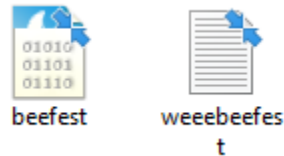


Network Forensic Challenge

Pada challenge ini, diberikan sebuah packet capture dan juga sebuah file text document.



Mari kita analisa file packet capture nya dengan wireshark.

The screenshot displays the Wireshark network protocol analyzer interface. The top menu bar includes File, Edit, View, Go, Capture, Analyze, Statistics, Telephony, Wireless, Tools, and Help. Below the menu is a toolbar with various icons for file operations, analysis, and visualization.

The main window shows a packet list on the left, a packet details pane in the center, and a packet bytes pane at the bottom. The packet list contains 26 packets, all of which are TLSv2. The selected packet is packet 7, which is a TLSv2 message (Seq=1 Ack=79 Win=257 Len=0). The packet details pane shows the following structure:

- Frame 10238:** 1128 bytes on wire (9024 bits), 1128 bytes captured (9024 bits) on interface Device\NPF_{1A6E4F7B-64BE-A46B-A7AB-8993470E670B}, id 0
- Ethernet II, Src: AzureNav_97:88:e5 (70:b6:00:55:97:88:e5), Dst: VmptNetc_98:75:d0 (d4:9a:a0:98:75:d0)**
- TLSv2**
 - Handshake (10), STREAM(0), HEADERS
 - Protected Payload (KPB), PKN: 8, STREAM(0), DATA
 - Protected Payload (KPB), PKN: 9, STREAM(0), PADDING

The packet bytes pane shows the raw data of the selected packet, which is a sequence of hexadecimal values representing the TLSv2 message structure.

Setelah dianalisa sebentar, ternyata tidak ada protocol HTTP dan juga semua data dari client ke server dan sebaliknya tidak ada yang jelas.

http						
No.	Time	Source	Destination	Protocol	Length	Info

The image shows a Wireshark packet capture of a TLSv1.3 handshake. The packet list on the left shows frames 9510 to 10221. The packet details pane on the right shows the TLSv1.3 Record Layer: Handshake Protocol: Client Hello. The packet bytes pane on the far right shows the raw data of the Client Hello message.

Nah, bisa dilihat disini bahwa ada percakapan yang terenkripsi antara client dan server, yaitu client berbicara dengan server dengan protocol website HTTPS atau HTTP Secure dengan enkripsi TLSv1.3.

The image shows a Wireshark packet capture of a TLSv1.3 handshake. The packet list on the left shows frame 8. The packet details pane on the right shows the TLSv1.3 Record Layer: Handshake Protocol: Client Hello. The packet bytes pane on the far right shows the raw data of the Client Hello message.

Juga percakapan yang terenkripsi tersebut datangnnya dari IP Address 192.168.1.4

Gathered Information

Sejauh ini, informasi yang kita dapat adalah :

