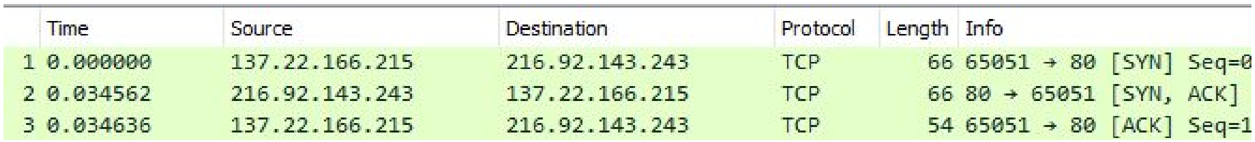
Sam Nozaki
Prof. Jeff Ondich
CS 231 Computer Security
January 10, 2018

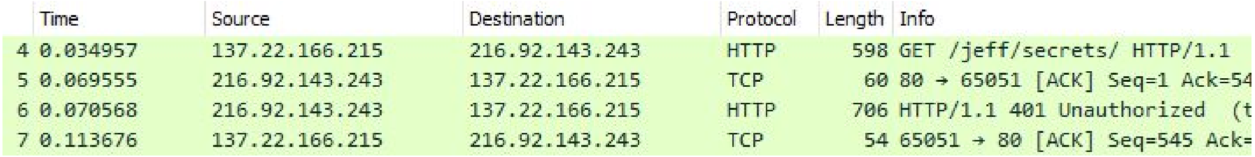
HTTP’s Basic Authentication

The first thing that occurs upon attempting to access Jeff’s secret site (http://sandbox.ultralingua.com/jeff/secrets/) is the execution of the TCP handshake. The client sends a SYN request, the server responds with a SYN, ACK, and the client responds with a final ACK, as shown in the screenshot below.



	Time	Source	Destination	Protocol	Length	Info
1	0.000000	137.22.166.215	216.92.143.243	TCP	66	65051 → 80 [SYN] Seq=0
2	0.034562	216.92.143.243	137.22.166.215	TCP	66	80 → 65051 [SYN, ACK]
3	0.034636	137.22.166.215	216.92.143.243	TCP	54	65051 → 80 [ACK] Seq=1

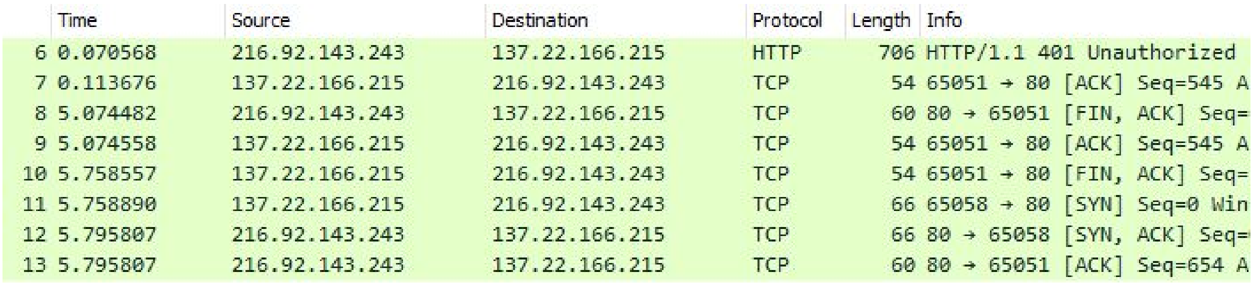
The next step involves the client sending an HTTP request for the webpage and the server acknowledges this request. However, in this case, the server does not respond with the usual 200 OK code, but instead responds with a 401 error because, of course, the client is unauthorized since the proper username and password have not been submitted. The client then acknowledges this error.



