

华东师范大学数据科学与工程学院实验报告

课程名称： 计算机网络与编程	年级： 21 级	上机实践成绩：
指导教师： 张召	姓名： 杨茜雅	学号： 10215501435
上机实践名称： Lab11		上机实践日期： 2023.5.19
上机实践编号： 11	组号：	上机实践时间： 5.19

一、实验目的

- 了解 TCP 协议的工作原理
- 学习TCP建立连接三次握手的过程
- 学习 TCP 断开连接四次挥手的过程

二、实验任务

- 使用 Wireshark 快速了解 TCP 协议

三、使用环境

Wireshark

四、实验过程

Task 1: 利用Wireshark抓取一个TCP数据包，查看其具体数据结构和实际的数据（要求根据报文结构正确标识每个部分），请将实验结果附在实验报告中。

TCP是因特网运输层的面向连接的可靠的运输协议。TCP被称为是面向连接的（connection.oriented），这是因为在一个应用进程可以开始 向另一个应用进程发送数据之前，这两个进程必须先相互“握手”，即它们必须相互发送某些预备报文段，以建立确保数据传输的参数。作为TCP连接建立的一部分，连接的双方都将初始化与 TCP 连接相关的许多 TCP 状态变量。如图是 TCP 的报文段结构



图 3-29 TCP 报文段结构

利用 Wireshark 抓包软件抓取一个 TCP 数据包并分析：

The image shows a Wireshark packet capture analysis of a TCP segment. The top panel displays a list of packets, with packet 1 selected. The middle panel shows the packet details for the selected TCP segment. The bottom panel shows the packet bytes in hexadecimal and ASCII. Handwritten annotations in Chinese explain the fields and their values.

Packet List:

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	110.242.70.51	172.30.131.208	TCP	60	80 → 54379 [ACK] Seq=1 Ack=1 Win=952 Len=0
2	0.000045	172.30.131.208	110.242.70.51	TCP	54	[TCP ACKed unseen segment] 54379 → 80 [ACK] Seq=1 Ack=2 Win=506 Len=0
3	0.543461	172.30.131.208	58.205.220.34	TLSv1.2	284	Application Data
4	0.563747	58.205.220.34	172.30.131.208	TCP	60	443 → 51791 [ACK] Seq=1 Ack=231 Win=4595 Len=0
5	0.575787	172.30.131.208	203.119.169.179	TCP	55	54761 → 443 [ACK] Seq=1 Ack=1 Win=510 Len=1 [TCP segment of a reassembled PDU]
6	0.609134	203.119.169.179	172.30.131.208	TCP	66	443 → 54761 [ACK] Seq=1 Ack=2 Win=264 Len=0 SLE=1 SRE=2
7	0.726452	58.205.220.34	172.30.131.208	TLSv1.2	288	Application Data

Packet Details:

- Transmission Control Protocol, Src Port: 80, Dst Port: 54379
 - [Stream index: 0]
 - [Conversation completeness: Incomplete (4)]
 - [TCP Segment Len: 0]
 - Sequence Number: 1 (relative sequence number)
 - Sequence Number (raw): 1290778301
 - [Next Sequence Number: 1 (relative sequence number)]
 - Acknowledgment Number: 1 (relative acknowledgment number)
 - Acknowledgment number (raw): 1094637638
 - 0101 = Header Length: 20 bytes (5)
 - 0101 = Header Length: 20 bytes (5)
 - 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: 952
 - [Calculated window size: 952]
 - [Window size scaling factor: 1 (unknown)]

Packet Bytes:

```

0000  10 3d 1c 9a dc a6 54 c6 ff 7b 38 02 08 00 45 00  ...T...{8...E...
0010  00 28 e8 ee 40 00 25 06 87 cd 6e f2 46 33 ac 1e  ...@...%...n.F3...
0020  83 d0 00 50 d4 6b 4c ef b6 bd 41 3e d8 46 50 10  ...P.kL...A>FP...
0030  03 b8 d5 1a 00 00 00 00 00 00 00 00 00 00 00 00  ...
  
