华东师范大学计算机科学技术系上机实践报告

课程名称：计算机网络	**年级**：2022级	**上机实践成绩**：
指导教师：洪道诚	**姓名**：朱宇笑	**创新实践成绩**：
实验名称：传输控制协议TCP	**学号**：10225001410	**上机实践日期**：2023/12/15
座位编号：F	**组号**：6	**上机实践时间**：2学时

1 实验目的

1. 掌握 TCP 协议的报文格式
2. 掌握 TCP 连接的建立和释放过程
3. 掌握 TCP 数据传输中编号与确认的过程
4. 掌握 TCP 协议校验和的计算方法
5. 理解 TCP 重传机制

2 实验环境

采用网络拓扑结构一

3 实验原理

3.1 TCP 协议简介

TCP（传输控制协议）协议是 TCP/IP 协议族中的面向连接的、可靠的传输层协议。TCP 与UDP 不同，它允许发送和接收字节流形式的数据。为了使服务器和客户端以不同的速度发送和接收数据，TCP 提供了发送和接收两个缓冲区。TCP 提供全双工服务，数据同时能双向流动。通信的每一方都有发送和接收两个缓冲区，可以双向发送数据。TCP 在报文中加上一个递增的确认序列号来告诉发送端，接收端期望收到的下一个报文，如果在规定时间内，没有收到关于这个包的确认响应，则重新发送此包，这保证了 TCP 是一种可靠的传输层协议。

TCP 的常用熟知端口如下表所示：

端口	协议	端口	协议
20	FTPData	80	HTTP
21	FTPControl	110	POP3
23	TELNET	143	IMAP
25	SMTP		

图 1：TCP 常用熟知端口

3.2 TCP 报文格式

TCP 报文包括 20 ～60 字节的首部，接着是应用程序的数据部分。首部在没有选项时是 20
字节，而当有选项时长度会增加，但是最大不会超过 60 字节。

TCP 报文的格式如下图所示：

<table>
<tr><td colspan="5">源端口（16 位）</td><td colspan="5">目的端口（16 位）</td></tr>
<tr><td colspan="10">序列号（32 位）</td></tr>
<tr><td colspan="10">确认号（32 位）</td></tr>
<tr><td>首部长度
（4 位）</td><td>保留
（4 位）</td><td>CWR</td><td>ECE</td><td>URG</td><td>ACK</td><td>PSH</td><td>RST</td><td>SYN</td><td>FIN</td><td>窗口大小（16 位）</td></tr>
<tr><td colspan="5">校验和（16 位）</td><td colspan="5">紧急指针（16 位）</td></tr>
<tr><td colspan="10">选项和填充</td></tr>
</table>

图 2：TCP 报文格式

- 源端口：该字段定义了在主机中发送这个报文的应用程序端口号。
- 目的端口：该字段定义了数据报发往的主机中接收这个报文的应用程序的端口号。
- 序列号：该字段定义了指派给本报文第一个数据字节的一个序号。TCP 是流式传输协议， 为了保证连通性，要在发送的每一个字节上编号。序号指定了这个序列中的哪一个字节是报文的第一个字节。在连接建立时，双方使用随机数产生器产生初始序号，通常每一方的初始序号都是不同的。
- 确认号：该字段定义了报文的接收端期望从对方接收的序号。如果报文的接收端成功地接收了对方发来的序号为 x 的报文，它就把确认号定义为 x+1。确认可以和数据一起发送。
- 首部长度：该字段指定 TCP 首部的长度，以 4 字节为单位。首部长度可以在 20 ～60 字节之间。因此，这个字段的值可以在 5 至 15 之间。
- 保留：这是 6 位字段，保留为今后使用。
- 控制：这个字段定义了 8 种不同的标志。如下图所示。在同一时间可设置一位或多位标志。

图 3：控制字段

这些标志用在 TCP 的流量控制、连接建立和终止以及数据传送的方式等方面。下表给出了每一位的简要说明。

标志	说明
CWR	拥塞窗口减小（用来表明发送主机接收到了设置 ECE 标志的 TCP 包。拥塞窗口是被 TCP 维护的一个内部变量，用来管理发送窗口大小）
ECE	经历拥塞回送（用来在 TCP3 次握手时表明一个 TCP 端是具备 ECN 功能的，并且表明接收到的 TCP 包的 IP 头部的 ECN 被设置为 11）
URG	紧急指针字段值有效
ACK	确认字段值有效
PSH	推送数据
RST	连接必须复位
SYN	在连接建立时对序号进行同步
FIN	终止连接

图 4：TCP 标志位

- 窗口大小：该字段定义对方必须维持的窗口值（以字节为单位）。这个字段的长度是 16 位，因此窗口值的最大长度是 65535 字节。这个值通常是作为接收窗口，并由接收端来确定。这时，发送端必须服从接收端的决定。
- 校验和：该字段的校验范围包括伪首部、TCP 首部和 TCP 数据部分。
- 紧急指针：只有当紧急标志置位时，这个 16 位字段才有效，这时的报文中包括紧急数据。
- 选项：在 TCP 首部中可以有多达 40 字节的可选信息。

3.3 TCP 封 装

TCP 报文封装在 IP 数据报中，然后再封装成数据链路层中的帧，如下图所示：

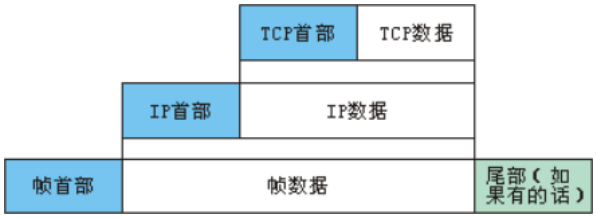

图 5：TCP 封装

3.4 TCP 校验和

TCP 的校验和与 UDP 的校验和计算过程是一样的。但是，UDP 是否使用校验和是可选的， 而 TCP 是否使用校验和则是强制性的。在计算 TCP 校验和时也要在报文上添加伪首部。对于TCP 的伪首部，高层协议类型字段的值是 6。如下图所示：