	Time	Source	Destination	Protocol	Length	Info
4	0.034957	137.22.166.215	216.92.143.243	HTTP	598	GET /jeff/secrets/ HTTP/1.1
5	0.069555	216.92.143.243	137.22.166.215	TCP	60	80 → 65051 [ACK] Seq=1 Ack=54
6	0.070568	216.92.143.243	137.22.166.215	HTTP	706	HTTP/1.1 401 Unauthorized (t
7	0.113676	137.22.166.215	216.92.143.243	TCP	54	65051 → 80 [ACK] Seq=545 Ack=

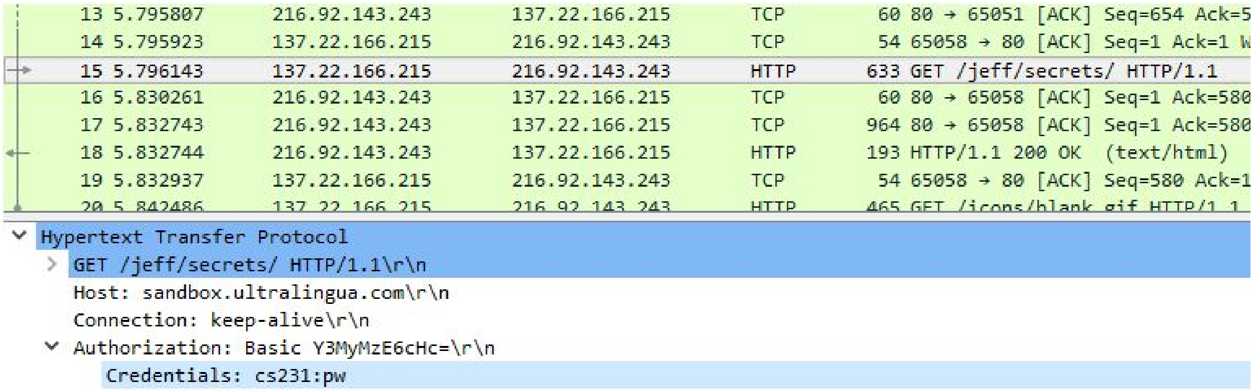
Now, the client must enter the correct username and password in order to gain the proper authorization to access the secrets hidden within Jeff’s website. Upon entering the information, the server then terminates the connection with a FIN, ACK. The client then responds with an

ACK followed by the their own FIN, ACK. Then, the connection is reinstated with another TCP handshake. The aforementioned packets are numbered 8-13 in the screenshot below.



	Time	Source	Destination	Protocol	Length	Info
6	0.070568	216.92.143.243	137.22.166.215	HTTP	706	HTTP/1.1 401 Unauthorized
7	0.113676	137.22.166.215	216.92.143.243	TCP	54	65051 → 80 [ACK] Seq=545 A
8	5.074482	216.92.143.243	137.22.166.215	TCP	60	80 → 65051 [FIN, ACK] Seq=
9	5.074558	137.22.166.215	216.92.143.243	TCP	54	65051 → 80 [ACK] Seq=545 A
10	5.758557	137.22.166.215	216.92.143.243	TCP	54	65051 → 80 [FIN, ACK] Seq=
11	5.758890	137.22.166.215	216.92.143.243	TCP	66	65058 → 80 [SYN] Seq=0 Win
12	5.795807	216.92.143.243	137.22.166.215	TCP	66	80 → 65058 [SYN, ACK] Seq=
13	5.795807	216.92.143.243	137.22.166.215	TCP	60	80 → 65051 [ACK] Seq=654 A

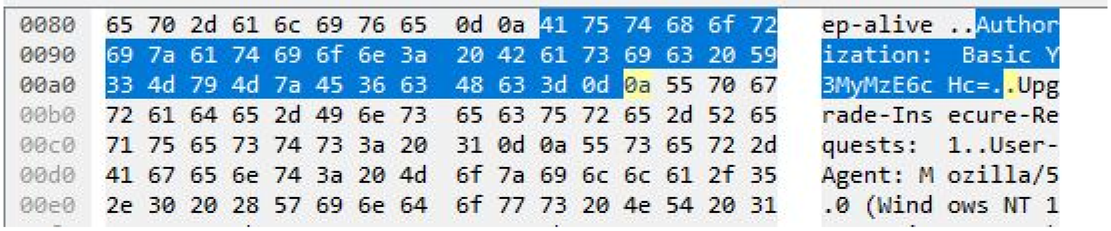
The next step is the second HTTP request for the secret webpage. This time, the client has full authorization since the correct username and password have been sent to the server along with the HTTP GET request. As shown in the screenshot below, the username and password are transmitted along with the GET request and are visible under the heading "Hypertext Transmission Protocol/Authorization/Credentials". Once this information is verified by the server, the server sends back the requested hypertext along with a 200 OK code.



