使用 wireshark 分析 TCP

计算机科学与技术专业　　1711436　　皮春莹

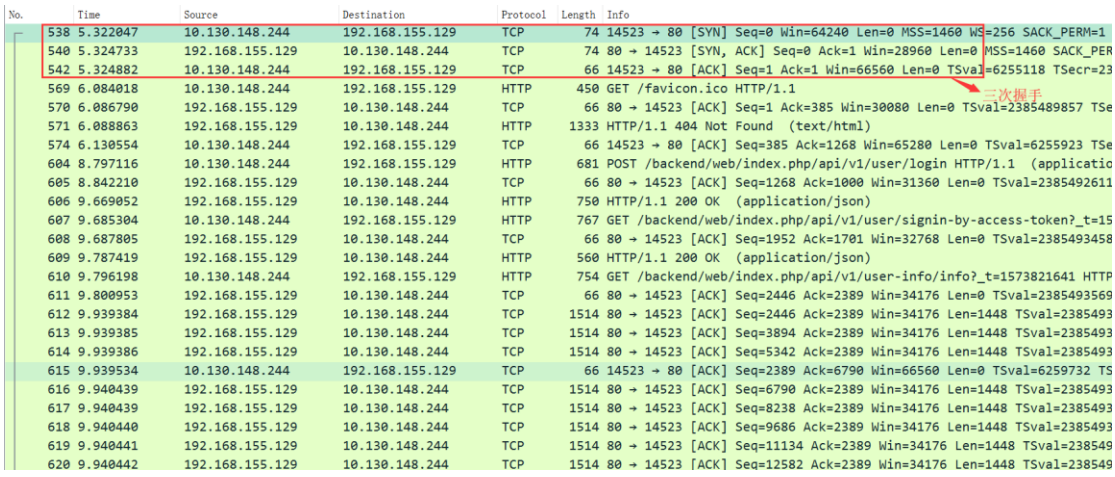
1. 实验内容

通过 HTTP 访问某个网页，使用 Wireshark 对整个过程中的数据段进行捕获，分析 TCP 连接建立、数据传输、连接关闭的全过程，至少对其中 5 个典型的 TCP 数据段进行详细分析。

2. 过程分析

2.1 Wireshark 捕获及过滤数据说明

以计算机学院网站为例，网址为 http://cc-backend.nankai.edu.cn，对应的 IP 地址为 192.168.155.129，打开 wireshark，在网站上依次进行：登录后台、点击页面、下载多个文件的操作，只保留 TCP 协议的捕获结果，设置 wireshark 的过滤条件为”ip.addr == 192.168.155.129”，找到客户端发给服务器的 SYN 那一条信息，右键选择“追踪流”，得到下面结果：



No.	Time	Source	Destination	Protocol	Length	Info
538	5.322047	10.130.148.244	192.168.155.129	TCP	74	14523 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
540	5.324733	192.168.155.129	10.130.148.244	TCP	74	80 → 14523 [SYN, ACK] Seq=0 Ack=1 Win=28960 Len=0 MSS=1460 SACK_PER
542	5.324882	10.130.148.244	192.168.155.129	TCP	66	14523 → 80 [ACK] Seq=1 Ack=1 Win=66560 Len=0 TSval=6255118 TSecr=23
569	6.084018	10.130.148.244	192.168.155.129	HTTP	450	GET /favicon.ico HTTP/1.1
570	6.086790	192.168.155.129	10.130.148.244	TCP	66	80 → 14523 [ACK] Seq=1 Ack=385 Win=30080 Len=0 TSval=2385489857 TSe
571	6.088863	192.168.155.129	10.130.148.244	HTTP	1333	HTTP/1.1 404 Not Found (text/html)
574	6.130554	10.130.148.244	192.168.155.129	TCP	66	14523 → 80 [ACK] Seq=385 Ack=1268 Win=65280 Len=0 TSval=6255923 TSe
604	8.797116	10.130.148.244	192.168.155.129	HTTP	681	POST /backend/web/index.php/api/v1/user/login HTTP/1.1 (applicatio
605	8.842210	192.168.155.129	10.130.148.244	TCP	66	80 → 14523 [ACK] Seq=1268 Ack=1000 Win=31360 Len=0 TSval=2385492611
606	9.669052	192.168.155.129	10.130.148.244	HTTP	750	HTTP/1.1 200 OK (application/json)
607	9.685304	10.130.148.244	192.168.155.129	HTTP	767	GET /backend/web/index.php/api/v1/user/signin-by-access-token?_t=15
608	9.687805	192.168.155.129	10.130.148.244	TCP	66	80 → 14523 [ACK] Seq=1952 Ack=1701 Win=32768 Len=0 TSval=2385493458
609	9.787419	192.168.155.129	10.130.148.244	HTTP	560	HTTP/1.1 200 OK (application/json)
610	9.796198	10.130.148.244	192.168.155.129	HTTP	754	GET /backend/web/index.php/api/v1/user-info/info?_t=1573821641 HTTP
611	9.800953	192.168.155.129	10.130.148.244	TCP	66	80 → 14523 [ACK] Seq=2446 Ack=2389 Win=34176 Len=0 TSval=2385493569
612	9.939384	192.168.155.129	10.130.148.244	TCP	1514	80 → 14523 [ACK] Seq=2446 Ack=2389 Win=34176 Len=1448 TSval=2385493
613	9.939385	192.168.155.129	10.130.148.244	TCP	1514	80 → 14523 [ACK] Seq=3894 Ack=2389 Win=34176 Len=1448 TSval=2385493
614	9.939386	192.168.155.129	10.130.148.244	TCP	1514	80 → 14523 [ACK] Seq=5342 Ack=2389 Win=34176 Len=1448 TSval=2385493
615	9.939534	10.130.148.244	192.168.155.129	TCP	66	14523 → 80 [ACK] Seq=2389 Ack=6790 Win=66560 Len=0 TSval=6259732 TS
616	9.940439	192.168.155.129	10.130.148.244	TCP	1514	80 → 14523 [ACK] Seq=6790 Ack=2389 Win=34176 Len=1448 TSval=2385493
617	9.940439	192.168.155.129	10.130.148.244	TCP	1514	80 → 14523 [ACK] Seq=8238 Ack=2389 Win=34176 Len=1448 TSval=2385493
618	9.940440	192.168.155.129	10.130.148.244	TCP	1514	80 → 14523 [ACK] Seq=9686 Ack=2389 Win=34176 Len=1448 TSval=2385493
619	9.940441	192.168.155.129	10.130.148.244	TCP	1514	80 → 14523 [ACK] Seq=11134 Ack=2389 Win=34176 Len=1448 TSval=238549
620	9.940442	192.168.155.129	10.130.148.244	TCP	1514	80 → 14523 [ACK] Seq=12582 Ack=2389 Win=34176 Len=1448 TSval=238549

