

Principles of Information Security

HW4 Using Wireshark



Anonymous

日期: 2024/03/27

2024-2025 春 学期

Table of Contents

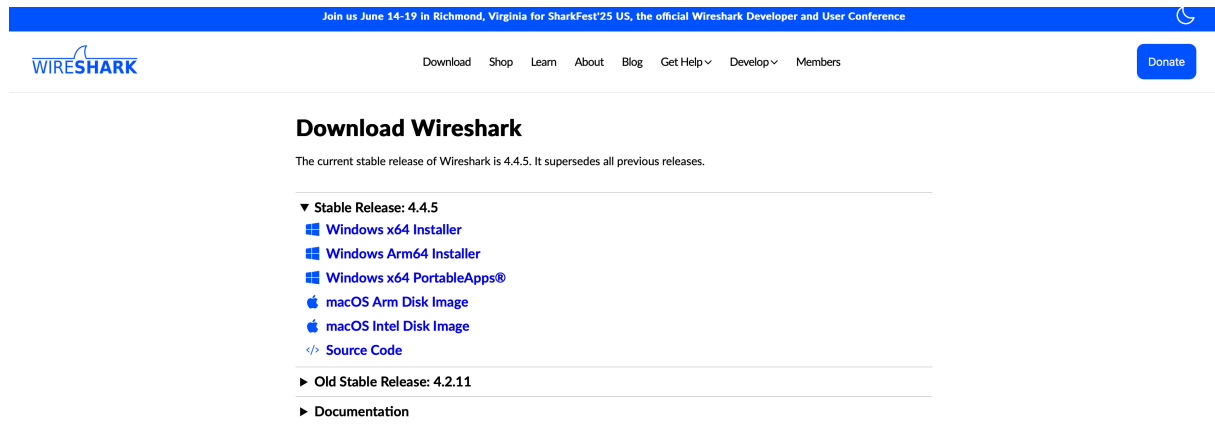
1	Download & Install Wireshark	3
1.1	Download Wireshark	3
1.2	Install Wireshark	3
2	Capture all packages when you access https://www.zju.edu.cn (using filter)	4
2.1	Get the IP Address of https://www.zju.edu.cn	4
2.2	Capture Packages when accessing the website	5
3	Using Wireshark to analyze the packets	5
3.1	Analyzing TCP 3-Way Handshake	6
3.1.1	First Stage: SYN (Synchronize) -> Client -> Server	6
3.1.2	Second Stage: SYN-ACK (Synchronize-Acknowledge) -> Server -> Client	7
3.1.3	Third Stage: ACK (Acknowledge) -> Client -> Server	7
3.2	Examining HTTPS Request/Response	8
3.2.1	TLS Handshake Step 1: Client Hello	8
3.2.2	TLS Handshake Step 2: Server Hello	9
3.2.3	TLS Handshake Step 3: Key Exchange & Encryption Setup	9
3.3	Analyzing TCP Connection Termination (4-Way Handshake)	10

1 Download & Install Wireshark

1.1 Download Wireshark

- Operating System: MacOS

Download from the official website of Wireshark. According to your specific operating system, choose different versions to download:



For MacOS users, simply click MacOS Arm Disk Image, and then a dmg file will be downloaded.

