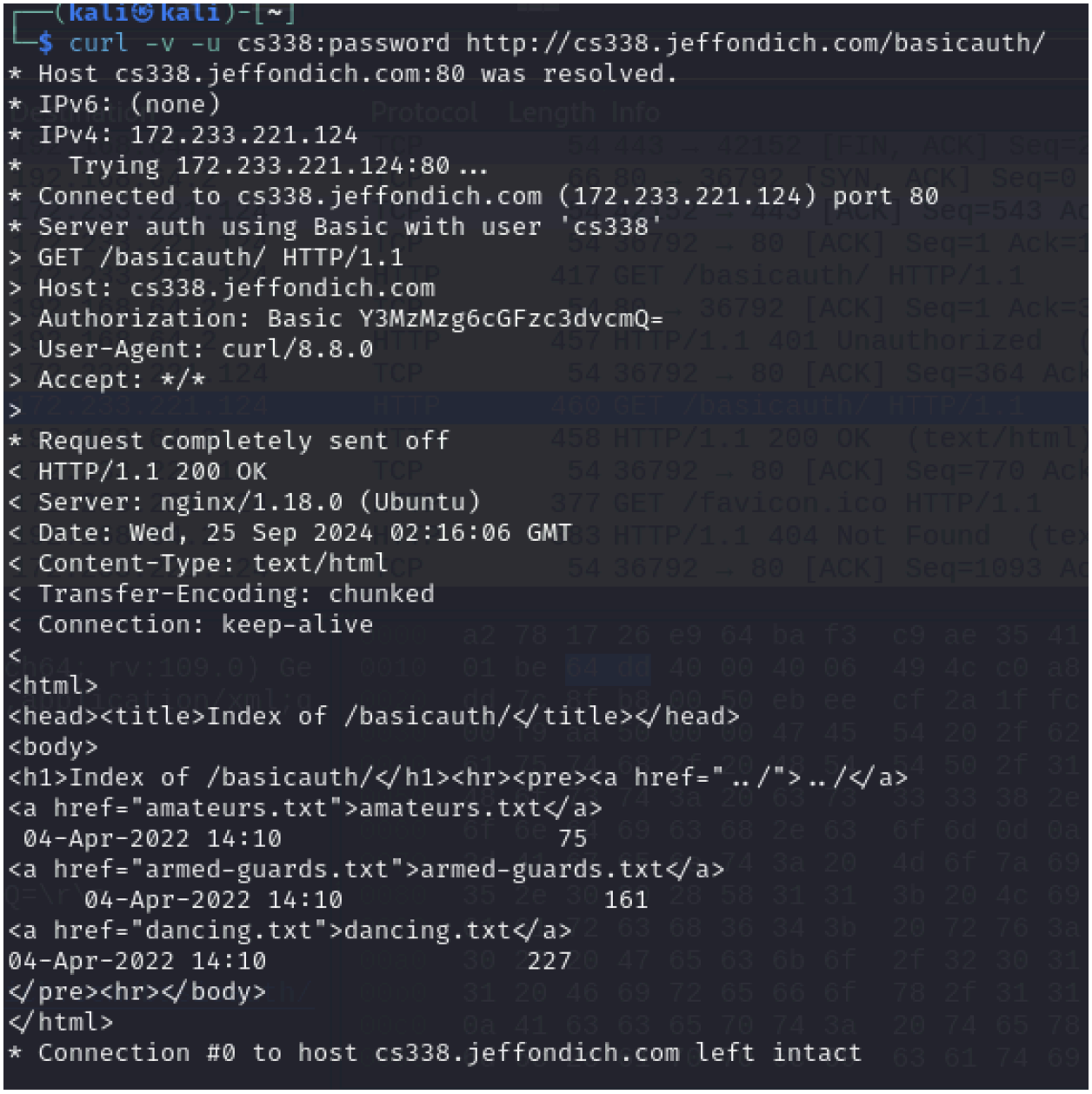
```
┌──(kali㉿kali)-[~]
└─$ curl -v http://cs338.jeffondich.com/basicauth/
* Host cs338.jeffondich.com:80 was resolved.
* IPv6: (none)
* IPv4: 172.233.221.124
*   Trying 172.233.221.124:80...
* Connected to cs338.jeffondich.com (172.233.221.124) port 80
> GET /basicauth/ HTTP/1.1
> Host: cs338.jeffondich.com
> User-Agent: curl/8.8.0
> Accept: */*
>
* Request completely sent off
< HTTP/1.1 401 Unauthorized
< Server: nginx/1.18.0 (Ubuntu)
< Date: Wed, 25 Sep 2024 02:25:07 GMT
< Content-Type: text/html
< Content-Length: 188
< 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>
* Connection #0 to host cs338.jeffondich.com left intact
```