图 1 wireshark 过滤结果

双击每一条“Info”时，在下面可以看到四个条目，从上往下，依次对应 TCP/IP 五层模型中的数据链路层、网络层和传输层。

```
> Frame 538: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface 0
> Ethernet II, Src: LiteonTe_19:f5:7d (f8:28:19:19:f5:7d), Dst: IETF-VRRP-VRID_0d (00:00:5e:00:01:0d)
> Internet Protocol Version 4, Src: 10.130.148.244, Dst: 192.168.155.129
> Transmission Control Protocol, Src Port: 14523, Dst Port: 80, Seq: 0, Len: 0
```

图 2 数据链路层、网络层和传输层

数据链路层使用以太网传输，以太网 MAC 帧格式如表 1 所示，对于本次试验的例子，以太网 MAC 帧数据如图 3 所示：

表 1 以太网 MAC 帧格式

目的地址 6 bytes	源地址 6 bytes	类型 2 bytes	数据 46~1500 bytes	FCS 4 bytes

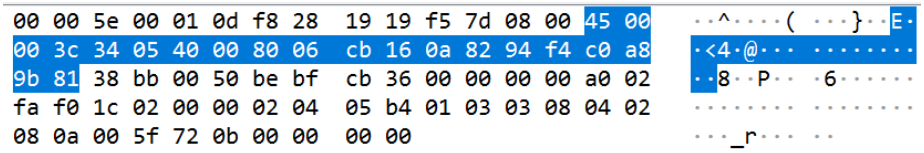


图 3 以太网 MAC 帧数据

从太网帧内容（十六进制）可以知道目的地址为：f8:28:19:19:f5:7d，源地址为：00:00:5e:00:01:0d，类型为：0x0800（即 IPv4），后面是数据（IP 数据报）。

IP 数据报首部的格式如表 2 所示：

表 2 IP 数据报首部的格式

版本 4 bit	首部长度 4 bit	区分服务 8 bit	总长度 16 bit	
标识 16 bit			标志 3 bit	片偏移 13bit
生存时间 8 bit		协议 8 bit	首部检验和 16 bit	
源地址 32 bit				
目的地址 32 bit				

从图 3 中可以看出 IP 数据报的以下信息：版本：4；首部长度：5(5*4 byte)；区分服务：00；总长度：003c(60 bytes)；标识：3405；标志：010(二进制，DF=1)；片偏移：00000 00000000；生存时间：80；协议：06(TCP)；首部检验和：cb16；源地址：0a.82.94.f4；目的地址：c0.a8.9b.81。

IP 数据报的数据部分即为 TCP 报文，TCP 报文格式如表 3 所示：

表 3 TCP 报文格式

源端口 16 bit								目的端口 16 bit
序号 32 bit								
确认号 32 bit								
数据 偏移 4 bit	保留 6 bit	U R G	A C K	P S H	P S T	S Y N	F I N	窗口 16 bit
检验和 16 bit								紧急指针 16 bit
选项 （长度可变）								填充
数据 （可变）								

点击查看第一条 TCP 数据包，如图 4 所示，可以知道以下信息：

源端口：38bb(14523)；目的端口：0050(HTTP 默认端口号 80)；序号：bebfcb36；确认号：00000000；数据偏移：a(10*4 bytes)；保留：000000（二进制）；URG：0；ACK：0；PSH：0；PST：0；SYN：1；FIN：0；窗口：faf0(64240)；检验和：

1c02；紧急指针：0000。

```
Transmission Control Protocol, Src Port: 14523, Dst Port: 80, Seq: 0, Len: 0
    Source Port: 14523
    Destination Port: 80
    [Stream index: 31]
    [TCP Segment Len: 0]
    Sequence number: 0    (relative sequence number)
    [Next sequence number: 0    (relative sequence number)]
    Acknowledgment number: 0
    1010 .... = Header Length: 40 bytes (10)
  > Flags: 0x002 (SYN)
    Window size value: 64240
    [Calculated window size: 64240]
    Checksum: 0x1c02 [unverified]
    [Checksum Status: Unverified]
    Urgent pointer: 0
  > Options: (20 bytes), Maximum segment size, No-Operation (NOP), Window scale,
  > [Timestamps]
```

```
0000  00 00 5e 00 01 0d f8 28  19 19 f5 7d 08 00 45 00   ··^····( ···}··E·
0010  00 3c 34 05 40 00 80 06  cb 16 0a 82 94 f4 c0 a8   ·<4·@··· ········
0020  9b 81 38 bb 00 50 be bf  cb 36 00 00 00 00 a0 02   ··8··P·· ·6······
0030  fa f0 1c 02 00 00 02 04  05 b4 01 03 03 08 04 02   ········ ········
0040  08 0a 00 5f 72 0b 00 00  00 00                     ···_r··· ··
```

