

Oliver Clay
4/8/22

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

eth0 (host 45.79.89.123)

Apply a display filter - <Ctrl>F

+

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	172.16.127.128	45.79.89.123	TCP	74	36152 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=3549383393 TSsecr=0 WS=128
2	0.000073386	172.16.127.128	45.79.89.123	TCP	74	36154 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=3549383393 TSsecr=0 WS=128
3	0.044960260	45.79.89.123	172.16.127.128	TCP	60	80 → 36152 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0
4	0.044960523	45.79.89.123	172.16.127.128	TCP	60	80 → 36154 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460
5	0.044955616	172.16.127.128	45.79.89.123	TCP	54	36152 → 80 [ACK] Seq=1 Ack=1 Win=64240 Len=0
6	0.045455556	172.16.127.128	45.79.89.123	TCP	54	36154 → 80 [ACK] Seq=1 Ack=1 Win=64240 Len=0
7	0.045274185	172.16.127.128	45.79.89.123	HTTP	395	GET /basicauth/ HTTP/1.1
8	0.045453669	45.79.89.123	172.16.127.128	TCP	60	80 → 36154 [ACK] Seq=1 Ack=342 Win=64240 Len=0
9	0.089956485	45.79.89.123	172.16.127.128	HTTP	457	HTTP/1.1 401 Unauthorized (text/html)
10	0.090973781	172.16.127.128	45.79.89.123	TCP	54	36154 → 80 [ACK] Seq=342 Ack=484 Win=63837 Len=0
11	0.091408439	172.16.127.128	45.79.89.123	TCP	54	36152 → 80 [FIN, ACK] Seq=1 Ack=1 Win=64240 Len=0
12	0.046465778	45.79.89.123	172.16.127.128	TCP	60	80 → 36152 [ACK] Seq=1 Ack=2 Win=64239 Len=0
13	0.091552781	45.79.89.123	172.16.127.128	TCP	60	80 → 36152 [FIN, PSH, ACK] Seq=1 Ack=2 Win=64239 Len=0
14	0.091714861	172.16.127.128	45.79.89.123	TCP	54	36152 → 80 [ACK] Seq=2 Ack=3 Win=64240 Len=0
15	0.071761725	172.16.127.128	45.79.89.123	HTTP	438	GET /basicauth/ HTTP/1.1
16	0.071562521	45.79.89.123	172.16.127.128	TCP	60	80 → 36154 [ACK] Seq=484 Ack=726 Win=64240 Len=0
17	0.177661291	45.79.89.123	172.16.127.128	HTTP	458	HTTP/1.1 200 OK (text/html)
18	0.117700225	172.16.127.128	45.79.89.123	TCP	54	36154 → 80 [ACK] Seq=726 Ack=908 Win=63837 Len=0
19	0.171458523	172.16.127.128	45.79.89.123	HTTP	355	GET /favicon.ico HTTP/1.1
20	0.172071488	45.79.89.123	172.16.127.128	TCP	60	80 → 36154 [ACK] Seq=808 Ack=1807 Win=64240 Len=0
21	0.217880977	45.79.89.123	172.16.127.128	TCP	383	HTTP/1.1 404 Not Found (text/html)
22	0.218000575	172.16.127.128	45.79.89.123	TCP	54	36154 → 80 [ACK] Seq=1027 Ack=1137 Win=63837 Len=0