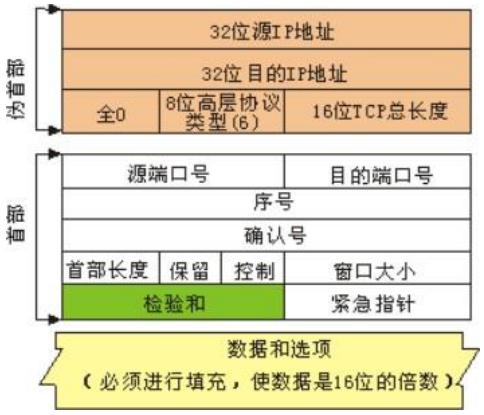

图 6：伪首部加到 TCP 报文上

3.5 TCP 连接建立与释放

3.5.1 连接建立

TCP 以全双工方式传送数据。当两个进程建立了 TCP 连接后，它们能够同时向对方发送数据。在传送数据之前，双方都要对通信进行初始化，得到对方的认可。

3.5.2 三次握手

TCP 的连接建立过程叫做三次握手。服务器程序首先准备好接受 TCP 连接，这个过程叫做被动打开请求。这时，服务器的 TCP 就已准备好接受任何一台主机的 TCP 连接了。客户程序发出 TCP 连接请求的过程叫做主动打开。然后服务器与客户端就开始三次握手过程，如下图所示

(在图中客户端与服务器端各使用一条时间线，并给出每个阶段的几个重要字段，包括序号、确认号、控制标志以及非零的窗口值)。这个过程有以下 3 个步骤。

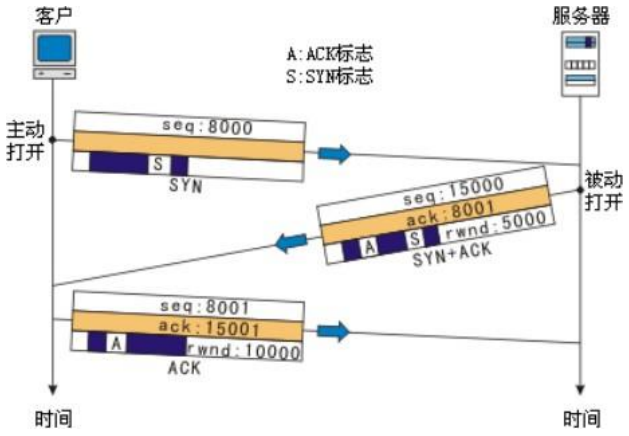

图 7：使用三次握手的连接建立

1. 客户发送第一个报文，这是一个 SYN 报文，在这个报文中只有 SYN 标志置为 1。这个报文的作用是使序号同步。

2. 服务器发送第二个报文，即 SYN+ACK 报文，其中 SYN 和 ACK 标志被置为 1。这个报文有两个目的。首先，它是一个用来和对方进行通信的 SYN 报文。服务器使用这个报文同步初始序号，以便从服务器向客户发送字节。服务器还使用 ACK 标志确认已从客户端收到了 SYN 报

文，同时给出期望从客户端收到的下一个序号。另外，服务器还定义了客户端要使用的接收窗口的大小。

3. 客户发送第三个报文。这仅仅是一个 ACK 报文。它使用 ACK 标志和确认号字段来确认收到了第二个报文。

3.5.3 连接终止

通信双方中的任何一方都可以关闭连接。当一方的连接被终止时，另一方还可继续向对方发送数据。TCP 的连接终止有两种方式：三次握手和具有半关闭的四次握手。

3.5.4 三次握手方式终止连接

使用三次握手的 TCP 终止过程如下图所示：

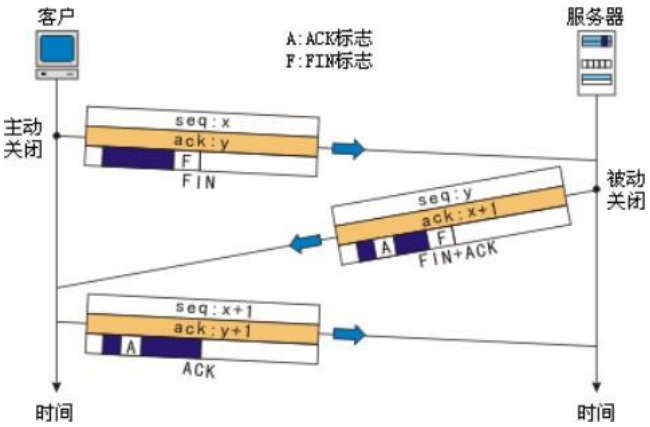

图 8：使用三次握手的连接终止

1. 当客户端想关闭 TCP 连接时，它发送一个 TCP 报文，把 FIN 标志位设置为 1。

2. 服务器端在收到这个 TCP 报文后，把 TCP 连接即将关闭的消息发送给相应的进程，并发送第二个报文——FIN+ACK 报文，以证实从客户端收到了 FIN 报文，同时也说明，另一个方向的连接也关闭了。

3. 客户端发送最后一个报文以证实从 TCP 服务器收到了 FIN 报文。这个报文包括确认号， 它等于从服务器收到的 FIN 报文的序号加 1。

3.5.5 半关闭的四次握手方式终止连接

在 TCP 连接中，一方可以终止发送数据，但仍然保持接收数据，这就叫做半关闭。半关闭通常是由客户端发起的。下图描绘了半关闭的过程。客户发送 FIN 报文，半关闭了这