图 4 TCP 报文

2.2 TCP 的连接管理

2.2.1 建立连接

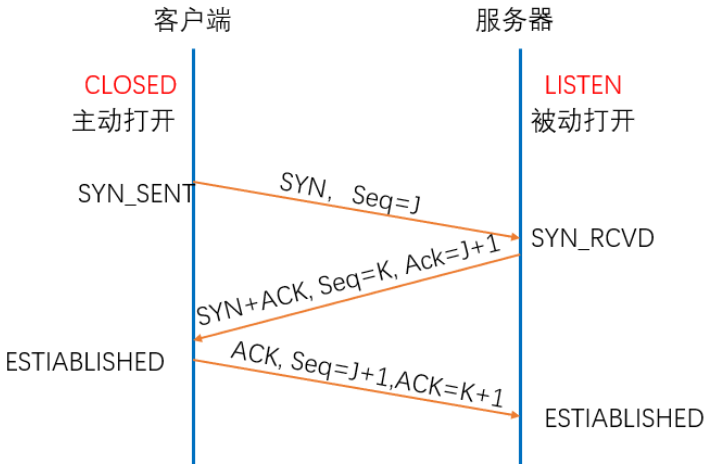


图 5 三次握手过程

第一次握手数据包如图 6 所示。客户端发送一个 TCP 数据包，标志位为 SYN，序列号为 0， 代表客户端请求建立连接。

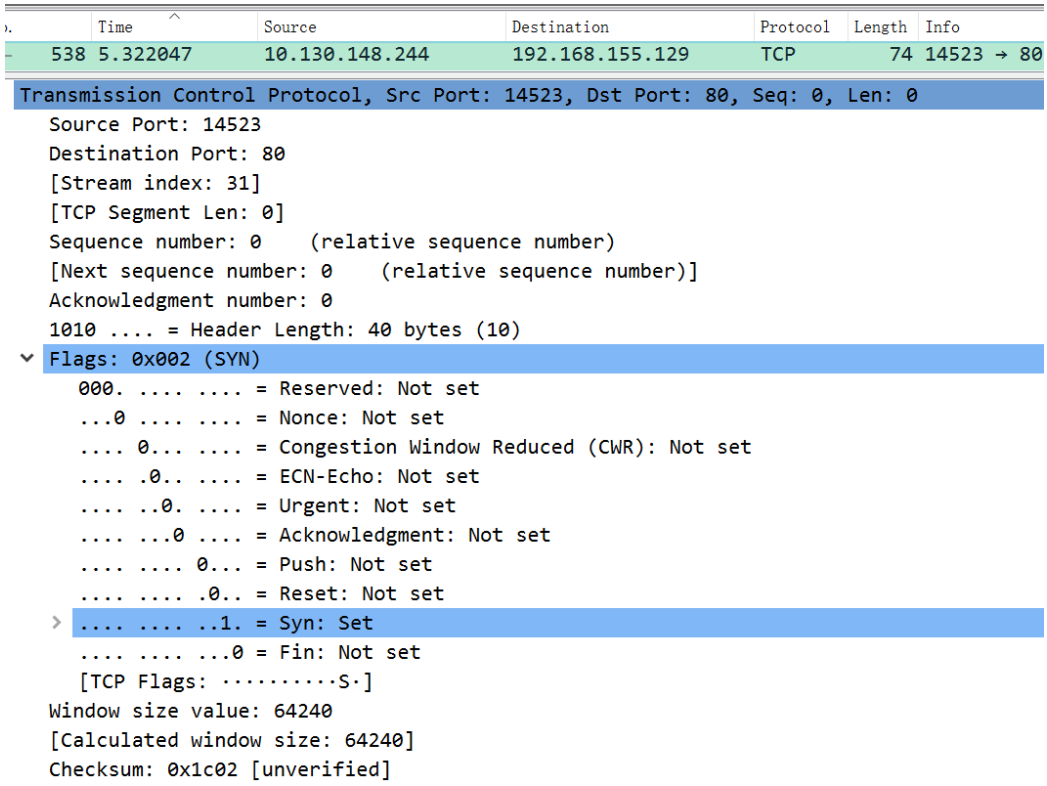


图 6 第一次握手

第二次握手的数据包如图 7 所示。服务器发回确认包，标志位为 SYN,ACK. 将确认序号(Acknowledgement Number)设置为客户的初始序号(ISN, Initial Sequence Number)加 1，即 0+1=1。