23	0.135708353	172.16.127.128	45.79.89.123	HTTP	434	GET / HTTP/1.1
24	0.135708469	45.79.89.123	172.16.127.128	TCP	60	80 → 36154 [ACK] Seq=1137 Ack=1407 Win=64240 Len=0
25	0.140334160	45.79.89.123	172.16.127.128	HTTP	639	HTTP/1.1 200 OK (text/html)
26	0.148336907	172.16.127.128	45.79.89.123	TCP	54	36154 → 80 [ACK] Seq=1407 Ack=1722 Win=63837 Len=0
27	0.1479447018	172.16.127.128	45.79.89.123	HTTP	354	GET /jffr_square_head.jpg HTTP/1.1
28	0.147291483	45.79.89.123	172.16.127.128	TCP	60	80 → 36154 [ACK] Seq=1722 Ack=1707 Win=64240 Len=0
29	0.147717557	45.79.89.123	172.16.127.128	TCP	14534	80 → 36154 [PSH, ACK] Seq=1722 Ack=1707 Win=64240 Len=14480 [TCP segment of a reassembled PDU]
30	0.177222195	172.16.127.128	45.79.89.123	TCP	54	36154 → 80 [ACK] Seq=1707 Ack=15080 Win=55480 Len=0
31	0.152436765	45.79.89.123	172.16.127.128	TCP	29014	80 → 36154 [PSH, ACK] Seq=16202 Ack=1707 Win=64240 Len=29960 [TCP segment of a reassembled PDU]
32	0.152420463	172.16.127.128	45.79.89.123	TCP	54	36154 → 80 [ACK] Seq=1707 Ack=41562 Win=45260 Len=0
33	0.117083517	45.79.89.123	172.16.127.128	HTTP	3049	HTTP/1.1 200 OK (JPEG image)
34	0.157085623	172.16.127.128	45.79.89.123	TCP	54	36154 → 80 [ACK] Seq=1707 Ack=82057 Win=39420 Len=0
35	0.459333390	172.16.127.128	45.79.89.123	HTTP	499	GET /basicauth/ameeturs.txt HTTP/1.1
36	0.4590626902	45.79.89.123	172.16.127.128	TCP	60	80 → 36154 [ACK] Seq=82057 Ack=2152 Win=64240 Len=0
37	0.505796395	45.79.89.123	172.16.127.128	HTTP	375	HTTP/1.1 200 OK (text/plain)
38	0.505811912	172.16.127.128	45.79.89.123	TCP	54	36154 → 80 [ACK] Seq=2152 Ack=82378 Win=105535 Len=0
39	0.494147183	45.79.89.123	172.16.127.128	HTTP	503	GET /basicauth/armed-guard.txt HTTP/1.1
40	0.494161667	45.79.89.123	172.16.127.128	TCP	60	80 → 36154 [ACK] Seq=82378 Ack=2604 Win=64240 Len=0
41	0.748959518	45.79.89.123	172.16.127.128	HTTP	482	HTTP/1.1 200 OK (text/plain)
42	0.749012291	172.16.127.128	45.79.89.123	TCP	54	36154 → 80 [ACK] Seq=2604 Ack=82786 Win=65535 Len=0
43	0.594565512	172.16.127.128	45.79.89.123	HTTP	498	GET /basicauth/dancing.txt HTTP/1.1
44	0.594549055	45.79.89.123	172.16.127.128	TCP	60	80 → 36154 [ACK] Seq=82786 Ack=3045 Win=64240 Len=0
45	0.140510191	45.79.89.123	172.16.127.128	HTTP	528	HTTP/1.1 200 OK (text/plain)
46	0.141120777	172.16.127.128	45.79.89.123	TCP	54	36154 → 80 [ACK] Seq=3045 Ack=83260 Win=105535 Len=0

