Nicholas Vlahos-Sten

Jeff Ondich

Computer Security

23 September 2025

Basic Authorization Story: a play in two acts

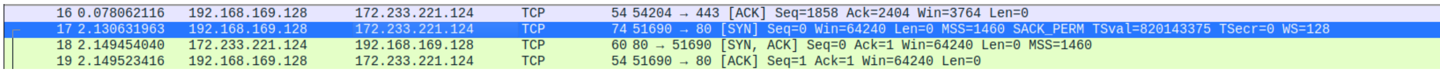
DRAMATIS PERSONAE:

BROWSER, a browser on an individual virtual computer. Seeks to gain access to the website hosted on the server.

SERVER, the hoster of Jeffondich.com. Seeks to grant access to Jeffondich.com, but only to those it deems authorized.

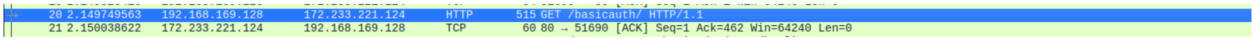
Act 1: TCP handshake

(Ignore everything in grey, that was because I used bootleg chrome on Kali and forgot to turn on the “don’t warn me before entering a non-secure site” feature, so all this grey stuff is before I clicked “yes I want to proceed” at 2 seconds).



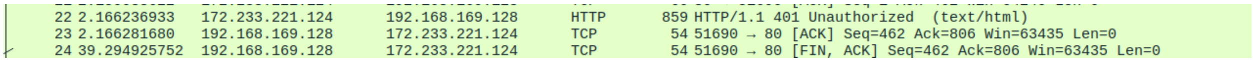
```
16 0.078062116   192.168.169.128   172.233.221.124   TCP   54 54204 → 443 [ACK] Seq=1858 Ack=2404 Win=3764 Len=0
17 2.130631963   192.168.169.128   172.233.221.124   TCP   74 51690 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=820143375 TSecr=0 WS=128
18 2.149454040   172.233.221.124   192.168.169.128   TCP   60 80 → 51690 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460
19 2.149523416   192.168.169.128   172.233.221.124   TCP   54 51690 → 80 [ACK] Seq=1 Ack=1 Win=64240 Len=0
```

The browser, using port 51690, starts up a conversation with the server, using port 80. They engage in the TCP handshake–the browser asks “am I talking to jeffondich.com” the server says “yes” and the browser says “okay cool.”

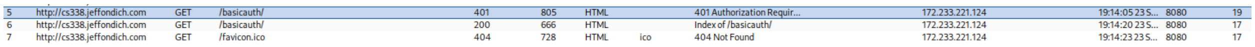


```
20 2.149749563   192.168.169.128   172.233.221.124   HTTP   515 GET /basicauth/ HTTP/1.1
21 2.150038622   172.233.221.124   192.168.169.128   TCP     60 80 → 51690 [ACK] Seq=1 Ack=462 Win=64240 Len=0
```

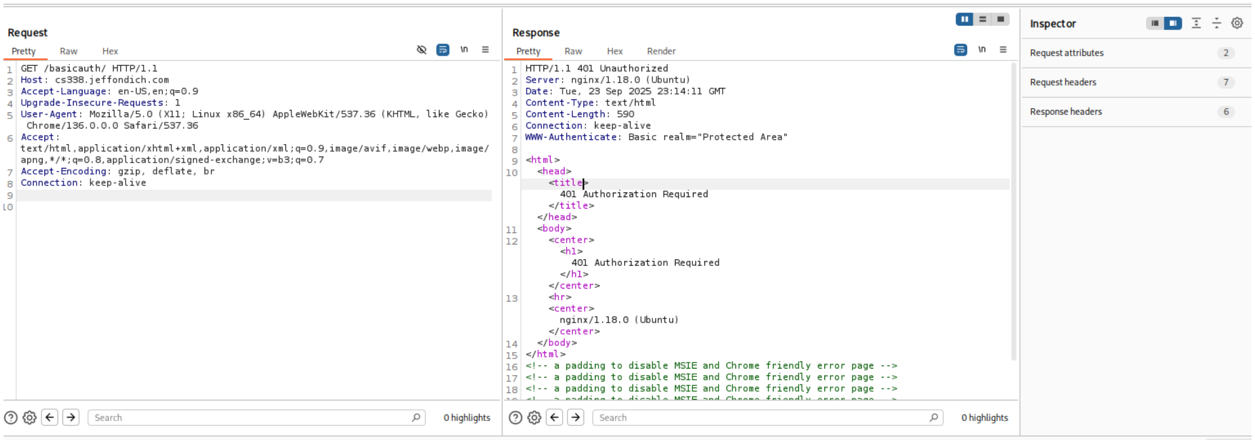
The browser then sends an HTTP GET request asking “hey can I have the info for jeffondich.com/basicauth?” And the server acknowledges that it got the message…



```
22 2.166236933   172.233.221.124    192.168.169.128    HTTP      859 HTTP/1.1 401 Unauthorized  (text/html)
23 2.166281680   192.168.169.128    172.233.221.124    TCP        54 51690 → 80 [ACK] Seq=462 Ack=806 Win=63435 Len=0
24 39.294925752  192.168.169.128    172.233.221.124    TCP        54 51690 → 80 [FIN, ACK] Seq=462 Ack=806 Win=63435 Len=0
```