```

Annotations:

- Source Port: 80 → 源端口号 00 50
- Destination Port: 54379 → 目的端口号 d4 6b
- Sequence Number: 1 (relative sequence number) → 序号 4c ef b6 bd
- Acknowledgment Number: 1 (relative acknowledgment number) → 确认号 41 3e d8 46
- Header Length: 20 bytes (5) → 首部长度 50
- Flags: 0x010 (ACK) → 保留未用
- Window: 952 → 接收窗口大小 03 b8
- Calculated window size: 952 → 因特网检验和 d5 1a
- Urgent Pointer: 0 → 紧急数据指针 00 00
- Options & Data → 选项&数据

Task 2: 根据TCP三次握手的交互图和抓到的TCP报文详细分析三次握手过程，请将实验结果附在实验报告中。

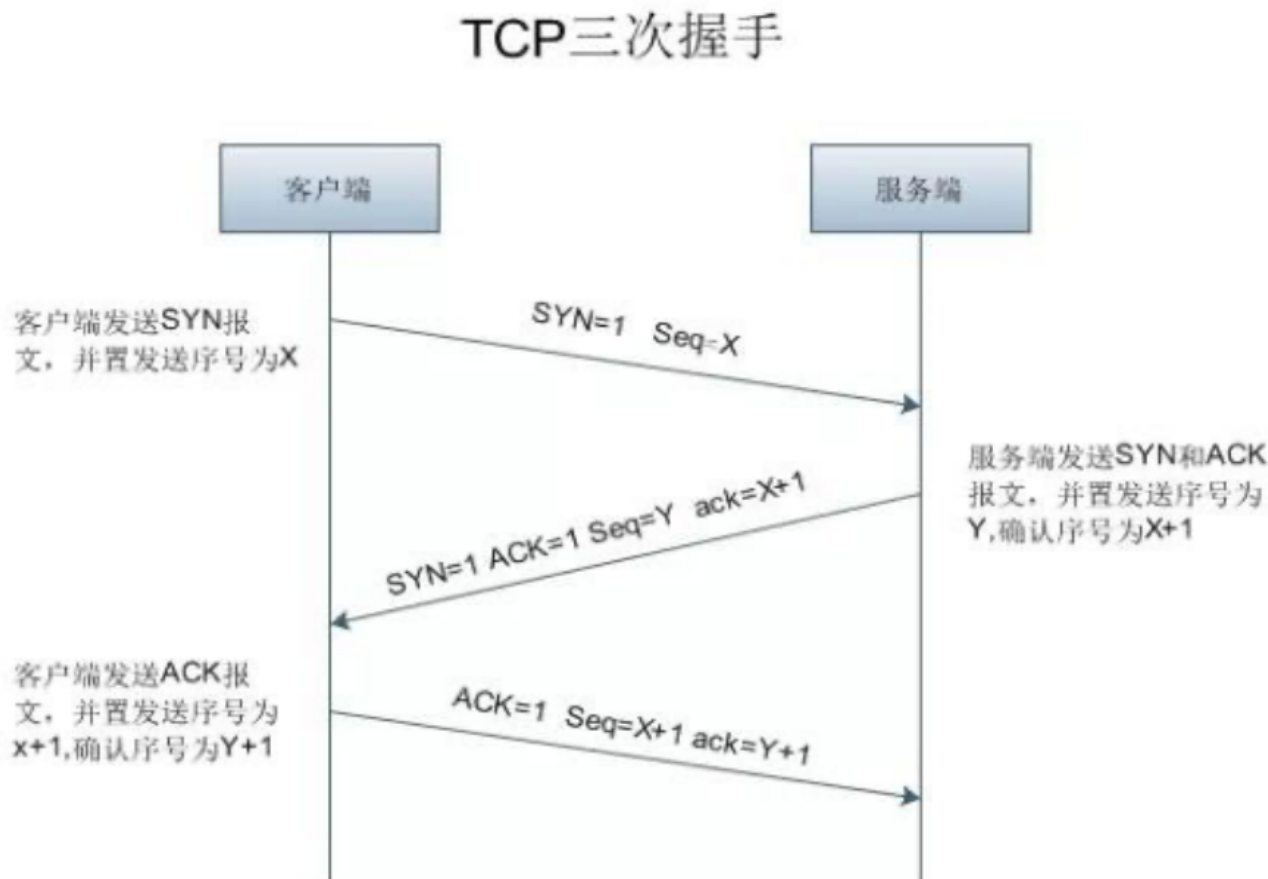
TCP建立连接时，会有三次握手过程，如下图所示，Wireshark截获到了三次握手的三个数据包。