13	5.795807	216.92.143.243	137.22.166.215	TCP	60	80 → 65051 [ACK] Seq=654 Ack=5
14	5.795923	137.22.166.215	216.92.143.243	TCP	54	65058 → 80 [ACK] Seq=1 Ack=1 W
15	5.796143	137.22.166.215	216.92.143.243	HTTP	633	GET /jeff/secrets/ HTTP/1.1
16	5.830261	216.92.143.243	137.22.166.215	TCP	60	80 → 65058 [ACK] Seq=1 Ack=580
17	5.832743	216.92.143.243	137.22.166.215	TCP	964	80 → 65058 [ACK] Seq=1 Ack=580
18	5.832744	216.92.143.243	137.22.166.215	HTTP	193	HTTP/1.1 200 OK (text/html)
19	5.832937	137.22.166.215	216.92.143.243	TCP	54	65058 → 80 [ACK] Seq=580 Ack=1
20	5.842486	137.22.166.215	216.92.143.243	HTTP	465	GET /icons/blank.gif HTTP/1.1

```
Hypertext Transfer Protocol
  GET /jeff/secrets/ HTTP/1.1\r\n
  Host: sandbox.ultralingua.com\r\n
  Connection: keep-alive\r\n
  Authorization: Basic Y3MyMzE6cHc=\r\n
    Credentials: cs231:pw
```

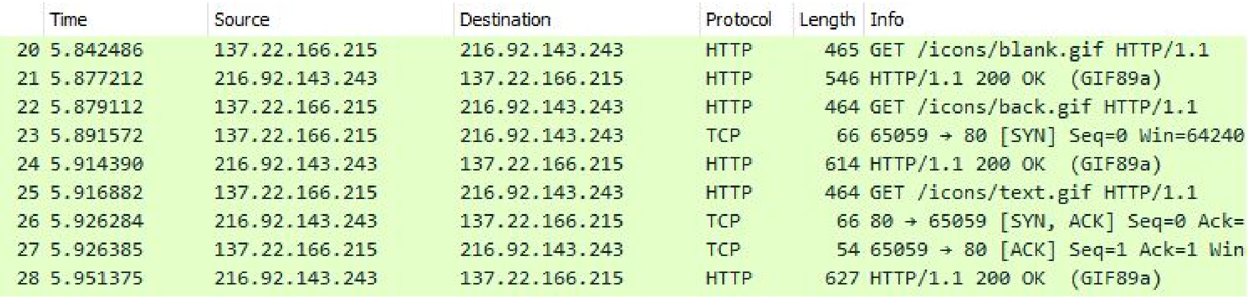
At first glance, it looks like the password is sent in plaintext, which would be a major security concern since anyone monitoring the packets being exchanged on the connection would be able to easily glean that information and possibly abuse it. However, upon further inspection of this transmission, the information under “Credentials” is not actually sent as plaintext in the packet; it is encrypted.



```
0080   65 70 2d 61 6c 69 76 65  0d 0a 41 75 74 68 6f 72    ep-alive ..Author
0090   69 7a 61 74 69 6f 6e 3a  20 42 61 73 69 63 20 59    ization:  Basic Y
00a0   33 4d 79 4d 7a 45 36 63  48 63 3d 0d 0a 55 70 67    3MyMzE6c Hc=..Upg
00b0   72 61 64 65 2d 49 6e 73  65 63 75 72 65 2d 52 65    rade-Ins ecure-Re
00c0   71 75 65 73 74 73 3a 20  31 0d 0a 55 73 65 72 2d    quests:  1..User-
00d0   41 67 65 6e 74 3a 20 4d  6f 7a 69 6c 6c 61 2f 35    Agent: M ozilla/5
00e0   2e 30 20 28 57 69 6e 64  6f 77 73 20 4e 54 20 31    .0 (Wind ows NT 1
```

