计算机网络实验八

传输控制协议（TCP）

信息学部 朱婉婷

主要内容

一、实验原理

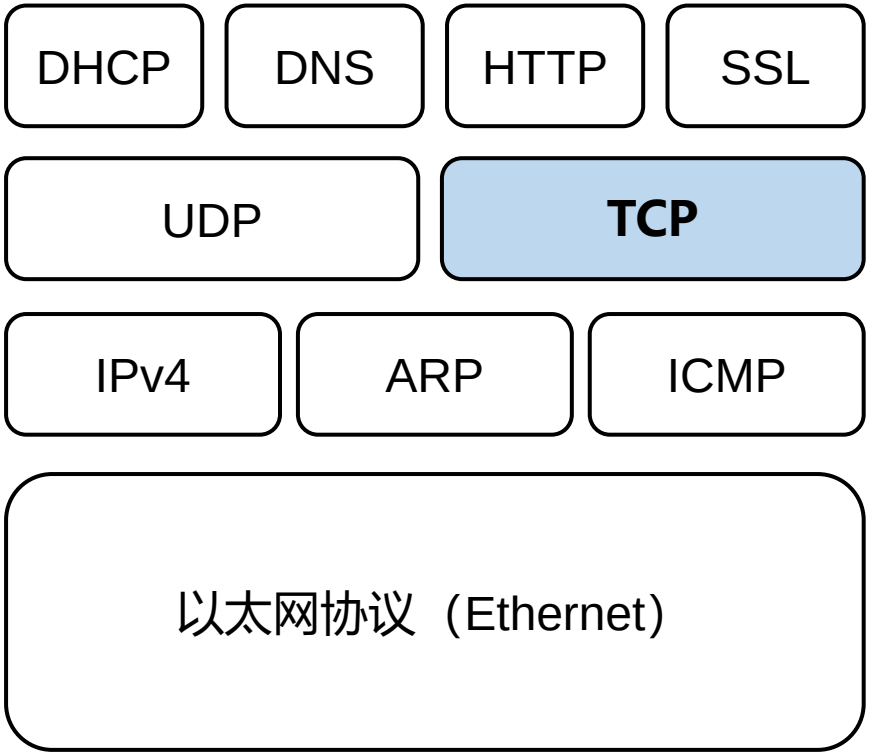
二、实验步骤

三、实验结果及分析

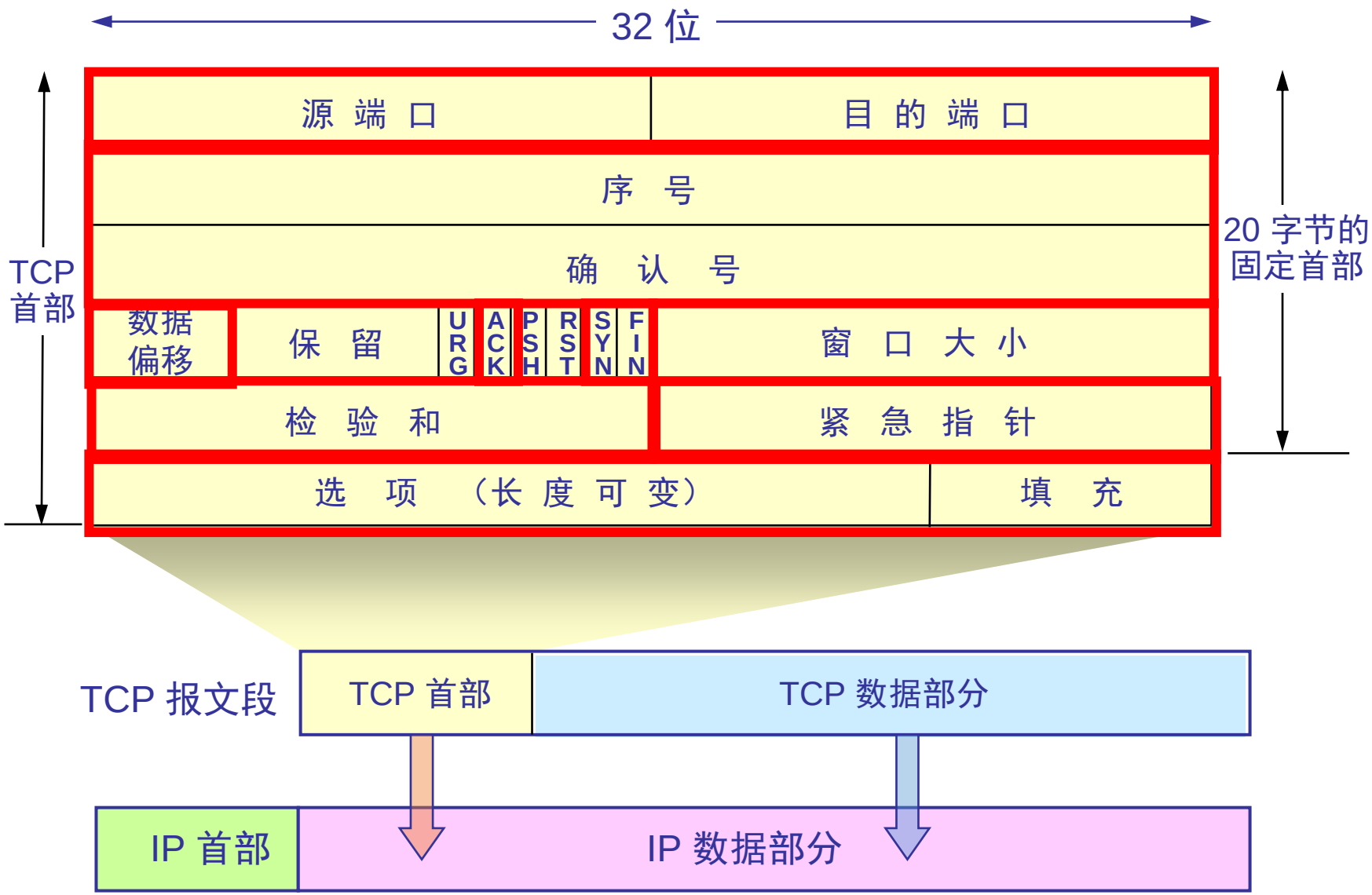
TCP简介

◆传输控制协议 TCP
(Transmission Control Protocol)

- TCP 是面向连接的传输层协议。
- 每一条 TCP 连接只能是点对点的（一对一）。
- TCP 提供全双工通信。
- TCP 是面向字节流。
- TCP 提供可靠交付的服务。
- 滑动窗口和超时重传。



TCP 报文段的首部格式



TCP 报文段的首部格式

- 源端口和目的端口字段：各占 2 字节，分别是数据发起者和接收者的端口号。
- 序号字段：占 4 字节。TCP 连接中传送的数据流中的每一个字节都编上一个序号。序号字段的值则指的是本报文段所发送的数据的第一个字节的序号。
- 确认号字段：占 4 字节，是期望收到对方的下一个报文段的数据的第一个字节的序号。
- 数据偏移（即首部长度）：占 4 位，它指出 TCP 报文段的数据起始处距离 TCP 报文段的起始处有多远。以 4 字节为计算单位。
- 保留字段：占 6 位，保留为今后使用，但目前应置为 0。