第四个包才 Client Hello 的，说明的确是使用 TCP 建立连接的。

No.	Time	Source	Destination	Protocol	Length	Info
18	0.117718	192.168.1.7	120.92.73.121	TCP	54	58603 → 80 [ACK] Seq=329 Ack=294 Win=130560 Len=0
19	0.140816	59.82.58.85	192.168.1.7	TCP	66	443 → 50745 [ACK] Seq=1 Ack=2 Win=146 Len=0 SLE=1 SRE=2
24	0.203863	192.168.1.7	44.230.35.225	TCP	66	58604 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
25	0.402150	44.230.35.225	192.168.1.7	TCP	66	443 → 58604 [SYN, ACK] Seq=0 Ack=1 Win=26883 Len=0 MSS=1440 SACK_PERM WS=256
26	0.402290	192.168.1.7	44.230.35.225	TCP	54	58604 → 443 [ACK] Seq=1 Ack=1 Win=132352 Len=0
27	0.402546	192.168.1.7	44.230.35.225	TLSv1.2	365	Client Hello
28	0.604700	44.230.35.225	192.168.1.7	TCP	54	443 → 58604 [ACK] Seq=1 Ack=312 Win=28160 Len=0
29	0.604700	44.230.35.225	192.168.1.7	TLSv1.2	158	Server Hello
30	0.604700	44.230.35.225	192.168.1.7	TCP	1494	443 → 58604 [ACK] Seq=105 Ack=312 Win=28160 Len=1440 [TCP segment of a reassembled PDU]
31	0.604700	44.230.35.225	192.168.1.7	TCP	1494	443 → 58604 [ACK] Seq=1545 Ack=312 Win=28160 Len=1440 [TCP segment of a reassembled PDU]

> Frame 1: 54 bytes on wire (432 bits), 54 bytes captured (432 bits) on interface 0	10 3d 1c 9a dc a6 34 24 3e ae b5 e8 08 00 45 04
> Ethernet II, Src: zte_ae:b5:e8 (34:24:3e:ae:b5:e8), Dst: 08:00:45:04:00:00	00 28 4f 5d 40 00 31 06 76 ea 78 5c 49 79 c0 a8
> Internet Protocol Version 4, Src: 120.92.73.121, Dst: 192.168.1.7	01 07 00 50 e4 e8 4b cf 29 1c e0 99 a9 0e 50 10
> Transmission Control Protocol, Src Port: 80, Dst Port: 443	00 3b 48 48 00 00;HH..

TCP 三次握手交互图：



第一次握手，客户端发送SYN报文，并置发送序号为0。在这个数据报中，Seq=0，说明一开始是从序号为0的包发送的。我们可以看到这里SYN 这一位已经被设为1了，因为还没收到服务器的确认信息，因此这里ACK 设为0。

```
> Frame 24: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface \Device\NPF_{90CB2279-4FDD-49D7-8293-23DC64008BEA}, id 0
> Ethernet II, Src: IntelCor_9a:dc:a6 (10:3d:1c:9a:dc:a6), Dst: zte_ae:b5:e8 (34:24:3e:ae:b5:e8)
> Internet Protocol Version 4, Src: 192.168.1.7, Dst: 44.230.35.225
< Transmission Control Protocol, Src Port: 58604, Dst Port: 443, Seq: 0, Len: 0
  Source Port: 58604
  Destination Port: 443
  [Stream index: 3]
  [Conversation completeness: Incomplete, DATA (15)]
  [TCP Segment Len: 0]
  Sequence Number: 0 (relative sequence number)
  Sequence Number (raw): 739516858
  [Next Sequence Number: 1 (relative sequence number)]
  Acknowledgment Number: 0
  Acknowledgment number (raw): 0
  1000 .... = Header Length: 32 bytes (8)
  < 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
    > ....1... = Syn: Set
    ....0... = Fin: Not set
    [TCP Flags: .....S.]
  Window: 64240
  [Calculated window size: 64240]
  Checksum: 0x129d [unverified]
  [Checksum Status: Unverified]
  Urgent Pointer: 0
  < Options: (12 bytes), Maximum segment size, No-Operation (NOP), Window scale, No-Operation (NOP), No-Operation (NOP), SACK permitted
    > TCP Option - Maximum segment size: 1460 bytes
    > TCP Option - No-Operation (NOP)
```