个连接。服务器发送 ACK 报文接受这个半关闭。但是，服务器仍然可以发送数据。当服务器已经把所有处理的数据都发送完毕时，就发送 FIN 报文，客户端发送 ACK 报文给予确认。

在半关闭一条连接后，客户端仍然可以接收服务器发送的数据，而服务器也可以接收客户端发送的确认。但是，客户端不能传送数据给服务器。

3.6 流量控制

在发送端收到接收端的确认报文之前，流量控制可以对发送端发送的数据量进行管理。

在不考虑流量控制的情况下，传输层协议可以每次只发送一个字节的数据，然后在发送下一个字节数据之前等待接收端的确认报文。这是一个非常缓慢的过程，如果数据要走很长的距离，发送端就要在等待确认报文时一直处在空闲状态。还有一种情况是传输层协议一次就将全部数据发送出去，而不理会确认报文。这样虽然加速了发送过程，但可能会使接收端来不及接收而瘫痪。此外，若有一部分数据丢失、重复、失序或损坏，发送端就要一直等到接收端将全部数据都检查完毕后才能知道。

TCP 的流量控制采用一种折中的方法。它在缓存上定义一个窗口。缓存是用来暂时存放将要发送的数据的。TCP 发送数据的多少由这个窗口决定。

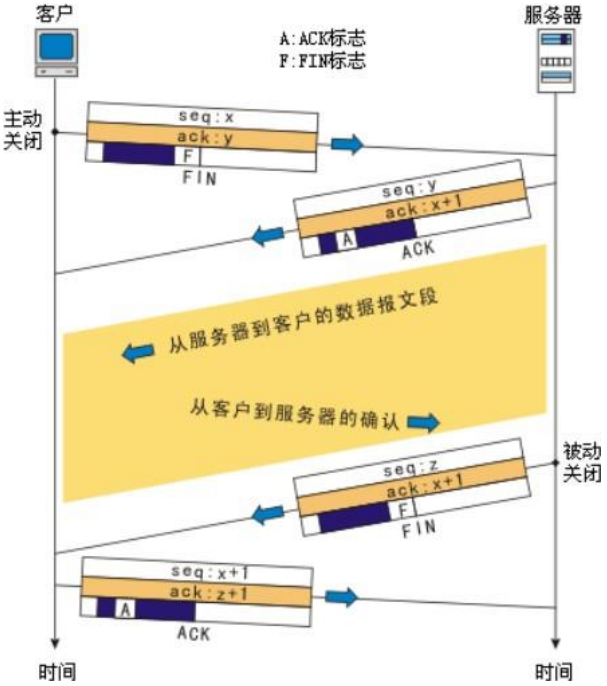

图 9：半关闭

3.6.1 滑动窗口协议

为了完成流量控制，TCP 使用滑动窗口协议。窗口覆盖了缓存的一部分，在这个窗口中的数据是可以发送而不必考虑确认的。窗口有两个沿：一个在左边，另一个在右边。因为左沿和右沿都是可以滑动的，所以这个窗口叫做滑动窗口。如下图所示：

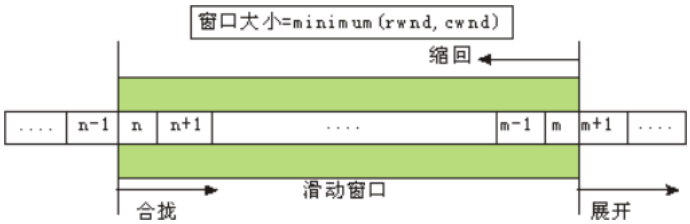

图 10：滑动窗口

窗口有三种动作：展开、合拢或缩回。这三种动作受接收端的控制而不是发送端的控制。展开窗口表示窗口的右沿向右移动，这样就可以从缓存中发送更多的数据。合拢窗口表示窗
口的左沿向右移动，这表示某些数据已经被确认了，发送端可以不再担心它们。缩回窗口表示窗

口的右沿向左移动，这在某些实现中是不允许的，因为这会使某些可以发送的数据变成不能发送的。如果发送端已经发送了这些字节，就会产生错误。窗口的左沿不能向左移动，因为这表示已经发送出去的并且经过确认的数据现在又要收回了。

窗口大小由接收窗口和拥塞窗口两者中的较小者决定。接收窗口大小由接收方发送的确认报文中的窗口大小字段值所确定。这是接收端在缓存溢出导致数据被丢弃之前所能接受的最大字节数。拥塞窗口大小是由网络根据拥塞情况而确定的。

3.7 差错控制

TCP 是可靠的传输层协议。应用程序把数据流交付给 TCP 后，就依靠 TCP 把整个数据流交付给接收端的应用程序，并且保证数据流是按序的、没有差错的、也没有任何一部分是丢失的或重复的。

TCP 使用差错控制提供可靠性。差错控制包括以下的一些机制：检测受到损伤的报文、丢失的报文、失序的报文和重复的报文。差错控制还包括检测出差错后纠正差错的机制。TCP 的差错检测和差错纠正是通过校验和、确认以及超时重传三种机制实现的。

3.7.1 校验和

每一个 TCP 报文都包括校验和字段，用来检查报文是否损坏。若报文损坏，接收端就将报文丢弃，并认为这个报文丢失了。

3.7.2 确认

TCP 采用确认报文的方法来证实收到了数据报文。确认报文不携带数据，但消耗一个序号。除了 ACK 报文之外，确认报文也需要被确认。

3.7.3 重传

差错控制的核心是报文的重传机制。当一个报文损坏、丢失或延迟时，就需要重传这个报文。有两种情况需要对报文进行重传：当重传超时计时器时间到期时，或当发送端收到了 3 个重复的确认报文时。

