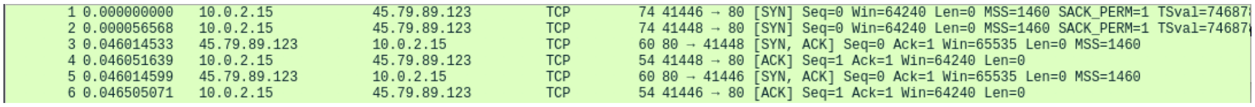
Authors: Anders Shenholm and Riaz Kelly

HTTP Basic Authentication: The Movie

We can analyze the steps of basic authentication by looking at frames captured by wireshark in a client-server interaction. In this case, the client is Kali running via VirtualBox on a personal computer and the server is http://cs231.jeffondich.com/basicauth/. Most frames capture the TCP interactions that form the basic connection between client and server, however all of the steps of basic of HTTP basic authentication occur, of course, in HTTP interactions.

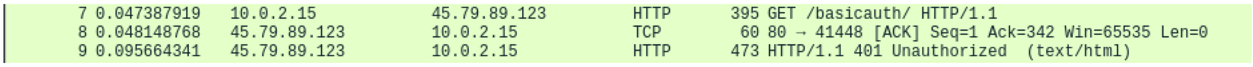
TCP Handshake:



```
      1 0.000000000   10.0.2.15             45.79.89.123          TCP        74 41446 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=74687
      2 0.000056568   10.0.2.15             45.79.89.123          TCP        74 41448 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=74687
      3 0.046014533   45.79.89.123          10.0.2.15             TCP        60 80 → 41448 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460
      4 0.046051639   10.0.2.15             45.79.89.123          TCP        54 41448 → 80 [ACK] Seq=1 Ack=1 Win=64240 Len=0
      5 0.046014599   45.79.89.123          10.0.2.15             TCP        60 80 → 41446 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460
      6 0.046505071   10.0.2.15             45.79.89.123          TCP        54 41446 → 80 [ACK] Seq=1 Ack=1 Win=64240 Len=0
```

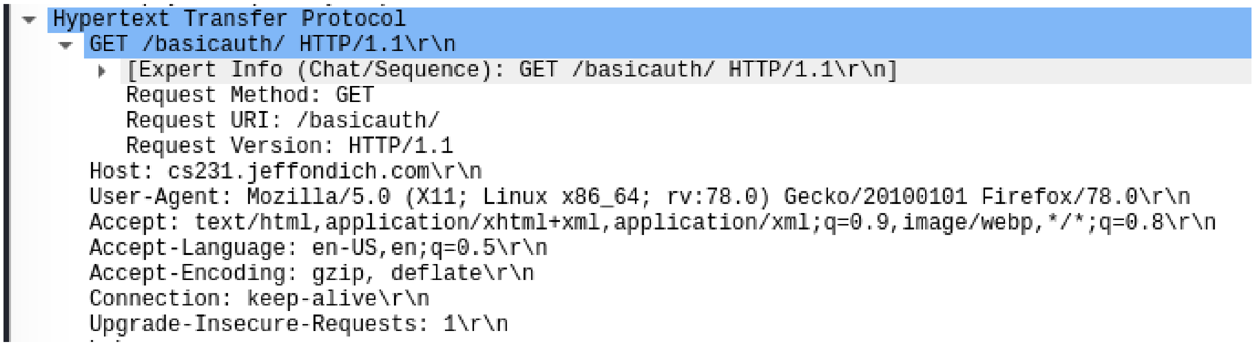
Frames 1, 2: Client sends initial request to make a TCP connection.
Frames 3, 5: The server acknowledges the client and agrees to make a connection
Frames 4, 6: The client acknowledges the connection

Initial HTTP Action:



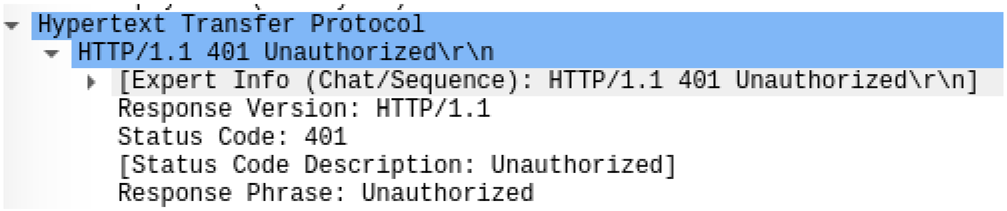
```
      7 0.047387919   10.0.2.15             45.79.89.123          HTTP       395 GET /basicauth/ HTTP/1.1
      8 0.048148768   45.79.89.123          10.0.2.15             TCP         60 80 → 41448 [ACK] Seq=1 Ack=342 Win=65535 Len=0
      9 0.095664341   45.79.89.123          10.0.2.15             HTTP       473 HTTP/1.1 401 Unauthorized  (text/html)
```

