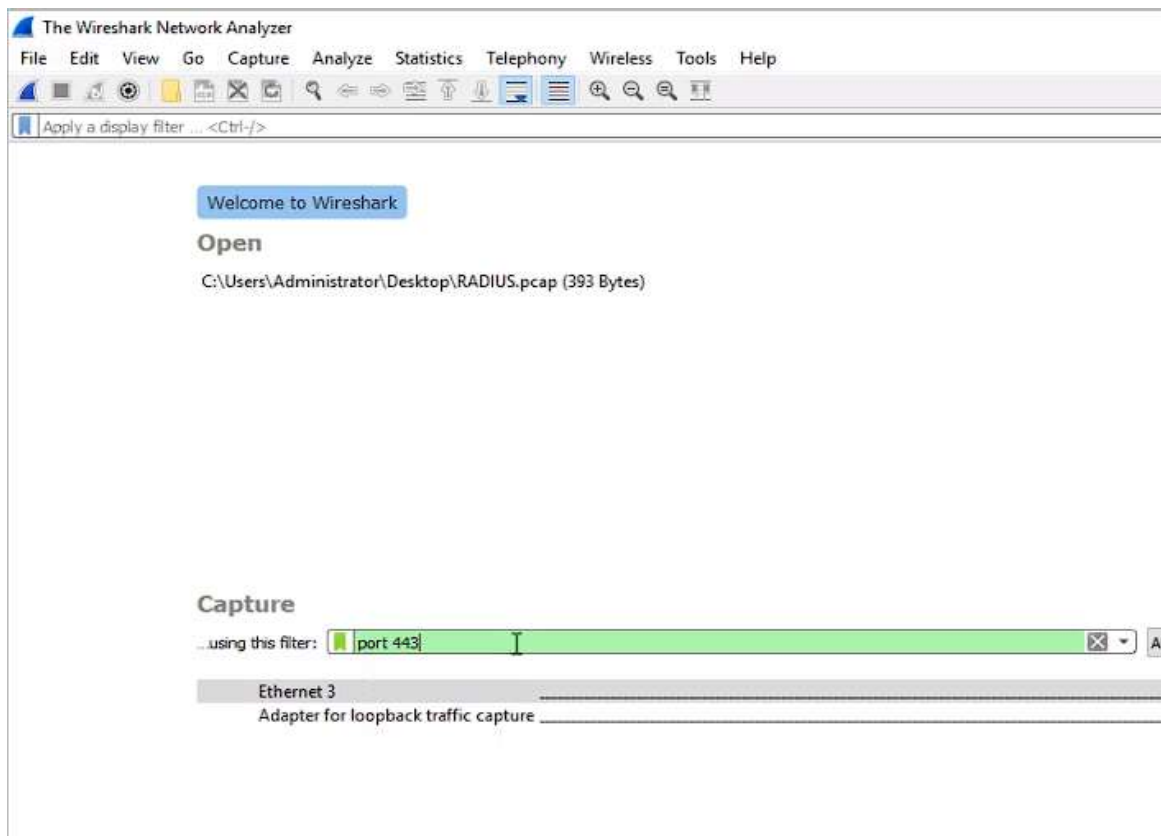


Manuela Kesten, april '24.