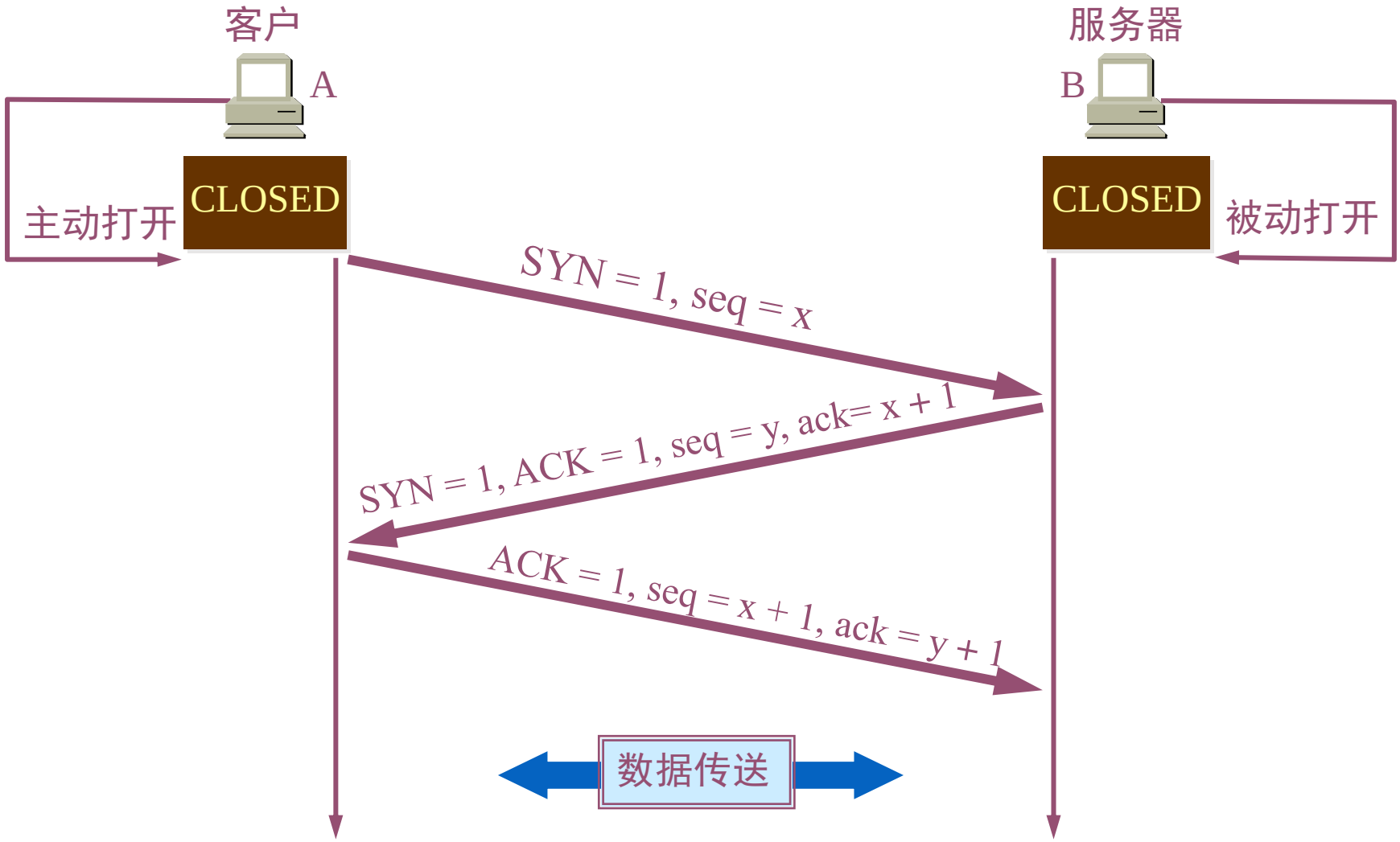
TCP 报文段的首部格式

- 紧急 URG： 当 URG = 1 时，表明紧急指针字段有效。它告诉系统此报文段中有紧急数据，应尽快传送(相当于高优先级的数据)。
- 确认 ACK： 只有当 ACK =1 时确认号字段才有效，代表这是一个确认（ACK）包。当 ACK=0 时，确认号无效。
- 推送 PSH： 接收到 PSH = 1 的报文段，接收方应尽快地将报文交付给应用层，不做队列处理。
- 复位 RST： 当 RST =1 时，表明 TCP 连接中出现严重差错（如由于主机崩溃或其他原因），必须释放连接，然后再重新建立运输连接。
- 同步 SYN： 同步 SYN = 1 表示这是一个连接请求或连接响应报文。
- 终止 FIN： 用来释放一个连接。FIN = 1 表明此报文段的发送端的数据已发送完毕，并要求释放运输连接。

TCP 报文段的首部格式

- 窗口字段： 占 2 字节，用来让对方设置发送窗口的依据，表示准备收到的每个TCP数据的大小，单位为字节。
- 检验和： 占 2 字节。检验和字段检验的范围包括首部和数据这两部分。在计算检验和时，要在 TCP 报文段的前面加上 12 字节的伪首部。
- 紧急指针字段：占 16 位，指出在本报文段中紧急数据共有多少个字节（紧急数据放在本报文段数据的最前面），当URG位取值为1时有效。
- 选项字段： 长度可变。
- 填充字段： 这是为了使整个首部长度是 4 字节的整数倍。

TCP 的连接建立（三次握手）

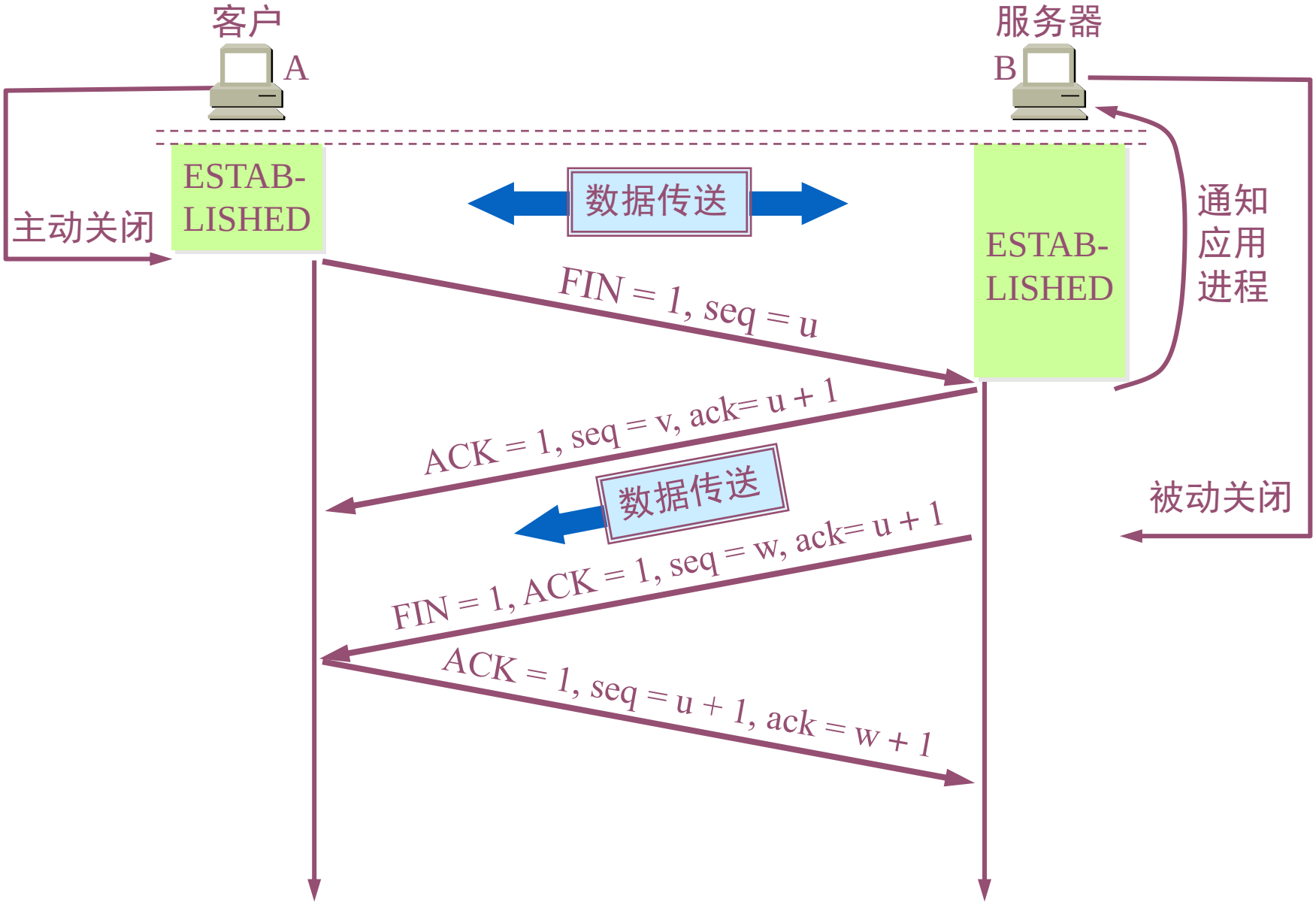


TCP 的连接建立

◆三次握手过程

- （1）A 的 TCP 向 B 发出连接请求报文段，其首部中的同步位 SYN = 1，并选择序号 seq = x，表明传送数据时的第一个数据字节的序号是 x。
- （2）B 的 TCP 收到连接请求报文段后，如同意，则发回确认。B 在确认报文段中应使 SYN = 1，使 ACK = 1，其确认号ack = x +1，自己选择的序号 seq = y。
- （3）A 收到此报文段后向 B 给出确认，其 ACK = 1，确认号 ack = y +1。A 的 TCP 通知上层应用进程，连接已经建立。B 的 TCP 收到主机 A 的确认后，也通知其上层应用进程，TCP 连接已经建立。

