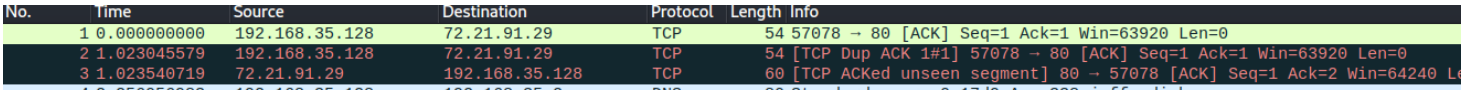
HTTP'S BASIC AUTHENTICATION

By: Sydney Nguyen

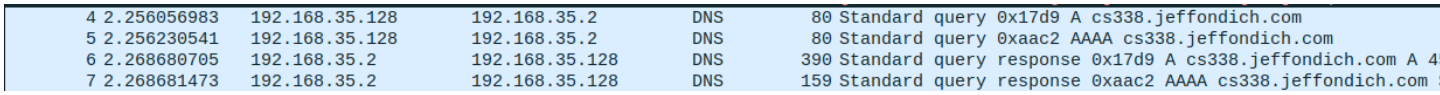
Background information (Aka before the authentication)

The image below shows the initial connection between my laptop and the firefox browser. Since the browser was already open. What is documented is the back half of a TCP handshake. As firefox is a major web browser there are many packages that Wireshark was unable to capture as to why there are unseen segments and duplicated ACKs because of the multiple SYN requests that were sent that are unseen in the image below.



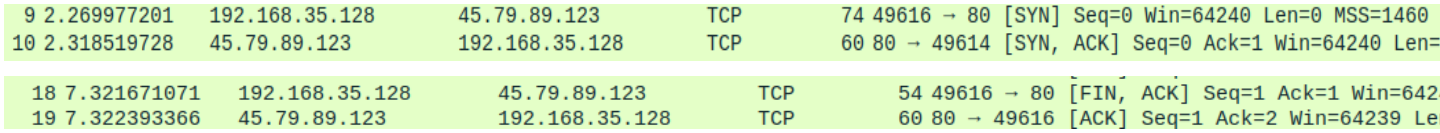
No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	192.168.35.128	72.21.91.29	TCP	54	57078 → 80 [ACK] Seq=1 Ack=1 Win=63920 Len=0
2	1.023045579	192.168.35.128	72.21.91.29	TCP	54	[TCP Dup ACK 1#1] 57078 → 80 [ACK] Seq=1 Ack=1 Win=63920 Len=0
3	1.023540719	72.21.91.29	192.168.35.128	TCP	60	[TCP ACKed unseen segment] 80 → 57078 [ACK] Seq=1 Ack=2 Win=64240 Le

After the initial connection is made with firefox we then enact several DNS, domain name systems, protocols. The frames highlighted in blue are the process of taking internet domain names and translating them into IP addresses that allows us to locate the website. Now that we have the IP address of the website we are able to access the link and direct our computer to the desired request.



4	2.256056983	192.168.35.128	192.168.35.2	DNS	80	Standard query 0x17d9 A cs338.jeffondich.com
5	2.256230541	192.168.35.128	192.168.35.2	DNS	80	Standard query 0xaac2 AAAA cs338.jeffondich.com
6	2.268680705	192.168.35.2	192.168.35.128	DNS	390	Standard query response 0x17d9 A cs338.jeffondich.com A 4
7	2.268681473	192.168.35.2	192.168.35.128	DNS	159	Standard query response 0xaac2 AAAA cs338.jeffondich.com

After we translate the domain names into IP addresses we are able to initiate the TCP handshake between the client and the server. The client's request is to access the website. As we see in the images the request being made to the server is from the IP address of my laptop to Jondich's server. We are also able to see the three steps of the TCP handshake the request the accept and the termination i.e syn(synchronize), ack(acknowledge), and fin(final). There are omitted lines/frames from the image as they are duplicate TCP handshakes made from our IP address to the desire website to attempt to gather other data simultaneously.



