489 → 8080 [SYN] Seq=0 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM
2	0.000066	192.168.1.224	192.168.1.224	TCP	56	8080 → 51489 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM
3	0.000105	192.168.1.224	192.168.1.224	TCP	44	51489 → 8080 [ACK] Seq=1 Ack=1 Win=2619648 Len=0
4	0.001505	192.168.1.224	192.168.1.224	HTTP	493	GET /redirect HTTP/1.1

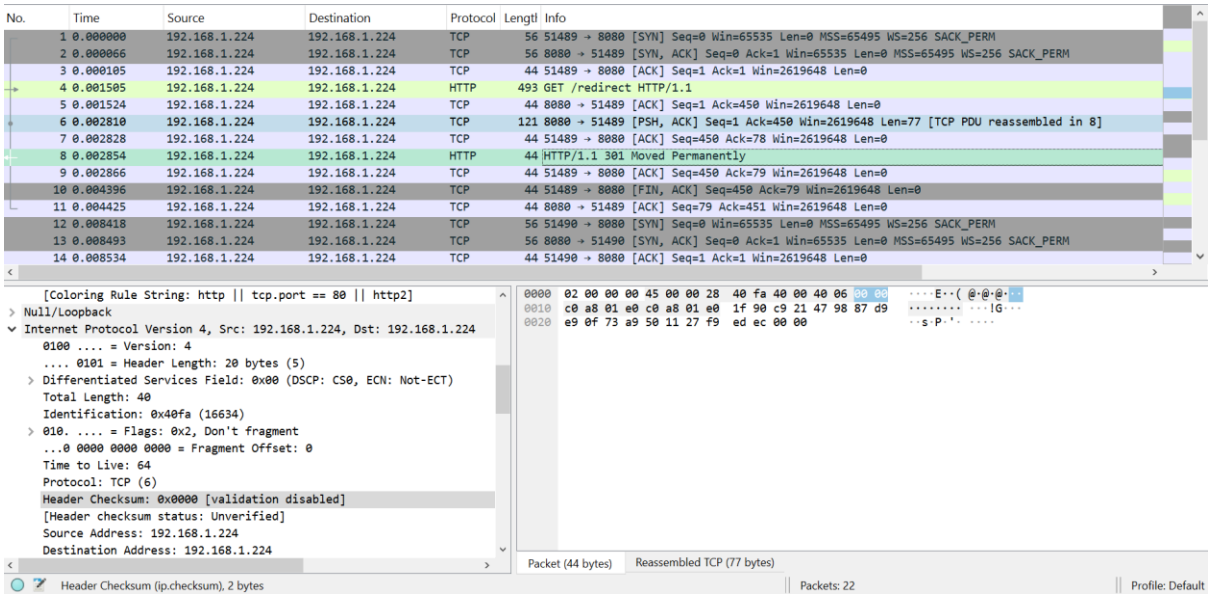
```
> Frame 4: Packet, 493 bytes on wire (3944 bits), 493 bytes captured (3944
> Null/Loopback
> Internet Protocol Version 4, Src: 192.168.1.224, Dst: 192.168.1.224
> Transmission Control Protocol, Src Port: 51489, Dst Port: 8080, Seq: 1,
v Hypertext Transfer Protocol
  v GET /redirect HTTP/1.1\r\n
      Request Method: GET
      Request URI: /redirect
      Request Version: HTTP/1.1
    Host: 192.168.1.224:8080\r\n
    Connection: keep-alive\r\n
    DNT: 1\r\n
    Upgrade-Insecure-Requests: 1\r\n
    User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.
    Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/av
    Accept-Encoding: gzip, deflate\r\n
    Accept-Language: en-US,en;q=0.9\r\n
    \r\n
    [Response in frame: 8]
    [Full request URI: http://192.168.1.224:8080/redirect]
```

```
0000  02 00 00 00 45 00 01 e9  40 f6 40 00 40 06 00 00   ····E··· @·@·@···
0010  c0 a8 01 e0 c0 a8 01 e0  c9 21 1f 90 e9 0f 71 e8   ········ ·!····q·
0020  47 98 87 8c 50 18 27 f9  a9 4f 00 00 47 45 54 20   G···P·'· ·O··GET
0030  2f 72 65 64 69 72 65 63  74 20 48 54 54 50 2f 31   /redirec t HTTP/1
0040  2e 31 0d 0a 48 6f 73 74  3a 20 31 39 32 2e 31 36   .1··Host : 192.16
0050  38 2e 31 2e 32 32 34 3a  38 30 38 30 0d 0a 43 6f   8.1.224: 8080··Co
0060  6e 6e 65 63 74 69 6f 6e  3a 20 6b 65 65 70 2d 61   nnection : keep-a
0070  6c 69 76 65 0d 0a 44 4e  54 3a 20 31 0d 0a 55 70   live··DN T: 1··Up
0080  67 72 61 64 65 2d 49 6e  73 65 63 75 72 65 2d 52   grade-In secure-R
0090  65 71 75 65 73 74 73 3a  20 31 0d 0a 55 73 65 72   equests:  1··User
00a0  2d 41 67 65 6e 74 3a 20  4d 6f 7a 69 6c 6c 61 2f   -Agent:  Mozilla/
00b0  35 2e 30 20 28 57 69 6e  64 6f 77 73 20 4e 54 20   5.0 (Win dows NT
00c0  31 30 2e 30 3b 20 57 69  6e 36 34 3b 20 78 36 34   10.0; Wi n64; x64
00d0  29 20 41 70 70 6c 65 57  65 62 4b 69 74 2f 35 33   ) AppleW ebKit/53
00e0  37 2e 33 36 20 28 4b 48  54 4d 4c 2c 20 6c 69 6b   7.36 (KH TML, lik
00f0  65 20 47 65 63 6b 6f 29  20 43 68 72 6f 6d 65 2f   e Gecko)  Chrome/
0100  31 34 33 2e 30 2e 30 2e  30 20 53 61 66 61 72 69   143.0.0. 0 Safari
0110  2f 35 33 37 2e 33 36 0d  0a 41 63 63 65 70 74 3a   /537.36· ·Accept:
0120  20 74 65 78 74 2f 68 74  6d 6c 2c 61 70 70 6c 69    text/ht ml,appli
0130  63 61 74 69 6f 6e 2f 78  68 74 6d 6c 2b 78 6d 6c   cation/x html+xml
0140  2c 61 70 70 6c 69 63 61  74 69 6f 6e 2f 78 6d 6c   ,applica tion/xml
0150  3b 71 3d 30 2e 39 2c 69  6d 61 67 65 2f 61 76 69   ;q=0.9,i mage/avi
0160  66 2c 69 6d 61 67 65 2f  77 65 62 70 2c 69 6d 61   f,image/ webp,ima
0170  67 65 2f 61 70 6e 67 2c  2a 2f 2a 3b 71 3d 30 2e   ge/apng, */*;q=0.
0180  38 2c 61 70 70 6c 69 63  61 74 69 6f 6e 2f 73 69   8,applic ation/si
0190  67 6e 65 64 2d 65 78 63  68 61 6e 67 65 3b 76 3d   gned-exc hange;v=
01a0  62 33 3b 71 3d 30 2e 37  0d 0a 41 63 63 65 70 74   b3;q=0.7 ··Accept
01b0  2d 45 6e 63 6f 64 69 6e  67 3a 20 67 7a 69 70 2c   -Encodin g: gzip,
01c0  20 64 65 66 6c 61 74 65  0d 0a 41 63 63 65 70 74    deflate ··Accept
01d0  2d 4c 61 6e 67 75 61 67  65 3a 20 65 6e 2d 55 53   -Languag e: en-US
01e0  2c 65 6e 3b 71 3d 30 2e  39 0d 0a 0d 0a            ,en;q=0. 9····
```