TCP 的连接释放（四次挥手）



TCP 的连接释放

◆四次挥手过程

- （1）数据传输结束后，通信的双方都可释放连接。现在 A 的应用进程先向其 TCP 发出连接释放报文段，并停止再发送数据，主动关闭 TCP连接。 A 把连接释放报文段首部的 FIN = 1，其序号seq = u，等待 B 的确认。
- （2）B 发出确认，确认号 ack = u + 1，而这个报文段自己的序号 seq = v。TCP 服务器进程通知高层应用进程。从 A 到 B 这个方向的连接就释放了，TCP 连接处于半关闭状态。B 若发送数据，A 仍要接收。
- （3）若 B 已经没有要向 A 发送的数据，其应用进程就通知 TCP 释放连接，发送一个FIN报文（序号seq = w，确认号 ack = u + 1）给客户端。
- （4）A 收到连接释放报文段后，必须发出确认。在确认报文段中 ACK = 1，确认号 ack = w + 1，序号 seq = u + 1。A 在发送完最后一个 ACK 报文段后，再经过时间 2MSL，才真正释放掉TCP连接。

主要内容

一、实验原理

二、实验步骤

三、实验结果及分析

实验环境搭建

列出本次实验所使用的平台和相关软件，以下为例：

（打开cmd指令窗口，输入指令“ipconfig /all”查看）

1、主机：联想笔记本（Win10系统）；主机IP地址：192.168.1.106；子网掩码：255.255.255.0；主机网卡MAC地址：34-F6-4B-C0-90-40。