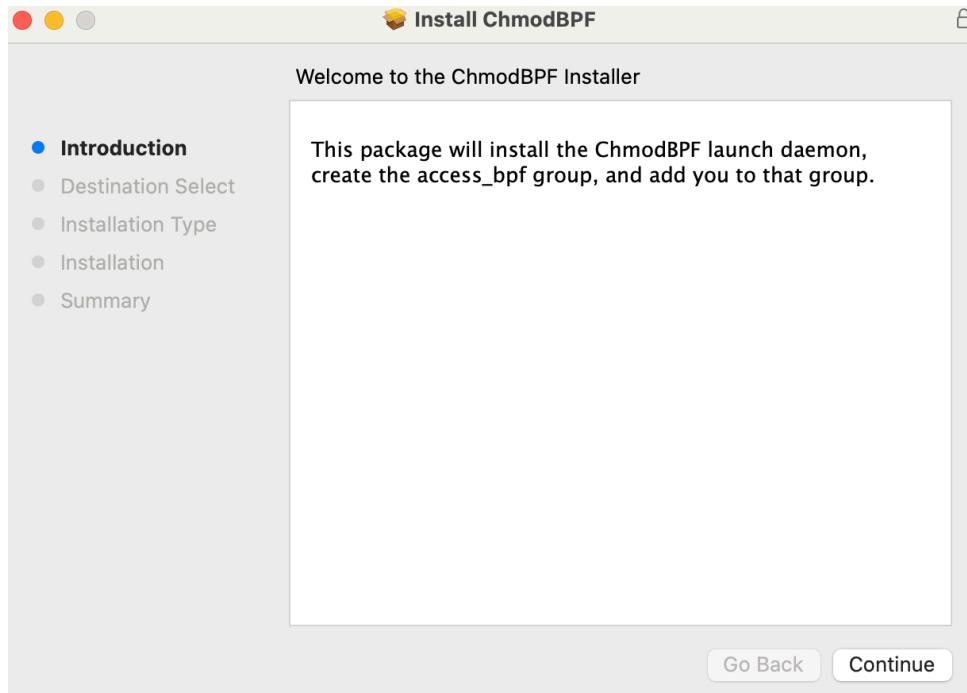
1.2 Install Wireshark

After successfully downloading the dmg file, double click to start the installation, an interface like the following will appear:



1. Drag the icon Wireshark.app to the folder Application.
2. Double click Install ChmodBPF.pkg to install ChmodBPF. ChmodBPF is a utility associated with Wireshark on macOS. It is used to manage permissions for capturing network packets.

Follow the instructions of Install ChmodBPF, and the installation process will finish successfully.



2 Capture all packages when you access https://www.zju.edu.cn (using filter)

2.1 Get the IP Address of https://www.zju.edu.cn

Open the terminal (zsh for instance), input ping www.zju.edu.cn and press enter.

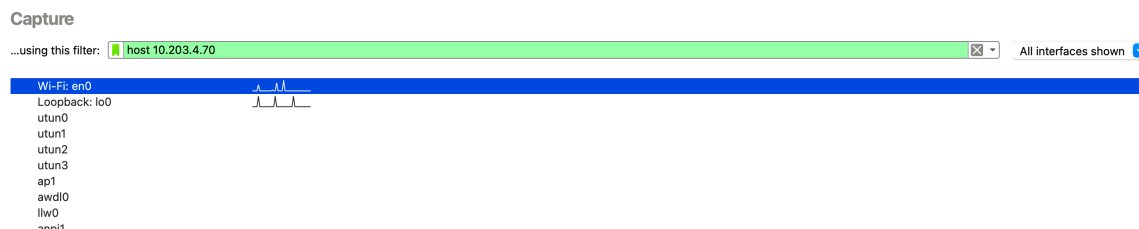
Result like the following will be acquired:

```
❯ ~/ ping www.zju.edu.cn
PING www.zju.edu.cn (10.203.4.70): 56 data bytes
64 bytes from 10.203.4.70: icmp_seq=0 ttl=60 time=18.706 ms
64 bytes from 10.203.4.70: icmp_seq=1 ttl=60 time=13.100 ms
64 bytes from 10.203.4.70: icmp_seq=2 ttl=60 time=17.235 ms
64 bytes from 10.203.4.70: icmp_seq=3 ttl=60 time=17.862 ms
64 bytes from 10.203.4.70: icmp_seq=4 ttl=60 time=18.854 ms
64 bytes from 10.203.4.70: icmp_seq=5 ttl=60 time=18.222 ms
64 bytes from 10.203.4.70: icmp_seq=6 ttl=60 time=13.622 ms
64 bytes from 10.203.4.70: icmp_seq=7 ttl=60 time=13.365 ms
64 bytes from 10.203.4.70: icmp_seq=8 ttl=60 time=15.261 ms
^C
--- www.zju.edu.cn ping statistics ---
9 packets transmitted, 9 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 13.100/16.247/18.854/2.271 ms
```