You should provide as much detail as you can figure out about the interactions between the browser and cs338.jeffondich.com's nginx server.

From the curl output, I can see that it pulled the IP address of the page (172.233.221.124) and connected to port 80. It also pulled the host of the page, which is cs338.jeffondich.com. It then tried to get the webpage with the GET /basicauth/ HTTP header. Due to me not adding the username or password in, I get the next line of 401 unauthorized and the line that says “basic realm = Protected area”. It only loads the html information from the 401 page that says Authorization required, and I know that 401 is the error code for unauthorized access.

I then tried again, updating my curl command to have the username and password. (Source:

https://stackoverflow.com/questions/2594880/using-curl-with-a-username-and-password
)



This time, it shows the authorization: Basic … which I explain later is the base64 encoding of my input. It also loads the html text of the webpage after the browser is authenticated.

What queries are sent from the browser, and what responses does it receive?

Using wireshark, I can see that first, the TCP request is sent by the client, which begins the TCP handshake, that includes [SYN]s (synchronization packets), [ACK]s (acknowledgements), and [FIN] (ending that particular handshake,which the server responds to to complete the initial handshake.

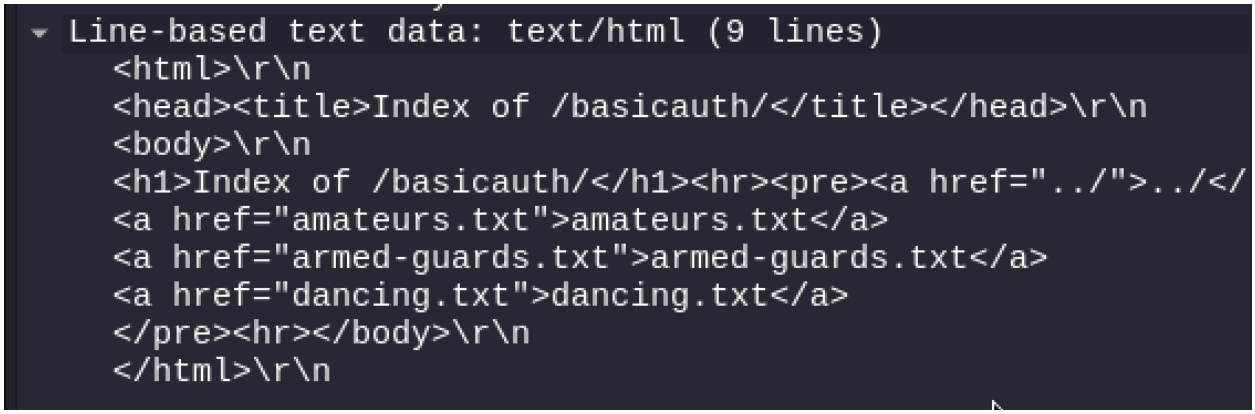
The next thing is a HTTP request to GET /basicauth/. This loads the html for the web page, which then prompts the following lines. A HTTP protocol from the server now follows, saying 401 unauthorized. This is due to me not having put the password in yet.