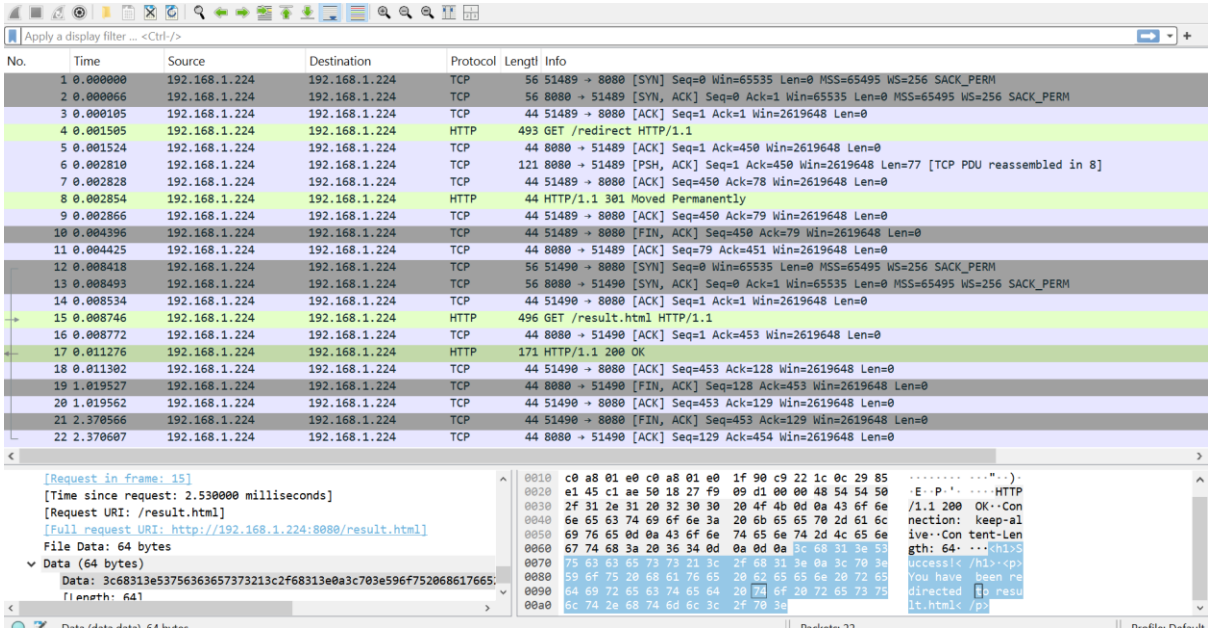
We receive an ack from the server and then the message to redirect us to /result and to close the connection, which we send ack as a response