In frame 7, The client requests the webpage extension /basicauth/



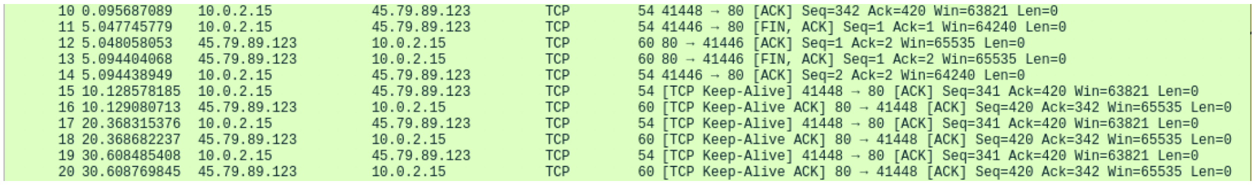
```
▾ Hypertext Transfer Protocol
  ▾ GET /basicauth/ HTTP/1.1\r\n
    ▸ [Expert Info (Chat/Sequence): GET /basicauth/ HTTP/1.1\r\n]
      Request Method: GET
      Request URI: /basicauth/
      Request Version: HTTP/1.1
    Host: cs231.jeffondich.com\r\n
    User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0\r\n
    Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8\r\n
    Accept-Language: en-US,en;q=0.5\r\n
    Accept-Encoding: gzip, deflate\r\n
    Connection: keep-alive\r\n
    Upgrade-Insecure-Requests: 1\r\n
```

In frame 9, the server communicates that authentication is required, sending HTTP status code 401.



```
▾ Hypertext Transfer Protocol
  ▾ HTTP/1.1 401 Unauthorized\r\n
    ▸ [Expert Info (Chat/Sequence): HTTP/1.1 401 Unauthorized\r\n]
      Response Version: HTTP/1.1
      Status Code: 401
      [Status Code Description: Unauthorized]
      Response Phrase: Unauthorized
```

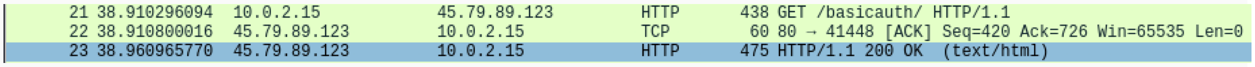
Frames 10-20:



```
10 0.095687089   10.0.2.15      45.79.89.123   TCP   54 41448 → 80 [ACK] Seq=342 Ack=420 Win=63821 Len=0
11 5.047745779   10.0.2.15      45.79.89.123   TCP   54 41446 → 80 [FIN, ACK] Seq=1 Ack=1 Win=64240 Len=0
12 5.048058053   45.79.89.123   10.0.2.15      TCP   60 80 → 41446 [ACK] Seq=1 Ack=2 Win=65535 Len=0
13 5.094404068   45.79.89.123   10.0.2.15      TCP   60 80 → 41446 [FIN, ACK] Seq=1 Ack=2 Win=65535 Len=0
14 5.094438949   10.0.2.15      45.79.89.123   TCP   54 41446 → 80 [ACK] Seq=2 Ack=2 Win=64240 Len=0
15 10.128578185  10.0.2.15      45.79.89.123   TCP   54 [TCP Keep-Alive] 41448 → 80 [ACK] Seq=341 Ack=420 Win=63821 Len=0
16 10.129080713  45.79.89.123   10.0.2.15      TCP   60 [TCP Keep-Alive ACK] 80 → 41448 [ACK] Seq=420 Ack=342 Win=65535 Len=0
17 20.368315376  10.0.2.15      45.79.89.123   TCP   54 [TCP Keep-Alive] 41448 → 80 [ACK] Seq=341 Ack=420 Win=63821 Len=0
18 20.368682237  45.79.89.123   10.0.2.15      TCP   60 [TCP Keep-Alive ACK] 80 → 41448 [ACK] Seq=420 Ack=342 Win=65535 Len=0
19 30.608485408  10.0.2.15      45.79.89.123   TCP   54 [TCP Keep-Alive] 41448 → 80 [ACK] Seq=341 Ack=420 Win=63821 Len=0
20 30.608769845  45.79.89.123   10.0.2.15      TCP   60 [TCP Keep-Alive ACK] 80 → 41448 [ACK] Seq=420 Ack=342 Win=65535 Len=0
```