Generate, Capture, Analyze then Decrypt HTTPS Traffic in Wireshark

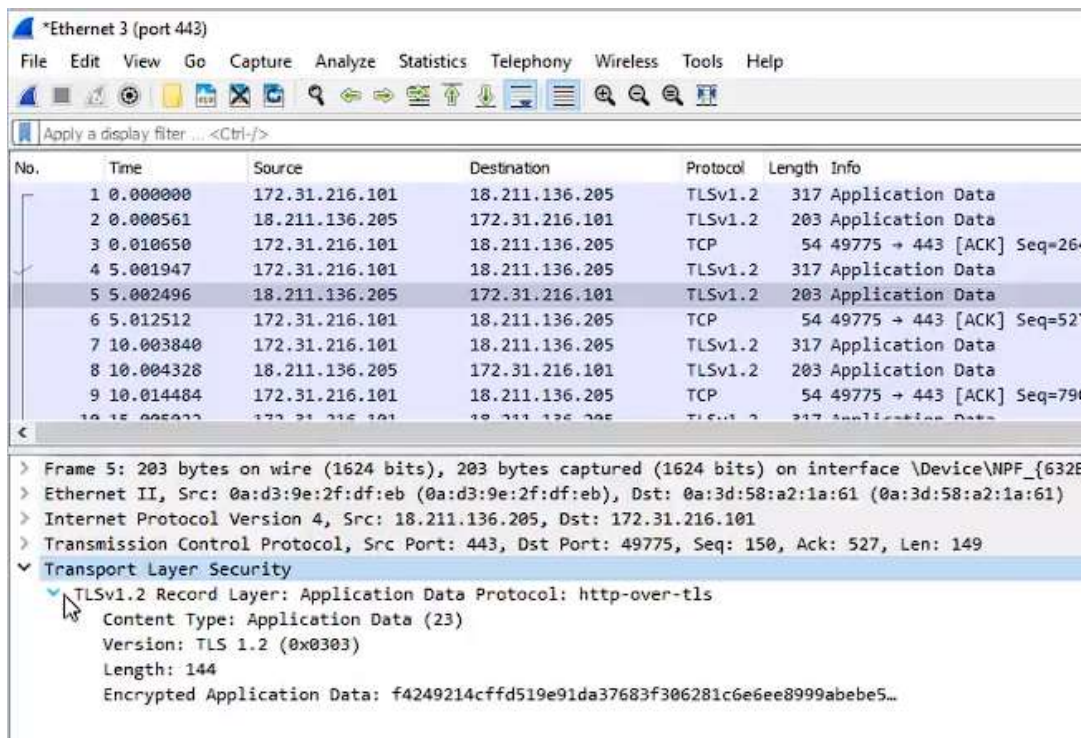
HTTPS is used for secure communication. It's http-over-tls. It operates on port 443.

I began with capture using the filter *port 443*.



Then clicked on interface **Ethernet 3**. I generated some traffic googling some random webpage.

Examining one Application Data, in section Transport over Security, it tells us that Application Data is encrypted, as seen in picture below.

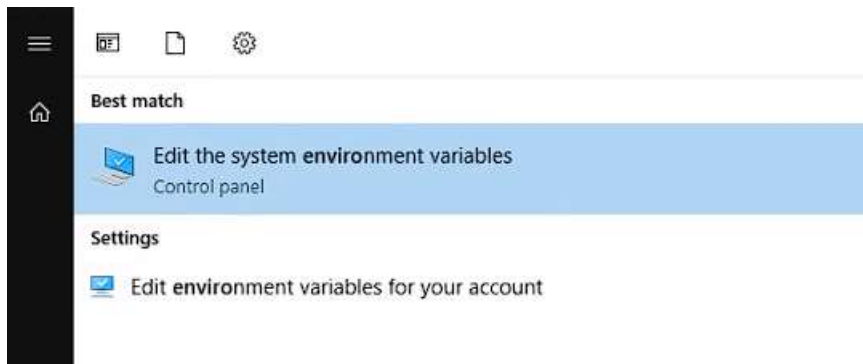


We won't be able to decrypt this data, as we don't have the private key used for encryption. That's why it's all secure. In http we would be able to see those details decrypted, such as username and password.

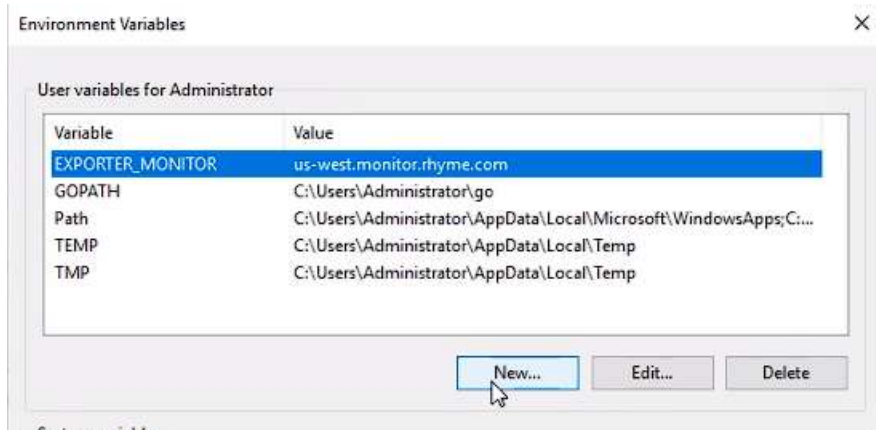
But sometimes we really need to decrypt encrypted traffic. For example, when you are trying to administer your network. So this is how we can decrypt the SSL traffic in Wireshark.