>> IP Address yang dicurigai = 192.168.1.4

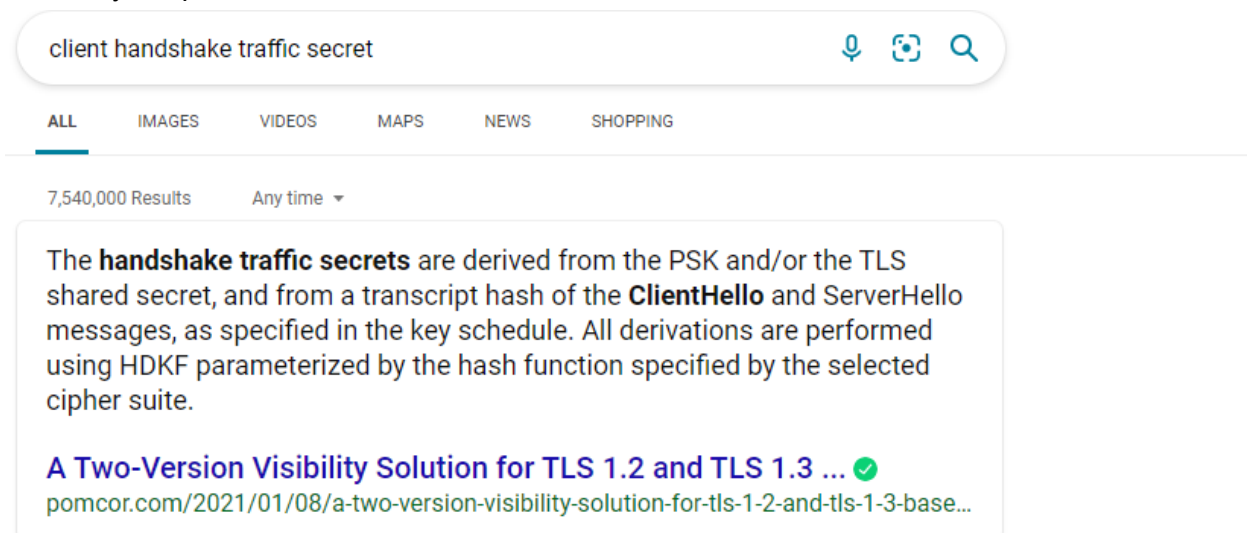
>> Protocol Cryptographic / Enkripsi Traffic nya = TLSv1.3

Selanjutnya, bisa di cek ke file kedua, yang text document.

Disitu ada banyak sekali kata abstrak, kecuali kata-kata terdepan.

```
CLIENT_HANDSHAKE_TRAFFIC_SECRET f95ff4f1f1c1433a8f344c862bbc4a6f37b6f849fa8dc75565a59ff93c37
SERVER_HANDSHAKE_TRAFFIC_SECRET f95ff4f1f1c1433a8f344c862bbc4a6f37b6f849fa8dc75565a59ff93c37
CLIENT_TRAFFIC_SECRET_0 f95ff4f1f1c1433a8f344c862bbc4a6f37b6f849fa8dc75565a59ff93c37fab6 68a
SERVER_TRAFFIC_SECRET_0 f95ff4f1f1c1433a8f344c862bbc4a6f37b6f849fa8dc75565a59ff93c37fab6 7c3l
EXPORTER_SECRET f95ff4f1f1c1433a8f344c862bbc4a6f37b6f849fa8dc75565a59ff93c37fab6 9d2a2c0a4ef
CLIENT_HANDSHAKE_TRAFFIC_SECRET b49d81715999369304edecb7eaba9294ecd0e7515b7b70cacba3acf78271a
SERVER_HANDSHAKE_TRAFFIC_SECRET b49d81715999369304edecb7eaba9294ecd0e7515b7b70cacba3acf78271a
CLIENT_TRAFFIC_SECRET_0 b49d81715999369304edecb7eaba9294ecd0e7515b7b70cacba3acf78271a815 6b4
SERVER_TRAFFIC_SECRET_0 b49d81715999369304edecb7eaba9294ecd0e7515b7b70cacba3acf78271a815 199f
EXPORTER_SECRET b49d81715999369304edecb7eaba9294ecd0e7515b7b70cacba3acf78271a815 7bf907e73e8
CLIENT_HANDSHAKE_TRAFFIC_SECRET 26b88884f753e9486bc1b4876b3fe78a02bd6f1ded423b2b0687cfd3bda9
SERVER_HANDSHAKE_TRAFFIC_SECRET 26b88884f753e9486bc1b4876b3fe78a02bd6f1ded423b2b0687cfd3bda9
CLIENT_TRAFFIC_SECRET_0 26b88884f753e9486bc1b4876b3fe78a02bd6f1ded423b2b0687cfd3bda94d91 8ee
SERVER_TRAFFIC_SECRET_0 26b88884f753e9486bc1b4876b3fe78a02bd6f1ded423b2b0687cfd3bda94d91 e51l
EXPORTER_SECRET 26b88884f753e9486bc1b4876b3fe78a02bd6f1ded423b2b0687cfd3bda94d91 90b5ee5e94f
CLIENT_RANDOM 65338a726792ff00f15316f860d716f2271a015a14d003a31aab86c5e8d8daae 1dc8ed88703df
CLIENT_HANDSHAKE_TRAFFIC_SECRET c85af6844088312a0df83b56360ed61c31e3213115b50db792f05ffc49cc
SERVER_HANDSHAKE_TRAFFIC_SECRET c85af6844088312a0df83b56360ed61c31e3213115b50db792f05ffc49cc
```