5	http://cs338.jeffondich.com	GET	/basicauth/	401	805	HTML		401 Authorization Requir...	172.233.221.124	19:14:05 23 S...	8080	19
6	http://cs338.jeffondich.com	GET	/basicauth/	200	666	HTML		Index of /basicauth/	172.233.221.124	19:14:20 23 S...	8080	17
7	http://cs338.jeffondich.com	GET	/favicon.ico	404	728	HTML	ico	404 Not Found	172.233.221.124	19:14:23 23 S...	8080	17



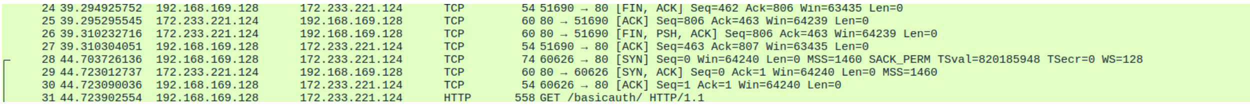
```
GET /basicauth/ HTTP/1.1
Host: cs338.jeffondich.com
Accept-Language: en-US,en;q=0.9
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko)
 Chrome/136.0.0.0 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/
apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Accept-Encoding: gzip, deflate, br
Connection: keep-alive
```

```
HTTP/1.1 401 Unauthorized
Server: nginx/1.18.0 (Ubuntu)
Date: Tue, 23 Sep 2025 23:14:11 GMT
Content-Type: text/html
Content-Length: 590
Connection: keep-alive
WWW-Authenticate: Basic realm="Protected Area"

<html>
  <head>
    <title>
      401 Authorization Required
    </title>
  </head>
  <body>
    <center>
      <h1>
        401 Authorization Required
      </h1>
    </center>
    <hr>
    <center>
      nginx/1.18.0 (Ubuntu)
    </center>
  </body>
</html>
<!-- a padding to disable MSIE and Chrome friendly error page -->
<!-- a padding to disable MSIE and Chrome friendly error page -->
<!-- a padding to disable MSIE and Chrome friendly error page -->
```

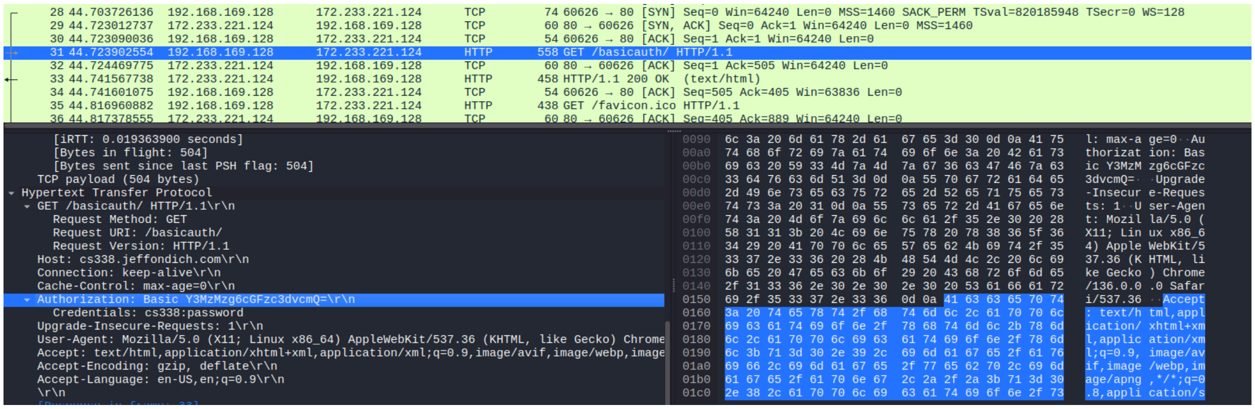
But it doesn't give back the website because wait! There is a problem. It requires authorization, so instead the server sends the browser a "401 Unauthorized" code and informs it that it wants to keep the connection alive. According to this blog post, the server knows the browser is not authorized to access the website because it checks if the browser's GET request contains an authorization header, and in this case it does not. The server also offers a field where the user can enter a username and password. In line 23 of wireshark, the browser acknowledges that it's not authorized, and then from the time jump from 2 to 39 seconds (the time it took for me to input the username and password) no other queries are sent, because the server is not giving anything else without authorization but is not quitting the conversation because it wants to give the browser the chance to authorize.