第二次握手，服务器发送SYN和ACK报文，并置发送序号为0，确认序号为1的报文。Seq=0，ACK = 1，因为TCP是全双工通信的，因此发送来的第一个报文段也是从seq=0开始的。但是这个报文段中包含了对我发的确认信息，因此这里ACK被设置为1，这说明1以前的包我全部都收到了，请本地发送1以后的包。

```
> Frame 25: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface \Device\NPF_{90CB2279-4FDD-49D7-8293-23DC64008BEA}, id 0
> Ethernet II, Src: zte_ae:b5:e8 (34:24:3e:ae:b5:e8), Dst: IntelCor_9a:dc:a6 (10:3d:1c:9a:dc:a6)
> Internet Protocol Version 4, Src: 44.230.35.225, Dst: 192.168.1.7
< Transmission Control Protocol, Src Port: 443, Dst Port: 58604, Seq: 0, Ack: 1, Len: 0
  Source Port: 443
  Destination Port: 58604
  [Stream index: 3]
  [Conversation completeness: Incomplete, DATA (15)]
  [TCP Segment Len: 0]
  Sequence Number: 0 (relative sequence number)
  Sequence Number (raw): 2707660783
  [Next Sequence Number: 1 (relative sequence number)]
  Acknowledgment Number: 1 (relative ack number)
  Acknowledgment number (raw): 739516859
  1000 .... = Header Length: 32 bytes (8)
  < 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
    > [Expert Info (Chat/Sequence): Connection establish acknowledge (SYN+ACK): server port 443]
    ....0... = Fin: Not set
    [TCP Flags: .....A..S.]
  Window: 26883
  [Calculated window size: 26883]
  Checksum: 0x7dd0 [unverified]
  [Checksum Status: Unverified]
  Urgent Pointer: 0
  < Options: (12 bytes), Maximum segment size, No-Operation (NOP), No-Operation (NOP), SACK permitted, No-Operation (NOP), window scale
    > TCP Option - Maximum segment size: 1440 bytes
    > TCP Option - No-Operation (NOP)
    > TCP Option - No-Operation (NOP)
    > TCP Option - SACK permitted
    > TCP Option - No-Operation (NOP)
    > TCP Option - window scale: 8 (multiply by 256)
  < [Timestamps]
    [Time since first frame in this TCP stream: 0.198287000 seconds]
    [Time since previous frame in this TCP stream: 0.198287000 seconds]
```

第三次握手，客户端发送ACK报文，并置发送序号为1，确认序号为1。Seq=1，说明这是本地发送的第二个包了(第一个包Seq=0)；ACK = 1 说明已经收到了来自服务端的1 以前的所有包。

```
> Frame 26: 54 bytes on wire (432 bits), 54 bytes captured (432 bits) on interface \Device\NPF_{90CB2279-4FDD-49D7-8293-23DC64008BEA}, id 0
> Ethernet II, Src: IntelCor_9a:dc:a6 (10:3d:1c:9a:dc:a6), Dst: zte_ae:b5:e8 (34:24:3e:ae:b5:e8)
> Internet Protocol Version 4, Src: 192.168.1.7, Dst: 44.230.35.225
v Transmission Control Protocol, Src Port: 58604, Dst Port: 443, Seq: 1, Ack: 1, Len: 0
  Source Port: 58604
  Destination Port: 443
  [Stream index: 3]
  [Conversation completeness: Incomplete, DATA (15)]
  [TCP Segment Len: 0]
  Sequence Number: 1 (relative sequence number)
  Sequence Number (raw): 739516859
  [Next Sequence Number: 1 (relative sequence number)]
  Acknowledgment Number: 1 (relative ack number)
  Acknowledgment number (raw): 2707660784
  0101 .... = Header Length: 20 bytes (5)
v 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....]
```

Client hello:

```
> Frame 27: 365 bytes on wire (2920 bits), 365 bytes captured (2920 bits) on interface \Device\NPF_{90CB2279-4FDD-49D7-8293-23DC64008BEA}, id 0
> Ethernet II, Src: IntelCor_9a:dc:a6 (10:3d:1c:9a:dc:a6), Dst: zte_ae:b5:e8 (34:24:3e:ae:b5:e8)
> Internet Protocol Version 4, Src: 192.168.1.7, Dst: 44.230.35.225
v Transmission Control Protocol, Src Port: 58604, Dst Port: 443, Seq: 1, Ack: 1, Len: 311
  Source Port: 58604
  Destination Port: 443
  [Stream index: 3]
  [Conversation completeness: Incomplete, DATA (15)]
  [TCP Segment Len: 311]
  Sequence Number: 1 (relative sequence number)
  Sequence Number (raw): 739516859
  [Next Sequence Number: 312 (relative sequence number)]
  Acknowledgment Number: 1 (relative ack number)
  Acknowledgment number (raw): 2707660784
  0101 .... = Header Length: 20 bytes (5)
v Flags: 0x018 (PSH, 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
  .... .... 1... = Push: Set
  .... .... .0.. = Reset: Not set
  .... .... ..0. = Syn: Not set
  .... .... ...0 = Fin: Not set
  [TCP Flags: .....AP...]
  Window: 517
  [Calculated window size: 132352]
  [Window size scaling factor: 256]
  Checksum: 0x13c8 [unverified]
  [Checksum Status: Unverified]
  Urgent Pointer: 0
  > [Timestamps]
  > [SEQ/ACK analysis]
  TCP payload (311 bytes)
v Transport Layer Security
v TLSv1.2 Record Layer: Handshake Protocol: Client Hello
  Content type: Handshake (22)
  Version: TLS 1.2 (0x0303)
  Length: 306
  > Handshake Protocol: Client Hello

0000 34 24 3e ae b5 e8 10 3d 1c 9a dc a6 08 00 45 00 4$>....= .....E
0010 01 5f e6 07 40 00 80 06 00 00 c0 a8 01 07 2c e6  _..@... .....,
```