Jika bingung, bisa dicek di Google.
Contohnya seperti ini :



Setelah di search, kita juga bisa mengetahui suatu hal seperti **Client Hello** ataupun **Server Hello**

Dan mengapa IP Address yang dicurigai adalah 192.168.1.4?

Seperti yang sudah tertera, Client Hello berarti dari Client menyampaikan pesan atau request ke server atau sebuah endpoint, sedangkan Server Hello berarti mengirimkan pesan balik kepada Client.

Bisa dilihat pada gambar berikut :

11	0.018009	192.168.1.6	114.4.168.117	TCP	54	49776 → 443 [ACK] Seq=1 Ack=1 Win=66560 Len=0
12	0.018657	192.168.1.6	114.4.168.117	TLSv1.3	678	Client Hello
21	0.027925	114.4.168.117	192.168.1.6	TCP	54	443 → 49776 [ACK] Seq=1 Ack=625 Win=30464 Len=0
22	0.028372	114.4.168.117	192.168.1.6	TLSv1.3	324	Server Hello, Change Cipher Spec, Application Data, A
23	0.029374	192.168.1.6	114.4.168.117	TLSv1.3	134	Change Cipher Spec, Application Data

IP Address sebelah kiri ada source, destination di sebelah kanan
 ...dengan Client Hello ada pada source yaitu 192.168.1.4

Kalau begini, yang harus kita lakukan adalah **mendecrypt traffic HTTPS**.

(Untuk referensi, bisa dilihat pada link berikut)

[Wireshark Tutorial: Decrypting HTTPS Traffic \(Includes SSL and TLS\) \(paloaltonetworks.com\)](https://www.paloaltonetworks.com/wireshark/tutorial/decrypting-https-traffic-includes-ssl-and-tls/)

