

Random Analysis of Wireshark packets:

First, applying the filter:

ip.dst == 192.168.1.10 and tcp.port == 80 and frame.len >= 40 and frame.len < 80

From (figure 1) packets no. 94 and 95 are part of the three-way TCP handshake process, which is used to establish a connection.

No.	Time	Source	Destination	Protocol	Length	Info
94	5.312063	39.156.69.79	192.168.1.10	TCP	76	80 → 59390 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0 MSS=1412 SACK_PERM=1 WS=32
95	5.312433	39.156.69.79	192.168.1.10	TCP	76	80 → 59388 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0 MSS=1412 SACK_PERM=1 WS=32
99	5.565066	39.156.69.79	192.168.1.10	TCP	62	80 → 59390 [ACK] Seq=1 Ack=419 Win=25856 Len=0
110	5.859567	39.156.69.79	192.168.1.10	TCP	62	80 → 59390 [ACK] Seq=387 Ack=775 Win=26880 Len=0
131	6.312074	104.193.88.123	192.168.1.10	TCP	76	80 → 49962 [SYN, ACK] Seq=0 Ack=1 Win=14480 Len=0 MSS=1412 SACK_PERM=1 TSval=2317630534 TSecr=3392175894 WS=
132	6.312451	104.193.88.123	192.168.1.10	TCP	68	80 → 49964 [SYN, ACK] Seq=0 Ack=1 Win=14600 Len=0 MSS=1412 SACK_PERM=1 WS=512
136	6.389575	104.193.88.123	192.168.1.10	TCP	68	80 → 49962 [ACK] Seq=1 Ack=451 Win=15616 Len=0 TSval=2317630612 TSecr=3392175966
150	6.573578	104.193.88.123	192.168.1.10	TCP	69	80 → 49962 [PSH, ACK] Seq=8193 Ack=451 Win=15616 Len=1 TSval=2317630785 TSecr=3392175966 [TCP segment of a ...
253	6.881080	104.193.88.123	192.168.1.10	HTTP	73	HTTP/1.1 200 OK (text/html)
276	7.589469	202.108.119.194	192.168.1.10	TCP	76	80 → 46288 [SYN, ACK] Seq=0 Ack=1 Win=14600 Len=0 MSS=1380 WS=128 SACK_PERM=1 TSval=4186285369 TSecr=190359...
279	7.802097	202.108.119.194	192.168.1.10	TCP	68	80 → 46288 [ACK] Seq=1 Ack=423 Win=6656 Len=0 TSval=4186285422 TSecr=190359857
319	8.020599	202.108.119.194	192.168.1.10	TCP	68	[TCP Dup ACK 279#1] 80 → 46288 [ACK] Seq=13681 Ack=423 Win=6656 Len=0 TSval=4186285475 TSecr=190360074

Figure 1

Now, as we have this information, we can use a Wireshark feature by going to:

Analyze> Conversation Filter> TCP.

When this filter is applied the whole conversation between the client and the webserver can be seen (figure 2) in the packets which is on the IP address 39.156.69.79.

No.	Time	Source	Destination	Protocol	Length	Info
93	5.065195	192.168.1.10	39.156.69.79	TCP	74	59390 → 80 [SYN] Seq=0 Win=65340 Len=0 MSS=1452 SACK_PERM=1 TSval=1464106120 TSecr=0 WS=128
94	5.312063	39.156.69.79	192.168.1.10	TCP	76	80 → 59390 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0 MSS=1412 SACK_PERM=1 WS=32
96	5.312572	192.168.1.10	39.156.69.79	TCP	54	59390 → 80 [ACK] Seq=1 Ack=1 Win=65408 Len=0
98	5.313444	192.168.1.10	39.156.69.79	HTTP	472	GET / HTTP/1.1
99	5.565066	39.156.69.79	192.168.1.10	TCP	62	80 → 59390 [ACK] Seq=1 Ack=419 Win=25856 Len=0
100	5.565438	39.156.69.79	192.168.1.10	TCP	361	80 → 59390 [PSH, ACK] Seq=1 Ack=419 Win=25856 Len=305 [TCP segment of a reassembled PDU]
101	5.565700	192.168.1.10	39.156.69.79	TCP	54	59390 → 80 [ACK] Seq=419 Ack=306 Win=65152 Len=0
102	5.565814	39.156.69.79	192.168.1.10	HTTP	137	HTTP/1.1 200 OK (text/html)
103	5.566071	192.168.1.10	39.156.69.79	TCP	54	59390 → 80 [ACK] Seq=419 Ack=387 Win=65152 Len=0

Figure 2

From the above screenshot:

First the client sends the SYN to the destination address 39.156.69.79.

In packet no. 94 the server responds with the SYN, ACK.

In third place the client sends an ACK and acknowledges that the connection is established.

In frame no. 98, the client is using the HTTP GET method to retrieve the webpage from the webserver.

By going to:

Analyze > Follow> TCP Stream.