1. 重传超时计时器到期之后的重传。发送端为每一个 TCP 报文都设置一个重传超时计时器。若计时器时间到期时还没有收到对这个报文的确认报文，就认为这个报文丢失了，于是重传这个报文，即使可能由于报文延迟到达，或确认报文延迟到达，或确认报文丢失等原因。重传超时计时器的值是动态的，它根据报文的往返时间而更新。报文的往返时间是报文离开发送端到发送端收到此报文的确认报文所需的时间。

2. 三个重复的确认报文之后的重传。一个报文的丢失会导致接收端收到的报文失序，这时接收端会发送对丢失报文的确认报文，当发送端收到 3 个重复的确认报文之后，发送

端立即重传这个报文，这叫做快重传。对不消耗序号的报文不进行重传。对所有 ACK 报文都不进行重传。

4 实验内容

4.1 察看 TCP 连接的建立和释放

各主机打开工具区的“拓扑验证工具”，选择相应的网络结构，配置网卡后，进行拓扑验证， 如果通过拓扑验证，关闭工具继续进行实验，如果没有通过，请检查网络连接。

本练习将主机 A 和 B 作为一组，主机 C 和 D 作为一组，主机 E 和 F 作为一组。现仅以主机 A、B 为例，其它组的操作参考主机 A、B 的操作。

1. 主机 B 启动协议分析器捕获数据，并设置过滤条件（提取 TCP 协议）。主机 B 在命令行下输入：netstat -a -n 命令来查看主机 B 的 TCP 端口号。

2. 主机 A 启动 TCP 工具连接主机 B。主机 A 启动实验平台工具栏中的“TCP 工具”。选中“客户端”单选框，在“地址”文本框中填入主机 B 的 IP 地址，在“端口”文本框中填入主机 B 的一个 TCP 端口，点击 [连接] 按钮进行连接。

3. 察看主机 B 捕获的数据，填写下表：

字段名称	报文 1	报文 2	报文 3
序列号			
确认号			
ACK			
SYN			

图 11：实验结果

- TCP 连接建立时，前两个报文的首部都有一个“最大字段长度”字段，它的值是多少？作用是什么？结合 IEEE802.3 协议规定的以太网最大帧长度分析此数据是怎样得出的。

4. 主机 A 断开与主机 B 的 TCP 连接。

5. 察看主机 B 捕获的数据，填写下表。

字段名称	报文 4	报文 5	报文 6	报文 7
序列号				
确认号				
ACK				
FIN				

图 12：实验结果

- 结合步骤 3、5 所填的表，理解 TCP 的三次握手建立连接和四次握手的释放连接过程，理解序号、确认号等字段在 TCP 可靠连接中所起的作用。

5 实验实骤

5.1 察看 TCP 连接的建立和释放

主机 F 在命令行下输入：netstat -a -n 命令来查看主机 B 的 TCP 端口号。如图可以看到 IP地址为 172.16.0.62 的 TCP 端口号为 135：