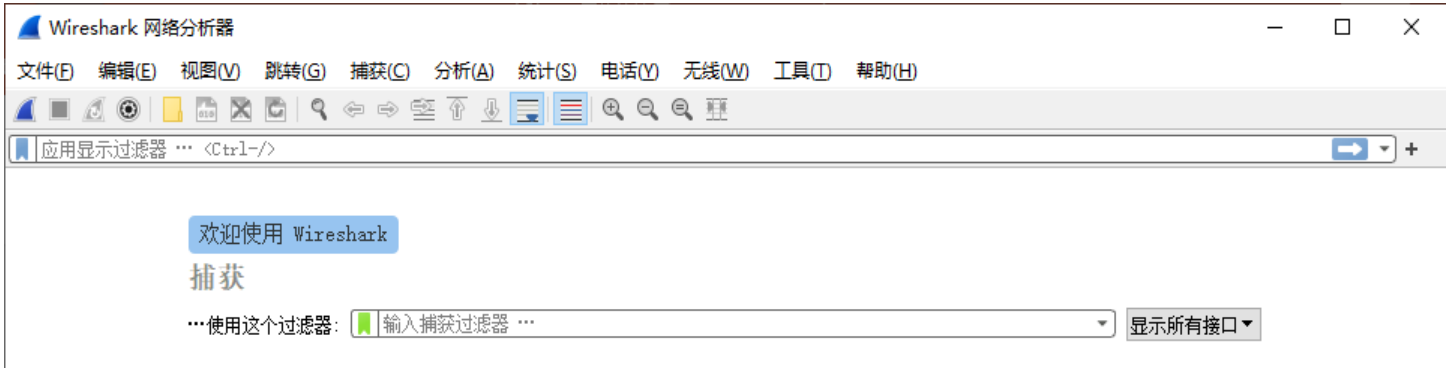
2、网络连接方式：无线连接；默认网关地址：192.168.1.1。

3、抓包工具：Wireshark （v3.6.2）。

实验具体步骤

- 当主机打开某个网页时，主机和Web服务器之间建立TCP连接，随后进行HTTP服务，并在最后释放TCP连接。

1、打开Wireshark软件，双击本次实验正在使用的网络接口，开始进行抓包。

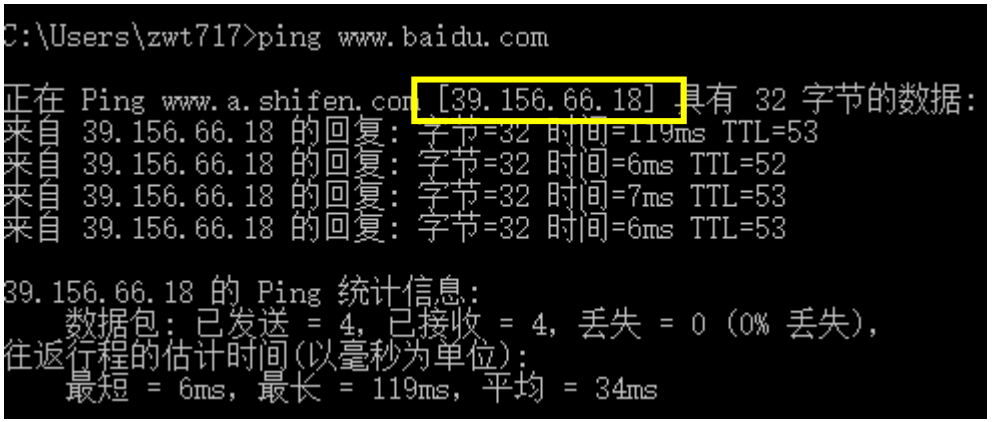


实验具体步骤

2、然后打开浏览器，在网页地址栏中输入网址，例如访问北工大官网并浏览新闻，或者访问百度官网等。

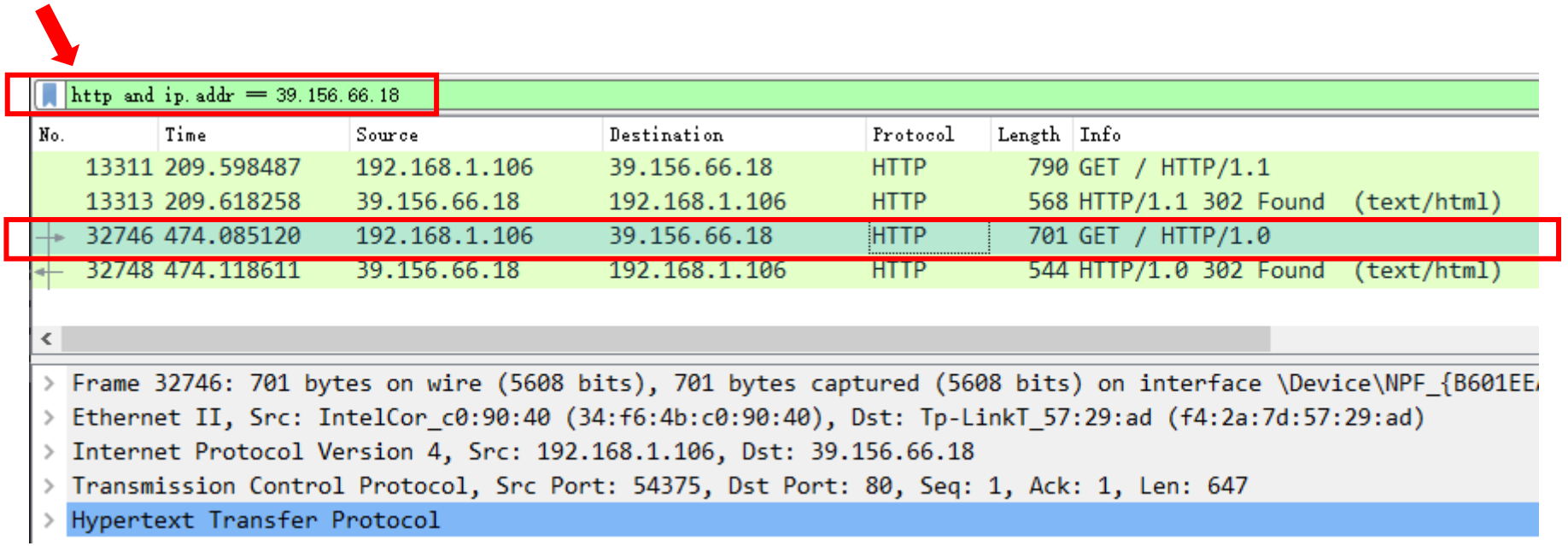
3、关闭浏览器，结束访问。

4、通过ping命令获取网站的IP地址39.156.66.14。



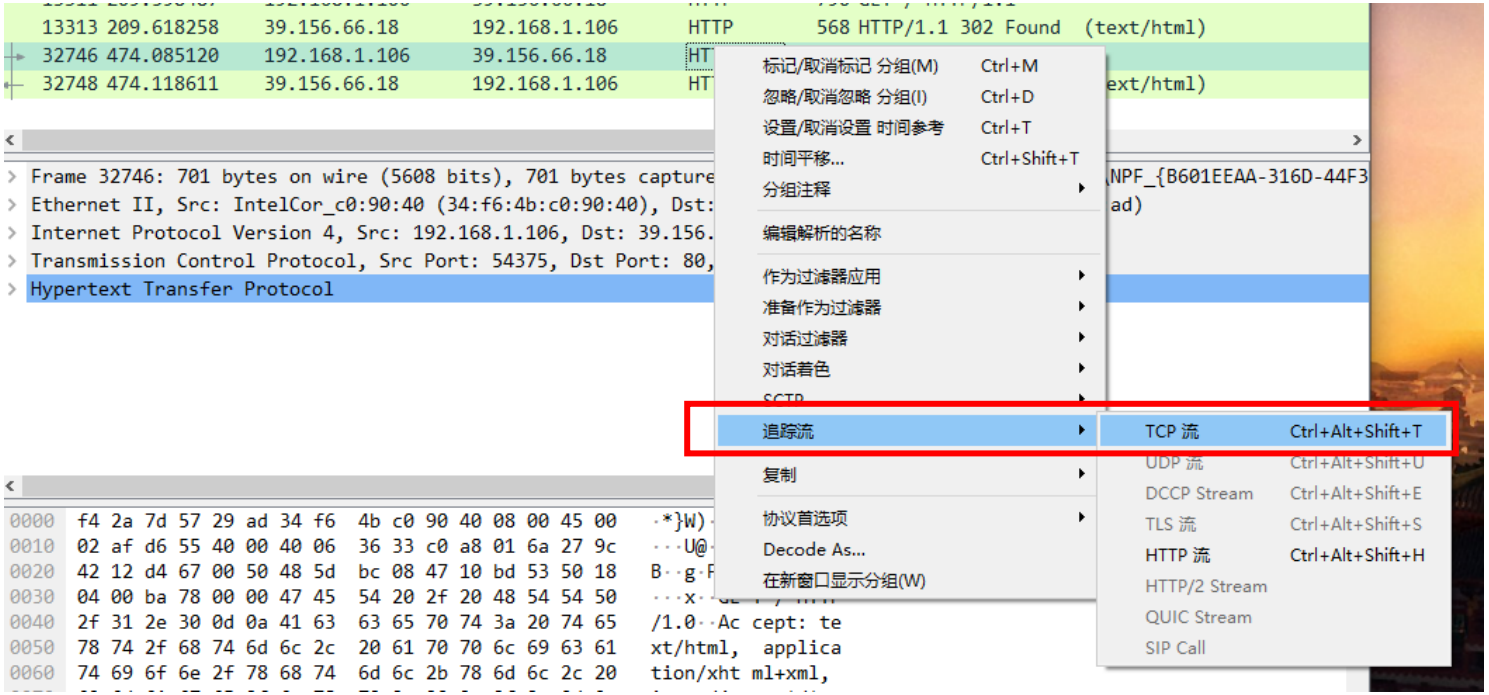
实验具体步骤

5、在过滤栏内输入“http and ip.addr == 39.156.66.18”进行包过滤。并找到HTTP协议包info项为“GET”形式的数据包。



实验具体步骤

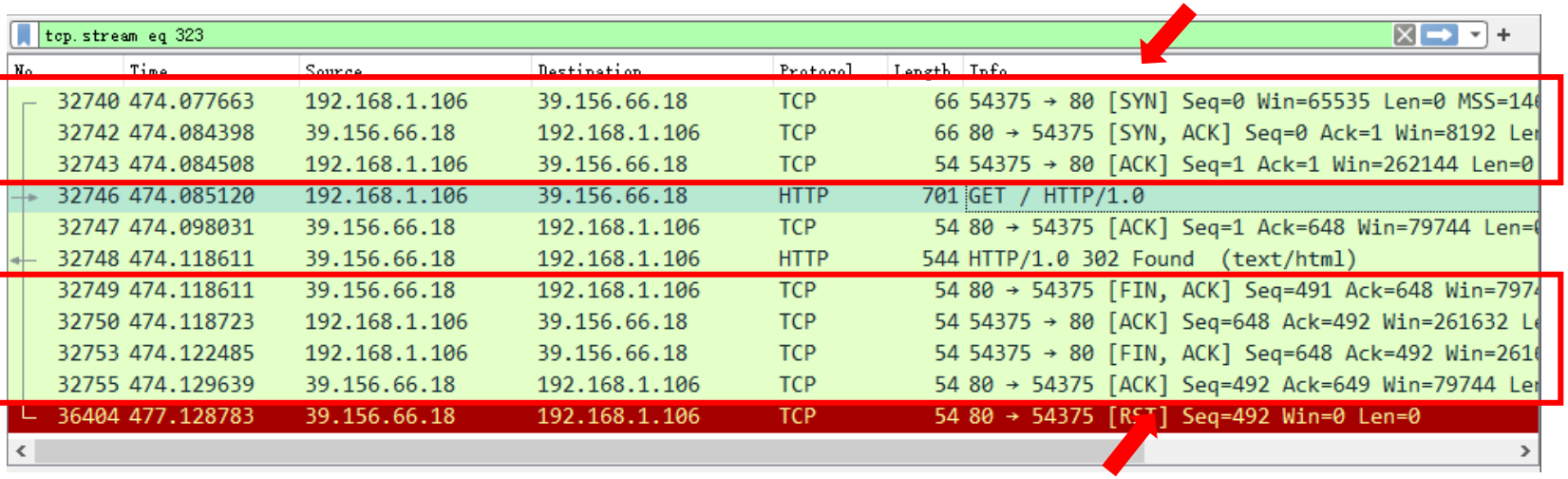
6、点击选中HTTP协议包Info项为“GET”形式的数据包，随后点击Wireshark导航栏内的“分析->追踪流->TCP流”，或右键选择“追踪流->TCP流”进行选中数据包的TCP流追踪。



实验具体步骤

6、点击选中HTTP协议包Info项为“GET”形式的数据包，随后点击Wireshark导航栏内的“分析->追踪流->TCP流”，或右键选择“追踪流->TCP流”进行选中数据包的TCP流追踪。