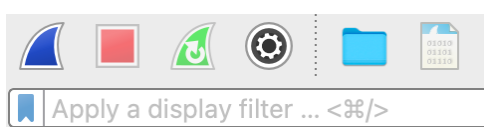
Notice that there is an IP address 10.203.4.70, remember the IP address which will be used in Wireshark app to capture packages.

2.2 Capture Packages when accessing the website

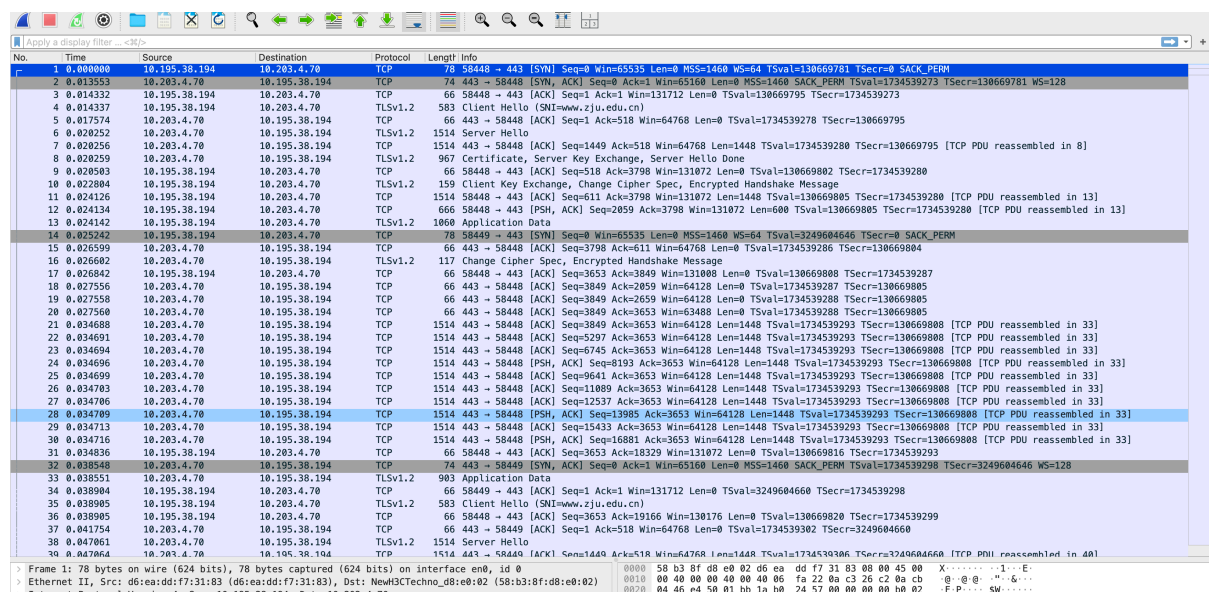
Open the Wireshark app, input the filter condition host 10.203.4.70 and select Wi-Fi: en0.



Press the blue shark fin on the left upper corner to start capturing:



Then open the website <https://www.zju.edu.cn> in the browser (Google Chrome for instance), Wireshark will capture all the packages when accessing the website:



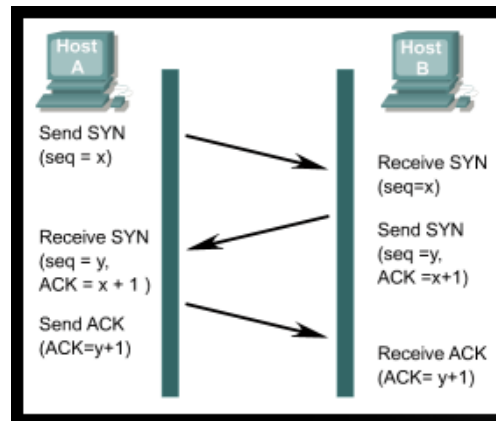
Click the red square button to stop capturing, and then click the form-like button to save the captured file.

