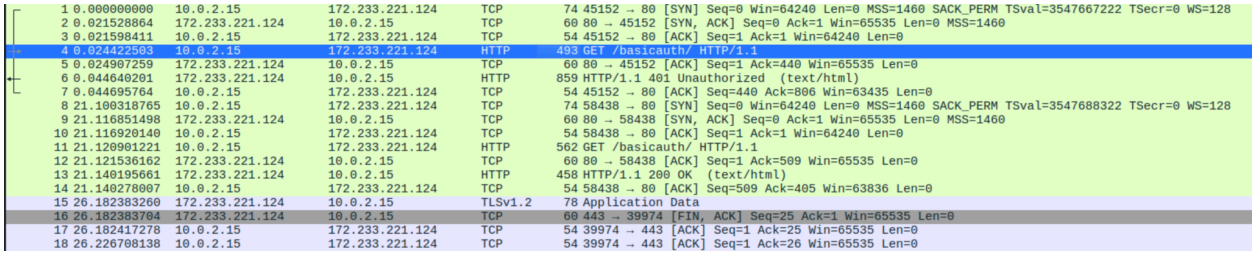
Devin Dennis - HTTP's Basic Authentication: A Story

There once was a cs student trying to break into Jeff's website. Little did he know that he had left the username and password on a sticky note on his desk. The student found it and logged in. The student attempted to determine how he had gained access to the site.

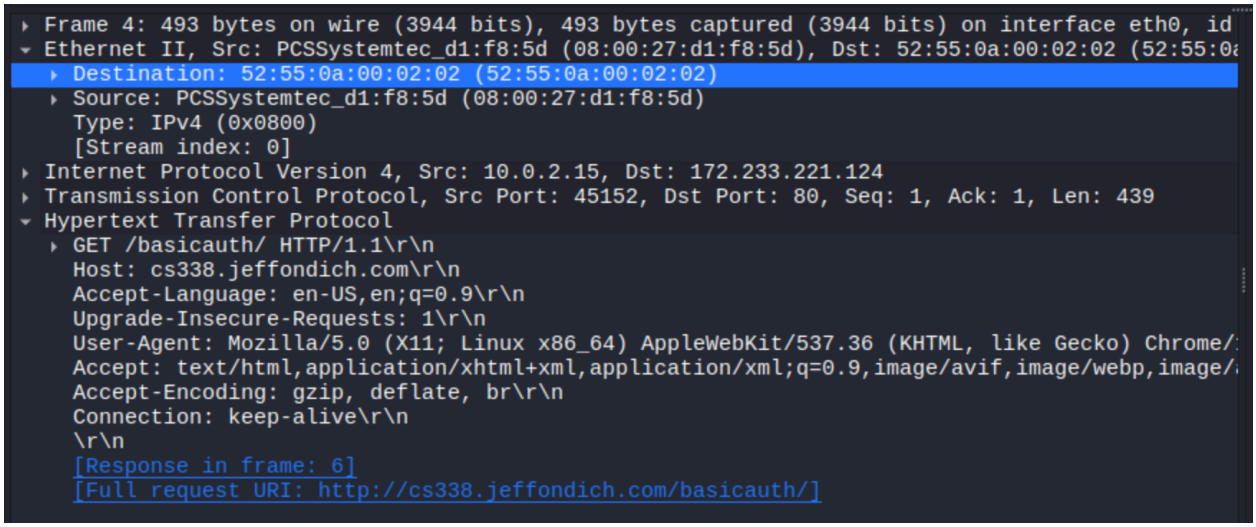
I recorded all the traffic from logging on Wireshark and Burp Suite to see the HTTP and TCP requests being sent.