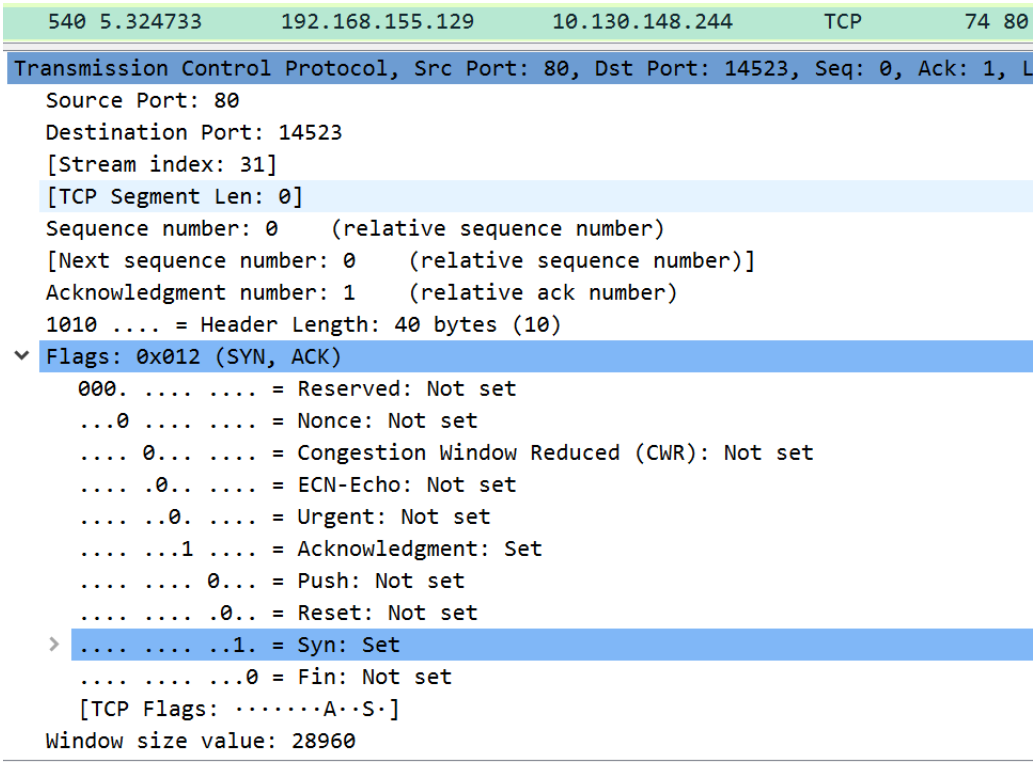


图 7 第二次握手

第三次握手的数据包如图 8 所示。客户端再次发送确认包(ACK) SYN 标志位为 0,ACK 标志位为 1.并且把服务器发来 TCP 数据包的序号字段+1,放在确定字段中发送给对方。

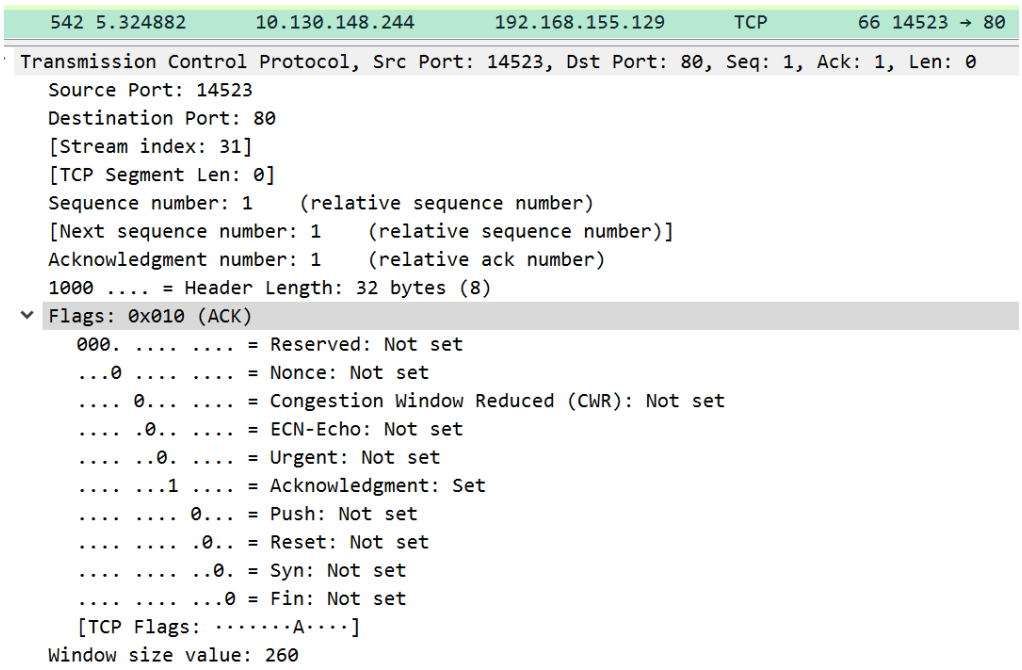


图 8 第三次握手

经过三次握手后，客户端和服务器建立了 TCP 连接。接着就可以发送数据了，因为建立连接使用了一个序列号 0，所以发送数据的第一个字节序号为 1。TCP 为应用层提供全双工服务，意味数据能在两个方向上独立地进行传输，因此连接的每一段都有各自的传输数据序号（对应于图 5 中的 J 和 K，这两个值是没有必然联系的）。

2.2.2 数据传输

客户端和服务器建立了 TCP 连接后，第四行的 HTTP 包说明已经成功建立连接，主机向服务器发送了一个 http 应用请求，如图 9 所示。在这之后，服务器收到请求，返回一个 tcp 确认帧，接着发送一个 http 应答给主机，主机收到服务器的 http 应答数据后，又发送一个 tcp 确认帧，确认收到了数据……如图 10 所示，经过多次数据传送。