3 Using Wireshark to analyze the packets

Http connection is based on TCP, so in order to construct an HTTP connection, TCP connection has to be established first.

3.1 Analyzing TCP 3-Way Handshake

The establishment of TCP can be concluded as 3-way handshake:



In order to find the packets corresponding to 3-way handshake, simply look for three consecutive packets with the following pattern:

1. SYN: Client → Server with [SYN] flag
2. SYN-ACK: Server → Client with [SYN, ACK] flags
3. ACK: Client → Server with [ACK] flag

As shown in the following:

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	10.195.38.194	10.203.4.70	TCP	78	58253 → 443 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=64 TSval=3470561282 TSecr=0 SACK_PERM
2	0.016902	10.203.4.70	10.195.38.194	TCP	74	443 → 58253 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM TSval=1726241421 TSecr=3470561282 WS=128
3	0.017246	10.195.38.194	10.203.4.70	TCP	66	58253 → 443 [ACK] Seq=1 Ack=1 Win=131712 Len=0 TSval=3470561300 TSecr=1726241421

3.1.1 First Stage: SYN (Synchronize) → Client → Server

The client sends a SYN packet to the server, indicating it wants to establish a connection. The packet includes:

- SYN flag = 1 (indicating a new connection request)
- A randomly generated initial sequence number (ISN) (e.g., Seq=0)

```
> Frame 1: 78 bytes on wire (624 bits), 78 bytes captured (624 bits) on interface en0, id 0
> Ethernet II, Src: d6:ea:dd:f7:31:83 (d6:ea:dd:f7:31:83), Dst: NewH3CTechno_d8:e0:02 (58:b3:8f:d8:e0:02)
> Internet Protocol Version 4, Src: 10.195.38.194, Dst: 10.203.4.70
< Transmission Control Protocol, Src Port: 58448, Dst Port: 443, Seq: 0, Len: 0
  Source Port: 58448
  Destination Port: 443
  [Stream index: 0]
  [Stream Packet Number: 1]
  [Conversation completeness: Complete, WITH_DATA (63)]
  [TCP Segment Len: 0]
  Sequence Number: 0 (relative sequence number)
  Sequence Number (raw): 447751255
  [Next Sequence Number: 1 (relative sequence number)]
  Acknowledgment Number: 0
  Acknowledgment number (raw): 0
  1011 .... = Header Length: 44 bytes (11)
  < Flags: 0x002 (SYN)
    000. .... = Reserved: Not set
    ...0 .... = Accurate ECN: Not set
    ....0... = Congestion Window Reduced: Not set
    ....0... = ECN-Echo: Not set
    ....0... = Urgent: Not set
    ....0... = Acknowledgment: Not set
    ....0... = Push: Not set
    ....0... = Reset: Not set
  > ....0...1. = Syn: Set
  ....0...0 = Fin: Not set
  [TCP Flags: .....S.]
  Window: 65535
  [Calculated window size: 65535]
  Checksum: 0xecb4 [unverified]
  [Checksum Status: Unverified]
  Urgent Pointer: 0
  > Options: (24 bytes), Maximum segment size, No-Operation (NOP), Window scale, No-Operation (NOP), No-0...
  > [Timestamps]
```

As is shown in the figure above, SYN flag is set to 1 and the randomly generated number is 447751255, while the source port is 58448, the destination port is 443.