First, I looked at the Wireshark input to see what was going on



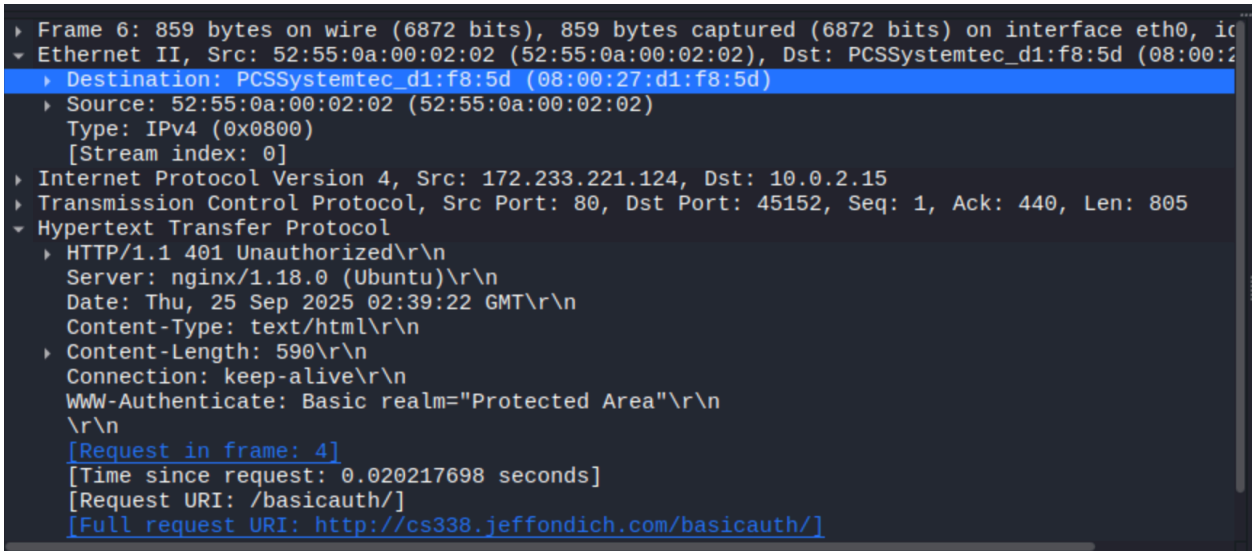
1	0.000000000	10.0.2.15	172.233.221.124	TCP	74 45152 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=3547667222 TSecr=0 WS=128
2	0.021528864	172.233.221.124	10.0.2.15	TCP	60 80 → 45152 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460
3	0.021598411	10.0.2.15	172.233.221.124	TCP	54 45152 → 80 [ACK] Seq=1 Ack=1 Win=64240 Len=0
4	0.024422503	10.0.2.15	172.233.221.124	HTTP	493 GET /basicauth/ HTTP/1.1
5	0.024907259	172.233.221.124	10.0.2.15	TCP	60 80 → 45152 [ACK] Seq=1 Ack=440 Win=65535 Len=0
6	0.044640201	172.233.221.124	10.0.2.15	HTTP	859 HTTP/1.1 401 Unauthorized (text/html)
7	0.044695764	10.0.2.15	172.233.221.124	TCP	54 45152 → 80 [ACK] Seq=440 Ack=806 Win=63435 Len=0
8	21.100318765	10.0.2.15	172.233.221.124	TCP	74 58438 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=3547688322 TSecr=0 WS=128
9	21.116851498	172.233.221.124	10.0.2.15	TCP	60 80 → 58438 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460
10	21.116920140	10.0.2.15	172.233.221.124	TCP	54 58438 → 80 [ACK] Seq=1 Ack=1 Win=64240 Len=0
11	21.120901221	10.0.2.15	172.233.221.124	HTTP	562 GET /basicauth/ HTTP/1.1
12	21.121536162	172.233.221.124	10.0.2.15	TCP	60 80 → 58438 [ACK] Seq=1 Ack=509 Win=65535 Len=0
13	21.140195661	172.233.221.124	10.0.2.15	HTTP	458 HTTP/1.1 200 OK (text/html)
14	21.140278007	10.0.2.15	172.233.221.124	TCP	54 58438 → 80 [ACK] Seq=509 Ack=405 Win=63836 Len=0
15	26.182383260	172.233.221.124	10.0.2.15	TLSv1.2	78 Application Data
16	26.182383704	172.233.221.124	10.0.2.15	TCP	60 443 → 39974 [FIN, ACK] Seq=25 Ack=1 Win=65535 Len=0
17	26.182417278	10.0.2.15	172.233.221.124	TCP	54 39974 → 443 [ACK] Seq=1 Ack=25 Win=65535 Len=0
18	26.226708138	10.0.2.15	172.233.221.124	TCP	54 39974 → 443 [ACK] Seq=1 Ack=26 Win=65535 Len=0

I see that an initial TCP connection was set up with the server. The browser then tried to make an HTTP GET request. Here's is the request:

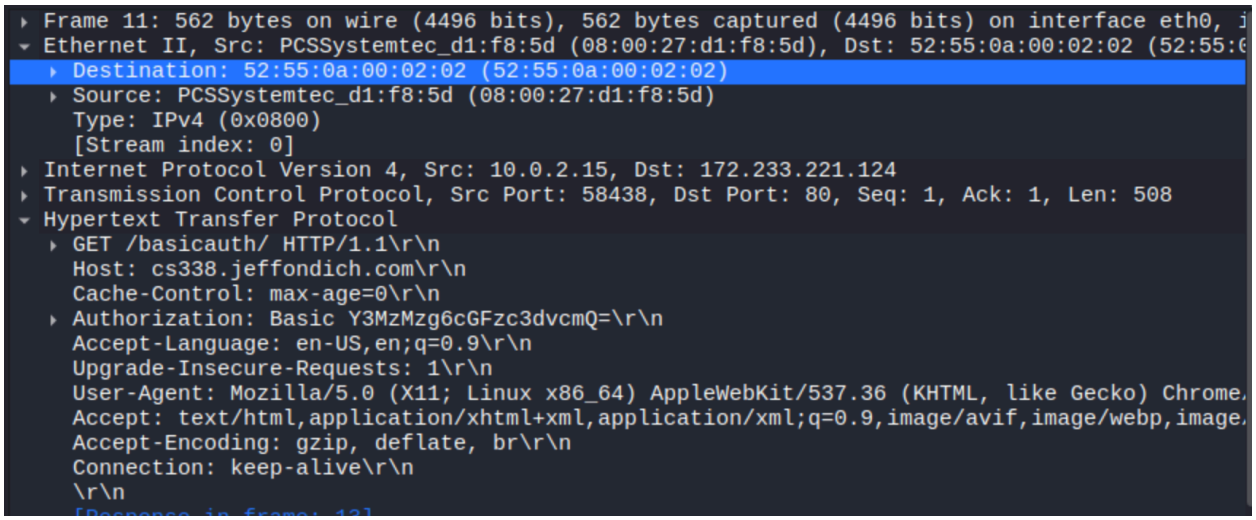


```
▸ Frame 4: 493 bytes on wire (3944 bits), 493 bytes captured (3944 bits) on interface eth0, id
▾ Ethernet II, Src: PCSSystemtec_d1:f8:5d (08:00:27:d1:f8:5d), Dst: 52:55:0a:00:02:02 (52:55:0a
  ▸ Destination: 52:55:0a:00:02:02 (52:55:0a:00:02:02)
  ▸ Source: PCSSystemtec_d1:f8:5d (08:00:27:d1:f8:5d)
    Type: IPv4 (0x0800)
    [Stream index: 0]
▸ Internet Protocol Version 4, Src: 10.0.2.15, Dst: 172.233.221.124
▸ Transmission Control Protocol, Src Port: 45152, Dst Port: 80, Seq: 1, Ack: 1, Len: 439
▾ Hypertext Transfer Protocol
  ▸ GET /basicauth/ HTTP/1.1\r\n
    Host: cs338.jeffondich.com\r\n
    Accept-Language: en-US,en;q=0.9\r\n
    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, br\r\n
    Connection: keep-alive\r\n
    \r\n
    [Response in frame: 6]
    [Full request URI: http://cs338.jeffondich.com/basicauth/]
```

The server sent back a response saying that you are not authorized to see this page with a HTTP/1.1 401 Unauthorized