Act Two: authorization



```
24 39.294925752  192.168.169.128   172.233.221.124   TCP    54 51690 → 80 [FIN, ACK] Seq=462 Ack=806 Win=63435 Len=0
25 39.295295545  172.233.221.124   192.168.169.128   TCP    60 80 → 51690 [ACK] Seq=806 Ack=463 Win=64239 Len=0
26 39.310232716  172.233.221.124   192.168.169.128   TCP    60 80 → 51690 [FIN, PSH, ACK] Seq=806 Ack=463 Win=64239 Len=0
27 39.310304051  192.168.169.128   172.233.221.124   TCP    54 51690 → 80 [ACK] Seq=463 Ack=807 Win=63435 Len=0
28 44.703726136  192.168.169.128   172.233.221.124   TCP    74 60626 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=820185948 TSecr=0 WS=128
29 44.723012737  172.233.221.124   192.168.169.128   TCP    60 80 → 60626 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460
30 44.723090036  192.168.169.128   172.233.221.124   TCP    54 60626 → 80 [ACK] Seq=1 Ack=1 Win=64240 Len=0
31 44.723902554  192.168.169.128   172.233.221.124   HTTP  558 GET /basicauth/ HTTP/1.1
```

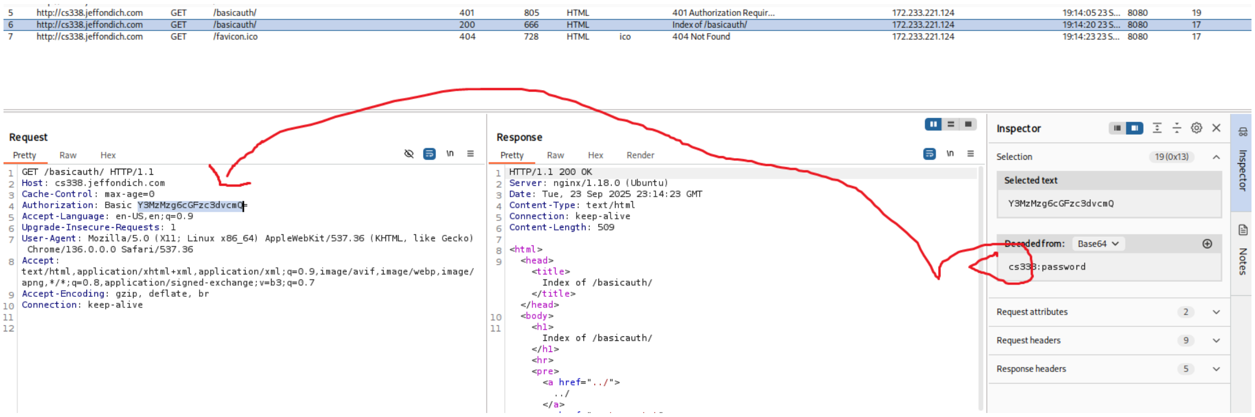
Once the user inputs the username and password, we have this strange sequence in which the browser's original port nopes out of the conversation, sending a FIN and the browser picks up the conversation on a new port, 60626, which starts by giving the server another TCP handshake.



```
28 44.703726136  192.168.169.128   172.233.221.124   TCP    74 60626 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=820185948 TSecr=0 WS=128
29 44.723012737  172.233.221.124   192.168.169.128   TCP    60 80 → 60626 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460
30 44.723090036  192.168.169.128   172.233.221.124   TCP    54 60626 → 80 [ACK] Seq=1 Ack=1 Win=64240 Len=0
31 44.723902554  192.168.169.128   172.233.221.124   HTTP  558 GET /basicauth/ HTTP/1.1
32 44.724469775  172.233.221.124   192.168.169.128   TCP    60 80 → 60626 [ACK] Seq=1 Ack=505 Win=64240 Len=0
33 44.741567738  172.233.221.124   192.168.169.128   HTTP  458 HTTP/1.1 200 OK  (text/html)
34 44.741601075  192.168.169.128   172.233.221.124   TCP    54 60626 → 80 [ACK] Seq=505 Ack=405 Win=63836 Len=0
35 44.816960882  192.168.169.128   172.233.221.124   HTTP  438 GET /favicon.ico HTTP/1.1
36 44.817378555  172.233.221.124   192.168.169.128   TCP    60 80 → 60626 [ACK] Seq=405 Ack=889 Win=64240 Len=0
```

```
      [iRTT: 0.019363900 seconds]
      [Bytes in flight: 504]
      [Bytes sent since last PSH flag: 504]
    TCP payload (504 bytes)
▾ Hypertext Transfer Protocol
  ▾ GET /basicauth/ HTTP/1.1\r\n