9	2.269977201	192.168.35.128	45.79.89.123	TCP	74	49616 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460
10	2.318519728	45.79.89.123	192.168.35.128	TCP	60	80 → 49614 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=
18	7.321671071	192.168.35.128	45.79.89.123	TCP	54	49616 → 80 [FIN, ACK] Seq=1 Ack=1 Win=642
19	7.322393366	45.79.89.123	192.168.35.128	TCP	60	80 → 49616 [ACK] Seq=1 Ack=2 Win=64239 Le

The Authentication

No.	Time	Source	Destination	Protocol	Length	Info
12	2.319333752	192.168.35.128	45.79.89.123	HTTP	395	GET /basicauth/ HTTP/1.1
16	2.448613106	45.79.89.123	192.168.35.128	HTTP	457	HTTP/1.1 401 Unauthorized (text/html)
22	10.860186374	192.168.35.128	45.79.89.123	HTTP	438	GET /basicauth/ HTTP/1.1
24	10.997375081	45.79.89.123	192.168.35.128	HTTP	458	HTTP/1.1 200 OK (text/html)
28	11.444468700	192.168.35.128	45.79.89.123	HTTP	355	GET /favicon.ico HTTP/1.1
32	11.600789547	45.79.89.123	192.168.35.128	HTTP	383	HTTP/1.1 404 Not Found (text/html)

The first initial get/basicauth is the initial response of the browser to the user trying to access the webpage. We receive in frame 16 a 401 Unauthorized because we are trying to access private, restricted data andwe need to authenticate our user. It is important to take notice that the first basicauth is sent without an authorization header.

What is an authorization header?

> The authorization header is only sent once the client attempts to access protected/restricted data without the proper credentials. In our case it is attempting to access the website. This header contains the attempted credentials user and password. It is to be noted that the authorization header in this context is a response to the receiving of credentials.

```
Upgrade-Insecure-Requests: 1\r\n
▸ Authorization: Basic Y3MzMzg6cGFzc3dvcmQ=\r\n
\r\n
```

The second, frame 22, basic authenticate is when the browser takes in our data. We can see in the HTTP tab the authentication header. The authorization header can look different for different types of authentication types. However for our basic authentication it needs credentials as user-id/password pairs encoded in Base64. It is important to realize that there is no encryption involved with the revival of our authentication only encoding so the computer is able to transfer the data. It should also be noted that the password is not confirmed or handled by the browser, it is received by the server and then checked in base 64.

```
Credentials: cs338:password
```

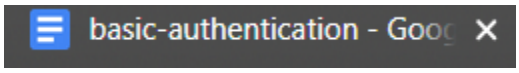
When the credentials pass the computer is sent from the website an ‘OK’ which grants access to the webpage. This access can also be seen from the connection the “Connection: Keep-alive” which is only enabled when the sequence received is exactly as expected. However it should be noted that when the incorrect credentials are imputed into the website the connection is still left on keep alive, but this is also because the website continually asks for the credentials until it is given the correct response.

```
    24 10.997375081  45.79.89.123        192.168.35.128      HTTP      458 HTTP/1.1 200 OK  (text/html)
```

Once we are granted access in fram 24 our computer is sent all of Jeff's secret files this can be seen in the line-based text data. Not that the client has been authroized we are now able to access all other files without the need for other credentials because the text files are not restricted/protected.

```
Line-based text data: text/html (9 lines)
    <html>\r\n
    <head><title>Index of /basicauth/</title></head>\r\n
    <body>\r\n
    <h1>Index of /basicauth/</h1><hr><pre><a href="../">../</a>\r\n
    <a href="amateurs.txt">amateurs.txt</a>                                    04-Apr-2022 14:10                75\r\n
    <a href="armed-guards.txt">armed-guards.txt</a>                                04-Apr-2022 14:10               161\r\n
    <a href="dancing.txt">dancing.txt</a>                                      04-Apr-2022 14:10               227\r\n
    </pre><hr></body>\r\n
    </html>\r\n
```

Next I would like to analyze the final HTTP protocols my computer received. The request for the favicon icon is the request for the images that appear when you open a browser, pictured below is an example of a favicon icon. We are then sent a 404 not found because Jeff's webpage does not have such icon associated.




```
28 11.444468700  192.168.35.128      45.79.89.123        HTTP      355 GET /favicon.ico HTTP/1.1
32 11.600789547  45.79.89.123        192.168.35.128      HTTP      383 HTTP/1.1 404 Not Found  (text/html)
```

It is important to not that we are not using the same port for the requests of the other text files because of a reset in other words the ending of our beginning connection with the original webpage. So when we request the other text pages it is a full new TCP handshake.

```
    33 11.600908051  192.168.35.128      45.79.89.123        TCP        54 49614 → 80 [RST] Seq=1028 Win=0 Len=0
```

Sources

Jeff Ondich

https://datatracker.ietf.org/doc/html/rfc7617#section-4

https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Authorization

https://auth0.com/docs/get-started/identity-fundamentals/authentication-and-authorization

https://datatracker.ietf.org/doc/html/rfc7617#section-4