Task 3: 根据TCP四次挥手的交互图和抓到的TCP报文详细分析四次挥手过程，请将实验结果附在实验报告中。

当通信双方完成数据传输，需要进行TCP连接的释放，由于TCP连接是全双工的，因此每个方向都必须单独进行关闭。这个原则是当一方完成它的数据发送任务后就能发送一个FIN来终止这个方向的连接。收到一个FIN只意味着这一方向上没有数据流动，一个TCP连接在收到一个FIN后仍能发送数据。首先进行关闭的一方将执行主动关闭，而另一方执行被动关闭。因为正常关闭过程需要发送4个TCP帧，因此这个过程也叫作4次挥手。如下图所示，Wireshark截获到了四次挥手的四个数据包。

No.	Time	Source	Destination	Protocol	Length	Info
1589	6.246003	192.168.1.7	36.150.23.48	TCP	66	56533 → 443 [ACK] Seq=611 Ack=6286 Win=131584 Len=0 SLE=6284 SRE=6285
1590	6.249942	36.150.23.48	192.168.1.7	TCP	60	[TCP Spurious Retransmission] 443 → 56533 [PSH, ACK] Seq=6282 Ack=611 Win=523648 Len=1
1591	6.249967	192.168.1.7	36.150.23.48	TCP	66	[TCP Dup ACK 1589#1] 56533 → 443 [ACK] Seq=611 Ack=6286 Win=131584 Len=0 SLE=6282 SRE=6283
1592	6.262846	101.198.2.196	192.168.1.7	TCP	66	80 → 56534 [SYN, ACK] Seq=0 Ack=1 Win=28200 Len=0 MSS=1402 SACK_PERM WS=128
1593	6.262975	192.168.1.7	101.198.2.196	TCP	54	56534 → 80 [ACK] Seq=1 Ack=1 Win=131584 Len=0
1594	6.263500	192.168.1.7	101.198.2.196	OCSP	524	Request
1595	6.293510	101.198.2.196	192.168.1.7	TCP	54	80 → 56534 [ACK] Seq=1 Ack=471 Win=29312 Len=0
1596	6.295287	101.198.2.196	192.168.1.7	OCSP	923	Response
1597	6.295590	101.198.2.196	192.168.1.7	TCP	54	80 → 56534 [FIN, ACK] Seq=870 Ack=471 Win=29312 Len=0
1598	6.295617	192.168.1.7	101.198.2.196	TCP	54	56534 → 80 [ACK] Seq=471 Ack=871 Win=130816 Len=0
1599	6.296119	192.168.1.7	101.198.2.196	TCP	54	56534 → 80 [FIN, ACK] Seq=471 Ack=871 Win=130816 Len=0
1600	6.324455	101.198.2.196	192.168.1.7	TCP	60	80 → 56534 [ACK] Seq=871 Ack=472 Win=29312 Len=0
1601	6.448468	2409:8c28:90a0:9::2	2409:8a1e:6e84:2340::	TCP	86	[TCP Retransmission] 443 → 56508 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1412 SACK_PERM WS=128
1602	6.450290	2409:8c28:90a0:9::2	2409:8a1e:6e84:2340::	TCP	86	[TCP Retransmission] 443 → 56509 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1412 SACK_PERM WS=128
1603	6.463324	2409:8c28:90a0:9::2	2409:8a1e:6e84:2340::	TCP	86	[TCP Retransmission] 443 → 56510 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1412 SACK_PERM WS=128
1604	6.468186	2409:8c28:90a0:9::2	2409:8a1e:6e84:2340::	TCP	86	[TCP Retransmission] 443 → 56511 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1412 SACK_PERM WS=128