wreshark_eth0MAROKI.pcapng

Packets: 46 · Displayed: 46 (100.0%) · Dropped: 0 (0.0%)

Profile: Default

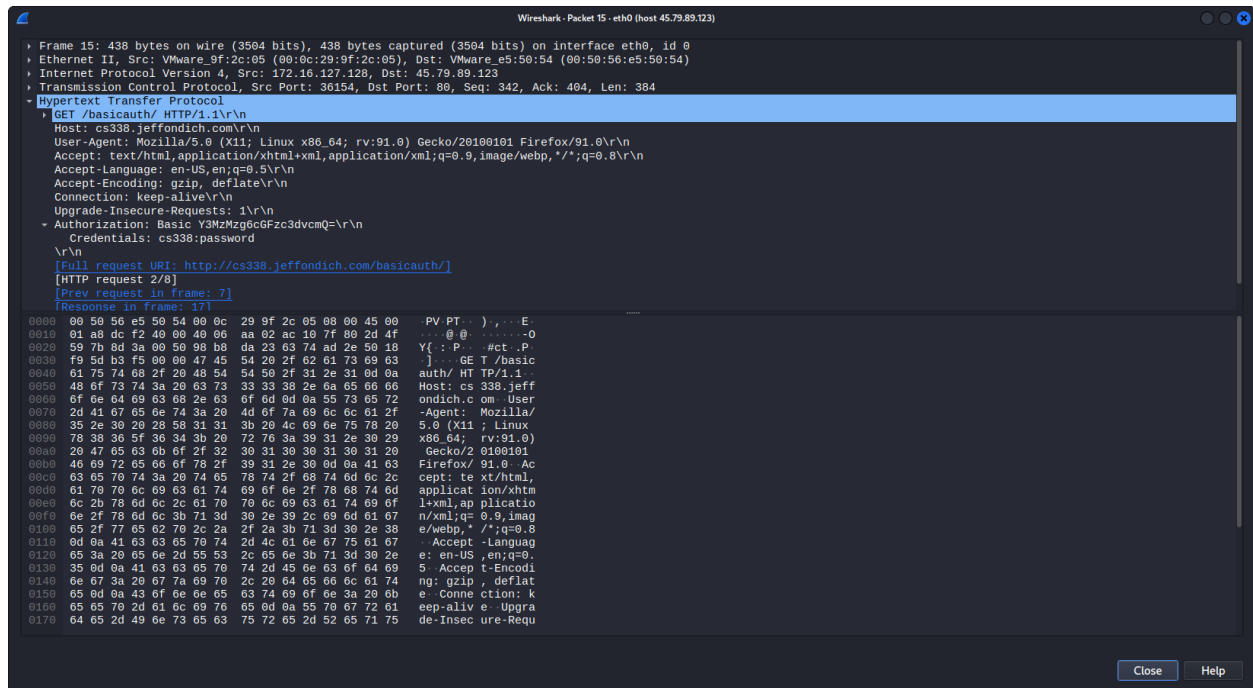
Above is an overview screenshot of the full client-server interaction when visiting the website <http://cs338.jeffondich.com/basicauth/>.

The first 6 frames detail the TCP handshake between the client and server. For whatever reason (perhaps multi-threading) the client sent two [SYN] requests, which, in turn, were answered by two [SYN, ACK] requests, and responded to by the client's own two [ACK] requests. Hence, the TCP handshake occurred twice, and this was consistent between multiple packet collection sessions.

In frame 7, however, things get interesting, the client is requesting “GET /basicauth/ HTTP/1.1”, this seems like the client requesting the mechanism for authentication with the server. The server acknowledges this in frame 8, and then sends “Unauthorized (text/html)” in frame 9. Following this, communications are a combination of [ACK] and [FIN] between the client and

server, indicating that communications could very well end here should proper authentication not be sent by the client.

Yet, low and behold, the client sends its credentials in frame 15.



The image shows a Wireshark packet capture window titled "Wireshark - Packet 15 - eth0 (host 45.79.89.123)". The packet list on the left shows "Frame 15: 438 bytes on wire (3504 bits), 438 bytes captured (3504 bits) on interface eth0, id 0". The packet details pane shows the following structure:

- Ethernet II, Src: VMware_Bf2c:05 (00:0c:29:9f:2c:05), Dst: VMware_e5:50:54 (00:50:56:e5:50:54)
- Internet Protocol Version 4, Src: 172.16.127.128, Dst: 45.79.89.123
- Transmission Control Protocol, Src Port: 36154, Dst Port: 80, Seq: 342, Ack: 404, Len: 384
- Hypertext Transfer Protocol
 - GET /basicauth/ HTTP/1.1\r\n
 - Host: cs338.jeffondich.com\r\n
 - User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0\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 Y3MzMzg6cGFzc3dvcmQ=\r\n
 - Credentials: cs338:password

The packet bytes pane shows the raw data in hexadecimal and ASCII. The ASCII portion shows the request line and headers, including the Basic authentication header: "Authorization: Basic Y3MzMzg6cGFzc3dvcmQ=".