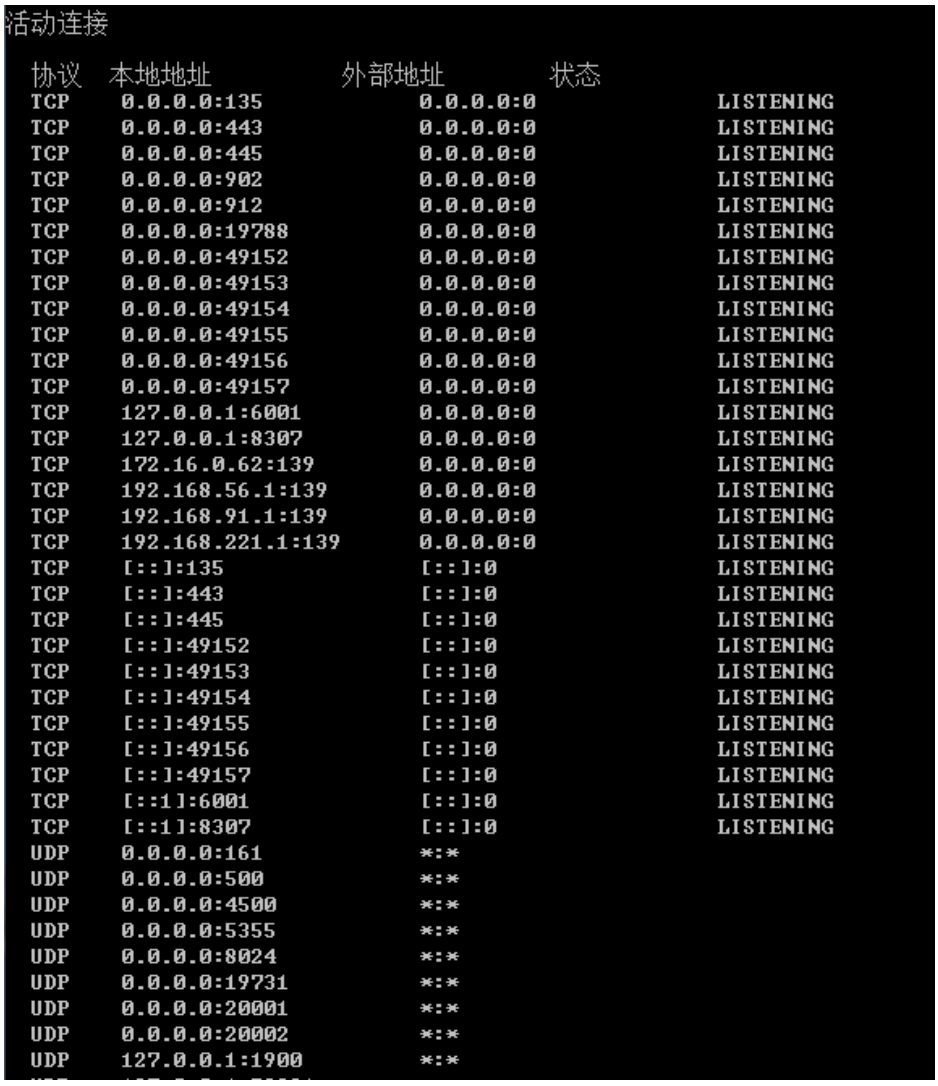

图 13：主机 B 的 TCP 端口号为 135

主机 A 启动实验平台工具栏中的“TCP 工具”。选中“客户端”单选框，在“地址”文本框中填入主机 B 的 IP 地址即]172.16.0.62，在“端口”文本框中填入主机 B 的一个 TCP 端口例如135，点击 [连接] 按钮进行连接。

主机 A 断开与主机 B 的 TCP 连接。主机 A主动点击断开后会看到主机 A 先发送给 B 一个 FIN 的请求，然后收到主机 B 的FIN，并回馈 ACK。（这次实验断开似乎丢包了）