3.1.2 Second Stage: SYN-ACK (Synchronize-Acknowledge) -> Server -> Client

The server acknowledges the client's request by sending:

- SYN flag = 1 (its own synchronization request)
- ACK flag = 1 (acknowledging the client's SYN)
- Acknowledgment number = Client's ISN + 1 (e.g., Ack=1)
- Its own initial sequence number (ISN) (e.g., Seq=0)

The image shows a Wireshark packet capture of a SYN-ACK packet. The packet list on the left shows 'Frame 2: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface en0, id 0'. The packet details pane on the right shows the following information:

- Ethernet II, Src: NewH3CTechno_d8:e0:02 (58:b3:8f:d8:e0:02), Dst: d6:ea:dd:f7:31:83 (d6:ea:dd:f7:31:83)
- Internet Protocol Version 4, Src: 10.203.4.70, Dst: 10.195.38.194
- Transmission Control Protocol, Src Port: 443, Dst Port: 58448, Seq: 0, Ack: 1, Len: 0
- Source Port: 443
- Destination Port: 58448
- [Stream index: 0]
- [Stream Packet Number: 2]
- [Conversation completeness: Complete, WITH_DATA (63)]
- [TCP Segment Len: 0]
- Sequence Number: 0 (relative sequence number)
- Sequence Number (raw): 1037957615
- [Next Sequence Number: 1 (relative sequence number)]
- Acknowledgment Number: 1 (relative ack number)
- Acknowledgment number (raw): 447751256
- 1010 = Header Length: 40 bytes (10)
- Flags: 0x012 (SYN, ACK)
- 000. = Reserved: Not set
- ...0 = Accurate ECN: Not set
- 0... = Congestion Window Reduced: Not set
-0.. = ECN-Echo: Not set
-0. = Urgent: Not set
-1 = Acknowledgment: Set
-0 = Push: Not set
-0.. = Reset: Not set
-1. = Syn: Set
-0 = Fin: Not set
- [TCP Flags:A..S.]
- Window: 65160
- [Calculated window size: 65160]
- Checksum: 0x67e5 [unverified]
- [Checksum Status: Unverified]
- Urgent Pointer: 0
- Options: (20 bytes), Maximum segment size, SACK permitted, Timestamps, No-Operation (NOP), Window scale
- [Timestamps]
- [SEQ/ACK analysis]

The source port is 443, the destination port is 58448, reverse of the previous one. As is shown, SYN flag is set to 1, so does ACK. Simultaneously, Acknowledge number (447751256) is set equal to Client's ISN (447751255) + 1, and it generated its own ISN 1037957615.

3.1.3 Third Stage: ACK (Acknowledge) -> Client -> Server

- The client confirms the server's SYN-ACK by sending:
 1. ACK flag = 1 (acknowledging the server's SYN)
 2. Acknowledgment number = Server's ISN + 1 (e.g., Ack=1)

```

> Frame 3: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface en0, id 0
> Ethernet II, Src: d6:ea:dd:f7:31:83 (d6:ea:dd:f7:31:83), Dst: NewH3CTechno_d8:e0:02 (58:b3:8f:d8:e0:02)
> Internet Protocol Version 4, Src: 10.195.38.194, Dst: 10.203.4.70
< Transmission Control Protocol, Src Port: 58448, Dst Port: 443, Seq: 1, Ack: 1, Len: 0
  Source Port: 58448
  Destination Port: 443
  [Stream index: 0]
  [Stream Packet Number: 3]
  [Conversation completeness: Complete, WITH_DATA (63)]
  [TCP Segment Len: 0]
  Sequence Number: 1 (relative sequence number)
  Sequence Number (raw): 447751256
  [Next Sequence Number: 1 (relative sequence number)]
  Acknowledgment Number: 1 (relative ack number)
  Acknowledgment number (raw): 1037957616
  1000 .... = Header Length: 32 bytes (8)
  < Flags: 0x010 (ACK)
    000. .... = Reserved: Not set
    ...0 .... = Accurate ECN: Not set
    ....0... = Congestion Window Reduced: Not set
    ....0.. = ECN-Echo: Not set
    ....0.. = Urgent: Not set
    ....1... = Acknowledgment: Set
    ....0... = Push: Not set
    ....0.. = Reset: Not set
    ....0.. = Syn: Not set
    ....0.. = Fin: Not set
    [TCP Flags: .....A....]
  Window: 2058
  [Calculated window size: 131712]
  [Window size scaling factor: 64]
  Checksum: 0x8d22 [unverified]
  [Checksum Status: Unverified]
  Urgent Pointer: 0
  > Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
  > [Timestamps]
  > [SEQ/ACK analysis]