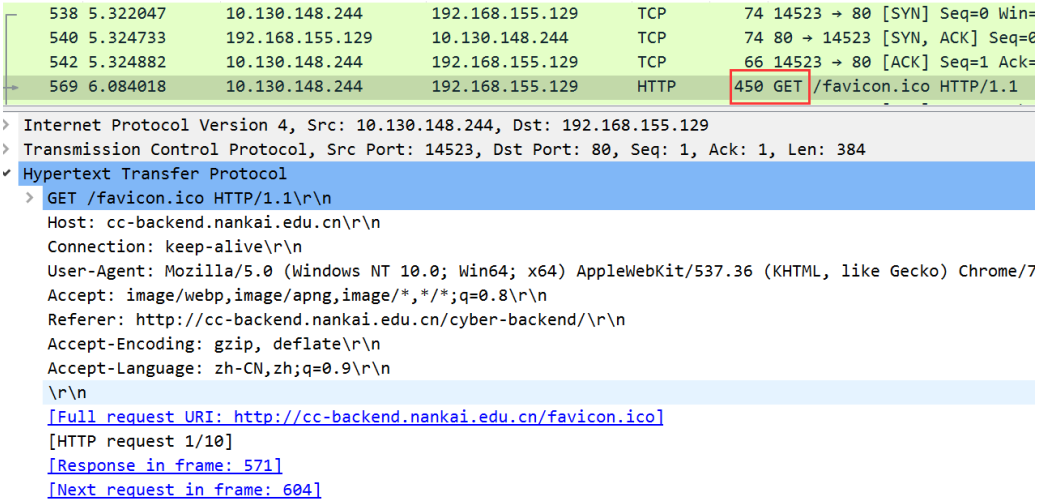


图 9 浏览器发出取文件命令

服务器给出响应，把数据发给浏览器，在后面会给出数据段的详细说明。

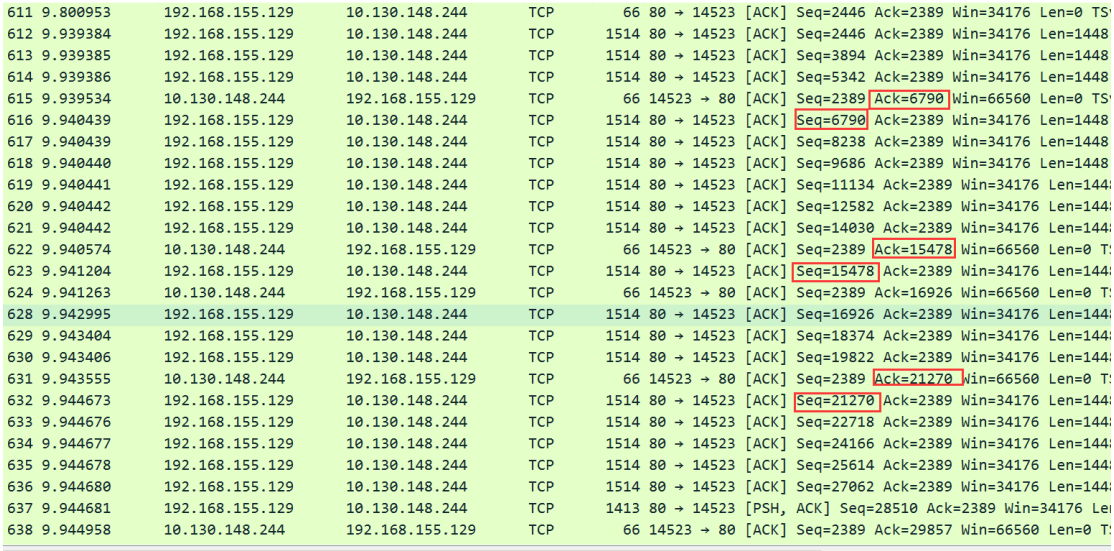


图 10 发送和接收数据包

2.2.3 关闭连接

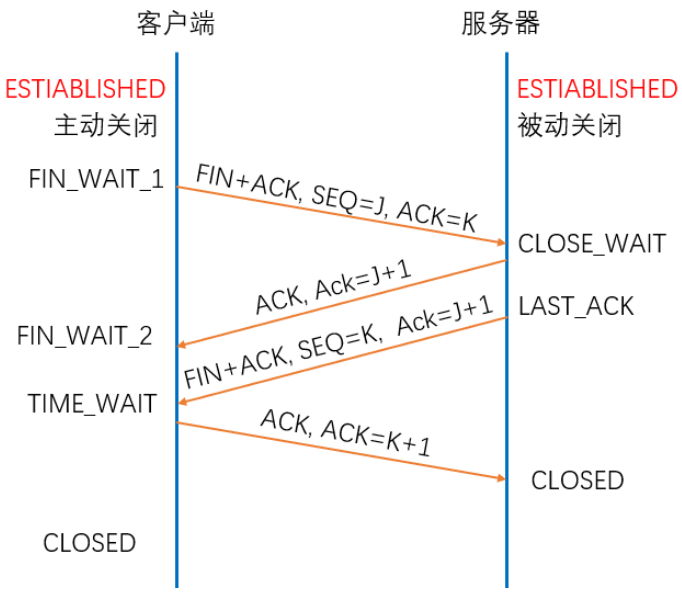


图 11 TCP 关闭连接

数据传输完毕后释放 TCP 连接，如图 12 所示。TCP 的连接是双向的，每一向流需要单独断开。关闭连接是四次挥手过程：

(1) 客户端向服务器发送 FIN 段，还会捎带之前的 ACK 包，此后，客户端不再向服务器发送数据，但仍可以接收数据。

(2) 服务器接收 FIN 端，回送 ACK 段，此时服务器仍然可以向客户端发送数据。

(3) 服务器向客户端发送 FIN 段，等待客户端返回 ACK。通过分析实际情况中的数据包，可以看出(2)(3)阶段的数据是一起发出的。

(4) 客户端接收 ACK 段，回送 ACK 段，等待两倍的段生存期，关闭连接。服务器接收 ACK，关闭连接。客户端最后一次发送 ACK 包后进入 TIME_WAIT 状态，而不是直接进入 CLOSED 状态关闭连接，这是因为客户端最后一次向服务器回传 ACK 包时，有可能会因为网络问题导致服务器收不到，服务器会再次发送 FIN 包，如果这时客户端完全关闭了连接，那么服务器无论如何也收不到 ACK 包了，所以客户端需要等待片刻、确认对方收到 ACK 包后才能进入 CLOSED 状态。