第一次挥手，主动关闭方发送 FIN=1, ACK=1, SEQ=870, ACK_SEQ=471, 状态 FIN_WAIT_1。

Wireshark - 分组 1597 - WLAN

> Frame 1597: 54 bytes on wire (432 bits), 54 bytes captured (432 bits) on interface \Device\NPF_{90CB2279-4FDD-49D7-8293-23DC64008BEA}, id 0

> Ethernet II, Src: zte_ae:b5:e8 (34:24:3e:ae:b5:e8), Dst: IntelCor_9a:dc:a6 (10:3d:1c:9a:dc:a6)

> Internet Protocol Version 4, Src: 101.198.2.196, Dst: 192.168.1.7

> Transmission Control Protocol, Src Port: 80, Dst Port: 56534, Seq: 870, Ack: 471, Len: 0

Source Port: 80

Destination Port: 56534

[Stream index: 63]

[Conversation completeness: Complete WITH_DATA (31)]

[TCP segment Len: 0]

Sequence Number: 870 (relative sequence number)

Sequence Number (raw): 3724154756

[Next Sequence Number: 871 (relative sequence number)]

Acknowledgment Number: 471 (relative ack number)

Acknowledgment number (raw): 380313123

0101 = Header Length: 20 bytes (5)

> Flags: 0x011 (FIN, 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

....1... = Fin: Set

[TCP flags: A . F]

Window: 229

[Calculated window size: 29312]

[Window size scaling factor: 128]

Checksum: 0x7d41 [unverified]

[Checksum Status: Unverified]

Urgent Pointer: 0

[Timestamp:]

0000 10 3d 1c 9a dc a6 34 24 3e ae b5 e8 08 00 45 04 ..4\$ >.....E

0010 00 28 67 78 40 00 2b 06 be 1a 65 c6 02 c4 c0 a8 .(g@+.e....

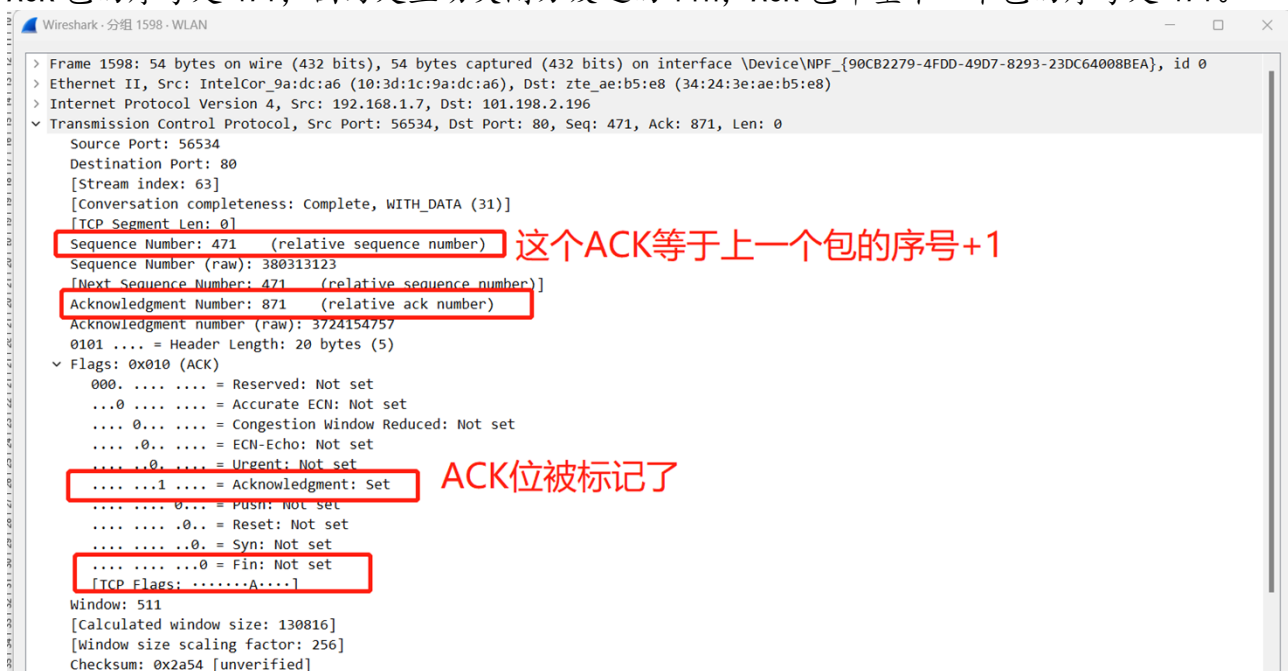
0020 01 07 00 50 dc d6 dd fa 17 84 16 ab 1e 23 50 11 .P.....#P.