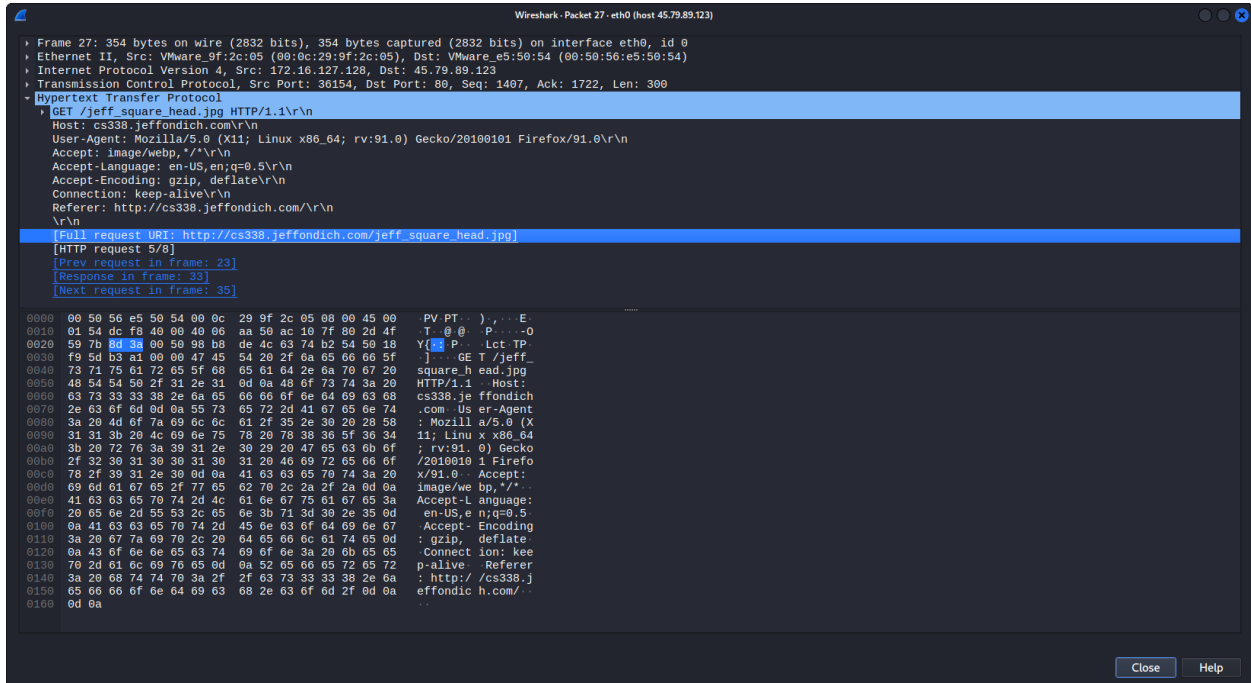
These credentials are found in the payload of the packet, under the HTTP headers, and are encoded in base64, (not encrypted, no keys were visibly sent, and this observation matches my expectations from the authentication's documentation) and are delivered to the server as plain text. I was expecting to have to dig around a little deeper to find these credentials, but apparently Basic HTTP Authentication is so readable that Wireshark saw the characters Y3MzMzg6cGFzc3dvcmQ=\r\n and felt enlightened enough to decode them for me then and there, putting the username:password under its own little tab.

After this, the server sends an [ACK] followed by an OK (text/html) in frame 17 signaling that it got the client's message and found it acceptable. And... we're in.

Following this authentication sequence, things seem to happen in the manner of standard TCP communication and HTTP GET requests for text or images. Nothing else happens that's super worth mentioning. I opened each of

the subsections of the website in turn, looking at the server's sensitive information, such as jeff_square_head.jpg.

Imagine, if I (a malicious third-party) had intercepted these packets; I would have wrongful access to this highly privy information.