6 实验总结与思考

6.1 察看 TCP 连接的建立和释放

连接过程 TCP 报文信息截图如下：

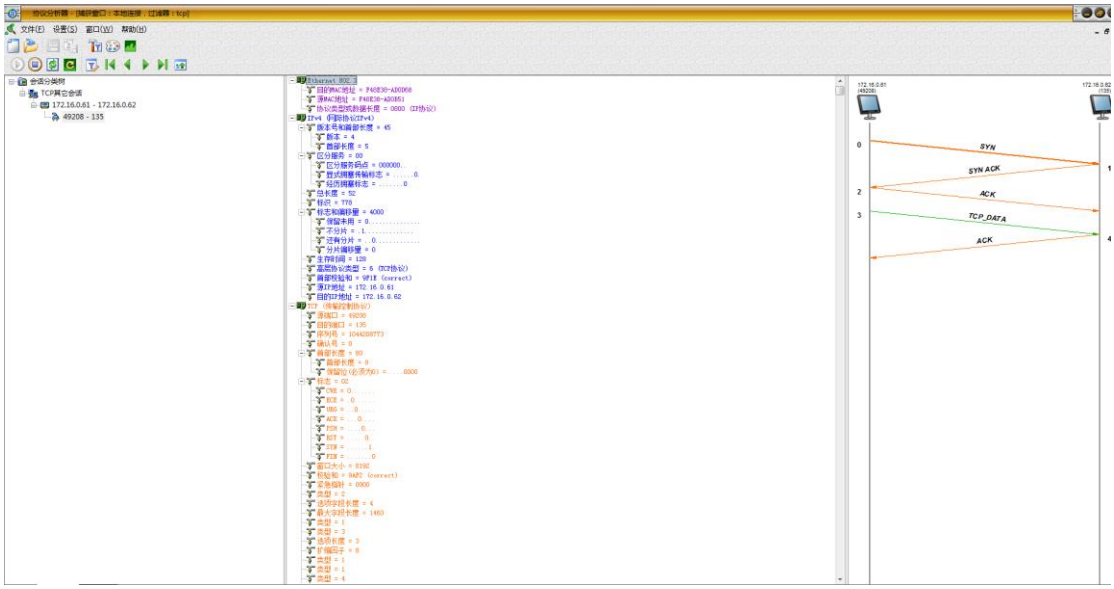

图 14：连接过程 TCP 报文信息（一）

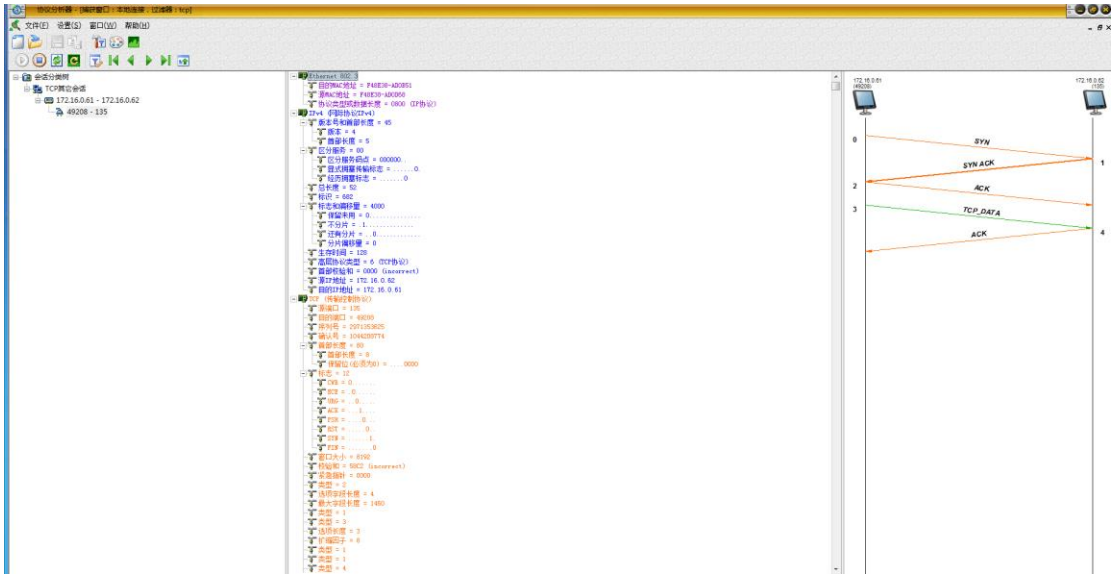

图 15：连接过程 TCP 报文信息（二）

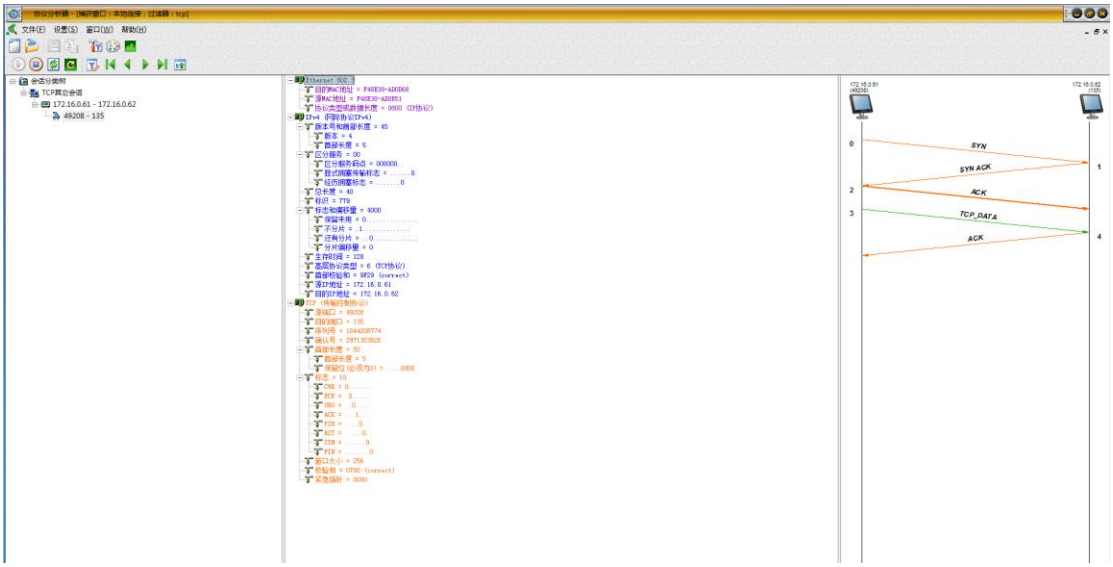

图 16：连接过程 TCP 报文信息（三）