      Request Method: GET
      Request URI: /basicauth/
      Request Version: HTTP/1.1
    Host: cs338.jeffondich.com\r\n
    Connection: keep-alive\r\n
    Cache-Control: max-age=0\r\n
  ▾ Authorization: Basic Y3MzMzg6cGFzc3dvcmQ=\r\n
      Credentials: cs338:password
    Upgrade-Insecure-Requests: 1\r\n
    User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome
    Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image
    Accept-Encoding: gzip, deflate\r\n
    Accept-Language: en-US,en;q=0.9\r\n
    \r\n
```

```
0090  6c 3a 20 6d 61 78 2d 61  67 65 3d 30 0d 0a 41 75   l: max-a ge=0··Au
00a0  74 68 6f 72 69 7a 61 74  69 6f 6e 3a 20 42 61 73   thorizat ion: Bas
00b0  69 63 20 59 33 4d 7a 4d  7a 67 36 63 47 46 7a 63   ic Y3MzM zg6cGFzc
00c0  33 64 76 63 6d 51 3d 0d  0a 55 70 67 72 61 64 65   3dvcmQ=· ·Upgrade
00d0  2d 49 6e 73 65 63 75 72  65 2d 52 65 71 75 65 73   -Insecur e-Reques
00e0  74 73 3a 20 31 0d 0a 55  73 65 72 2d 41 67 65 6e   ts: 1··U ser-Agen
00f0  74 3a 20 4d 6f 7a 69 6c  6c 61 2f 35 2e 30 20 28   t: Mozil la/5.0 (
0100  58 31 31 3b 20 4c 69 6e  75 78 20 78 38 36 5f 36   X11; Lin ux x86_6
0110  34 29 20 41 70 70 6c 65  57 65 62 4b 69 74 2f 35   4) Apple WebKit/5
0120  33 37 2e 33 36 20 28 4b  48 54 4d 4c 2c 20 6c 69   37.36 (K HTML, li
0130  6b 65 20 47 65 63 6b 6f  29 20 43 68 72 6f 6d 65   ke Gecko ) Chrome
0140  2f 31 33 36 2e 30 2e 30  2e 30 20 53 61 66 61 72   /136.0.0 .0 Safar
0150  69 2f 35 33 37 2e 33 36  0d 0a 41 63 63 65 70 74   i/537.36 ··Accept
0160  3a 20 74 65 78 74 2f 68  74 6d 6c 2c 61 70 70 6c   : text/h tml,appl
0170  69 63 61 74 69 6f 6e 2f  78 68 74 6d 6c 2b 78 6d   ication/ xhtml+xm
0180  6c 2c 61 70 70 6c 69 63  61 74 69 6f 6e 2f 78 6d   l,applic ation/xm
0190  6c 3b 71 3d 30 2e 39 2c  69 6d 61 67 65 2f 61 76   l;q=0.9, image/av
01a0  69 66 2c 69 6d 61 67 65  2f 77 65 62 70 2c 69 6d   if,image /webp,im
01b0  61 67 65 2f 61 70 6e 67  2c 2a 2f 2a 3b 71 3d 30   age/apng ,*/*;q=0
01c0  2e 38 2c 61 70 70 6c 69  63 61 74 69 6f 6e 2f 73   .8,appli cation/s
```



```
5  http://cs338.jeffondich.com  GET  /basicauth/   401  805  HTML       401 Authorization Requir...  172.233.221.124  19:14:05 23 S...  8080  19
6  http://cs338.jeffondich.com  GET  /basicauth/   200  666  HTML       Index of /basicauth/         172.233.221.124  19:14:20 23 S...  8080  17
7  http://cs338.jeffondich.com  GET  /favicon.ico  404  728  HTML  ico  404 Not Found                172.233.221.124  19:14:23 23 S...  8080  17
```