Now we send a message that we moved permanently 301 close the connection from the client side (server side already knows to close the connection) then we open a new connection



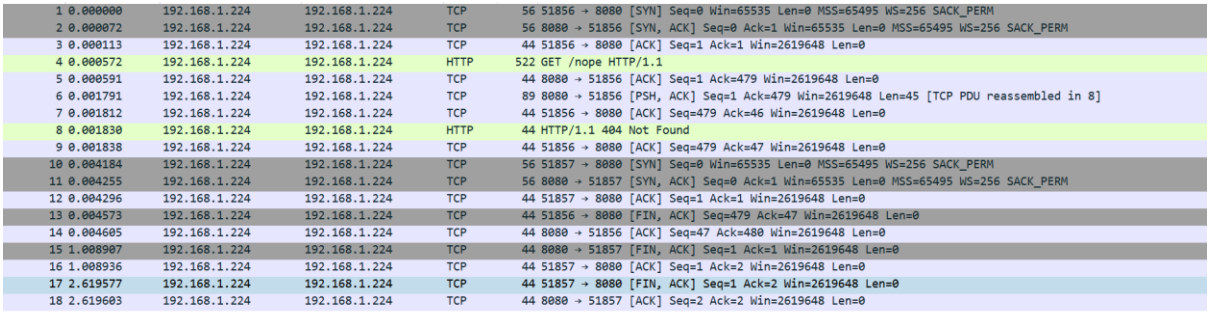
From the new connection we send a new http get for /result we receive it from the server and after a second we close the connection



No.	Time	Source	Destination	Protocol	Lengtl	Info
1	0.000000	192.168.1.224	192.168.1.224	TCP	56	51489 → 8080 [SYN] Seq=0 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM
2	0.000066	192.168.1.224	192.168.1.224	TCP	56	8080 → 51489 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM
3	0.000105	192.168.1.224	192.168.1.224	TCP	44	51489 → 8080 [ACK] Seq=1 Ack=1 Win=2619648 Len=0
4	0.001505	192.168.1.224	192.168.1.224	HTTP	493	GET /redirect HTTP/1.1
5	0.001524	192.168.1.224	192.168.1.224	TCP	44	8080 → 51489 [ACK] Seq=1 Ack=450 Win=2619648 Len=0
6	0.002810	192.168.1.224	192.168.1.224	TCP	121	8080 → 51489 [PSH, ACK] Seq=1 Ack=450 Win=2619648 Len=77 [TCP PDU reassembled in 8]
7	0.002828	192.168.1.224	192.168.1.224	TCP	44	51489 → 8080 [ACK] Seq=450 Ack=78 Win=2619648 Len=0
8	0.002854	192.168.1.224	192.168.1.224	HTTP	44	HTTP/1.1 301 Moved Permanently
9	0.002866	192.168.1.224	192.168.1.224	TCP	44	51489 → 8080 [ACK] Seq=450 Ack=79 Win=2619648 Len=0
10	0.004396	192.168.1.224	192.168.1.224	TCP	44	51489 → 8080 [FIN, ACK] Seq=450 Ack=79 Win=2619648 Len=0
11	0.004425	192.168.1.224	192.168.1.224	TCP	44	8080 → 51489 [ACK] Seq=79 Ack=451 Win=2619648 Len=0
12	0.008418	192.168.1.224	192.168.1.224	TCP	56	51490 → 8080 [SYN] Seq=0 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM
13	0.008493	192.168.1.224	192.168.1.224	TCP	56	8080 → 51490 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM
14	0.008534	192.168.1.224	192.168.1.224	TCP	44	51490 → 8080 [ACK] Seq=1 Ack=1 Win=2619648 Len=0
15	0.008746	192.168.1.224	192.168.1.224	HTTP	496	GET /result.html HTTP/1.1
16	0.008772	192.168.1.224	192.168.1.224	TCP	44	8080 → 51490 [ACK] Seq=1 Ack=453 Win=2619648 Len=0
17	0.011276	192.168.1.224	192.168.1.224	HTTP	171	HTTP/1.1 200 OK
18	0.011302	192.168.1.224	192.168.1.224	TCP	44	51490 → 8080 [ACK] Seq=453 Ack=128 Win=2619648 Len=0
19	1.019527	192.168.1.224	192.168.1.224	TCP	44	8080 → 51490 [FIN, ACK] Seq=128 Ack=453 Win=2619648 Len=0
20	1.019562	192.168.1.224	192.168.1.224	TCP	44	51490 → 8080 [ACK] Seq=453 Ack=129 Win=2619648 Len=0
21	2.370566	192.168.1.224	192.168.1.224	TCP	44	51490 → 8080 [FIN, ACK] Seq=453 Ack=129 Win=2619648 Len=0
22	2.370607	192.168.1.224	192.168.1.224	TCP	44	8080 → 51490 [ACK] Seq=129 Ack=454 Win=2619648 Len=0