I will use a very simple way called Pre-Master Secret Key, where our browser which is the client, will generate this Pre-Master Secret Key and we will log it in an SSL key log file. Then this key is used by the server which is the webpage, to generate a Master Secret Key that encrypts all the traffic. And since we have this Pre-Master Secret Key logged in a file, we can give it to a Wireshark to decrypt the data with.

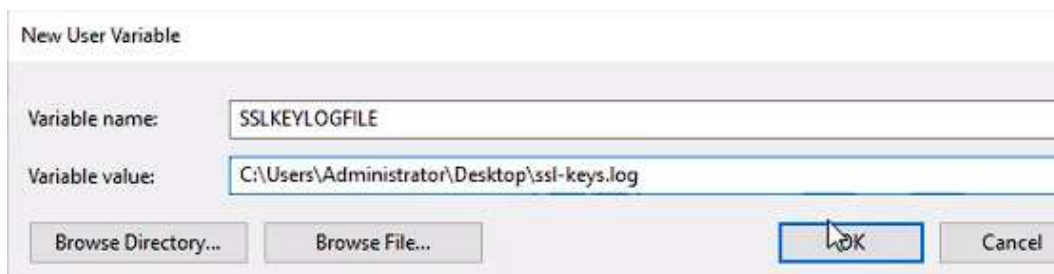
First, I will set an environment variable in Windows so that we can tell the browser to log this Pre-Master Secret Keys in a certain file.



After this, we click on **Environment Variables**. Then, option **New** as seen below.



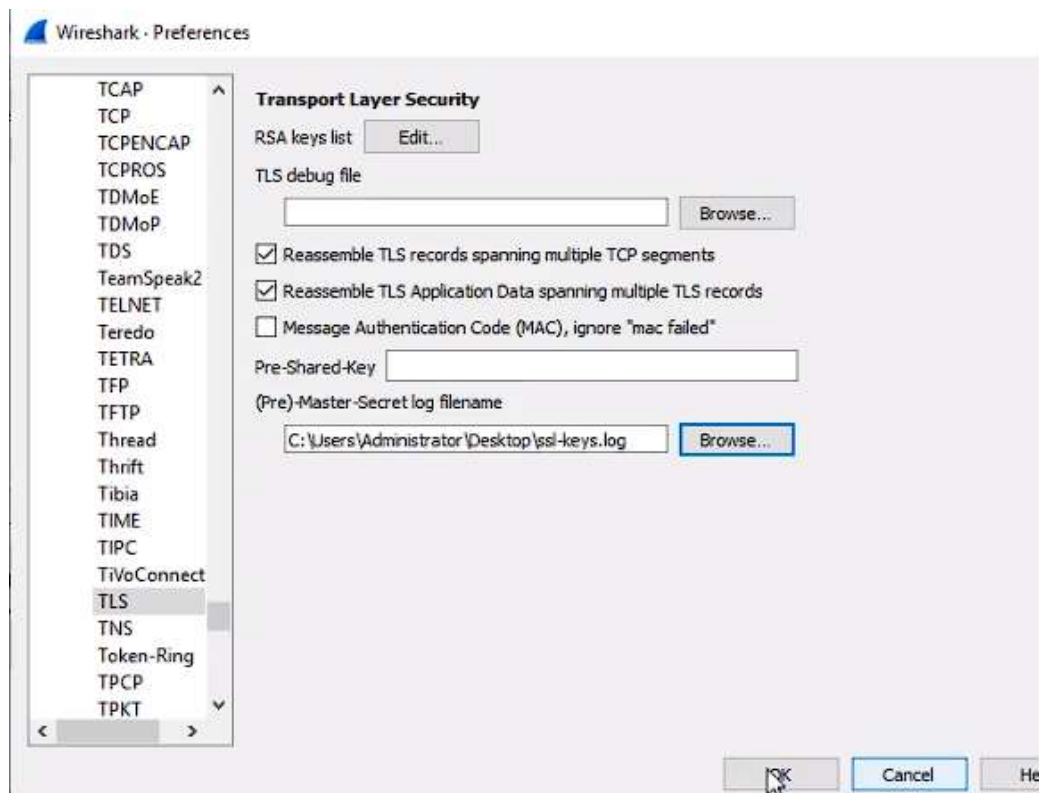
In Variable name, we put **SSLKEYLOGFILE**, and in Variable value, we should put the location the key will be logged in. I will put it on **Desktop** and also name of the file **ssl-keys.log**.