```

Same as the first stage, the source port is 58448 while the destination port is 443. Additionally, ACK flag (Acknowledgement) is set to 1, while the acknowledge number is set to $1037957616 = 1037957615 + 1$.

At this point, the connection is established, and data transfer can begin.

3.2 Examining HTTPS Request/Response

In order to get HTTPS (TLS) traffic, use the filter `tls` in the display filter:

No.	Time	Source	Destination	Protocol	Length	Info
4	0.017248	10.195.38.194	10.203.4.70	TLSv1.2	583	Client Hello (SNI=www.zju.edu.cn)
6	0.023939	10.203.4.70	10.195.38.194	TLSv1.2	1514	Server Hello
8	0.023946	10.203.4.70	10.195.38.194	TLSv1.2	967	Certificate, Server Key Exchange, Server Hello Done
10	0.027221	10.195.38.194	10.203.4.70	TLSv1.2	159	Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
13	0.028088	10.195.38.194	10.203.4.70	TLSv1.2	1060	Application Data
15	0.030985	10.203.4.70	10.195.38.194	TLSv1.2	117	Change Cipher Spec, Encrypted Handshake Message
30	0.042707	10.203.4.70	10.195.38.194	TLSv1.2	895	Application Data
34	0.073653	10.195.38.194	10.203.4.70	TLSv1.2	1202	Application Data
36	0.083385	10.203.4.70	10.195.38.194	TLSv1.2	272	Application Data
40	0.770315	10.195.38.194	10.203.4.70	TLSv1.2	1205	Application Data
44	0.783347	10.203.4.70	10.195.38.194	TLSv1.2	501	Application Data

This will show all TLS-encrypted traffic between local machine and the server.

3.2.1 TLS Handshake Step 1: Client Hello

Filter: `tls.handshake.type == 1`.

The client (browser) sends a Client Hello, which includes:

- Supported TLS versions (e.g., TLS 1.2, TLS 1.3)
- Cipher suites (encryption methods it can use)
- Server Name Indication (SNI) → Reveals the domain (www.zju.edu.cn) even though the traffic is encrypted.


```
its.handshake_type == 1
No.    Time    Source                Destination          Protocol  Length  Info
 3    0.814337  10.195.38.194        10.203.4.70         TLSv1.2   583    Client Hello (SNI=www.zju.edu.cn)
35    0.838905   10.195.38.194        10.203.4.70         TLSv1.2   583    Client Hello (SNI=www.zju.edu.cn)
11753 18.217822   10.195.38.194        10.203.4.70         TLSv1.2   390    Client Hello (SNI=www.zju.edu.cn)

> Frame 4: 583 bytes on wire (4664 bits), 583 bytes captured (4664 bits) on interface en0, id 0
Ethernet II, Src: d6eaa:d7:f3:83 (d6eaa:d7:f3:83), Dst: NewH3Techno.d8:e0:e02 (58:b3:8f:d8:e0:e02)
Internet Protocol Version 4, Src: 10.195.38.194, Dst: 10.203.4.70
Transmission Control Protocol, Src Port: 58448, Dst Port: 443, Seq: 1, Ack: 1, Len: 517
Transport Layer Security
  TLSv1.2 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: 9c9c3f8efed11247ede5933406c7cd849d6021834ed22f72c69ccfafabac60
      Session ID: 32
      Session ID: d8fef7ba0918ad17bc8249dc7ec11d94f51feda9a18af2c958614f8b5974
      Cipher Suites Length: 42
      Cipher Suites (21 suites)
        Compression Methods Length: 1
        Compression Methods (1 method)
        Extensions Length: 393
        Extension: Reserved (GREASE) (len=0)
        Extension: server_name (len=19) www.zju.edu.cn
        Extension: extended_master_secret (len=0)
        Extension: renegotiation_info (len=1)
        Extension: supported_groups (len=12)
        Extension: ec_point_formats (len=2)
        Extension: application_layer_protocol_negotiation (len=14)
        Extension: status_request (len=5)
        Extension: signature_algorithms (len=22)
        Extension: signed_certificate_timestamp (len=0)
        Extension: key_share (len=43) x25519
        Extension: psk_key_exchange_modes (len=12)
        Extension: supported_versions (len=11) TLS 1.3, TLS 1.2, TLS 1.1, TLS 1.0
```

