

Encryption Report; Google.com

Introduction

This report analyzes network traffic sent to Google.com using Wireshark on a Kali Linux virtual machine. The analysis focused on timestamps, IP addresses, port numbers, and encryption status of the packets. The findings confirm all traffic used HTTPS (port 443) and TLSv1.3 encryption for secure communication.

Methodology

A. Tool used

Wireshark: Network protocol analyzer

Mozilla Browser: Web browser for generating traffic

Kali Linux Virtual Machine: Secure environment for analysis

B. Procedure Followed:

Step 1: Launch Wireshark on Kali Linux VM.

2. Select the Network Interface:

On the input page, I selected the network interface to connect to the internet (in this instance, Ethernet, Wi-Fi). I ensured that the Promiscuous mode on all interfaces was enabled. On output, I created a permanent file to capture for future reference.

Step 2: Start Capturing Traffic:

Click on the `Start` button to begin capturing packets.

Step 3: Initiate the HTTPS Connection

1. Open a Web Browser:

For this analysis, Mozilla Firefox was used.

2. Visit Google:

Navigated to <https://www.google.com>.

Step 4: Stop Capturing Traffic

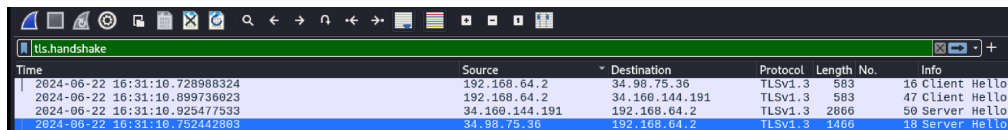
- After the page had loaded, I returned to Wireshark and clicked on the `Stop` button to stop capturing packets.

Step 5: Filter and Analyze the TLS Handshake

1. Apply Display Filter:

- the display filter was used to isolate the TLS handshake packets. In the Wireshark filter bar, the keyword 'tls.Handshake' was used to filter the tls handshake.

2. Result of the filtered packets URL:



The image shows a Wireshark packet capture window with the display filter 'tls.handshake'. The packet list shows four packets, all of which are TLSv1.3 handshake messages. The first two are 'Client Hello' messages and the next two are 'Server Hello' messages. The packet details pane shows the structure of the first 'Client Hello' packet, including the handshake type, version, random, session ID, cipher suites, and extensions.

Time	Source	Destination	Protocol	Length	No.	Info
2024-06-22 16:31:10.728988324	192.168.64.2	34.98.75.36	TLSv1.3	583	16	Client Hello
2024-06-22 16:31:10.899736023	192.168.64.2	34.100.144.191	TLSv1.3	583	47	Client Hello
2024-06-22 16:31:10.925477533	34.100.144.191	192.168.64.2	TLSv1.3	2866	50	Server Hello
2024-06-22 16:31:10.752442803	34.98.75.36	192.168.64.2	TLSv1.3	1466	18	Server Hello

- Client Hello (SNI=classify-client.service.mozilla.com)
- Server Hello, Change Cipher Spec
- Client Hello(SNI=content-signature-2.cdn.mozilla.net)
- Server Hello, Change Cipher Spec