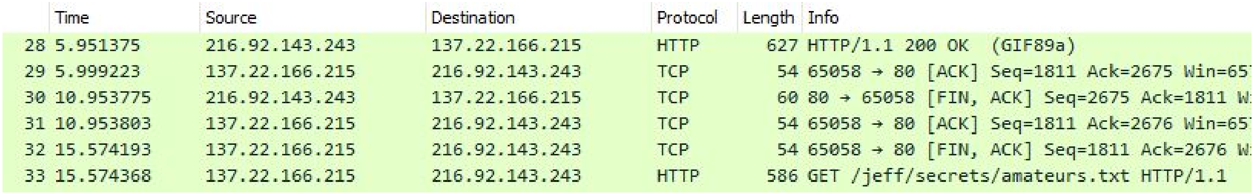
Some quick research on the Basic encryption method used by the client reveals that it is a simple Base64 substitution cipher and can be easily decrypted using any one of a number of online resources (https://www.base64decode.org). This method of encryption provides a *very* superficial layer of obscurity since any packet sniffer can very easily decode the information being transmitted.

After getting that initial 200 OK, the client then requests the remainder of the visual elements that are present on the webpage with GET requests for three images: blank.gif, back.gif, and text.gif, as shown below in packets 20, 22, and 25. These requests are met by the usual 200 OK message from the server along with the requested content as shown in packets 21, 24, and 28.



```
   Time         Source           Destination      Protocol  Length Info
20 5.842486     137.22.166.215   216.92.143.243   HTTP         465 GET /icons/blank.gif HTTP/1.1
21 5.877212     216.92.143.243   137.22.166.215   HTTP         546 HTTP/1.1 200 OK  (GIF89a)
22 5.879112     137.22.166.215   216.92.143.243   HTTP         464 GET /icons/back.gif HTTP/1.1
23 5.891572     137.22.166.215   216.92.143.243   TCP           66 65059 → 80 [SYN] Seq=0 Win=64240
24 5.914390     216.92.143.243   137.22.166.215   HTTP         614 HTTP/1.1 200 OK  (GIF89a)
25 5.916882     137.22.166.215   216.92.143.243   HTTP         464 GET /icons/text.gif HTTP/1.1
26 5.926284     216.92.143.243   137.22.166.215   TCP           66 80 → 65059 [SYN, ACK] Seq=0 Ack=
27 5.926385     137.22.166.215   216.92.143.243   TCP           54 65059 → 80 [ACK] Seq=1 Ack=1 Win
28 5.951375     216.92.143.243   137.22.166.215   HTTP         627 HTTP/1.1 200 OK  (GIF89a)
```

Now that the page is loaded in its entirety, the server then terminates the connection with the client since all relevant information has been successfully transferred between the two entities. Just like before, the server sends a FIN, ACK that is answered by an ACK from the client. Since the client may need to access additional information, the client does not send its own FIN, ACK until it requests a different webpage, in this case the page in question is http://sandbox.ultralingua.com/jeff/secrets/amateurs.txt. Note the timestamps for packets 31-33 in the screenshot below-- the client's FIN, ACK and the next GET request are sent almost simultaneously, about five seconds after the server's FIN, ACK is received by the client.



	Time	Source	Destination	Protocol	Length	Info
28	5.951375	216.92.143.243	137.22.166.215	HTTP	627	HTTP/1.1 200 OK (GIF89a)
29	5.999223	137.22.166.215	216.92.143.243	TCP	54	65058 → 80 [ACK] Seq=1811 Ack=2675 Win=65
30	10.953775	216.92.143.243	137.22.166.215	TCP	60	80 → 65058 [FIN, ACK] Seq=2675 Ack=1811 W
31	10.953803	137.22.166.215	216.92.143.243	TCP	54	65058 → 80 [ACK] Seq=1811 Ack=2676 Win=65
32	15.574193	137.22.166.215	216.92.143.243	TCP	54	65058 → 80 [FIN, ACK] Seq=1811 Ack=2676 W
33	15.574368	137.22.166.215	216.92.143.243	HTTP	586	GET /jeff/secrets/amateurs.txt HTTP/1.1

The same processes, minus the authentication, occur in the same manner when any other webpage on the server is accessed by the client. Thus, we have the framework of HTTP's basic authentication process.