As is shown in the figure above, TLS version is 1.2, cipher suites is 21 and SNI is www.zju.edu.cn.

3.2.2 TLS Handshake Step 2: Server Hello

Filter: tls.handshake.type == 2

The server responds with:

- Selected TLS version & cipher suite
- Its SSL certificate

No.	Time	Source	Destination	Protocol	Length	Info
6	0.820252	10.203.4.70	10.195.38.194	TLSv1.2	1514	Server Hello
38	0.047061	10.203.4.70	10.195.38.194	TLSv1.2	1514	Server Hello
11755	18.225305	10.203.4.70	10.195.38.194	TLSv1.2	1514	Server Hello
<p>Frame 61: 1514 bytes on wire (12112 bits), 1514 bytes captured (12112 bits) on interface en0, id 0</p> <p>Ethernet II, Src: New4Techno_d8:e0:02 (58:03:b3:d8:e0:02), Dst: d6:eadc:f7:31:83 (d6:eadc:f7:31:83)</p> <p>Internet Protocol Version 4, Src: 10.203.4.70, Dst: 10.195.38.194</p> <p>Transmission Control Protocol, Src Port: 443, Dst Port: 58448, Seq: 1</p> <p>Transport Layer Security</p> <p>TLSv1.2 Record Layer: Handshake Protocol: Server Hello</p> <p>Content Type: Handshake (22)</p> <p>Version: TLS 1.2 (0x0303)</p> <p>Length: 112</p> <p>Handshake Protocol: Server Hello</p> <p>Handshake Type: Server Hello (2)</p> <p>Length: 108</p> <p>Version: TLS 1.2 (0x0303)</p> <p>Random: 68b1e29244b31999e4a229ac5d3c358cd5b1a486724b476783900b7e8</p> <p>Session ID Length: 32</p> <p>Session ID: 6542358f7c02796b130b186c128dc3dfe38d6720b73864130748f38190</p> <p>Cipher Suite: TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 (0xc030)</p> <p>Compression Method: null (0)</p> <p>Extensions Length: 36</p> <p>Extension: renegotiation_info (len=1)</p> <p>Extension: server_name (len=0)</p> <p>Extension: ec_point_formats (len=4)</p> <p>Extension: application_layer_protocol_negotiation (len=11)</p> <p>Extension: extended_master_secret (len=0)</p> <p>Type: extended_master_secret (23)</p> <p>Length: 0</p> <p>[JAXS Fullstring: 771,49200,65281-0-11-16-23]</p> <p>[JAXS: 263:859c53912030774bc0599793d915]</p> <p>TLS segment data (1331 bytes)</p>						