Now we have to tell Wireshark that we need to use that file.

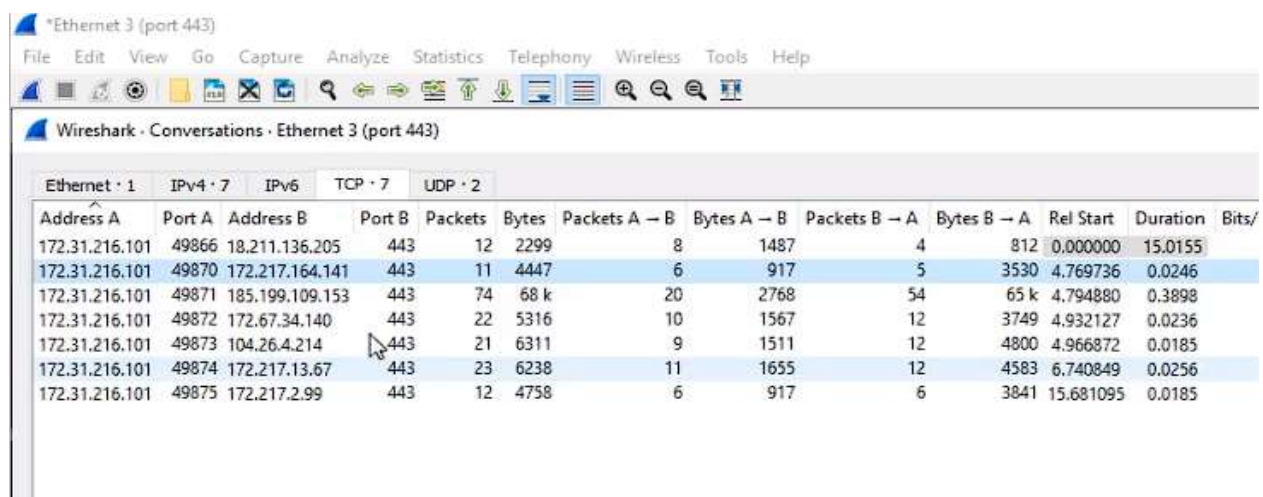
We go to **Edit -> Preferences -> Protocols -> TLS (or SSL)**.

There under **Pre-Master-Secret log filename**, we choose the file we just made.



In the last step, let's capture the HTTPS traffic and let's test it with a simple site.

In Wireshark, we start the capture on *port 443* -> **Ethernet 3**. Now we open some website, and stop the capture in Wireshark. Then we go **Statistics** -> **Conversations**, now we ping the server we connected to so we can check which conversation exactly that is.