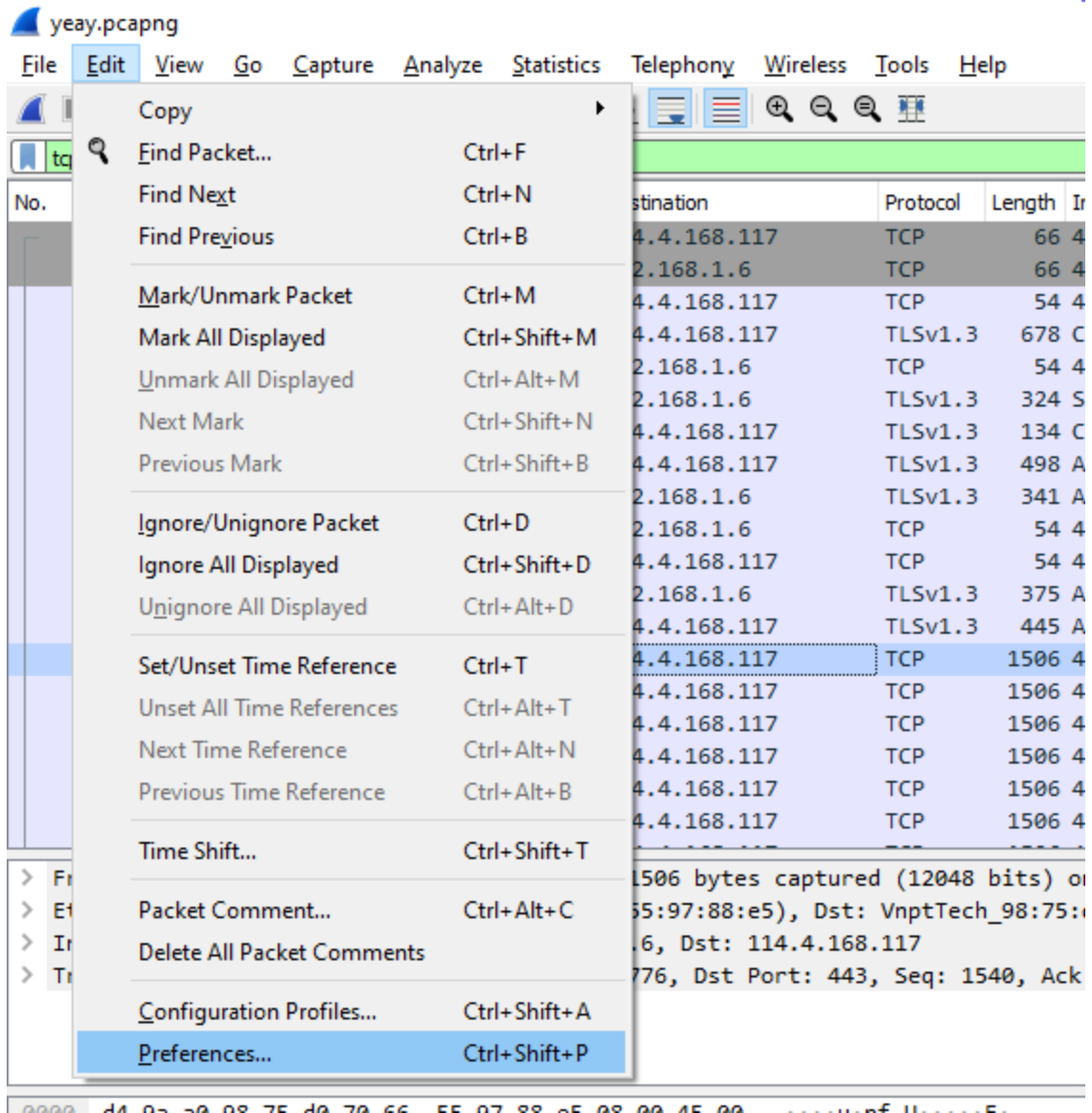
HTTPS Decryption

Dilihat dari link referensi sebelumnya, file document tersebut yang kita punya, berisikan SSL Key Log yang akan digunakan untuk mendecrypt TLS Encryptionnya. Tanpa adanya SSL Key Log terbuat selama proses packet capture dilakukan, maka traffic HTTPS yang berhasil dicapture pun akan menjadi sia-sia, kecuali hanya untuk melihat IP Address, Port, berapa banyak packet pada setiap send-data, dan sebagainya.

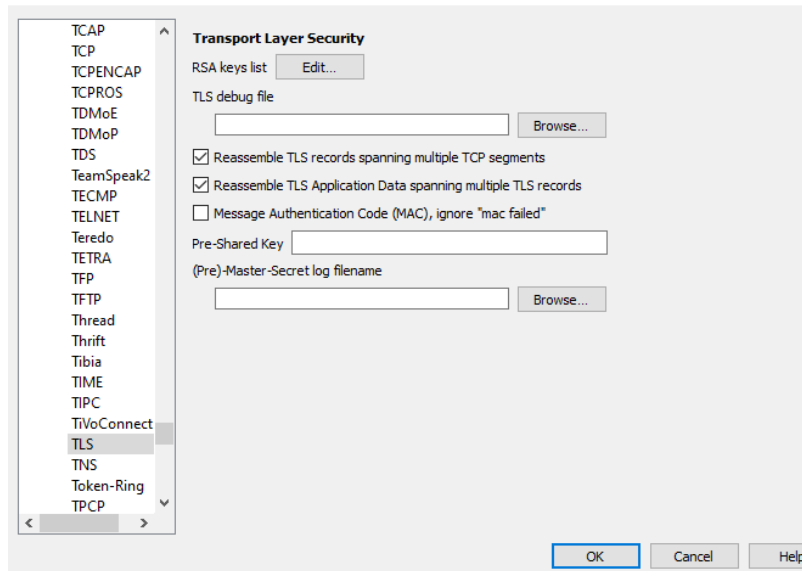
Karena yang terpenting adalah data-data apa saja yang bergerak dari Client ke Server dan sebaliknya yang terdapat pada header dan body dari website.