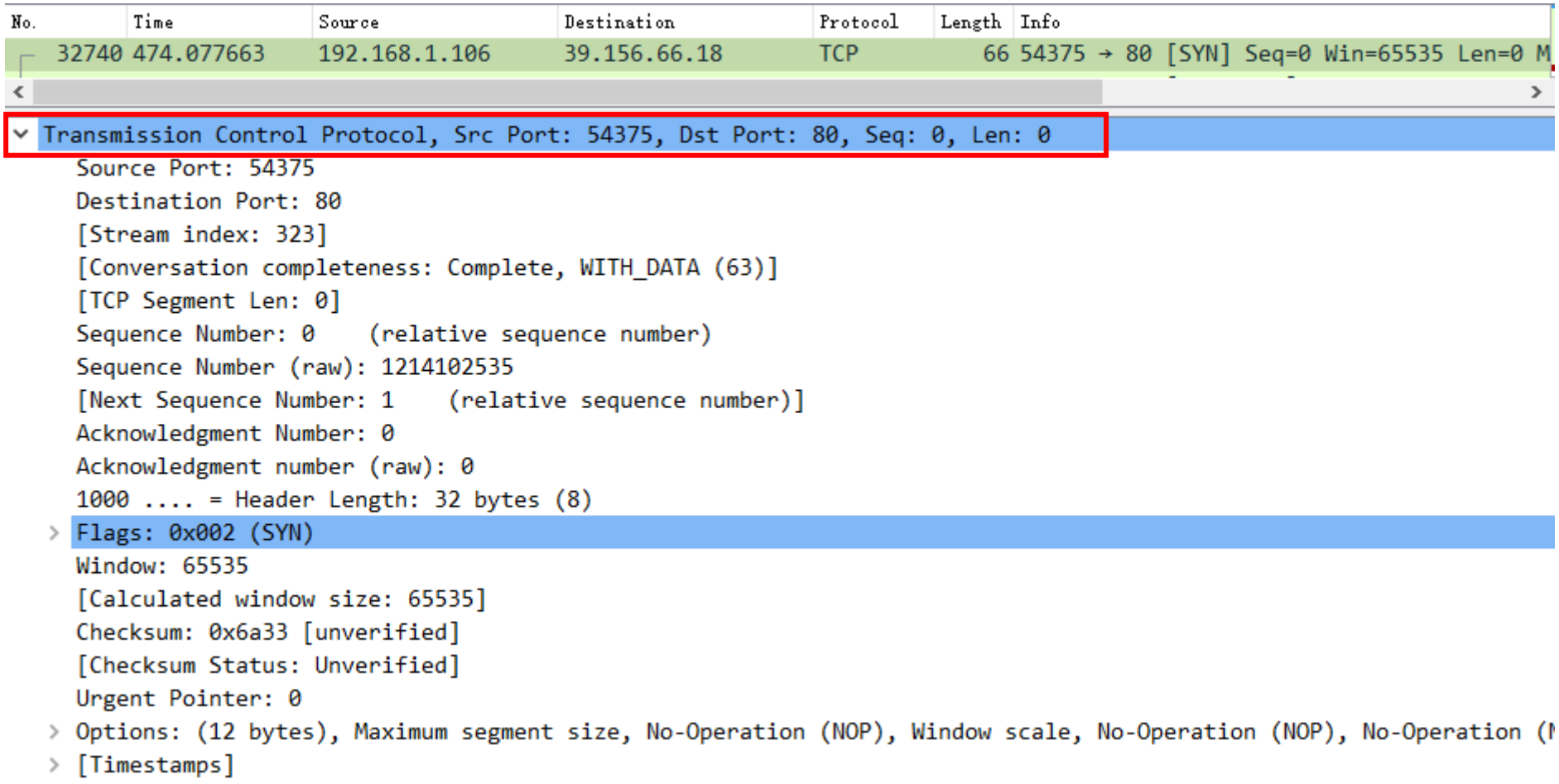
TCP连接建立：三次握手



TCP连接释放：四次挥手

实验具体步骤

7、从中选取任一TCP报文段进行TCP首部格式分析。



实验具体步骤

8、从中选取一组TCP连接建立的三次握手进行观察、分析。

No.	Time	Source	Destination	Protocol	Length	Info
32740	474.077663	192.168.1.106	39.156.66.18	TCP	66	54375 → 80 [SYN] Seq=0 Win=65535 Len=0 M
32742	474.084398	39.156.66.18	192.168.1.106	TCP	66	80 → 54375 [SYN, ACK] Seq=0 Ack=1 Win=81
32743	474.084508	192.168.1.106	39.156.66.18	TCP	54	54375 → 80 [ACK] Seq=1 Ack=1 Win=262144

```
Transmission Control Protocol, Src Port: 54375, Dst Port: 80, Seq: 0, Len: 0
    Source Port: 54375
    Destination Port: 80
    [Stream index: 323]
    [Conversation completeness: Complete, WITH_DATA (63)]
    [TCP Segment Len: 0]
    Sequence Number: 0    (relative sequence number)
    Sequence Number (raw): 1214102535
    [Next Sequence Number: 1    (relative sequence number)]
    Acknowledgment Number: 0
    Acknowledgment number (raw): 0
    1000 .... = Header Length: 32 bytes (8)
  > Flags: 0x002 (SYN)
    Window: 65535
    [Calculated window size: 65535]
    Checksum: 0x6a33 [unverified]
    [Checksum Status: Unverified]
    Urgent Pointer: 0
  > Options: (12 bytes), Maximum segment size, No-Operation (NOP), Window scale, No-Operation (NOP), No-Operation (N
  > [Timestamps]
```

实验具体步骤

9、从中选取一组TCP连接释放的四次挥手进行观察、分析。

No.	Time	Source	Destination	Protocol	Length	Info
32749	474.118611	39.156.66.18	192.168.1.106	TCP	54	80 → 54375 [FIN, ACK] Seq=491 Ack=648 W
32750	474.118723	192.168.1.106	39.156.66.18	TCP	54	54375 → 80 [ACK] Seq=648 Ack=492 Win=261
32753	474.122485	192.168.1.106	39.156.66.18	TCP	54	54375 → 80 [FIN, ACK] Seq=648 Ack=492 W
32755	474.129639	39.156.66.18	192.168.1.106	TCP	54	80 → 54375 [ACK] Seq=492 Ack=649 Win=797

```
Transmission Control Protocol, Src Port: 80, Dst Port: 54375, Seq: 491, Ack: 648, Len: 0
    Source Port: 80
    Destination Port: 54375
    [Stream index: 323]
    [Conversation completeness: Complete, WITH_DATA (63)]
    [TCP Segment Len: 0]
    Sequence Number: 491    (relative sequence number)
    Sequence Number (raw): 1192279869
    [Next Sequence Number: 492    (relative sequence number)]
    Acknowledgment Number: 648    (relative ack number)
    Acknowledgment number (raw): 1214103183
    0101 .... = Header Length: 20 bytes (5)
  > Flags: 0x011 (FIN, ACK)
    Window: 2492
    [Calculated window size: 79744]
    [Window size scaling factor: 32]
    Checksum: 0x9864 [unverified]
    [Checksum Status: Unverified]
    Urgent Pointer: 0
```

三次挥手也可以

◆四次挥手变三次挥手

- 第二次和第三次被合并了。
- TCP延迟确认机制。

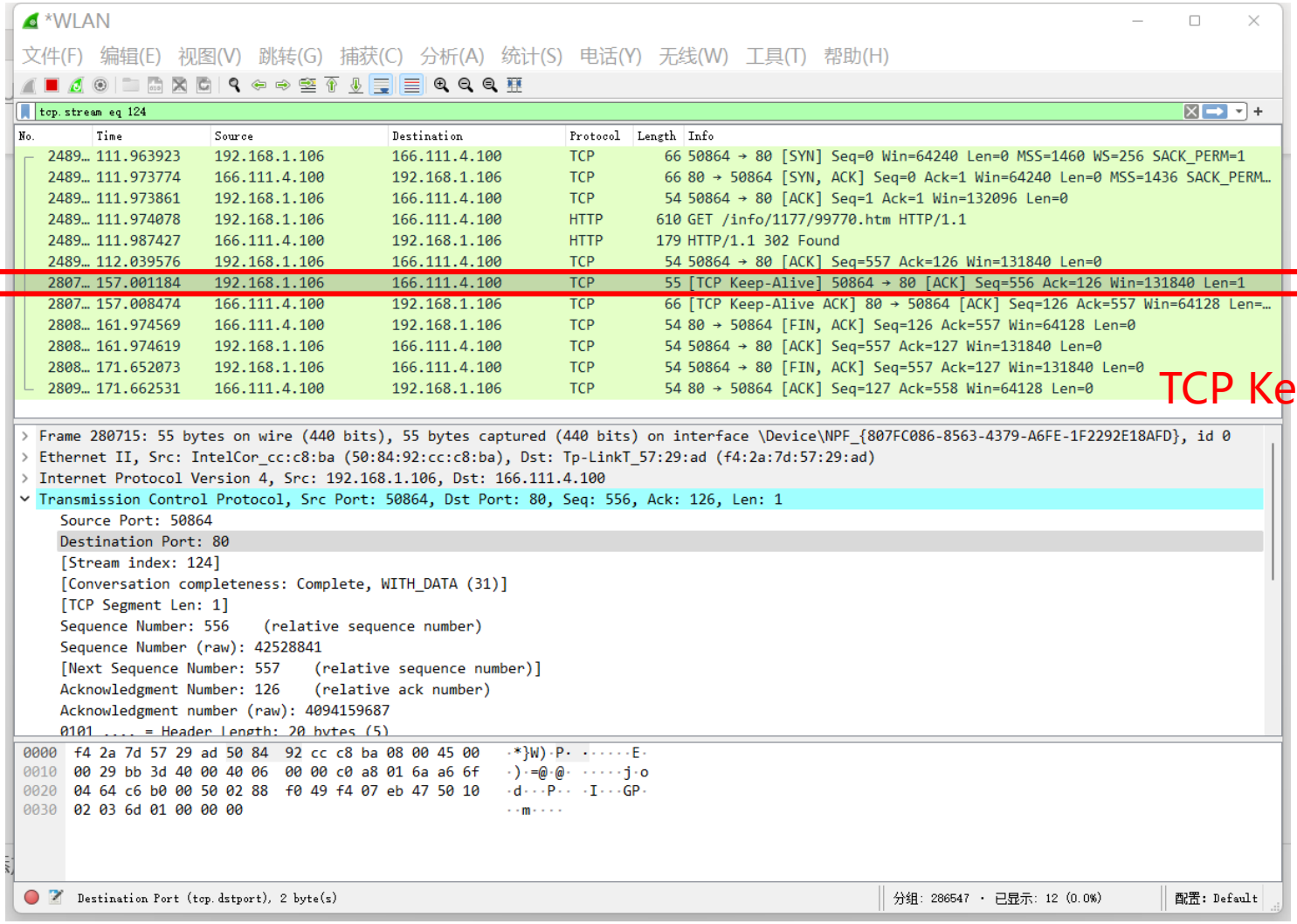
◆短连接和长连接（持续连接）

- 在http1.0中默认是短连接：每次与服务器交互，都需要新开一个连接。耗费资源，已经很少使用了。
- 在http1.1中默认使用持续连接（keep-alive）来解决：建立一次连接，多次请求均由这个连接完成。现在大部分用的是http1.1，并向http2.0过渡。