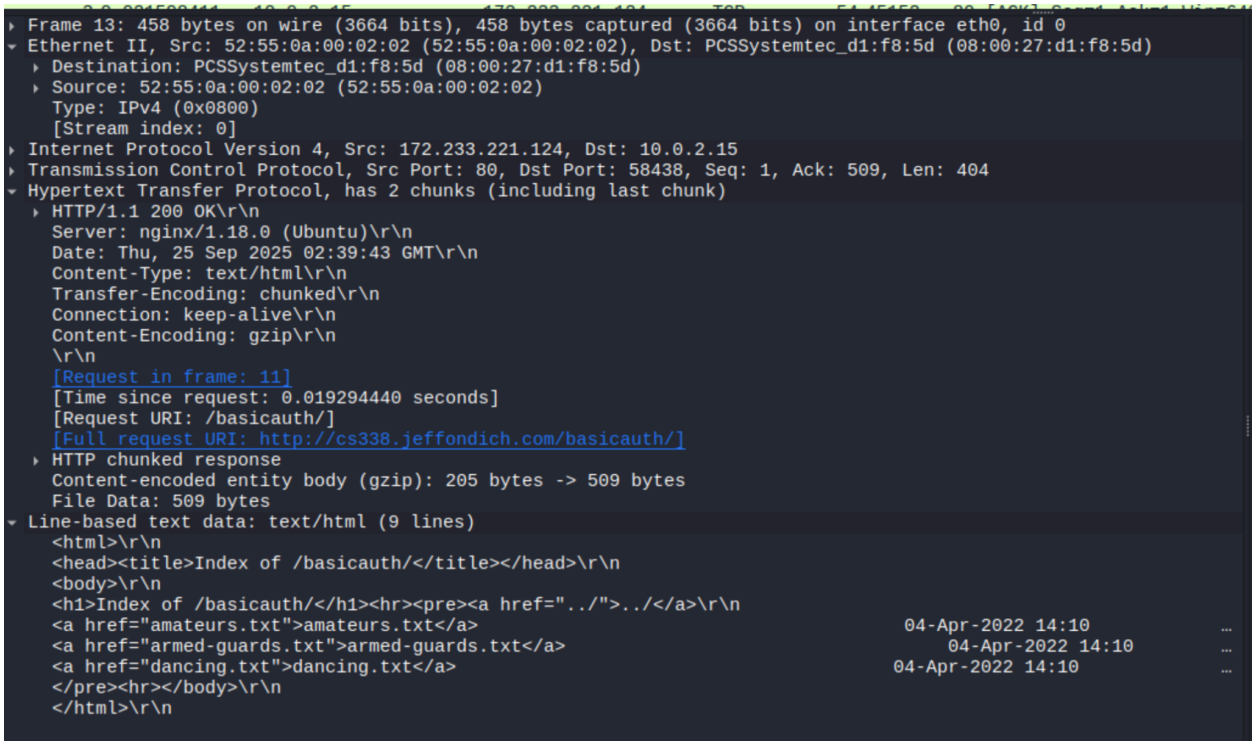


This is when the browser prompts the user to enter the username and password. When I did so, it sent this HTTP request:

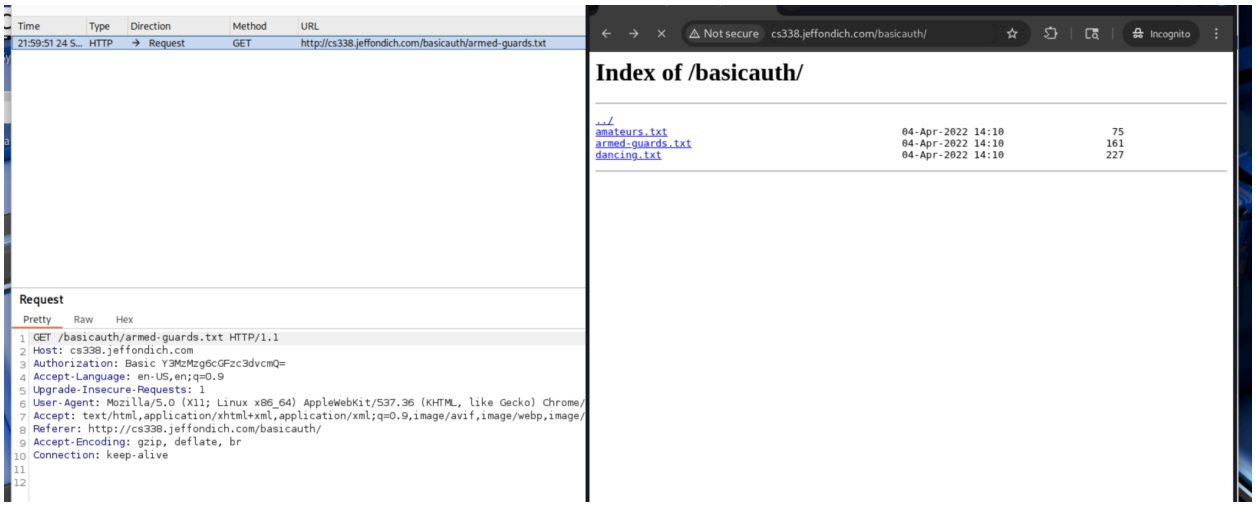


This time, the GET request had the Authorization header inside the payload. But the confusing part is why it's just a bunch of characters. I figured out that it was in base64 and decoded it:

The Authorization header follows this format for basic authentication: Authorization: Basic username:password. I realized that it's the password being sent from the browser in the HTTP request to log into the webpage. Then the server sent back an HTML page with a 200 OK code:



Every time I log into the site now, the browser automatically adds the authorization to the page since the browser has cached the credentials.



Now when I tried to look at one of Jeff's secret files, I didn't need to log in again.

Hehehee, I have all Jeff's secrets.