Berikut cara mendecrypt TLS dengan wireshark :

1. Ke Edit -> References



- Selanjutnya, pada Protocols klik tombol drop-down dan pilih TLS

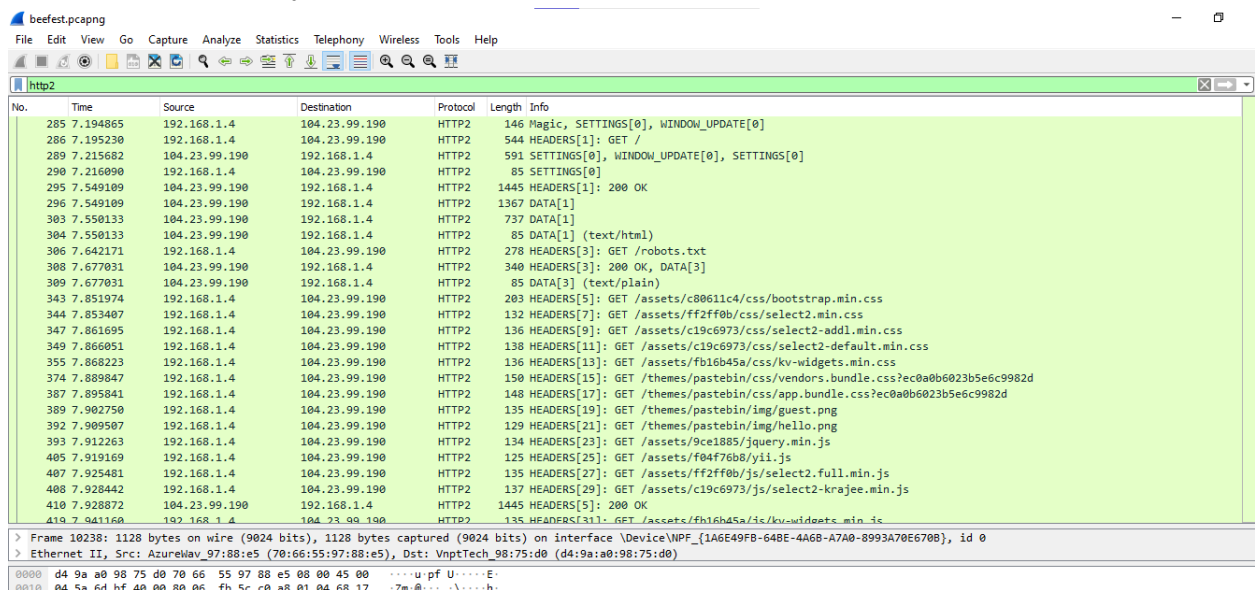


3. Kemudian klik browse dan masukkan SSL Key Log nya di bagian (Pre)-Master Secret log filename, yaitu file text berikut :



4. Setelah sudah, klik OK

Kemudian bisa di scroll-scroll dan sudah terlihat ada Traffic HTTP2 yang berarti HTTPS nya sudah berhasil di decrypt.