三次挥手也可以

◆示例：三次挥手时，可能出现的抓包结果如下图所示。

TCP Keep-Alive



实验要求

- 本次实验以下两种情况都可以：
  - 分析任意TCP首部格式+三次握手+四次挥手；
  - 分析任意TCP首部格式+三次握手+三次挥手。

主要内容

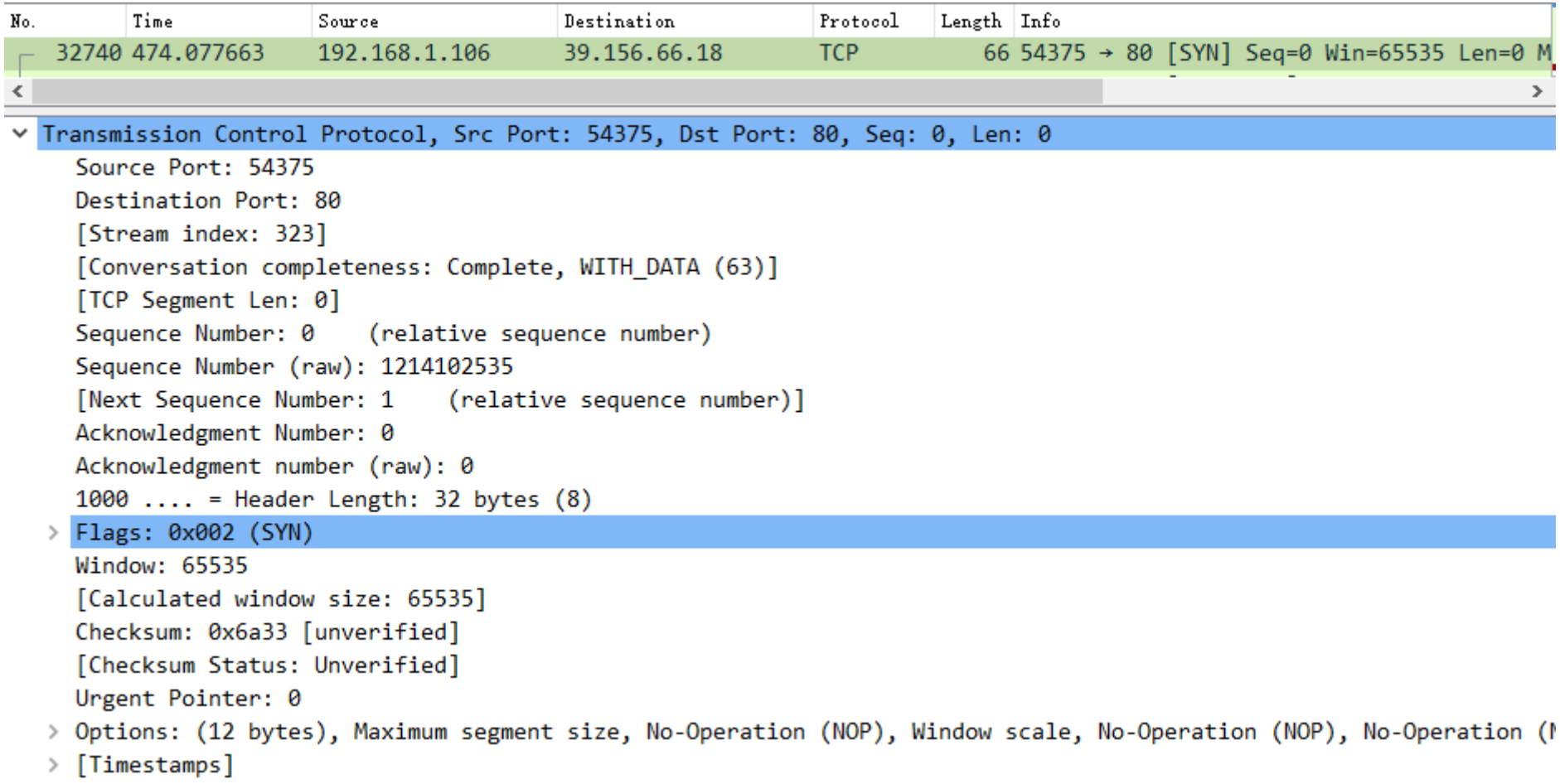
一、实验原理

二、实验步骤

三、实验结果及分析

TCP首部格式分析

实验结果：



实验结果与分析

实验分析：

- 源端口号 Source Port: 54375，定义了发送这个报文段的主机中的应用程序的端口号。
- 目的端口号Destination Port: 80，定义了接收这个报文段的主机中的应用程序的端口号，因为主机要向服务器请求HTTP服务，所以此处为HTTP协议的端口80。
- 序号Sequence number: 0，定义了指派给本报文段第一个数据字节的编号。为了保证连接性，要发送的每一个字节都要编上号。序号可以告诉终点，报文段中的第一个字节是这个序列中的哪一个字节。
- 确认号 Acknowledgment number: 0，定义了报文段的接收方期望从对方接收的字节编号。如果报文段的接收方成功地接收了对方发来的编号为X的字节，那么它就返回所期望接收的字节序列作为确认号。
- 首部长度 1000 .... = Header Length: 32 bytes (8)

实验结果与分析

- 标志位Flags: 0x002 (SYN)
- 000. .... = Reserved: Not set #保留位
- ...0 .... = Nonce: Not set #Noce位
- .... 0... .... = Congestion Window Reduced (CWR): Not set #CER位
- .... .0.. .... = ECN-Echo: Not set #ECN位
- .... ..0. .... = Urgent: Not set #紧急指针有效位
- ....0 .... = Acknowledgment: Not set #确认有效位
- .... 0... = Push: Not set #请求推送位
- .... .0.. = Reset: Not set #连接复位位
- ....1. = Syn: Set #同步序号位
- ....0 = Fin: Not set #终止连接位

实验结果与分析

- 窗口大小Window size value: 65535，并由接受方来决定。
- 检验和 Checksum: 0x6a33 [unverified]
- 紧急指针Urgent pointer: 0，只有当紧急标志置位时，这个16位的字段才有效。
- 选项Options: (12 bytes), Maximum segment size, No-Operation (NOP), Window scale, No-Operation (NOP), No-Operation (NOP), SACK permitted

实验结果与分析

绘制TCP首部字段格式（SYN）：

<table>
<tr><td colspan="6">Source Port: 54375</td><td colspan="5">Destination Port: 80</td></tr>
<tr><td colspan="11">Sequence number: 0</td></tr>
<tr><td colspan="11">Acknowledgment number: 0</td></tr>
<tr><td>Header Length：32 bytes</td><td>0
0
0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td colspan="5">Window size value: 65535</td></tr>
<tr><td colspan="6">Checksum:0x6a33</td><td colspan="5">Urgent pointer: 0</td></tr>
<tr><td colspan="11">Options: (12 bytes), Maximum segment size, No-Operation (NOP), Window scale, No-Operation (NOP), No-Operation (NOP), SACK permitted</td></tr>
</table>

TCP连接建立（三次握手）实验分析

实验结果：

No.	Time	Source	Destination	Protocol	Length	Info
32740	474.077663	192.168.1.106	39.156.66.18	TCP	66	54375 → 80 [SYN] Seq=0 Win=65535 Len=0 M
32742	474.084398	39.156.66.18	192.168.1.106	TCP	66	80 → 54375 [SYN, ACK] Seq=0 Ack=1 Win=81
32743	474.084508	192.168.1.106	39.156.66.18	TCP	54	54375 → 80 [ACK] Seq=1 Ack=1 Win=262144

```
Sequence Number: 0    (relat[illegible]equence number)
Sequence Number (raw): 1214102535
[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 .... .... = Nonce: Not set
   .... 0... .... = Congestion Window Reduced (CWR): 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: 65535
[Calculated window size: 65535]
```