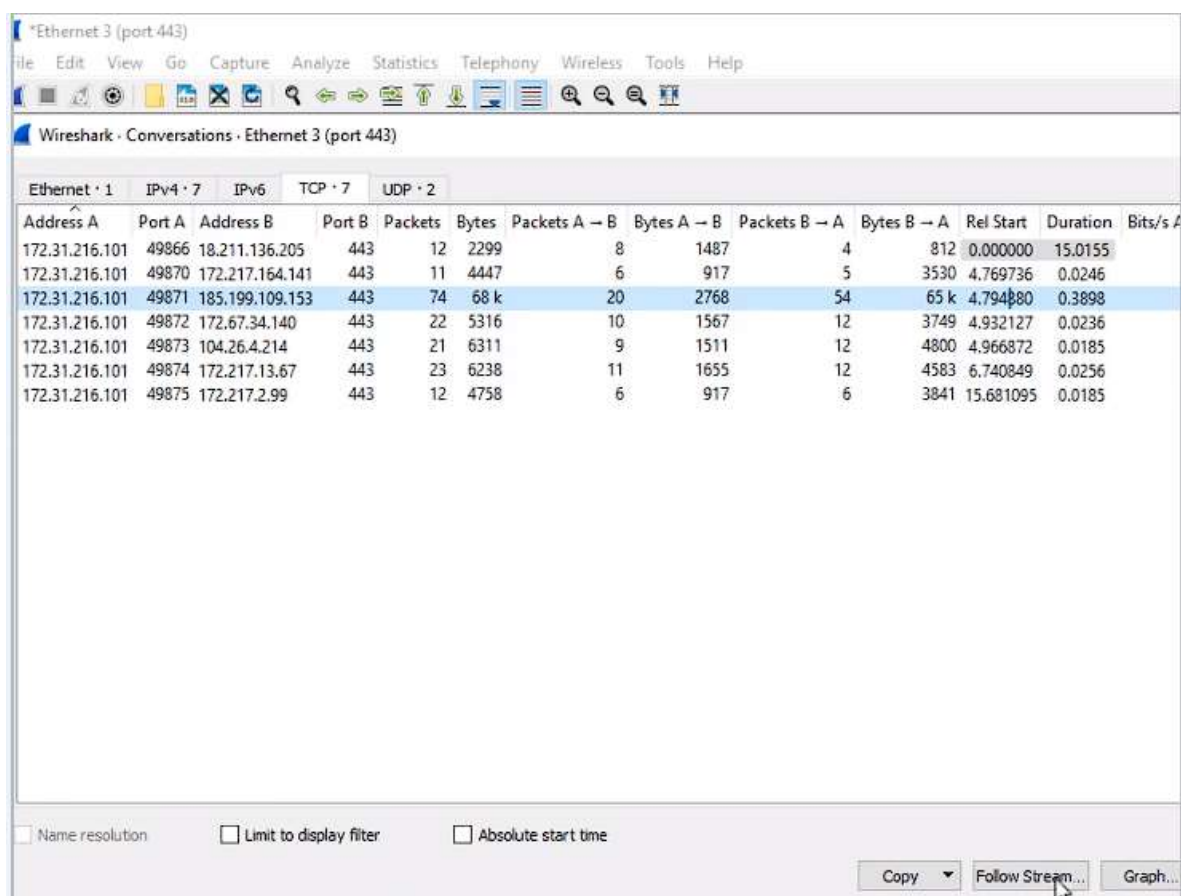
Then, we get the host name from the site that we opened and paste it to **Command Prompt**.

```
Administrator: Command Prompt
Microsoft Windows [Version 10.0.17763.1098]