Lalu, bisa gunakan filter untuk mempercepat pencarian dengan syntax berikut :

>> http2 && ip.src_host == 192.168.1.4

Artinya, kita hanya akan menglist packet dengan protocol HTTP2 dan akan menempatkan IP Address 192.168.1.6 di kolom source.

Berikut hasilnya :

http2 && ip.src_host == 192.168.1.4					
No.	Time	Source	Destination	Protocol	Length Info
285	7.194865	192.168.1.4	104.23.99.190	HTTP2	146 Magic, SETTINGS[0], WINDOW_UPDATE[0]
286	7.195230	192.168.1.4	104.23.99.190	HTTP2	544 HEADERS[1]: GET /
290	7.216090	192.168.1.4	104.23.99.190	HTTP2	85 SETTINGS[0]
306	7.642171	192.168.1.4	104.23.99.190	HTTP2	278 HEADERS[3]: GET /robots.txt
343	7.851974	192.168.1.4	104.23.99.190	HTTP2	203 HEADERS[5]: GET /assets/c80611c4/css/bootstrap.min.css
344	7.853407	192.168.1.4	104.23.99.190	HTTP2	132 HEADERS[7]: GET /assets/ff2ff0b/css/select2.min.css
347	7.861695	192.168.1.4	104.23.99.190	HTTP2	136 HEADERS[9]: GET /assets/c19c6973/css/select2-addl.min.css
349	7.866051	192.168.1.4	104.23.99.190	HTTP2	138 HEADERS[11]: GET /assets/c19c6973/css/select2-default.min.css
355	7.868223	192.168.1.4	104.23.99.190	HTTP2	136 HEADERS[13]: GET /assets/fb16b45a/css/kv-widgets.min.css
374	7.889847	192.168.1.4	104.23.99.190	HTTP2	150 HEADERS[15]: GET /themes/pastebin/css/vendors.bundle.css?ec0a0b6023b5e6c9982d
387	7.895841	192.168.1.4	104.23.99.190	HTTP2	148 HEADERS[17]: GET /themes/pastebin/css/app.bundle.css?ec0a0b6023b5e6c9982d
389	7.902750	192.168.1.4	104.23.99.190	HTTP2	135 HEADERS[19]: GET /themes/pastebin/img/guest.png
392	7.909507	192.168.1.4	104.23.99.190	HTTP2	129 HEADERS[21]: GET /themes/pastebin/img/hello.png
393	7.912263	192.168.1.4	104.23.99.190	HTTP2	134 HEADERS[23]: GET /assets/9ce1885/jquery.min.js
405	7.919169	192.168.1.4	104.23.99.190	HTTP2	125 HEADERS[25]: GET /assets/f04f76b8/yii.js
407	7.925481	192.168.1.4	104.23.99.190	HTTP2	135 HEADERS[27]: GET /assets/ff2ff0b/js/select2.full.min.js
408	7.928442	192.168.1.4	104.23.99.190	HTTP2	137 HEADERS[29]: GET /assets/c19c6973/js/select2-krajee.min.js
419	7.941160	192.168.1.4	104.23.99.190	HTTP2	135 HEADERS[31]: GET /assets/fb16b45a/js/kv-widgets.min.js
420	7.942165	192.168.1.4	104.23.99.190	HTTP2	132 HEADERS[33]: GET /assets/f04f76b8/yii.activeForm.js
421	7.949540	192.168.1.4	104.23.99.190	HTTP2	150 HEADERS[35]: GET /themes/pastebin/js/vendors.bundle.js?ec0a0b6023b5e6c9982d
432	7.951158	192.168.1.4	104.23.99.190	HTTP2	147 HEADERS[37]: GET /themes/pastebin/js/app.bundle.js?ec0a0b6023b5e6c9982d
513	8.069758	192.168.1.4	104.26.14.238	HTTP2	146 Magic, SETTINGS[0], WINDOW_UPDATE[0]
514	8.070164	192.168.1.4	104.26.14.238	HTTP2	488 HEADERS[1]: GET /adv1/?q=adf050ece17b957604b4bbfc1829059f
562	8.120910	192.168.1.4	104.26.14.238	HTTP2	85 SETTINGS[0]