This shows the 301 redirect

3. **Now we will anlyze notfound.pcapng where we executed get/nope which doesn't exist**
   We can see that we sent the get /nope and received a 404 http code meaning the page was not found, everything else is similar to previous analysis.



1	0.000000	192.168.1.224	192.168.1.224	TCP	56 51856 → 8080 [SYN] Seq=0 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM
2	0.000072	192.168.1.224	192.168.1.224	TCP	56 8080 → 51856 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM
3	0.000113	192.168.1.224	192.168.1.224	TCP	44 51856 → 8080 [ACK] Seq=1 Ack=1 Win=2619648 Len=0
4	0.000572	192.168.1.224	192.168.1.224	HTTP	522 GET /nope HTTP/1.1
5	0.000591	192.168.1.224	192.168.1.224	TCP	44 8080 → 51856 [ACK] Seq=1 Ack=479 Win=2619648 Len=0
6	0.001791	192.168.1.224	192.168.1.224	TCP	89 8080 → 51856 [PSH, ACK] Seq=1 Ack=479 Win=2619648 Len=45 [TCP PDU reassembled in 8]
7	0.001812	192.168.1.224	192.168.1.224	TCP	44 51856 → 8080 [ACK] Seq=479 Ack=46 Win=2619648 Len=0
8	0.001830	192.168.1.224	192.168.1.224	HTTP	44 HTTP/1.1 404 Not Found
9	0.001838	192.168.1.224	192.168.1.224	TCP	44 51856 → 8080 [ACK] Seq=479 Ack=47 Win=2619648 Len=0
10	0.004184	192.168.1.224	192.168.1.224	TCP	56 51857 → 8080 [SYN] Seq=0 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM
11	0.004255	192.168.1.224	192.168.1.224	TCP	56 8080 → 51857 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM
12	0.004296	192.168.1.224	192.168.1.224	TCP	44 51857 → 8080 [ACK] Seq=1 Ack=1 Win=2619648 Len=0
13	0.004573	192.168.1.224	192.168.1.224	TCP	44 51856 → 8080 [FIN, ACK] Seq=479 Ack=47 Win=2619648 Len=0
14	0.004605	192.168.1.224	192.168.1.224	TCP	44 8080 → 51856 [ACK] Seq=47 Ack=480 Win=2619648 Len=0
15	1.008907	192.168.1.224	192.168.1.224	TCP	44 8080 → 51857 [FIN, ACK] Seq=1 Ack=1 Win=2619648 Len=0
16	1.008936	192.168.1.224	192.168.1.224	TCP	44 51857 → 8080 [ACK] Seq=1 Ack=2 Win=2619648 Len=0
17	2.619577	192.168.1.224	192.168.1.224	TCP	44 51857 → 8080 [FIN, ACK] Seq=1 Ack=2 Win=2619648 Len=0
18	2.619603	192.168.1.224	192.168.1.224	TCP	44 8080 → 51857 [ACK] Seq=2 Ack=2 Win=2619648 Len=0