The whole HTTP communication can be viewed, which reveals that the website that was accessed is baidu.com, in this TCP stream (figure 3).

```

GET / HTTP/1.1
Host: baidu.com
Connection: keep-alive
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/88.0.4324.182 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9

HTTP/1.1 200 OK
Date: Fri, 12 Mar 2021 17:58:26 GMT
Server: Apache
Last-Modified: Tue, 12 Jan 2010 13:48:00 GMT
ETag: "51-47cf7e6ee8400"
Accept-Ranges: bytes
Content-Length: 81
Cache-Control: max-age=86400
Expires: Sat, 13 Mar 2021 17:58:26 GMT
Connection: Keep-Alive
Content-Type: text/html

<html>
<meta http-equiv="refresh" content="0;url=http://www.baidu.com/">
</html>

```

Figure 3

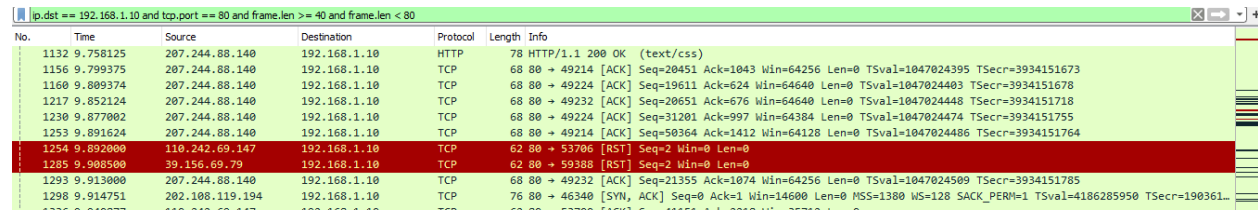
Continuing our analysis from the (figure 4), when the http request is made by the client to GET the favicon.ico of the website, the reply from the server is interrupted due to some reason which is shown by Wireshark in black. These packets are defective, means that the information is not properly transmitted. In packet no. 107 the client requests the packet again.

105	5.609698	192.168.1.10	39.156.69.79	HTTP	410 GET /favicon.ico HTTP/1.1
106	5.767067	39.156.69.79	192.168.1.10	TCP	361 [TCP Out-Of-Order] 80 → 59390 [PSH, ACK] Seq=1 Ack=419 Win=25856 Len=305
107	5.767453	192.168.1.10	39.156.69.79	TCP	66 [TCP Dup ACK 103#1] 59390 → 80 [ACK] Seq=775 Ack=387 Win=65152 Len=0 SLE=1 SRE=306
108	5.811941	39.156.69.79	192.168.1.10	TCP	137 [TCP Spurious Retransmission] 80 → 59390 [PSH, ACK] Seq=306 Ack=419 Win=25856 Len=81[Reassembly error, prot...
109	5.812326	192.168.1.10	39.156.69.79	TCP	66 [TCP Dup ACK 103#2] 59390 → 80 [ACK] Seq=775 Ack=387 Win=65152 Len=0 SLE=306 SRE=387
110	5.859567	39.156.69.79	192.168.1.10	TCP	62 80 → 59390 [ACK] Seq=387 Ack=775 Win=26880 Len=0
111	5.859941	39.156.69.79	192.168.1.10	TCP	294 80 → 59390 [PSH, ACK] Seq=387 Ack=775 Win=26880 Len=238 [TCP segment of a reassembled PDU]

Figure 4

In packet no.110 the server sends the ACK to confirm that the packets are now transmitted successfully and in the next packet, which is PSH, ACK it is ready to transmit the new information.

From (figure 5) the packets in red are the TCP termination packets. In packet no. 1254 the server (110.242.69.147) sends an RST to the client because the client has closed the opened tab in his browser. And so is the case in packet no. 1285 which is also closed by the client. By following the TCP Stream, it can be seen that the websites closed were baiddu.com and ss.bdimg.com.



The image shows a Wireshark packet capture window with a filter applied: `ip.dst == 192.168.1.10 and tcp.port == 80 and frame.len >= 40 and frame.len < 80`. The packet list shows several packets, with packets 1254 and 1285 highlighted in red, indicating they are RST (Reset) packets. The packet details for these red packets show `[RST] Seq=2 Win=0 Len=0`.

No.	Time	Source	Destination	Protocol	Length	Info
1132	9.758125	207.244.88.140	192.168.1.10	HTTP	78	HTTP/1.1 200 OK (text/css)
1156	9.799375	207.244.88.140	192.168.1.10	TCP	68	80 → 49214 [ACK] Seq=20451 Ack=1043 Win=64256 Len=0 TSval=1047024395 TSecr=3934151673
1160	9.809374	207.244.88.140	192.168.1.10	TCP	68	80 → 49224 [ACK] Seq=19611 Ack=624 Win=64640 Len=0 TSval=1047024403 TSecr=3934151678
1217	9.852124	207.244.88.140	192.168.1.10	TCP	68	80 → 49232 [ACK] Seq=20651 Ack=676 Win=64640 Len=0 TSval=1047024448 TSecr=3934151718
1230	9.877902	207.244.88.140	192.168.1.10	TCP	68	80 → 49224 [ACK] Seq=31201 Ack=997 Win=64384 Len=0 TSval=1047024474 TSecr=3934151755
1253	9.891624	207.244.88.140	192.168.1.10	TCP	68	80 → 49214 [ACK] Seq=50364 Ack=1412 Win=64128 Len=0 TSval=1047024486 TSecr=3934151764
1254	9.892000	110.242.69.147	192.168.1.10	TCP	62	80 → 53786 [RST] Seq=2 Win=0 Len=0
1285	9.908500	39.156.69.79	192.168.1.10	TCP	62	80 → 59388 [RST] Seq=2 Win=0 Len=0
1293	9.913000	207.244.88.140	192.168.1.10	TCP	68	80 → 49232 [ACK] Seq=21355 Ack=1074 Win=64256 Len=0 TSval=1047024509 TSecr=3934151785
1298	9.914751	202.108.119.194	192.168.1.10	TCP	76	80 → 46340 [SYN, ACK] Seq=0 Ack=1 Win=14600 Len=0 MSS=1380 WS=128 SACK_PERM=1 TSval=4186285950 TSecr=190361...

Figure 5

Conclusion:

From analysis of the packets, all the small sized packets that are transmitted by the client are either to request a service, to acknowledge that the data sent by the server has been received successfully or to request a packet that has not reached the destination properly. We can say that these small packets are used to make a connection and maintain it. And all the webserver used port 80 for communication.