Dari sini, bisa terlihat langsung bahwa IP 192.168.1.4 mengunjungi **website pastebin.com**

```
GET /  
  
GET /robots.txt  
GET /assets/c80611c4/css/bootstrap.min.css  
GET /assets/ff2ff0b/css/select2.min.css  
GET /assets/c19c6973/css/select2-addl.min.css  
GET /assets/c19c6973/css/select2-default.min.css  
GET /assets/fb16b45a/css/kv-widgets.min.css  
GET /themes/pastebin/css/vendors.bundle.css?ec0a0b6023b5e6c9982d  
GET /themes/pastebin/css/app.bundle.css?ec0a0b6023b5e6c9982d  
GET /themes/pastebin/img/guest.png  
GET /themes/pastebin/img/hello.png  
GET /assets/9ce1885/jquery.min.js
```

Sekarang coba kita follow HTTP/2 Stream nya

Time	Source	Destination	Protocol	Length	Info
285	7.194865	192.168.1.4	104.23.99.190	HTTP2	146 Magic, SETTINGS[0], WINDOW_UPDATE[0]
286	7.195230	192.168.1.4	104.23.99.190	HTTP2	544 HEADERS[1]: GET /
290	7.216090	192.168.1.4	104.23.99.190	HTTP2	85 SETTINGS[0]
306	7.642171	192.168.1.4	104.23.99.190	HTTP2	278 HEADERS[3]: GET /rob
343	7.851974	192.168.1.4	104.23.99.190	HTTP2	203 HEADERS[5]: GET /ass
344	7.853407	192.168.1.4	104.23.99.190	HTTP2	132 HEADERS[7]: GET /ass
347	7.861695	192.168.1.4	104.23.99.190	HTTP2	136 HEADERS[9]: GET /ass
349	7.866051	192.168.1.4	104.23.99.190	HTTP2	138 HEADERS[11]: GET /ass
355	7.868223	192.168.1.4	104.23.99.190	HTTP2	136 HEADERS[13]: GET /ass
374	7.889847	192.168.1.4	104.23.99.190	HTTP2	150 HEADERS[15]: GET /thi
387	7.895841	192.168.1.4	104.23.99.190	HTTP2	148 HEADERS[17]: GET /thi
389	7.902750	192.168.1.4	104.23.99.190	HTTP2	135 HEADERS[19]: GET /thi
392	7.909507	192.168.1.4	104.23.99.190	HTTP2	129 HEADERS[21]: GET /thi
393	7.912263	192.168.1.4	104.23.99.190	HTTP2	134 HEADERS[23]: GET /ass
405	7.919169	192.168.1.4	104.23.99.190	HTTP2	125 HEADERS[25]: GET /ass
407	7.925481	192.168.1.4	104.23.99.190	HTTP2	135 HEADERS[27]: GET /ass
408	7.928442	192.168.1.4	104.23.99.190	HTTP2	137 HEADERS[29]: GET /ass
419	7.941160	192.168.1.4	104.23.99.190	HTTP2	135 HEADERS[31]: GET /ass
420	7.942165	192.168.1.4	104.23.99.190	HTTP2	132 HEADERS[33]: GET /ass
421	7.949540	192.168.1.4	104.23.99.190	HTTP2	150 HEADERS[35]: GET /thi
432	7.951158	192.168.1.4	104.23.99.190	HTTP2	147 HEADERS[37]: GET /thi
513	8.069758	192.168.1.4	104.26.14.238	HTTP2	146 Magic, SETTINGS[0],
514	8.070164	192.168.1.4	104.26.14.238	HTTP2	488 HEADERS[1]: GET /adv
562	8.120910	192.168.1.4	104.26.14.238	HTTP2	85 SETTINGS[0]
730	8.501116	192.168.1.4	52.54.154.179	HTTP2	153 Magic, SETTINGS[0],
731	8.501421	192.168.1.4	52.54.154.179	HTTP2	582 HEADERS[11]: POST /log/bu

Jika dilihat ke stream-stream selanjutnya, memang ada yang masih terenkripsi karena wireshark mungkin juga mengcapture beberapa iklan atau gambar atau video yang memang tidak human-readable yang ter-load pada website.