28	0.165225555	192.168.64.2	172.233.221.124	HTTP	417	GET /basicauth/ HTTP/1.1
29	0.189106128	172.233.221.124	192.168.64.2	TCP	54	80 → 36792 [ACK] Seq=1 Ack=364 Win=64128 L
30	0.189106336	172.233.221.124	192.168.64.2	HTTP	457	HTTP/1.1 401 Unauthorized (text/html)
31	0.189167794	192.168.64.2	172.233.221.124	TCP	54	36792 → 80 [ACK] Seq=364 Ack=404 Win=31872
32	7.074815596	192.168.64.2	172.233.221.124	HTTP	460	GET /basicauth/ HTTP/1.1
33	7.108177107	172.233.221.124	192.168.64.2	HTTP	458	HTTP/1.1 200 OK (text/html)

Then there is an HTTP 200 okay from the server, which comes after I have put the username and password in, and the rest of the webpage data is received, as I am now authorized to view the page after being authenticated.

```
Line-based text data: text/html (7 lines)
    <html>\r\n
    <head><title>401 Authorization Required</title></head>\r\
    <body>\r\n
    <center><h1>401 Authorization Required</h1></center>\r\n
    <hr><center>nginx/1.18.0 (Ubuntu)</center>\r\n
    </body>\r\n
    </html>\r\n
```

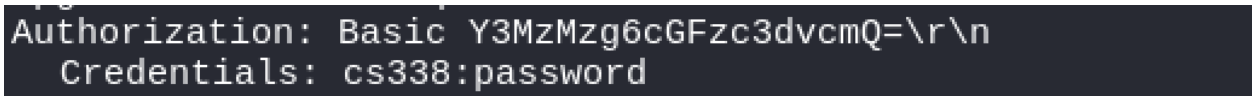
This first screenshot shows what the I can see before I am authenticated, and the one after shows the html information for the page after the 200 OK (when I am authenticated).



Then, there is a request for the favicon, which results in a 404 error as that is not found.

After the password is typed by the user, what sequence of queries and responses do you see? Is the password sent by the browser to the server, or does the browser somehow do the password checking itself? If the former, is the password sent in clear text or is it encrypted or something else?

The password is sent from the browser to the server, and the password is sent in clear text, as seen below.

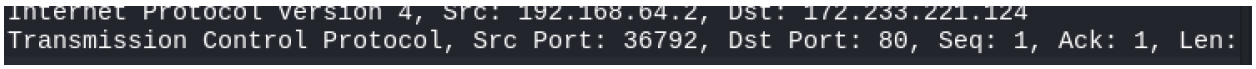


If it's encrypted, where did the encryption key come from?

I do not think that it is encrypted yet as we are using http and not https. HTTPS involves the TLS sessions which will encrypt the data, which HTTP does not. I do think that the password is encoded, (source: https://en.wikipedia.org/wiki/Basic_access_authentication) due to the Authorization: Basic <credentials> which is the encoding of the credentials in base64.

How does what you observe via Wireshark connect to the relevant sections of the HTTP and HTTP Basic Authentication specification documents?

The Wireshark output shows the TCP handshakes, which occur at the beginning of the connection as well as when the server is transmitting data to the browser. We can also see which port my browser is using as well as the port the server is using.

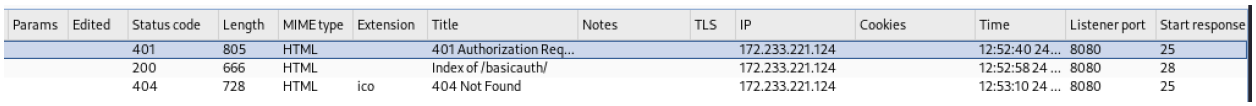


I can also see how HTTP is used by my browser to request information from the server and how the server responds with okay once it has been authenticated.

HTTP protocol is used to engage in exchanges between the client and server, and through the exchange of packets, I am able to eventually view the webpage.

I want you to do everything you can to understand what's going on in this browser/server interaction, and tell me the story of the interaction and your understanding of its mechanisms.

I also put it through burp suite, which provides less information, but also shows the status codes (401 when I was unauthorized, 200 okay when I was, and the 404 I'm assuming is associated with the no favicon as I got that error with wireshark as well).



Params	Edited	Status code	Length	MIME type	Extension	Title	Notes	TLS	IP	Cookies	Time	Listener port	Start response
		401	805	HTML		401 Authorization Req...			172.233.221.124		12:52:40 24...	8080	25
		200	666	HTML		Index of /basicauth/			172.233.221.124		12:52:58 24 ...	8080	28
		404	728	HTML	ico	404 Not Found			172.233.221.124		12:53:10 24 ...	8080	25