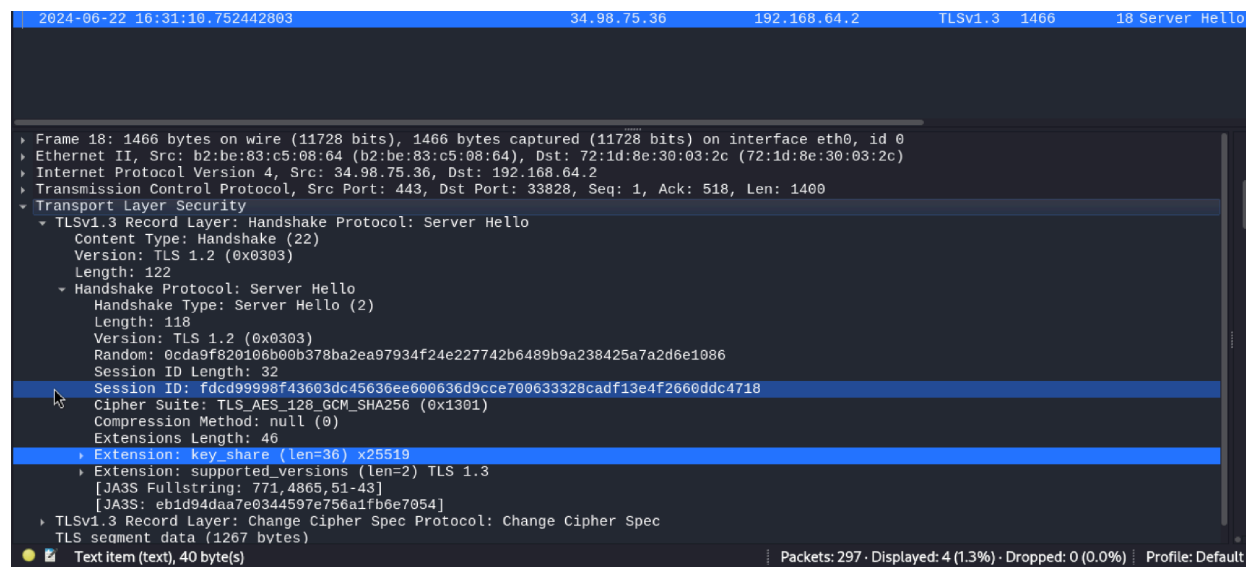
TLS Handshake Analysis

- ClientHello Message

```
▼ TLSv1.3 Record Layer: Handshake Protocol: Client Hello
  Content Type: Handshake (22)
  Version: TLS 1.0 (0x0301)
  Length: 512
  ▼ Handshake Protocol: Client Hello
    Handshake Type: Client Hello (1)
    Length: 508
    Version: TLS 1.2 (0x0303)
    Random: a3d0cc00f6b41ddd8792e0b872a630a3f31b1963d0f5a21534d10e74a7cf6bd1
    Session ID Length: 32
    Session ID: fdc99998f43603dc45636ee600636d9cce700633328cadf13e4f2660ddc4718
    Cipher Suites Length: 34
    ► Cipher Suites (17 suites)
    Compression Methods Length: 1
    ► Compression Methods (1 method)
    Extensions Length: 401
    ► Extension: server_name (len=41) name=classify-client.services.mozilla.com
    ► Extension: extended_master_secret (len=0)
    ► Extension: renegotiation_info (len=1)
    ► Extension: supported_groups (len=14)
    ► Extension: ec_point_formats (len=2)
    ► Extension: session_ticket (len=0)
    ► Extension: application_layer_protocol_negotiation (len=14)
    ► Extension: status_request (len=5)
```

In the TLS handshake, the ClientHello message acts as an initiation and introduction. Here's a breakdown of its purpose and significance:

- **Purpose:**
 - Initiates the secure communication process between a client (your browser) and a server (website).
 - Informs the server about the client's capabilities for secure communication.
- **Significance:**
 - Establishes a foundation for secure communication by initiating the negotiation process.
 - Allows the server to choose the most compatible and secure encryption settings from the options offered by the client.
 - Contains essential information like:
 - Encrypted: Yes (TLS handshake initiated)
 - Supported TLS versions (e.g., TLSv1.3)
 - Preferred cipher suites (combinations of encryption algorithms)
 - A random number used to generate encryption keys
- **ServerHello Message**



Purpose:

- Respond to the client's initiation (ClientHello message) and acknowledge the request for a secure connection.
- Informs the client about the server's chosen settings for secure communication.

Significance:

- Completes the initial negotiation phase by confirming the chosen encryption parameters.

- Establishes a foundation for secure communication by agreeing on the specific algorithms and keys used for encryption.
 - Contains essential information like:
 - - Encrypted: Yes (TLS handshake response)
 - The selected TLS version (e.g., TLSv1.3) - must be compatible with what the client offered.
 - Chosen cipher suite (encryption algorithms) - picked from the options proposed by the client based on compatibility and server policy.
 - The server's random number - it is used along with the client's random number to generate encryption keys for the session.
 - Server certificate (optional): If server authentication is required, the certificate containing the server's public key is sent for client verification.
-
- **Key Exchange Message**
 -
 - Purpose and Significance
 - Explain the purpose and significance of the ServerHello message in the TLS handshake process.