Tapi, karena streamnya belum habis, kita bisa lanjutkan pencarian apakah ada sesuatu yang menarik. Dan ternyata ada body HTTP yang human readable pada stream ke 87.

23.115238	192.168.1.4	104.23.99.190	HTTP2	671	HEADERS[87]: POST /
23.115824	192.168.1.4	104.23.99.190	HTTP2	1128	DATA[87]

Wireshark - Follow HTTP2 Stream (tcp.stream eq 4 and http2.streamid eq 87) - beefest.pcapng					
<pre> ..J.\$..W.....\..gX..~V.....j.bX..R..H.....,....{..(j.vz.{.....U..t.....@.AH..I'Z... 1.....'..O.f..@8..>/Q..Z..(..O~u..r.....8?h-----F-6F.?I.#hN.S.N.....T5~Fx...=.h...OP \$.V.RQ=.84..Jw...e.....S.....'..O.v.%.....h..}.C'.._.....m..BQ3{... ..J.y.....\$.;8.o.Z...[W.....Z#o.....<...>[G.....3..#z}.O...).i...}S...n.....W.....w~9...'.t.F.<...?~.l..h.....N..'. .o..o..2 8>.h.8....w1..+. '.....xx].~.....s.1a"M..&.K`..D.00.J.. ..k ..7 ?J..../... ..j..h2b..J...'.a.).9...A...O..l..?.....W-----WebKitFormBoundarybT7ArRCVineU5mIf Content-Disposition: form-data; name="_csrf-frontend" J5vazH08paQn_3IL-Ygp3gX67t2N-NfyYjEhLA2_av9tr0Lh8Bv16nOG1z0TunmkN7aMqqG3mKc1R1tOfQtQhAA== -----WebKitFormBoundarybT7ArRCVineU5mIf Content-Disposition: form-data; name="PostForm[text]" BeeFest{w0w_s0_y0u_kn0w_h0w_t0_d3crypt_HTTPS_4m4zinG!!} -----WebKitFormBoundarybT7ArRCVineU5mIf Content-Disposition: form-data; name="PostForm[format]" 1 -----WebKitFormBoundarybT7ArRCVineU5mIf Content-Disposition: form-data; name="PostForm[expiration]" N -----WebKitFormBoundarybT7ArRCVineU5mIf Content-Disposition: form-data; name="PostForm[status]" 0 -----WebKitFormBoundarybT7ArRCVineU5mIf Content-Disposition: form-data; name="PostForm[is_password_enabled]" 0 -----WebKitFormBoundarybT7ArRCVineU5mIf </pre>					
<pre> a a0 98 75 d0 70 66 55 97 88 e5 08 00 45 00u.pf U....E 1 6d bd 40 00 80 06 fd 27 c0 a8 01 04 68 17m @....'....h e fc 38 01 bb 33 be 3b 0e 16 34 38 1a 50 18 c..8..3..;..48.P f fb d2 00 00 17 03 03 02 64 4c 9b 86 77 34dL..w4 9 a3 99 68 36 60 54 40 9d 37 b6 35 e5 04 25h6'T @.7.5.% </pre>					

Jika dilihat dengan teliti, maka didapatkan Flagnya.

Flag : BeeFest{w0w_s0_y0u_kn0w_h0w_t0_d3crypt_HTTPS_4m4zinG!!}

Challenge selesai

=====