0030 00 e5 7d 41 00 00 ..}A..

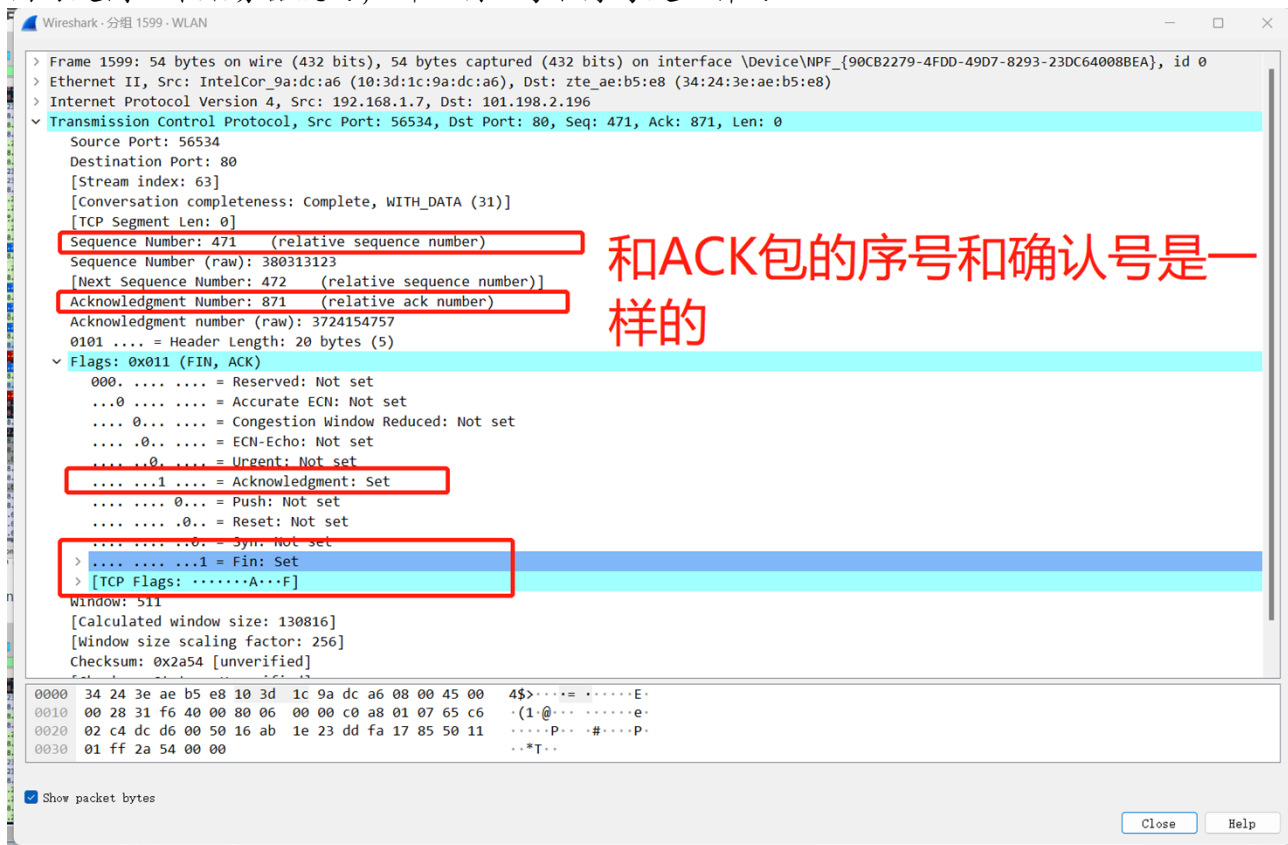
Show packet bytes

Close Help

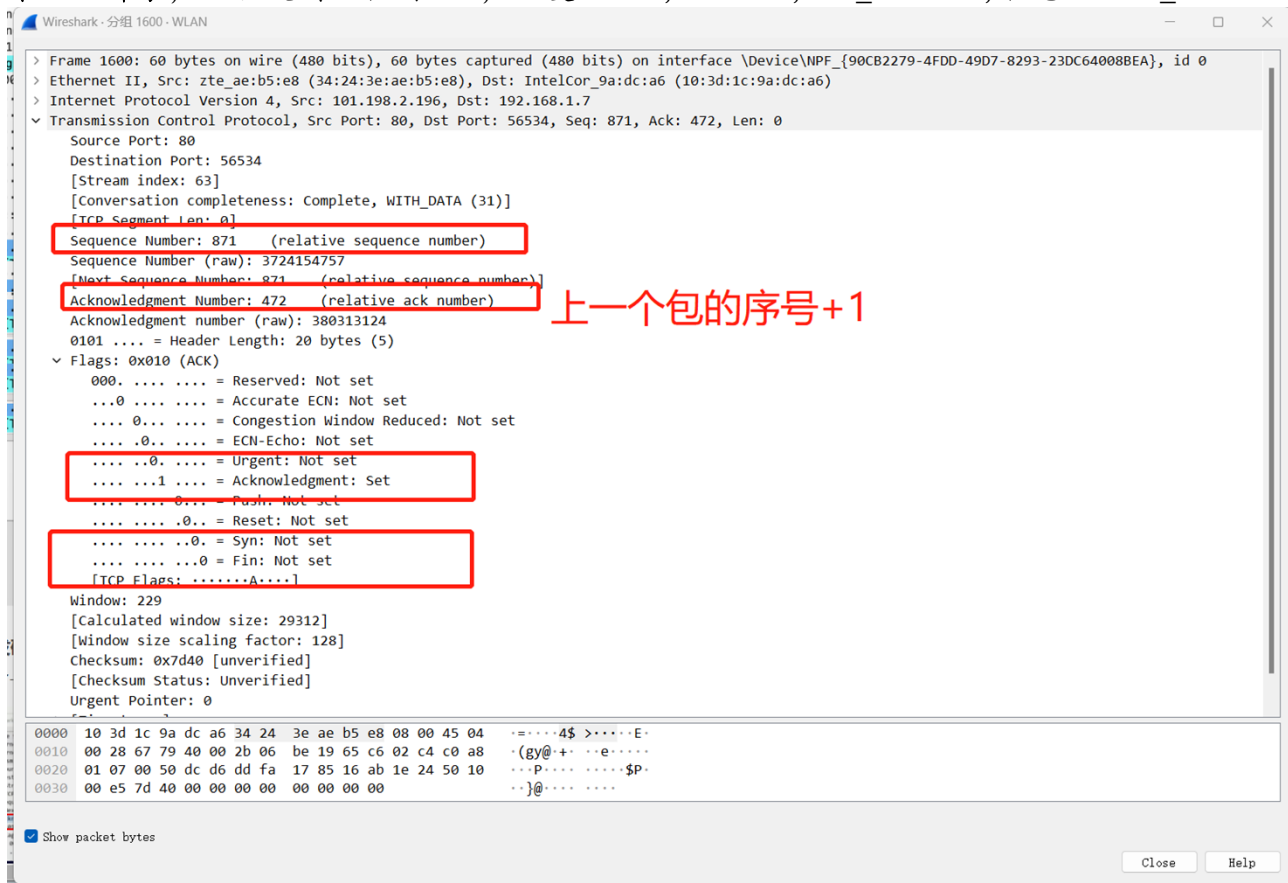
第二次挥手，被动关闭方收到 FIN，回复 ACK=1, SEQ=471, ACK_SEQ=871, 状态 CLOSE_WAIT。
只有 ACK 位被标记了，其他位没有被标记，因为这就是一个确认消息。
ACK 包的序号是 471，因为是主动关闭方发送的 fin，ACK 包希望下一个包的序号是 471。



第三次挥手：close，被动关闭方发送 FIN=1（告诉主动关闭方我也要释放连接了），ACK=1. SEQ=471, ACK_SEQ=871, 状态：LAST_ACK
因为是同一个服务器发的，所以确认号和序号是一样的



第四次挥手，主动关闭方收到 FIN，回复 ACK=1, SEQ=871, ACK_SEQ=472, 状态：TIME_WAIT。



五、总结

通过本次实验，我使用Wireshark 了解到了TCP 协议，明确了TCP协议的工作原理，更直观地认识到了TCP 建立连接三次握手和断开连接四次握手的过程，并对具体的数据包进行分析，对TCP协议有了更深入的理解。