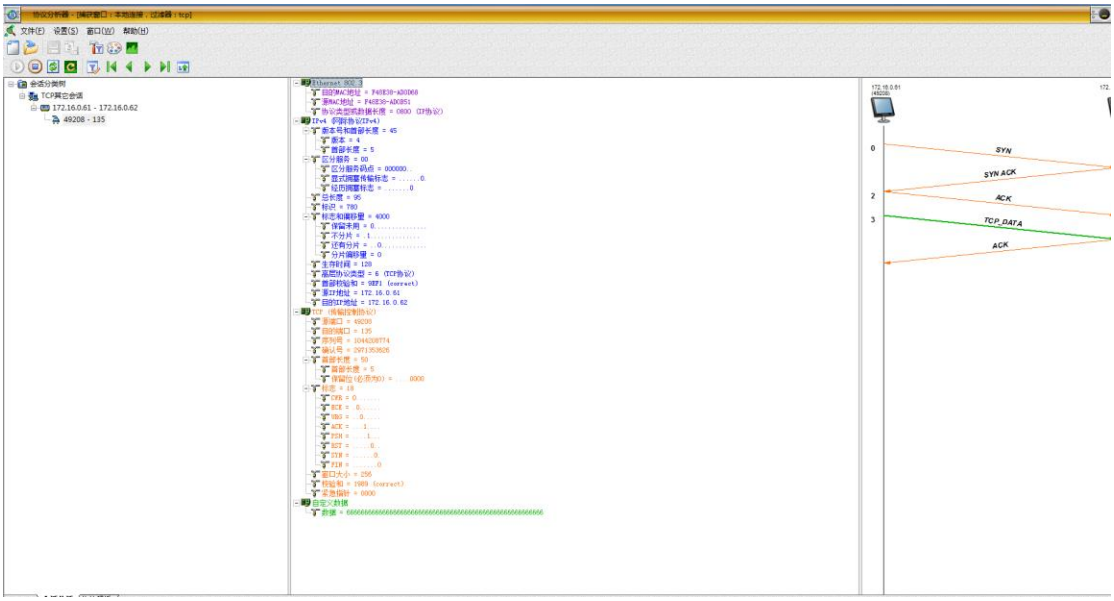

图 17：连接过程 TCP 报文信息（四）

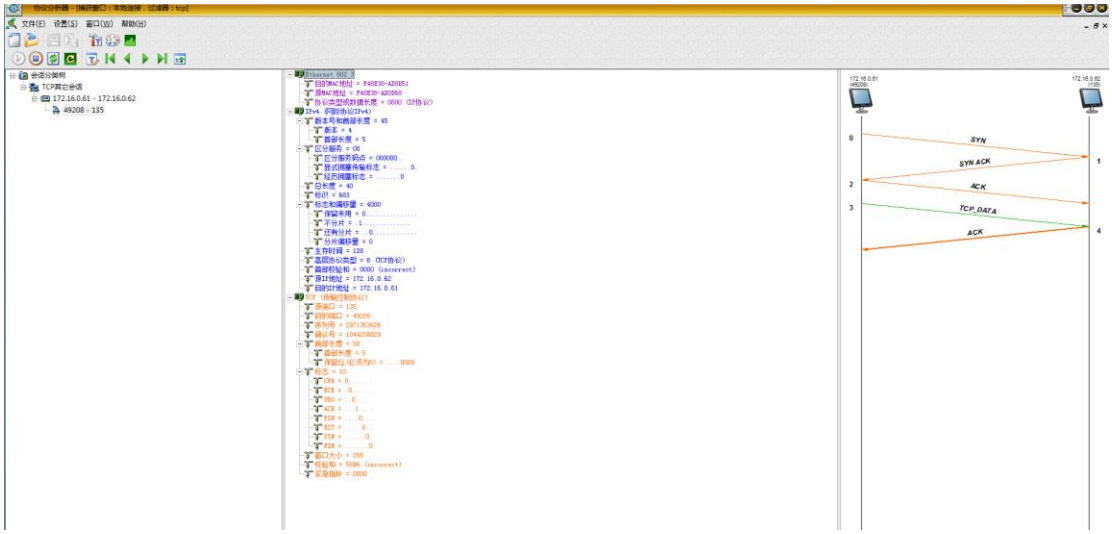

图 18：连接过程 TCP 报文信息（五）

根据 TCP 报文中显示的信息将 TCP 连接过程的实验记录如下表：

字段名称	报文 1	报文 2	报文 3
序列号	1044208773	2971353625	1044208774
确认号	0	1044208774	2971353626
ACK	0	1	1
SYN	1	1	0