(c) 2018 Microsoft Corporation. All rights reserved.

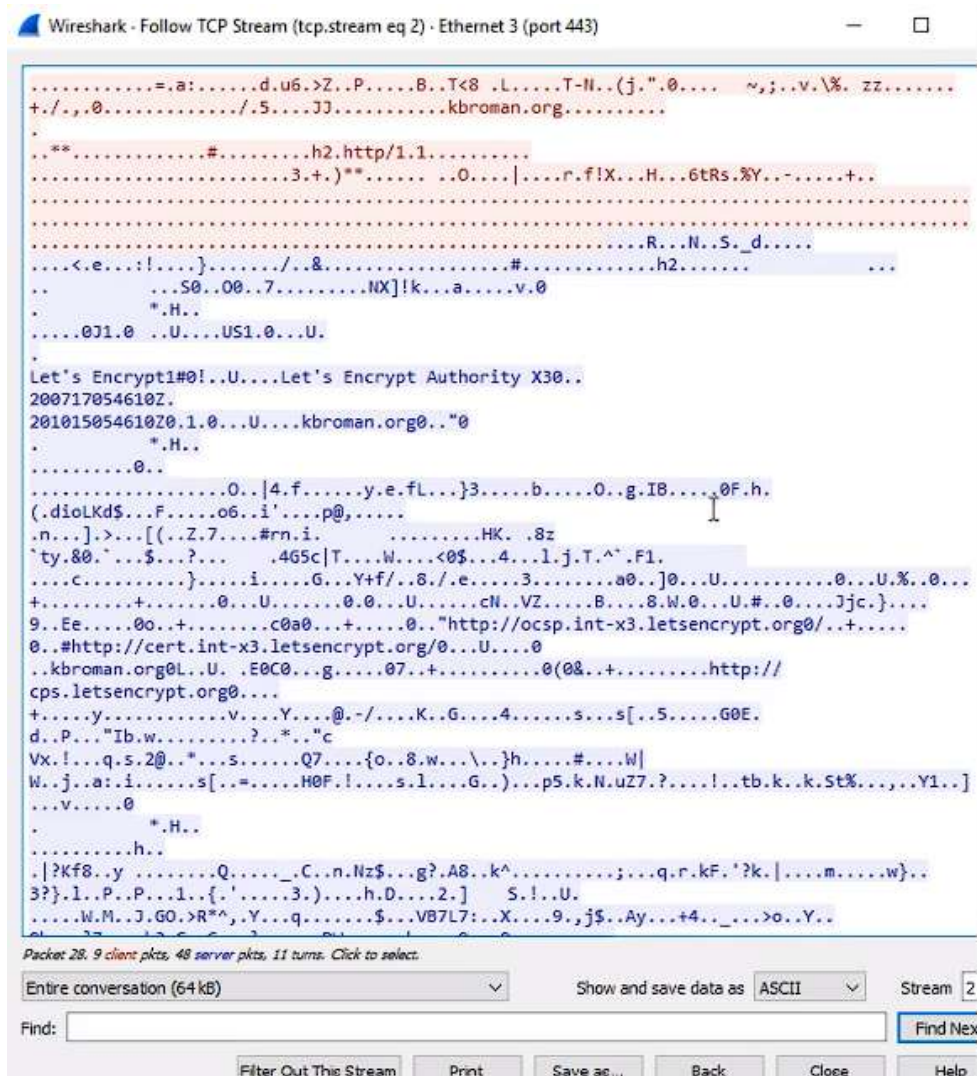
C:\Users\Administrator>ping kbroman.org

Pinging kbroman.org [185.199.111.153] with 32 bytes of data:
Control-C
^C
C:\Users\Administrator>_
```