3.2.3 TLS Handshake Step 3: Key Exchange & Encryption Setup

No.	Time	Source	Destination	Protocol	Length	Info
4	0.014337	10.195.38.194	10.203.4.70	TLSv1.2	583	Client Hello (SNI=www.zju.edu.cn)
6	0.020252	10.203.4.70	10.195.38.194	TLSv1.2	1514	Server Hello
8	0.020259	10.203.4.70	10.195.38.194	TLSv1.2	967	Certificate, Server Key Exchange, Server Hello Done
10	0.022084	10.195.38.194	10.203.4.70	TLSv1.2	1559	Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
13	0.024162	10.195.38.194	10.203.4.70	TLSv1.2	1060	Application Data
18	0.026602	10.203.4.70	10.195.38.194	TLSv1.2	117	Change Cipher Spec, Encrypted Handshake Message
33	0.038551	10.203.4.70	10.195.38.194	TLSv1.2	903	Application Data
35	0.038905	10.195.38.194	10.203.4.70	TLSv1.2	583	Client Hello (SNI=www.zju.edu.cn)
38	0.047061	10.203.4.70	10.195.38.194	TLSv1.2	1514	Server Hello
40	0.047067	10.203.4.70	10.195.38.194	TLSv1.2	967	Certificate, Server Key Exchange, Server Hello Done
42	0.049417	10.195.38.194	10.203.4.70	TLSv1.2	159	Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
45	0.050961	10.195.38.194	10.203.4.70	TLSv1.2	1202	Application Data
47	0.055462	10.203.4.70	10.195.38.194	TLSv1.2	117	Change Cipher Spec, Encrypted Handshake Message
50	0.062956	10.203.4.70	10.195.38.194	TLSv1.2	272	Application Data
51	0.065665	10.195.38.194	10.203.4.70	TLSv1.2	1705	Application Data

As shown, No.8 to 47 are the corresponding key exchange and encryption setup steps. The client and server exchange keys (Diffie-Hellman, RSA, etc.) to establish encryption.

After this, all HTTP traffic is encrypted and appears as “Application Data” in Wireshark.

3.3 Analyzing TCP Connection Termination (4-Way Handshake)

A proper TCP connection termination involves 4 steps (though sometimes it appears as 3 packets if FIN and ACK are combined).

- `curl -v https://www.zju.edu.cn` to terminate the connection.

Step	Direction	Packet Type	Description
1	Client → Server	FIN + ACK	The client (or server) initiates closure by sending FIN (Finish)
2	Server → Client	ACK	The other side acknowledges the FIN but may still send remaining data
3	Server → Client	FIN + ACK	The server (or client) sends its own FIN to fully close the connection
4	Client → Server	ACK	Final acknowledgment; the connection is now closed

Here is the Screenshot from Wireshark that corresponds to the above procedure:

11832	18.254548	10.195.38.194	10.203.4.70	TCP	66	58797 → 443	[FIN, ACK] Seq=555 Ack=83399 Win=131072 Len=0 TSval=1748800899 TSecr=1734557510
11833	18.258156	10.203.4.70	10.195.38.194	TCP	66	443 → 58797	[ACK] Seq=83399 Ack=555 Win=64896 Len=0 TSval=1734557518 TSecr=1748800897
11834	18.258158	10.203.4.70	10.195.38.194	TCP	66	443 → 58797	[FIN, ACK] Seq=83399 Ack=555 Win=64896 Len=0 TSval=1734557518 TSecr=1748800897
11835	18.258160	10.203.4.70	10.195.38.194	TCP	66	443 → 58797	[ACK] Seq=83400 Ack=556 Win=64896 Len=0 TSval=1734557518 TSecr=1748800899

1. Packet 11832: [FIN, ACK] (Client → Server)
 - Seq=555, Ack=83399
 - Indicates the client wants to close the connection.
2. Packet 11833: [ACK] (Server → Client)
 - Seq=83399, Ack=555
 - Server acknowledges the FIN but may still send data.
3. Packet 11834: [FIN, ACK] (Server → Client)
 - Seq=83399, Ack=555
 - Server initiates its own closure.
4. Packet 11835: [ACK] (Client → Server)
 - Seq=83400, Ack=556
 - Final acknowledgment; connection fully closed.