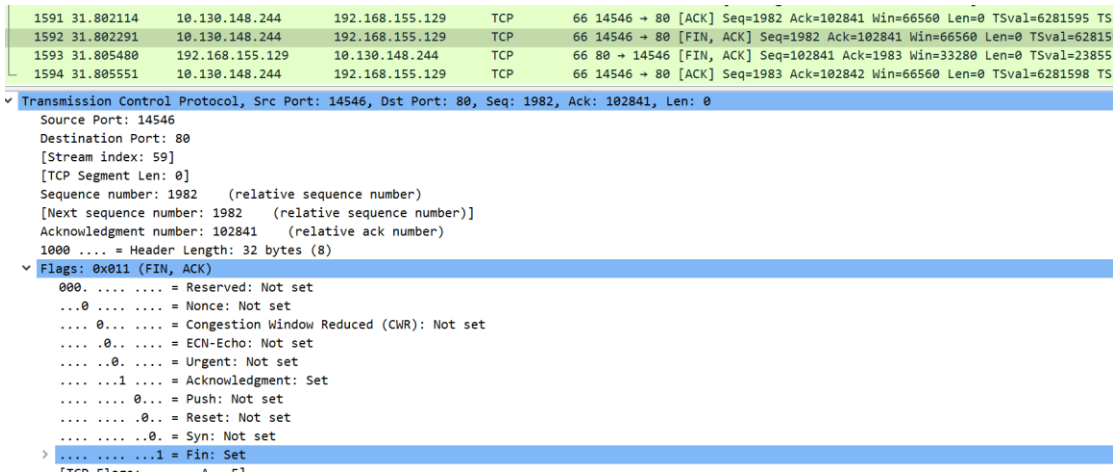


图 12 释放 TCP 连接

从上面的三个过程中可以发现 TCP 中 ACK 不占 seq，FIN 和 SYN 各占一个 seq。

另外，在课上讲的 TCP 关闭连接，是客户端主动断开，但在实验中，我发现也有服务器主动断开连接的情况，如图 13 所示，所以在网上学习了这种情况。从最开始的分析中可以知道本次实验中使用的是 http1.1，对于 http1.1 协议来说，如果响应头中的 Transfer-encoding 为 chunked 传输，则表示 body 是流式输出，body 会被分成多个块，每块的开始会标识出当前块的长度，此时，body 不需要通过长度来指定。如果是非 chunked 传输，而且有 content-length，则按照 content-length 来接收数据。否则，如果是非 chunked，并且没有 content-length，则客户端接收数据，直到服务端主动断开连接。

图 13 服务器主动断开连接

3. TCP 重要数据段说明

在这里对数据传输过程中的 TCP 数据的重要数据段做以说明。

TCP 的最大报文段长度（MSS）表示 TCP 传往另一端的最大块数据的长度。当一个连接建立时，连接的双方都要通告各自的 MSS。一般来说，MSS 越大越好，因为报文段越大允许每个报文段传送的数据就越多，相对 IP 和 TCP 首部有更高的网络利用率。MSS 选项只能出现在 SYN 报文段中，所以只能在 SYN=1 的帧中才会有 MSS 选项说明报文的最大段长度，如图 14 所示，本次 TCP 的 MSS=1460 bytes。

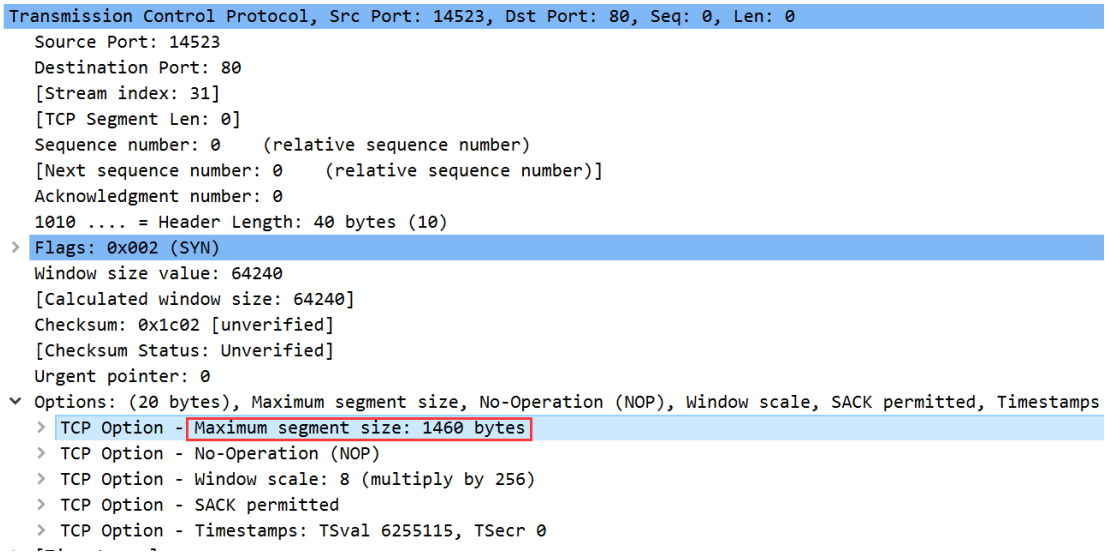


图 14 MSS 选项

TCP 数据段格式在前边的表 3 中已经给出，图 15 再次展示了 TCP 数据段格式。

其中，序号指 TCP 数据段中的“数据”部分（不包含“数据段头”部分）的第一个字节的编号，占 32 位。

确认号指期望接收到对方下一个数据段中“数据”部分的第一个字节序号，占 32 位。“序号”和“确认号”两个字段共同用于 TCP 服务中的差错控制，确保 TCP 数据传输的可靠性。