The image shows a Wireshark packet capture window titled "Wireshark - Packet 27: eth0 (host 45.79.89.123)". The packet list on the left shows "Frame 27: 354 bytes on wire (2832 bits), 354 bytes captured (2832 bits) on interface eth0, id 0". The packet details pane shows the following structure:

- Frame 27: 354 bytes on wire (2832 bits), 354 bytes captured (2832 bits) on interface eth0, id 0
- Ethernet II, Src: VMware 9f:2c:05 (00:0c:29:9f:2c:05), Dst: VMware e5:50:54 (00:50:56:e5:50:54)
- Internet Protocol Version 4, Src: 172.16.127.128, Dst: 45.79.89.123
- Transmission Control Protocol, Src Port: 36154, Dst Port: 80, Seq: 1487, Ack: 1722, Len: 380
- Hypertext Transfer Protocol
 - GET /jeff_square_head.jpg HTTP/1.1\r\n
 - Host: cs338.jeffondich.com\r\n
 - User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20190101 Firefox/91.0\r\n
 - Accept: image/webp,*/*\r\n
 - Accept-Language: en-US,en;q=0.5\r\n
 - Accept-Encoding: gzip, deflate\r\n
 - Connection: keep-alive\r\n
 - Referer: http://cs338.jeffondich.com/\r\n
 - \r\n

The packet bytes pane shows the raw data of the packet, including the HTTP request line and headers.

```
0000  00 50 56 e5 50 54 00 0c 29 9f 2c 05 08 00 45 00  PV PT . . . . E
0010  01 54 dc f8 49 00 40 06 aa 50 ac 10 7f 89 2d 4f  T . @ . P . . . -O
0020  59 7b 8d 3a 00 50 98 b8 de 4c 63 74 b2 54 50 18  Y { [ P . . Lct TP
0030  f9 5d b3 a1 00 00 47 45 54 20 2f 6a 65 66 66 5f  ] . . . GE T /jeff_
0040  73 71 75 61 72 65 5f 68 65 61 64 2e 6a 70 67 20  square_h ead.jpg
0050  48 54 54 50 2f 31 2e 31 0d 0a 48 6f 73 74 3a 20  HTTP/1.1 . Host:
0060  63 73 33 33 38 2e 6a 65 66 66 6f 6e 64 69 63 68  cs338.je ffondich
0070  2e 63 6f 6d 0d 0a 55 73 65 72 2d 41 67 65 6e 74  .com .Us er-Agent
0080  3a 20 4d 6f 7a 69 6c 6c 61 2f 35 2e 30 20 28 58  : Mozill a/5.0 (X
0090  31 31 3b 20 4c 69 6e 75 78 20 78 38 36 5f 36 34  11; Linu x x86_64
00a0  3b 20 72 70 3a 39 31 2e 30 29 29 47 65 63 6b 6f  / rv:91. 0) Gecko
00b0  2f 32 30 31 30 30 31 30 31 20 46 69 72 65 66 6f  /2019010 1 Firefo
00c0  78 2f 39 31 2e 30 0d 0a 41 63 63 65 70 74 3a 20  x/91.0 . Accept:
00d0  69 6d 61 67 65 2f 77 65 62 70 2c 2a 2f 2a 0d 0a  image/we bp,*/*
00e0  41 63 63 65 70 74 2d 4c 61 6e 67 75 61 67 65 3a  Accept-L anguage:
00f0  20 65 6e 2d 55 53 2c 65 6e 3b 71 3d 30 2e 35 0d  en-US,e n;q=0.5
0100  0a 41 63 63 65 70 74 2d 45 6e 63 6f 64 69 6e 67  Accept- Encoding
0110  3a 20 67 7a 69 70 2c 20 64 65 66 6c 61 74 65 0d  : gzip, deflate
0120  0a 43 6f 6e 6e 65 63 74 69 6f 6e 3a 20 6b 65 65  Connect ion: kee
0130  70 2d 61 6c 69 76 65 0d 0a 52 65 66 65 72 65 72  p-alive . Referer
0140  3a 20 68 74 74 70 3a 2f 2f 63 73 33 33 38 2e 6a  : http:/ /cs338.j
0150  65 66 66 6f 6e 64 69 63 68 2e 63 6f 6d 2f 0d 0a  effondic h.com/
0160  0d 0a  ..
```