第一次握手：客户端向服务器发送连接请求报文段，标志位SYN置为1，序列号为0。

TCP连接建立（三次握手）实验分析

绘制第一次握手TCP报文首部：

<table>
<tr><td colspan="6">Source Port: 54375</td><td colspan="5">Destination Port: 80</td></tr>
<tr><td colspan="11">Sequence number: 0</td></tr>
<tr><td colspan="11">Acknowledgment number: 0</td></tr>
<tr><td>Header Length：32 bytes</td><td>0
0
0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td colspan="2">Window size value: 65535</td></tr>
<tr><td colspan="6">Checksum:0x6a33</td><td colspan="5">Urgent pointer: 0</td></tr>
<tr><td colspan="11">Options: (12 bytes), Maximum segment size, No-Operation (NOP), Window scale, No-Operation (NOP), No-Operation (NOP), SACK permitted</td></tr>
</table>

TCP连接建立（三次握手）实验分析

实验结果：

No.	Time	Source	Destination	Protocol	Length	Info
32740	474.077663	192.168.1.106	39.156.66.18	TCP	66	54375 → 80 [SYN] Seq=0 Win=65535 Len=0 M
32742	474.084398	39.156.66.18	192.168.1.106	TCP	66	80 → 54375 [SYN, ACK] Seq=0 Ack=1 Win=81
32743	474.084508	192.168.1.106	39.156.66.18	TCP	54	54375 → 80 [ACK] Seq=1 Ack=1 Win=262144

```
Sequence Number: 0    (relat sequence number)
Sequence Number (raw): 1192279378
[Next Sequence Number: 1    (relative sequence number)]
Acknowledgment Number: 1    (relative ack number)
Acknowledgment number (raw): 1214102536
1000 .... = Header Length: 32 bytes (8)
Flags: 0x012 (SYN, ACK)
   000. .... .... = Reserved: Not set
   ...0 .... .... = Nonce: Not set
   .... 0... .... = Congestion Window Reduced (CWR): 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: 8192
[Calculated window size: 8192]
```

第二次握手：服务器端收到客户端发过来的请求连接报文段，由SYN=1知道客户端要求建立连接。随后服务器端向客户端发送一个SYN和ACK都置为1的TCP报文段，并且在这个报文段当中，设置初始序列号0，将确认号设置为客户端的序列号加1。

TCP连接建立（三次握手）实验分析

绘制第二次握手TCP报文首部：

<table>
<tr><td colspan="11">Source Port: 80</td><td colspan="5">Destination Port: 54375</td></tr>
<tr><td colspan="16">Sequence number: 0</td></tr>
<tr><td colspan="16">Acknowledgment number: 1</td></tr>
<tr><td colspan="3">Header Length：32 bytes</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>1</td><td>0</td><td colspan="5">Window size value: 8192</td></tr>
<tr><td colspan="11">Checksum:0x45da</td><td colspan="5">Urgent pointer: 0</td></tr>
<tr><td colspan="16">Options: (12 bytes), Maximum segment size, No-Operation (NOP), Window scale, No-Operation (NOP), No-Operation (NOP), SACK permitted</td></tr>
</table>

TCP连接建立（三次握手）实验分析

实验结果：

No.	Time	Source	Destination	Protocol	Length	Info
32740	474.077663	192.168.1.106	39.156.66.18	TCP	66	54375 → 80 [SYN] Seq=0 Win=65535 Len=0 M
32742	474.084398	39.156.66.18	192.168.1.106	TCP	66	80 → 54375 [SYN, ACK] Seq=0 Ack=1 Win=81
32743	474.084508	192.168.1.106	39.156.66.18	TCP	54	54375 → 80 [ACK] Seq=1 Ack=1 Win=262144

```
Sequence Number: 1    (relat  equence number)
Sequence Number (raw): 1214102536
[Next Sequence Number: 1    (relative sequence number)]
Acknowledgment Number: 1    (  tive ack number)
Acknowledgment number (raw): 1192279379
0101 .... = Header Length: 20 bytes (5)
Flags: 0x010 (ACK)
   000. .... .... = Reserved: Not set
   ...0 .... .... = Nonce: Not set
   .... 0... .... = Congestion Window Reduced (CWR): 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: 1024
[Calculated window size: 262144]
```

第三次握手：客户端收到服务器发来的SYN+ACK报文段之后，客户端发送ACK报文段，ACK标志位为1，SYN标志位为0。确认号是服务器端来的SYN+ACK报文段中的序号加1，发送序号为SYN+ACK报文段中的确认号 。服务器端收到客户端发来的ACK报文段，随后连接建成进行数据传输。

TCP连接建立（三次握手）实验分析

绘制第三次握手TCP报文首部：

<table>
<tr><td colspan="10">Source Port: 54375</td><td colspan="2">Destination Port: 80</td></tr>
<tr><td colspan="12">Sequence number: 1</td></tr>
<tr><td colspan="12">Acknowledgment number: 1</td></tr>
<tr><td>Header Length：20 bytes</td><td>0
0
0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td colspan="2">Window size value: 1024</td></tr>
<tr><td colspan="10">Checksum:0xa292</td><td colspan="2">Urgent pointer: 0</td></tr>
</table>

TCP连接释放（四次挥手）实验分析

实验结果：本次TCP连接释放是HTTP服务端首先发起的。

No.	Time	Source	Destination	Protocol	Length	Info
32749	474.118611	39.156.66.18	192.168.1.106	TCP	54	80 → 54375 [FIN, ACK] Seq=491 Ack=648 Wi
32750	474.118723	192.168.1.106	39.156.66.18	TCP	54	54375 → 80 [ACK] Seq=648 Ack=492 Win=261
32753	474.122485	192.168.1.106	39.156.66.18	TCP	54	54375 → 80 [FIN, ACK] Seq=648 Ack=492 Wi
32755	474.129639	39.156.66.18	192.168.1.106	TCP	54	80 → 54375 [ACK] Seq=492 Ack=649 Win=797

```
Sequence Number: 491    (relative sequence number)
Sequence Number (raw): 1192279869
[Next Sequence Number: 492    (relative sequence number)]
Acknowledgment Number: 648    (relative ack number)
Acknowledgment number (raw): 1214103183
0101 .... = Header Length: 20 bytes (5)
Flags: 0x011 (FIN, ACK)
   000. .... .... = Reserved: Not set
   ...0 .... .... = Nonce: Not set
   .... 0... .... = Congestion Window Reduced (CWR): 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: 2492
```

第一次挥手：HTTP服务器端发送一个FIN，关闭服务器端到客户端的数据传送（服务器端不再发送报文给客户端，但可接受客户端报文）。FIN报文的FIN位设置为1。Seq序号为491，Ack号为648。

TCP连接释放（四次挥手）实验分析

绘制第一次挥手TCP报文首部：

<table>
<tr><td colspan="10">Source Port: 80</td><td colspan="2">Destination Port: 54375</td></tr>
<tr><td colspan="12">Sequence number: 491</td></tr>
<tr><td colspan="12">Acknowledgment number: 648</td></tr>
<tr><td>Header Length: 20 bytes</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>1</td><td colspan="2">Window size value: 2492</td></tr>
<tr><td colspan="10">Checksum:0x9864</td><td colspan="2">Urgent pointer: 0</td></tr>
</table>

TCP连接释放（四次挥手）实验分析

实验结果：

No.	Time	Source	Destination	Protocol	Length	Info
32749	474.118611	39.156.66.18	192.168.1.106	TCP	54	80 → 54375 [FIN, ACK] Seq=491 Ack=648 Wi
32750	474.118723	192.168.1.106	39.156.66.18	TCP	54	54375 → 80 [ACK] Seq=648 Ack=492 Win=261
32753	474.122485	192.168.1.106	39.156.66.18	TCP	54	54375 → 80 [FIN, ACK] Seq=648 Ack=492 Wi
32755	474.129639	39.156.66.18	192.168.1.106	TCP	54	80 → 54375 [ACK] Seq=492 Ack=649 Win=797

```
Sequence Number: 648    (relative sequence number)
Sequence Number (raw): 1214103183
[Next Sequence Number: 648    (relative sequence number)]
Acknowledgment Number: 492    (relative ack number)
Acknowledgment number (raw): 1192279870
0101 .... = Header Length: 20 bytes (5)
Flags: 0x010 (ACK)
   000. .... .... = Reserved: Not set
   ...0 .... .... = Nonce: Not set
   .... 0... .... = Congestion Window Reduced (CWR): 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: 1022
```

第二次挥手：客户端收到这个FIN之后发回一个ACK报文，它的Seq序号是收到的服务器端发送的FIN报文的Ack号648，它的Ack号是收到的服务器端发送的FIN报文的Seq序号加1，即491+1为492。