数据偏移指数据段中的“数据”部分起始处距离 TCP 数据段起始处的字节偏移量，占 4 位。因为 TCP 数据段头中有不确定的“可选项”字段，所以数据偏移字段是非常必要的。但是注意的是，数据偏移量是以 32 位（即 4 字节）为单位来计算的，4 个比特位可以表示的最大数为 15，所以数据偏移量最大为 60 字节，这也是 TCP 数据段头部分的最大长度。

ACK：Acknowledgement（确认）控制位，指示 TCP 数据段中的“确认号”字段是否有效，占 1 位。PSH：Push（推）控制位，指示是否需要立即把收到的该数据段提交给应用进程，而置 0 时没有这个要求，可以先缓存起来。RST：Reset（重置）控制位，用于重置、释放一个已经混乱的传输连接，然后重建新的传输连接，占 1 位。当 RST 位置 1 时，释放当前传输连接，然后可以重新建立新的传输连接。SYN：Synchronization（同步）控制位，用来在传输连接建立时同步传输连接序号，占 1 位。当 SYN 位置 1 时，表示这是一个连接请求或连接确认报文。当 SYN=1，而 ACK=0 时，表明这是一个连接请求数据段。如果对方同意建立连接，则对方会返回一个 SYN=1、ACK=1 的确认。FIN：Final（最后）控制位，用于释放一个传输连接，占 1 位。当 FIN 位置 1 时，表示数据已经全部传输完成，发送端没有数据要传输了，要求释放当前信号，但是接收端仍然可以继续接受还没有接受完的数据。在正常传输时，该位置 0。

“窗口大小”字段的值告诉接受本数据段的主机，从本数据段中所设置的“确认号”值算起，本端目前允许对端发送的字节数，是作为让对方设置其发送窗口大小的依据。

检验和是对“数据段头”、“数据”、和“伪头部”这三部分进行校验，占 16

位。“伪头部”包括源主机和目的主机的 32 位 IP 地址、TCP 协议号（6），以及 TCP 数据段长度。

可选项可以包括窗口缩放选项（Windows Scale Option， WSopt）、MSS（最大数据段大小）、SACK（选择性确认）选项、时间戳（Timestamp）选项等。

数据是应用层的应用进程提交的数据，作为 TCP 数据段的“数据”（有效载荷）部分。

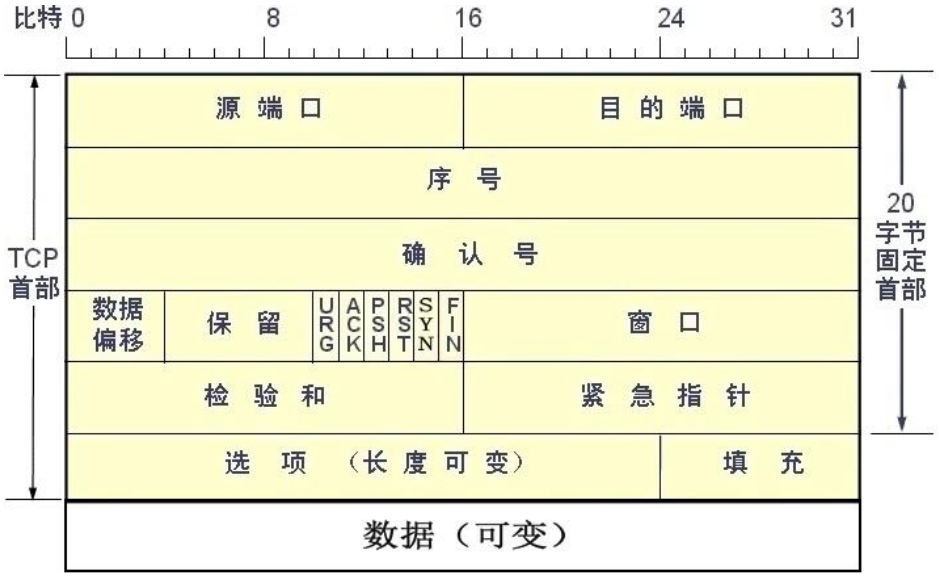


图 15 TCP 数据段格式

图片来自：http://blog.csdn.net/terrysg/article/details/47058041

从 Wireshark 捕获的数据中找一条数据，查看他的详细情况，如图 16、图 17 所示。

```
No.     Time           Source          Destination     Protocol  Length  Info
    616 9.940439       192.168.155.129 10.130.148.244  TCP       1514    80 → 14523 [ACK] Seq=6790 Ack=2389

▾ Transmission Control Protocol, Src Port: 80, Dst Port: 14523, Seq: 6790, Ack: 2389, Len: 1448
    Source Port: 80
    Destination Port: 14523
    [Stream index: 31]
    [TCP Segment Len: 1448]
    Sequence number: 6790    (relative sequence number)
    [Next sequence number: 8238    (relative sequence number)]
    Acknowledgment number: 2389    (relative ack number)
    1000 .... = Header Length: 32 bytes (8)
  › Flags: 0x010 (ACK)
    Window size value: 267
    [Calculated window size: 34176]
    [Window size scaling factor: 128]
    Checksum: 0xf5ab [unverified]
    [Checksum Status: Unverified]
    Urgent pointer: 0
  › Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
  › [SEQ/ACK analysis]
  ▾ [Timestamps]
      [Time since first frame in this TCP stream: 4.618392000 seconds]
      [Time since previous frame in this TCP stream: 0.000905000 seconds]
    TCP payload (1448 bytes)
    [Reassembled PDU in frame: 639]
    TCP segment data (1448 bytes)
```