We see that IP is 185.199.111.153, so we go back to Wireshark to search for that IP. When we find it, we click on it and then **Follow Stream**, as seen below.



What we get next is that everything is unencrypted.



The screenshot shows the Wireshark interface with the 'Follow TCP Stream' window open for 'Ethernet 3 (port 443)'. The stream is displayed in ASCII format, showing a series of unencrypted characters. The data includes a mix of lowercase letters, numbers, and special characters, some of which are highlighted in blue. The stream starts with a series of characters that appear to be a mix of random noise and some recognizable patterns like 'kbroman.org'. The interface at the bottom shows the 'Find' bar and various action buttons like 'Filter Out This Stream', 'Print', 'Save as...', 'Back', 'Close', and 'Help'.

```
.....=,a:.....d.u6.>Z..P.....B..T<8 .L.....T-N..(j..0.... ~,;..v.\%. zz.....
+./.,0...../5....JJ.....kbroman.org.....
.
.**.....#.....h2.http/1.1.....
.....3.+.)**.....O....|....r.f!X...H...6tRs.%Y..-.....+.
.....
.....R...N..S..d....
...<.e...!.....}/..&.....#.....h2.....
..
...S0..00..7.....NX]!k...a....v.0
.
..H..
.....0J1.0 ..U....US1.0...U.
.
Let's Encrypt1#0!..U....Let's Encrypt Authority X30..
200717054610Z.
201015054610Z0.1.0...U....kbroman.org0.."0
.
..H..
.....0..
.....0..[4.f.....y.e.fl...}3.....b.....0..g.IB.....0F.h.
(.diolKd$.F.....o6..i'....p@,....
.n...]>...[(..Z.7....#rn.i. ....HK. .8z
`ty.&0."..$.?... .465c|T....W....<0$...4...l.j.T.^".F1.
...C.....}.....i.....G...Y+f/.8./e.....3.....a0..]0...U.....0...U.%..0...
+.....+.....0...U.....0...U.....cN..VZ....B....8.W.0...U.#..0....Jjc.}....
9..Ee....0o..+.....c0a0..+.....0.."http://ocsp.int-x3.letsencrypt.org/..+....
0..#http://cert.int-x3.letsencrypt.org/0...U....0
..kbroman.org0L..U. .E0C0...g.....07..+.....0(0&..+.....http://
cps.letsencrypt.org0....
+.....y.....V....Y....@-/-...K..G...4.....s...s[.5....G0E.
d..P... "Ib.w.....?..*.."c
Vx.!...q.s.2@.."...s.....Q7....{o..8.w...\.}h....#....W|
W..j..a:i.....s[.=-.....H0F.!...s.l...G..).p5.k.N.uZ7.?....!..tb.k..k.St%....Y1..]
...v.....0
.
..H..
.....h..
..|?Kf8..y .....Q....._C..n.Nz$...g?.A8..k^.....;...q.r.kF.'?k.|....m....w}..
3?}.1..P..P..1..{.'.....3.)...h.D....2.] S.!..U.
.....W.M..J.GO.>R*^..Y...q.....$.VB7L7:..X...9.,j$.Ay...+4.._...>o..Y..
```

Even if we go back to where we see the packets, there is no Application Data anymore under column Info, you can also see HTTP2, what are the Headers, and everything.

*Ethernet 3 (port 443)

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

tcp.stream eq 2

No.	Time	Source	Destination	Protocol	Length	Info
42	4.805389	185.199.109.153	172.31.216.101	TCP	60	443 → 49871 [ACK] Seq=3821 Ack=1126 W:
44	4.883605	185.199.109.153	172.31.216.101	HTTP2	1514	HEADERS[1]: 200 OK
45	4.883605	185.199.109.153	172.31.216.101	HTTP2	1087	DATA[1] (text/html)
46	4.883694	172.31.216.101	185.199.109.153	TCP	54	49871 → 443 [ACK] Seq=1126 Ack=6314 W:
47	4.924920	172.31.216.101	185.199.109.153	HTTP2	221	HEADERS[3]: GET /simple_site/assets/tl
48	4.925305	172.31.216.101	185.199.109.153	HTTP2	148	HEADERS[5]: GET /simple_site/assets/tl
49	4.925694	172.31.216.101	185.199.109.153	HTTP2	144	HEADERS[7]: GET /simple_site/assets/tl
50	4.925829	185.199.109.153	172.31.216.101	TCP	60	443 → 49871 [ACK] Seq=6314 Ack=1387 W:
65	4.946395	172.31.216.101	185.199.109.153	TCP	144	[TCP Retransmission] 49871 → 443 [PSH
70	4.946010	185.199.109.153	172.31.216.101	TCP	60	443 → 49871 [ACK] Seq=6314 Ack=1477 W:

> Frame 23: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface \Device\NPF_{632B7902-F0FF-488D-9...}

> Ethernet II, Src: 0a:3d:58:a2:1a:61 (0a:3d:58:a2:1a:61), Dst: 0a:d3:9e:2f:df:eb (0a:d3:9e:2f:df:eb)

> Internet Protocol Version 4, Src: 172.31.216.101, Dst: 185.199.109.153

> Transmission Control Protocol, Src Port: 49871, Dst Port: 443, Seq: 0, Len: 0

In the beginning, it was only Application Data, now we can see everything, even the source code for the websites.