TCP连接释放（四次挥手）实验分析

绘制第二次挥手TCP报文首部：

<table>
<tr><td colspan="10">Source Port: 54375</td><td colspan="2">Destination Port: 80</td></tr>
<tr><td colspan="12">Sequence number: 648</td></tr>
<tr><td colspan="12">Acknowledgment number: 492</td></tr>
<tr><td>Header Length: 20 bytes</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td colspan="2">Window size value: 1022</td></tr>
<tr><td colspan="10">Checksum:0x9e22</td><td colspan="2">Urgent pointer: 0</td></tr>
</table>

TCP连接释放（四次挥手）实验分析

实验结果：

No.	Time	Source	Destination	Protocol	Length	Info
32749	474.118611	39.156.66.18	192.168.1.106	TCP	54	80 → 54375 [FIN, ACK] Seq=491 Ack=648 Wi
32750	474.118723	192.168.1.106	39.156.66.18	TCP	54	54375 → 80 [ACK] Seq=648 Ack=492 Win=261
32753	474.122485	192.168.1.106	39.156.66.18	TCP	54	54375 → 80 [FIN, ACK] Seq=648 Ack=492 Wi
32755	474.129639	39.156.66.18	192.168.1.106	TCP	54	80 → 54375 [ACK] Seq=492 Ack=649 Win=797

```
Sequence Number: 648    (relative sequence number)
Sequence Number (raw): 1214103183
[Next Sequence Number: 649    (relative sequence number)]
Acknowledgment Number: 492    (relative ack number)
Acknowledgment number (raw): 1192279870
0101 .... = Header Length: 20 bytes (5)
Flags: 0x011 (FIN, ACK)
   000. .... .... = Reserved: Not set
   ...0 .... .... = Nonce: Not set
   .... 0... .... = Congestion Window Reduced (CWR): 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: 1022
```

第三次挥手：当服务器端关闭TCP连接之后，接下来进行的是客户端关闭TCP连接。客户端发送一个FIN，关闭客户端到HTTP服务器端的数据传送。FIN数据报的FIN位设置为1，ACK位设置为1。

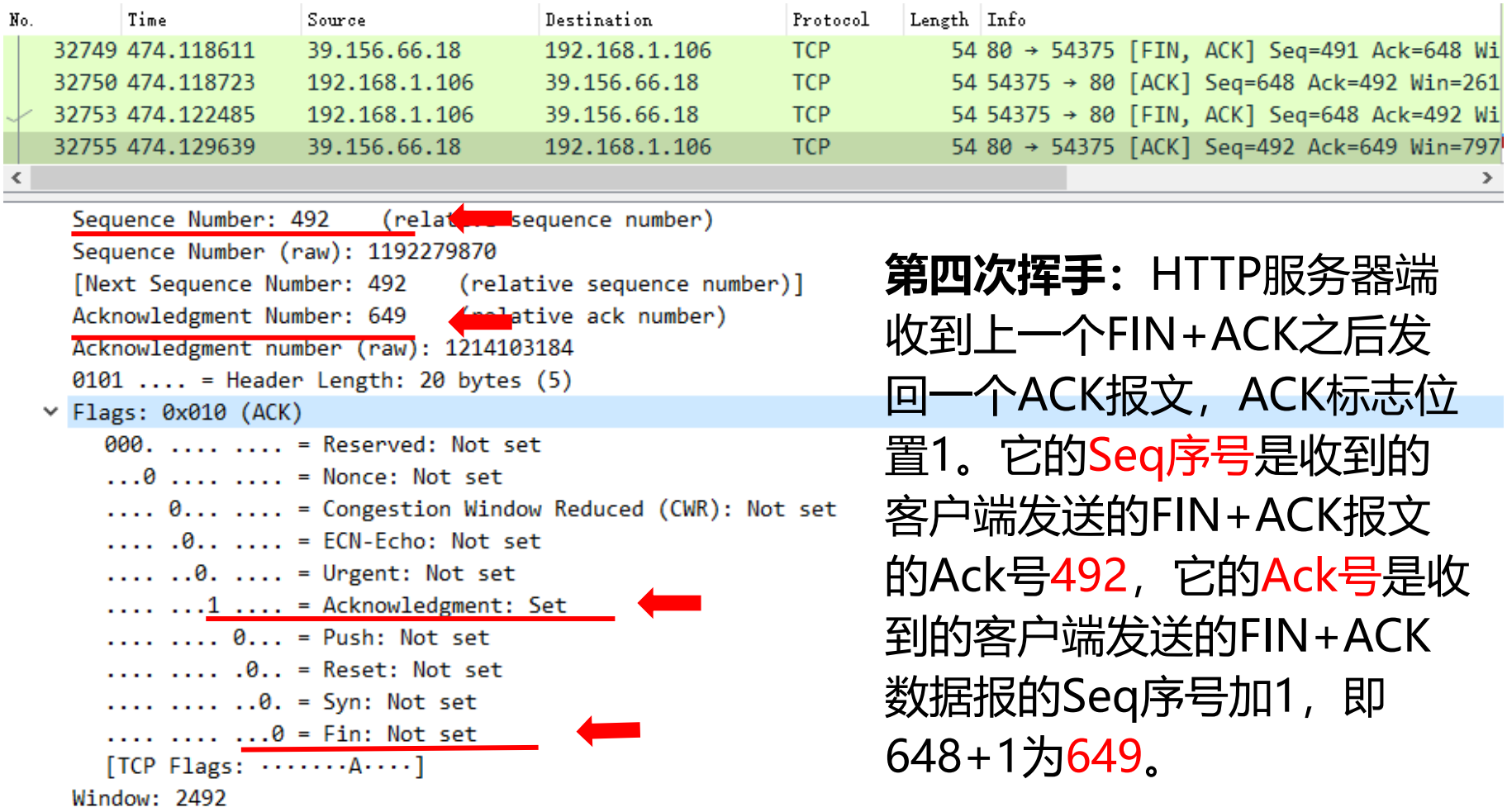
TCP连接释放（四次挥手）实验分析

绘制第三次挥手TCP报文首部：

<table>
<tr><td colspan="10">Source Port: 54375</td><td colspan="2">Destination Port: 80</td></tr>
<tr><td colspan="12">Sequence number: 648</td></tr>
<tr><td colspan="12">Acknowledgment number: 492</td></tr>
<tr><td>Header Length: 20 bytes</td><td>0
0
0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>1</td><td colspan="2">Window size value: 1022</td></tr>
<tr><td colspan="10">Checksum:0x9e21</td><td colspan="2">Urgent pointer: 0</td></tr>
</table>

TCP连接释放（四次挥手）实验分析

实验结果：



No.	Time	Source	Destination	Protocol	Length	Info
32749	474.118611	39.156.66.18	192.168.1.106	TCP	54	80 → 54375 [FIN, ACK] Seq=491 Ack=648 Wi
32750	474.118723	192.168.1.106	39.156.66.18	TCP	54	54375 → 80 [ACK] Seq=648 Ack=492 Win=261
32753	474.122485	192.168.1.106	39.156.66.18	TCP	54	54375 → 80 [FIN, ACK] Seq=648 Ack=492 Wi
32755	474.129639	39.156.66.18	192.168.1.106	TCP	54	80 → 54375 [ACK] Seq=492 Ack=649 Win=797

```
Sequence Number: 492    (relative sequence number)
Sequence Number (raw): 1192279870
[Next Sequence Number: 492    (relative sequence number)]
Acknowledgment Number: 649    (relative ack number)
Acknowledgment number (raw): 1214103184
0101 .... = Header Length: 20 bytes (5)
Flags: 0x010 (ACK)
   000. .... .... = Reserved: Not set
   ...0 .... .... = Nonce: Not set
   .... 0... .... = Congestion Window Reduced (CWR): 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: 2492
```

第四次挥手：HTTP服务器端收到上一个FIN+ACK之后发回一个ACK报文，ACK标志位置1。它的Seq序号是收到的客户端发送的FIN+ACK报文的Ack号492，它的Ack号是收到的客户端发送的FIN+ACK数据报的Seq序号加1，即648+1为649。

TCP连接释放（四次挥手）实验分析

绘制第四次挥手TCP报文首部：

<table>
<tr><td colspan="5">Source Port: 80</td><td colspan="5">Destination Port: 54375</td></tr>
<tr><td colspan="10">Sequence number: 492</td></tr>
<tr><td colspan="10">Acknowledgment number: 649</td></tr>
<tr><td>Header Length：20 bytes</td><td>0
0
0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td colspan="3">Window size value: 2492</td></tr>
<tr><td colspan="5">Checksum:0x9863</td><td colspan="5">Urgent pointer: 0</td></tr>
</table>

思考题

思考题：抓取并观察TCP Keep-Alive报文，分析其使用意图。

谢谢