Packet Details

A. Source and Destination Information

1. ClientHello Packet
 - Timestamp: 2024-06-22 16:31:10.728988324
 - - Source IP: `192.168.64.2`
 - - Destination IP: `34.98.75.36`
 - - Port: `443`
2. ServerHello Packet
 - Timestamp: 2024-06-22 16:31:10.899736023
 - Source IP: `34.98.75.36`
 - Destination IP: 192.168.64.2`
 - Port: 443
3. Key Exchange Packet
 - Source IP:
 - Source Port:
 - Destination IP:
 - Destination Port:

B. Encryption Verification

Wireshark - Packet 66 - google.packets.pcapng

▶ Frame 66: 1294 bytes on wire (10352 bits), 1294 bytes captured (10352 bits) on interface eth0, id 0
▶ Ethernet II, Src: b2:be:83:c5:08:64 (b2:be:83:c5:08:64), Dst: 72:1d:8e:30:03:2c (72:1d:8e:30:03:2c)
▶ Internet Protocol Version 4, Src: 34.160.144.191, Dst: 192.168.64.2
▶ Transmission Control Protocol, Src Port: 443, Dst Port: 58746, Seq: 8240, Ack: 1013, Len: 1228
▼ Transport Layer Security
 ▼ TLSv1.3 Record Layer: Application Data Protocol: Hypertext Transfer Protocol
 Opaque Type: Application Data (23)
 Version: TLS 1.2 (0x0303)
 Length: 1223
 Encrypted Application Data [truncated]: f8a000daba371fc152e03102b86b7cb6d6a22c9340d656e091023da8ebc8bf1
 [Application Data Protocol: Hypertext Transfer Protocol]

0000	72 1d 8e 30 03 2c b2 be	83 c5 08 64 08 00 45 00	r . . 0 . , d . E .
0010	05 00 3e 68 00 00 78 06	4b 86 22 a0 90 bf c0 a8	. . > h . x . K . "
0020	40 02 01 bb e5 7a 44 f9	e1 22 bc 35 c1 f5 80 18	@ z D . . " . 5
0030	01 0d 85 2f 00 00 01 01	08 0a 9e 86 2f 07 cd eb	. . . / / . . .
0040	21 a5 17 03 03 04 c7 f8	a0 00 da ba 37 1f c1 52	! 7 . . R
0050	e0 31 02 b8 6b 7c b6 d6	a2 2c 93 40 d6 56 e0 91	. 1 . . k , . . @ . V . .
0060	02 3d a8 eb c8 bf 13 c9	16 40 71 db 27 8c 0a af	. = @ q . '
0070	09 8d 0f 87 0f c8 39 e1	c2 15 5e f2 ec 4a 17 05 9 ^ . . J . .
0080	a0 97 06 a5 8b 8f 33 d0	65 ff e0 ba 95 b1 f6 c2 3 . . e
0090	37 ba d9 8c cc 8f ce 0f	c4 09 b4 86 30 12 83 7c	7 0 . .
00a0	d5 92 96 d3 d5 96 73 ec	10 a7 2a c0 4e a6 3f 60 s . . . * . N . ? .
00b0	c8 9d 20 d8 9b 9e 1b 9f	6a 7e 21 8b 66 10 d5 23 j ~ ! . f . . #
00c0	aa 76 de 15 16 ab 2f f7	8c e3 e5 ad eb bb a4 15	. v /
00d0	66 53 a6 a6 4b 7c 70 19	66 2c 12 d4 e1 86 5f 29	f S . . K p . f , _)
00e0	aa 0c c4 3a c4 37 ff c1	76 d2 52 c4 4e 20 e2 f1 7 . . v . R . N . .

Time: 2024-06-22 16:31:10.959604798 · Source: 34.160.144.191 · Destination: 192.168.64.2 · Protocol: TLSv1.3 · Length: 1294 · No.: 66 · Info: Application Data

✓ Show packet bytes

× Close Help