Request

```
GET /basicauth/ HTTP/1.1
Host: cs338.jeffondich.com
Cache-Control: max-age=0
Authorization: Basic Y3MzMzg6cGFzc3dvcmQ=
Accept-Language: en-US,en;q=0.9
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko)
 Chrome/136.0.0.0 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=0.7
Accept-Encoding: gzip, deflate, br
Connection: keep-alive
```

Response

```
HTTP/1.1 200 OK
Server: nginx/1.18.0 (Ubuntu)
Date: Tue, 23 Sep 2025 23:14:23 GMT
Content-Type: text/html
Connection: keep-alive
Content-Length: 509

<html>
  <head>
    <title>
      Index of /basicauth/
    </title>
  </head>
  <body>
    <h1>
      Index of /basicauth/
    </h1>
    <hr>
    <pre>
      <a href="../">
        ../
      </a>
```

Inspector

Selection 19 (0x13)

Selected text: Y3MzMzg6cGFzc3dvcmQ

Decoded from: Base64

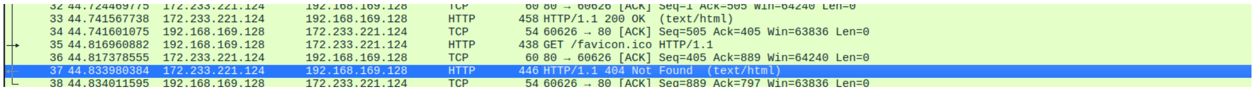
cs338:password

Request attributes 2

Request headers 9

Response headers 5

The browser then sends another GET request for the website, but now it adds an authorization header. The authorization header starts with basic to confirm it contains a username and password for basic authorization. Then it puts said username and password in the form *username*:*password* . It encodes the data using base64, but does not bother to encrypt it (it does not send an encryption key; I was surprised by this so I looked it up to make sure I wasn’t crazy and low and behold, there’s a big red warning box declaring this lack of encryption to be the case). The server receives this info, reverses the base64 encoding process to decode it, and checks to see that the username and password it received are in fact the correct username and password. If so, it grants authorization, and sends the website information the browser originally asked for, along with a 200 OK code.



```
32 44.724469775  172.233.221.124  192.168.169.128  TCP    60 80 → 60626 [ACK] Seq=1 Ack=505 Win=64240 Len=0
33 44.741567738  172.233.221.124  192.168.169.128  HTTP  458 HTTP/1.1 200 OK  (text/html)
34 44.741601075  192.168.169.128  172.233.221.124  TCP    54 60626 → 80 [ACK] Seq=505 Ack=405 Win=63836 Len=0
35 44.816960882  192.168.169.128  172.233.221.124  HTTP  438 GET /favicon.ico HTTP/1.1
36 44.817378555  172.233.221.124  192.168.169.128  TCP    60 80 → 60626 [ACK] Seq=405 Ack=889 Win=64240 Len=0
37 44.833980384  172.233.221.124  192.168.169.128  HTTP  446 HTTP/1.1 404 Not Found  (text/html)
38 44.834011595  192.168.169.128  172.233.221.124  TCP    54 60626 → 80 [ACK] Seq=889 Ack=797 Win=63836 Len=0
```

From here, we just do the browser’s HTTP “give me your favicon,” the server’s HTTP “no I don’t have that 404 error,” like a normal webpage. For all extent and purposes, once we authorize the user, this website behaves like a website that doesn’t require authorization in the first place. We chill here, the webpage is accessed, everyone is happy, and the characters won’t converse more until I close the page.

FIN