图 16 TCP 数据（1）

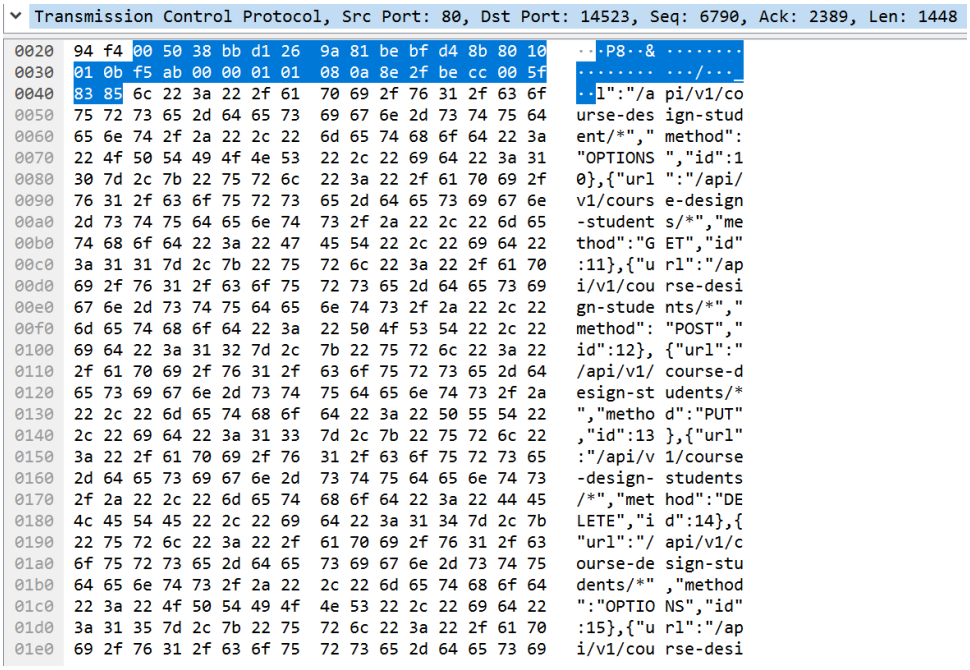


图 17 TCP 数据（2）

以图 17 中的 TCP 数据段为例，根据上面的格式可以知道：

源端口：0050(HTTP 默认端口号 80)；目的端口：38bb(14523)；序号：d1269a81；确认号：dedfd48b；数据偏移：8(8*4 bytes)，说明 TCP 数据段头的长度是 32 个字节，即图 16 中蓝色部分所示，并且由此可知选项和填充部分的长度为 12 个字节；保留：000000（二进制）；URG：0；ACK：1；PSH：0；PST：0；SYN：0；FIN：0；窗口：010b(267)；检验和：f5ab；紧急指针：0000。蓝色部分之后的即为有效载荷。对照图 15 中的信息，与分析结果是完全吻合的。

以图 18 为例，可以看见客户端与服务器的数据发送过程：客户端向服务器发送 ACK=6790，服务器回送 Seq=6790 的数据段，之后又乱序地发出了 Seq=8238、9686、11134、12582、14030 的数据段，此时客户端向服务器发送 ACK=15478，服务器回送 Seq=15478 的数据段，客户端立刻收到后，发送 ACK=16926，服务器回送 Seq=16926 的数据段……

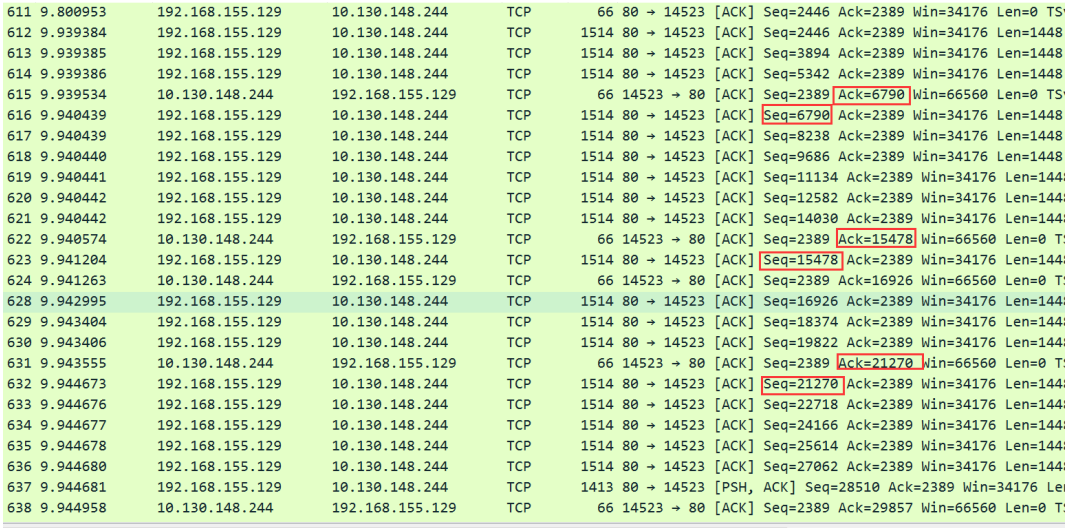


图 18 数据发送