- **TCP 连接建立时，前两个报文的首部都有一个“最大字段长度”字段，它的值是多少？**

  1460。

- **作用是什么？**

  指出最大报文的长度，加上首部 40，就是最大传输单元的大小 MTU=1500。

- **结合 IEEE802.3 协议规定的以太网最大帧长度分析此数据是怎样得出的。**

  IEEE802.3 协议规定的以太网最大帧长度规定的最大传输单元是 1500，减去 IP 首部 20、TCP首部 20，所以最大报文长度是 1460。

断开过程 TCP 报文信息截图如下：

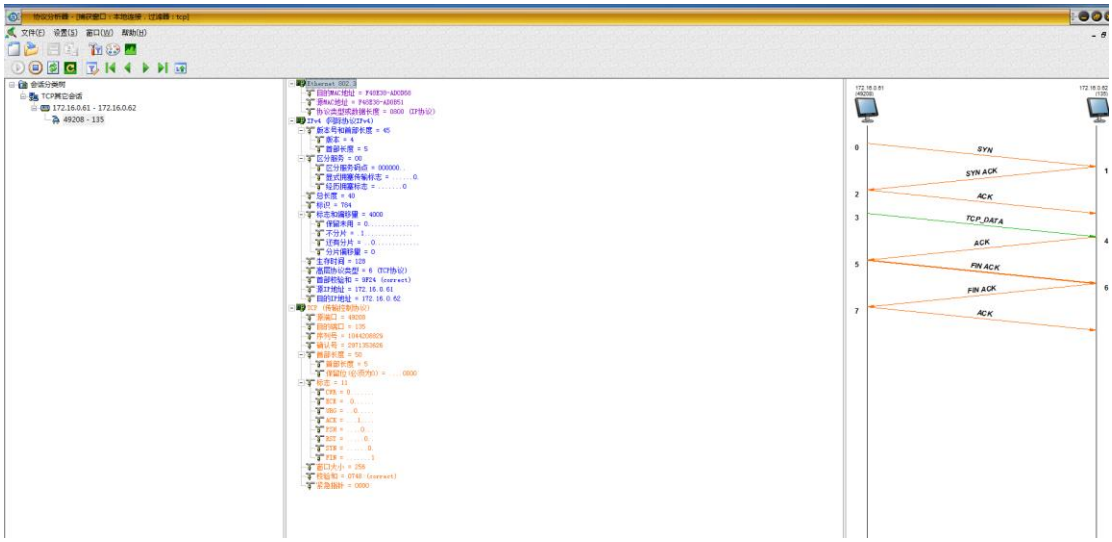

图 19：断开过程 TCP 报文信息（一）

图 19：断开

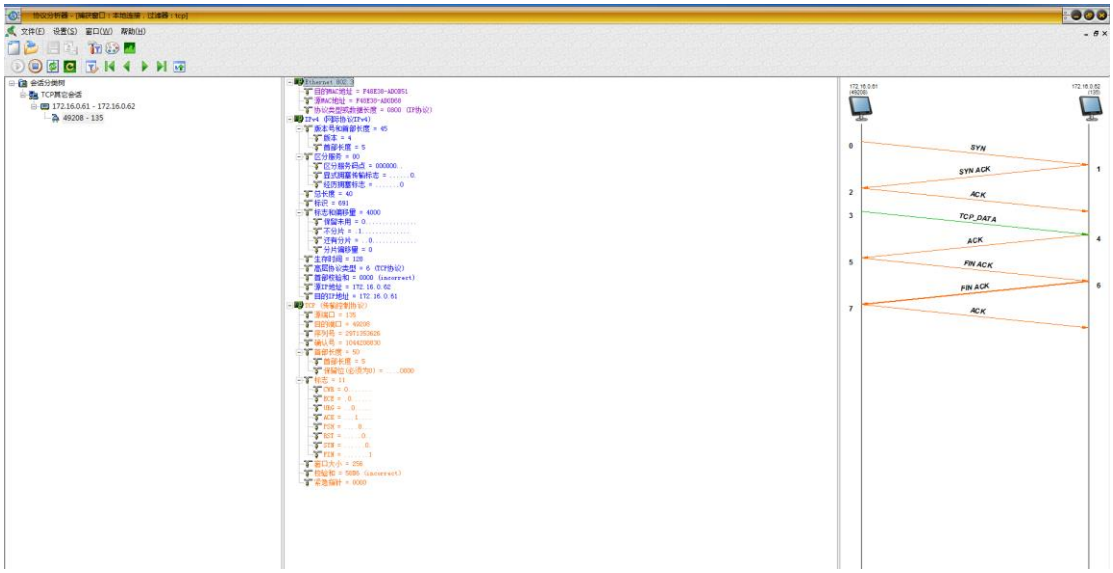

图 20：断开过程 TCP 报文信息（二）

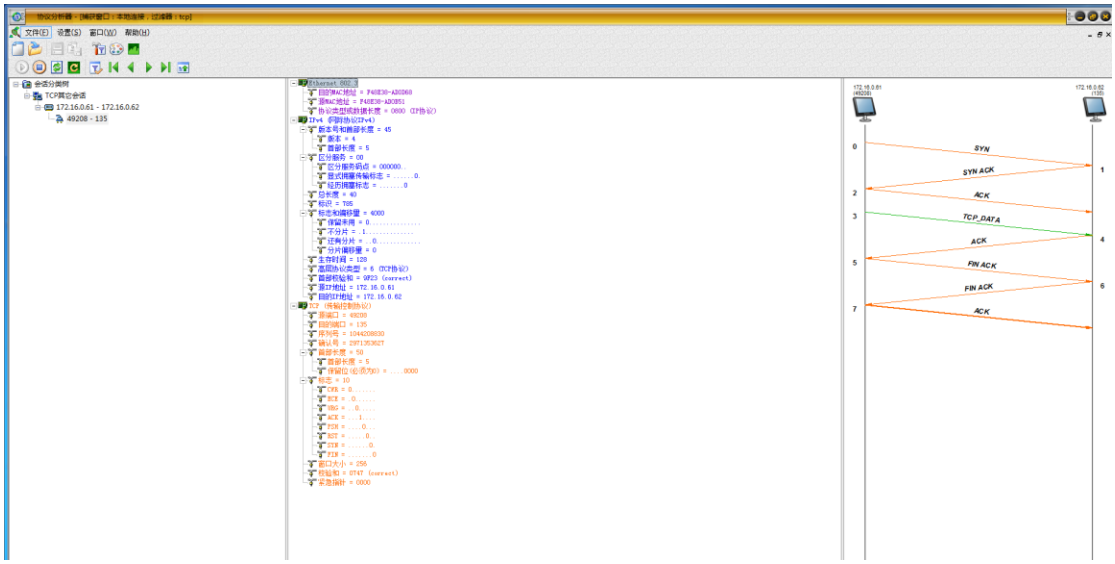

图 21：断开过程 TCP 报文信息（三）

根据 TCP 报文中显示的信息将 TCP 断开过程的实验记录如下表：

字段名称	报文 4	报文 5	报文 6
序列号	1044208829	2971353626	1044208830
确认号	2971353626	1044208830	2971353627
ACK	1	1	1
SYN	0	0	0

- **结合步骤 3、5 所填的表，理解 TCP 的三次握手建立连接和四次握手的释放连接过程，理解序号、确认号等字段在 TCP 可靠连接中所起的作用。**

  序号和确认号可以唯一的标识并确定当前的报文，由于 TCP 是可靠的协议，但是可能出现分组的丢失和乱序，为了保证当前的报文是按照正确顺序的，所以需要序号和确认号。

- **为什么在 TCP 连接中需要 3 次握手，如不这样做可能会出现什么情况？**

  为了实现可靠数据传输，TCP 协议的通信双方，都必须维护一个序列号，以标识发送出去的数据包中，哪些是已经被对方收到的。三次握手的过程即是通信双方相互告知序列号起始值，并确认对方已经收到了序列号起始值的必经步骤如果只是两次握手，至多只有连接发起方的起始序列号能被确认，另一方选择的序列号则得不到确认。

  如果两次握手就可以建立连接：

  那么 A 的一个超时连接发送给 B，B 就会认为这个连接是新的建立连接的请求并没有超时，然后建立连接，但是超时连接是对于 A 来说的，A 知道它发送的连接已经超时，只是B 不知道，此时 B 单方面建立连接，不过 A 并没有，此时不仅是连接超时，而且 B 还建立了脏连接。

  如果是三次握手的话：

  A 的超时连接发送给 B，B 发送确认连接，因为 A 知道已经超时，所以 A 不会理会 B

  发来的握手，那么 B 等待一段时间后发现自己的连接超时，连接就没有建立。

- **解释 TCP 协议的释放过程。**

1）客户端 A 的 TCP 进程先向服务端发出连接释放报文段，并停止发送数据，主动关闭 TCP 连接。

2）服务器端 B 收到连接释放报文段后即发出确认释放连接的报文段。然后服务器端 B 进入 CLOSE—WAIT（关闭等待）状态，此时 TCP 服务器进程应该通知上层的应用进程，因而客户端 A 到服务器端 B 这个方向的连接就释放了，这时 TCP 处于半关闭状态，即客户端 A 已经没有数据要发了，但服务器端 B 若发送数据，客户端 A 仍要接受，也就是说从服务器端 B 到客户端 A 这个方向的连接并没有关闭，这个状态可能会持续一些时间。

3）客户端 A 收到服务器端 B 的确认后，就进入了 FIN—WAIT（终止等待 2）状态，等待服务器端 B 发出连接释放报文段，如果服务器端 B 已经没有要向客户端 A 发送的数据了，其应用进程就通知 TCP 释放连接。这时服务器端 B 发出的链接释放报文段。这时服务器端 B 进入 LAST—ACK（最后确认）状态，等待客户端 A 的确认。

4）客户端 A 收到服务器端 B 的连接释放请求后，必须对此发出确认。这时候，TCP 连接还没有释放掉，必须经过时间等待计时器设置的时间 2MSL 后，客户端 A 才进入 CLOSED 状态，时间 MSL 叫做最长报文寿命，RFC 建议设为 2 分钟，因此从 客户端 A 进入 TIME—WAIT 状态后，要经过 4 分钟才能进入到 CLOSED 状态，而服务器端 B 只要收到了客户端 A 的确认后，就进入了 CLOSED 状态。二者都进入 CLOSED 状态后，连接就完全释放了。