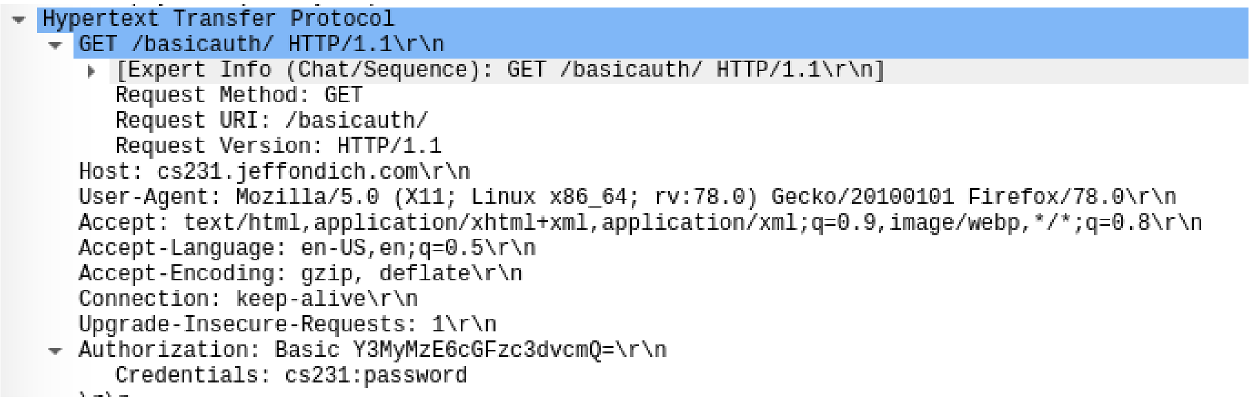
In the time between the authentication prompt and the entrance of the authentication information, the client and server continue to communicate with TCP packets. We can see in frames 15-20 that the client checks its connection to the server by sending keepalive packets. These packets are scheduled probes to the server to check that the connection is still intact. The server responds with a keepalive ack packet confirming the connection (source: https://tldp.org/HOWTO/TCP-Keepalive-HOWTO/overview.html)

Credentials passed, access granted:



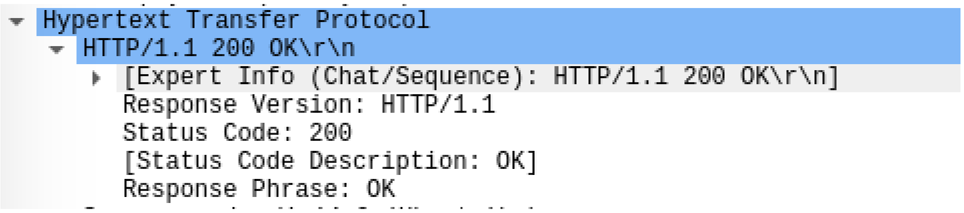
```
21 38.910296094  10.0.2.15      45.79.89.123    HTTP    438 GET /basicauth/ HTTP/1.1
22 38.910800016  45.79.89.123   10.0.2.15       TCP      60 80 → 41448 [ACK] Seq=420 Ack=726 Win=65535 Len=0
23 38.960965770  45.79.89.123   10.0.2.15       HTTP    475 HTTP/1.1 200 OK  (text/html)
```

Frame 21: The client repeats the request GET /basicauth/, this time sending the requested credentials. The password does not get encrypted when it is sent from the client to the server. However, the password is transformed into base64 notation in order to ensure that the input from the user is reported correctly to the server. Since base64 notation is standardized and can reversed to normal text, the password is not encrypted and therefore, is not confidential (source: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Authorization).



```
▾ Hypertext Transfer Protocol
  ▾ GET /basicauth/ HTTP/1.1\r\n
    ▸ [Expert Info (Chat/Sequence): GET /basicauth/ HTTP/1.1\r\n]
      Request Method: GET
      Request URI: /basicauth/
      Request Version: HTTP/1.1
    Host: cs231.jeffondich.com\r\n
    User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0\r\n
    Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8\r\n
    Accept-Language: en-US,en;q=0.5\r\n
    Accept-Encoding: gzip, deflate\r\n
    Connection: keep-alive\r\n
    Upgrade-Insecure-Requests: 1\r\n
  ▾ Authorization: Basic Y3MyMzE6cGFzc3dvcmQ=\r\n
      Credentials: cs231:password
```

Frame 23: This time, the client does have authorization to access the content, so the server responds with the HTTP status code 200 OK. Since the credentials were sent from the client to the server, it would make sense that the server somehow checks them itself. This is probably a relatively simple process and is not shown in Wireshark.



```
▾ Hypertext Transfer Protocol
  ▾ HTTP/1.1 200 OK\r\n
    ▸ [Expert Info (Chat/Sequence): HTTP/1.1 200 OK\r\n]
      Response Version: HTTP/1.1
      Status Code: 200
      [Status Code Description: OK]
      Response Phrase: OK
```

Frames beyond 23: What we see in wireshark after frame 23 (when our client gains access to the webpage) is the remaining TCP and HTTP interaction that occurred as our client attempts to access the contents of the website.

Observations vs HTTP Documentation:

Inside HTTP document 2617, it explicitly says “The most serious flaw in Basic authentication is that it results in the essentially cleartext transmission of the user's password over the physical network.” We observed this in Wireshark inside frame 21. The client sends the inputted username and password using base64, which is not a form of encryption. It is interesting that the HTTP document says the biggest danger that comes from this lack of security is that users typically use the same password for multiple accounts. The document also says that if the browser chooses a password for the user and does not allow them to change it, then the danger is minimized (given there is no important information on the server itself). Source: https://tools.ietf.org/html/rfc2617#page-19.