Aiden Chang, Eli Arbogast

April 8th, 2021

CS231, Spring 2021

The Story of HTTP Authentication

For our filter in Wireshark, we put down “host cs231.jeffondich.com”. We first monitored the Wireshark responses for when the website asked for our username and password. The first thing we noticed was the 3-way TCP handshake. Interestingly, our device was starting the TCP handshake with 3 separate ports shown below.

1 0.000000000 10.0.2.15 45.79.89.123 TCP 74 34486 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=2583857701 TSecr=0 WS=128
2 0.004333368 10.0.2.15 45.79.89.123 TCP 74 34488 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=2583857705 TSecr=0 WS=128
3 0.257669108 10.0.2.15 45.79.89.123 TCP 74 34490 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=2583857958 TSecr=0 WS=128

This was then followed by the acknowledgement from the server(port80) to 2 of our ports.

4 0.332020712 45.79.89.123 10.0.2.15 TCP 60 80 → 34486 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460
5 0.332063664 10.0.2.15 45.79.89.123 TCP 54 34486 → 80 [ACK] Seq=1 Ack=1 Win=64240 Len=0
6 0.332184697 45.79.89.123 10.0.2.15 TCP 60 80 → 34488 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460
7 0.332193864 10.0.2.15 45.79.89.123 TCP 54 34488 → 80 [ACK] Seq=1 Ack=1 Win=64240 Len=0

Our machine then proceeded to request the content of /basicauth/ on line 8. Within line 8, we see “Transmission Control Protocol, Src Port: 34486, Dst Port: 80, Seq: 1, Ack: 1, Len: 341”
So the starting port that made this request was port 34486 from our device. Line 9 acknowledges the request made on line 8. On line 10, Jeff’s website acknowledges our third port, port 34490.

8 0.332381644 10.0.2.15 45.79.89.123 HTTP 395 GET /basicauth/ HTTP/1.1
9 0.332479534 45.79.89.123 10.0.2.15 TCP 60 80 → 34486 [ACK] Seq=1 Ack=342 Win=65535 Len=0

10 0.379062812 45.79.89.123 10.0.2.15 TCP 60 80 → 34490 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460
11 0.379097675 10.0.2.15 45.79.89.123 TCP 54 34490 → 80 [ACK] Seq=1 Ack=1 Win=64240 Len=0
After this we got an unauthorization response from the website.
12 0.381148626 45.79.89.123 10.0.2.15 HTTP 473 HTTP/1.1 401 Unauthorized (text/html)

This was also received by port 34486. The following lines after were just our device sending on TCP FIN packet from port 34488,34490 and both devices acknowledging it. Only port 34486 was kept connected.

22 10.583398361 10.0.2.15 45.79.89.123 TCP 54 [TCP Keep-Alive] 34486 → 80 [ACK] Seq=341 Ack=420 Win=63821 Len=0
23 10.583710897 45.79.89.123 10.0.2.15 TCP 60 [TCP Keep-Alive ACK] 80 → 34486 [ACK] Seq=420 Ack=342 Win=65535 Len=0

So at this point, we are very confused as to what the two other ports do.

Next up, we will actually log in with the given username and password and access one of the secret texts. The beginning was the same as above, except that there were two ports that opened instead of three. Similarly to above, all the other ports were closed by our device besides the first one after the line

9 0.093165903 45.79.89.123 10.0.2.15 HTTP 473 HTTP/1.1 401 Unauthorized (text/html)

Our assumption is that the other ports are backup ports to make requests just in case the first port fails. When we type in the username and password, we make the request of get /basicauth/ through this line

15 5.675430037 10.0.2.15 45.79.89.123 HTTP 438 GET /basicauth/ HTTP/1.1

But this time with a correct username and password. The server sends an acknowledgment then sends the fact that we are good to access the content and sends the content itself through this line.

17 5.722939831 45.79.89.123 10.0.2.15 HTTP 475 HTTP/1.1 200 OK (text/html)

Our device then proceeds to request favicon.io which is the icon on the left side of the url bar. Jeff's webpage then acknowledges our request for the favicon. Our port then proceeds to send a FIN request to Jeff's website. Jeff's website acknowledges this but does not send a FIN request back. We then get these lines.

23 5.860600308 45.79.89.123 10.0.2.15 HTTP 401 HTTP/1.1 404 Not Found (text/html)
24 5.860622239 10.0.2.15 45.79.89.123 TCP 54 34530 → 80 [RST] Seq=979 Win=0 Len=0
25 7.455452300 10.0.2.15 45.79.89.123 TCP 74 34534 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=2585757067 TSecr=0 WS=128
26 7.502192041 45.79.89.123 10.0.2.15 TCP 60 80 → 34534 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460
27 7.502231214 10.0.2.15 45.79.89.123 TCP 54 34534 → 80 [ACK] Seq=1 Ack=1 Win=64240 Len=0

We think that line 24 is a request resulting from line 23, with the 404 error Jeff's website sends back. We think the causation of this request is because we sent a FIN request but are still sending requests after. Our device then establishes a new connection with a new 3-way TCP handshake.

25 7.455452300 10.0.2.15 45.79.89.123 TCP 74 34534 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=2585757067 TSecr=0 WS=128
26 7.502192041 45.79.89.123 10.0.2.15 TCP 60 80 → 34534 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460
27 7.502231214 10.0.2.15 45.79.89.123 TCP 54 34534 → 80 [ACK] Seq=1 Ack=1 Win=64240 Len=0

To finalize, we open the secret text.

28 7.983970262 10.0.2.15 45.79.89.123 HTTP 499 GET /basicauth/amateurs.txt HTTP/1.1
29 7.984186944 45.79.89.123 10.0.2.15 TCP 60 80 → 34534 [ACK] Seq=1 Ack=446 Win=65535 Len=0
30 8.031960493 45.79.89.123 10.0.2.15 HTTP 375 HTTP/1.1 200 OK (text/plain)

31 8.031987706 10.0.2.15 45.79.89.123 TCP 54 34534 → 80 [ACK] Seq=446 Ack=322 Win=63919 Len=0

We make the request, Jeff's website receives it and acknowledges it, sends back the content, and my computer acknowledges that too.