# 华东师范大学软件工程学院实验报告

|  |  |  |
| --- | --- | --- |
| **实验课程**：计算机网络实践 | **年级**：2023级 | **实验成绩**： |
| **实验名称**：TCP | **姓名**：陆尚辰 |  |
| **实验编号**：Lab6 | **学号**：10235101545 | **实验日期**：2024.12.27 |
| **指导教师**：刘献忠 | **组号**： | **实验时间**： |

**一、实验目的**

1. 学会通过Wireshark获取TCP消息

2. 掌握TCP数据包结构

3. 掌握TCP数据包各字段的含义

4. 掌握TCP连接建立和释放的步骤

5. 掌握TCP数据传输阶段的过程

**二、实验内容与实验步骤**

**1.获取TCP消息**

（1）以 http://old.ecnu.edu.cn/site/xiaoli/2017.jpg为例，用wget确认该链接有效。或者wget www.baidu.com

（2） 启动Wireshark，在菜单栏的捕获->选项中进行设置，选择已连接的以太网，设置捕获过滤器为tcp port 80，我们主要观察客户端与服务器之间的tcp流

（3）捕获开始后，重复第一步，重新发送请求

（4）当wget命令结束后，停止wireshark捕获

**2.分析TCP包**

选择一个tcp帧，观察其协议层，画出TCP数据段的结构。

**3. TCP连接的建立**

问题：

（1）观察客户端与服务器的连接建立过程，画出三次握手协议的步骤图

（2）观察TCP数据段中的option字段。

**4.TCP连接的释放**

问题：

观察客户端与服务器连接的释放过程，画出释放连接的步骤图

**5.TCP数据传输**

问题：

1）实验中的下载速率是多少？ Bits/s & packets/s

2）实验中的上传速率，即ACK消息的发送速率是多少？

3）观察数据传输过程中Acknowledgment number，sequence number以及 Segment Len之间的变化。如果最近从服务器收到的TCP数据段的序列号 是X，那么下一个发送的ACK是多少？

**三、实验环境**

1. Wireshark v2.0.2

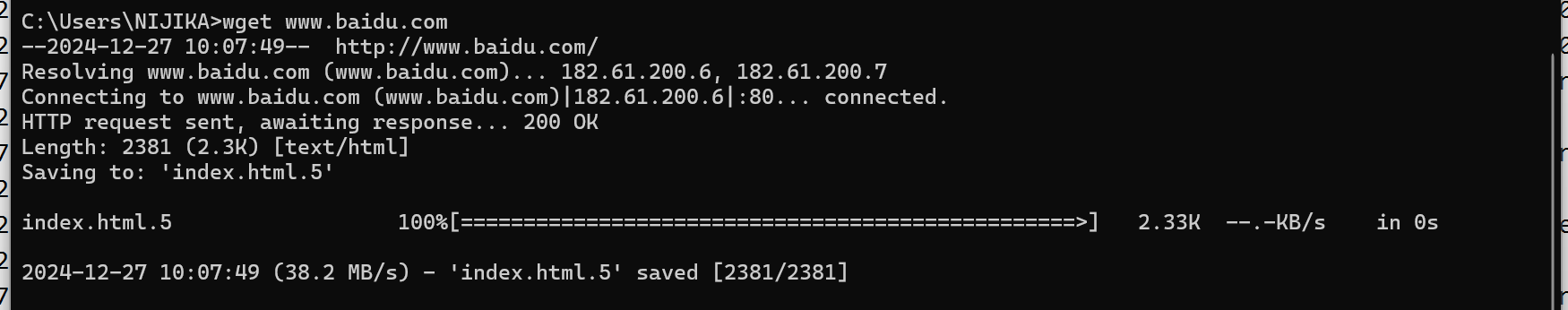
2. Windows操作系统

3. wget

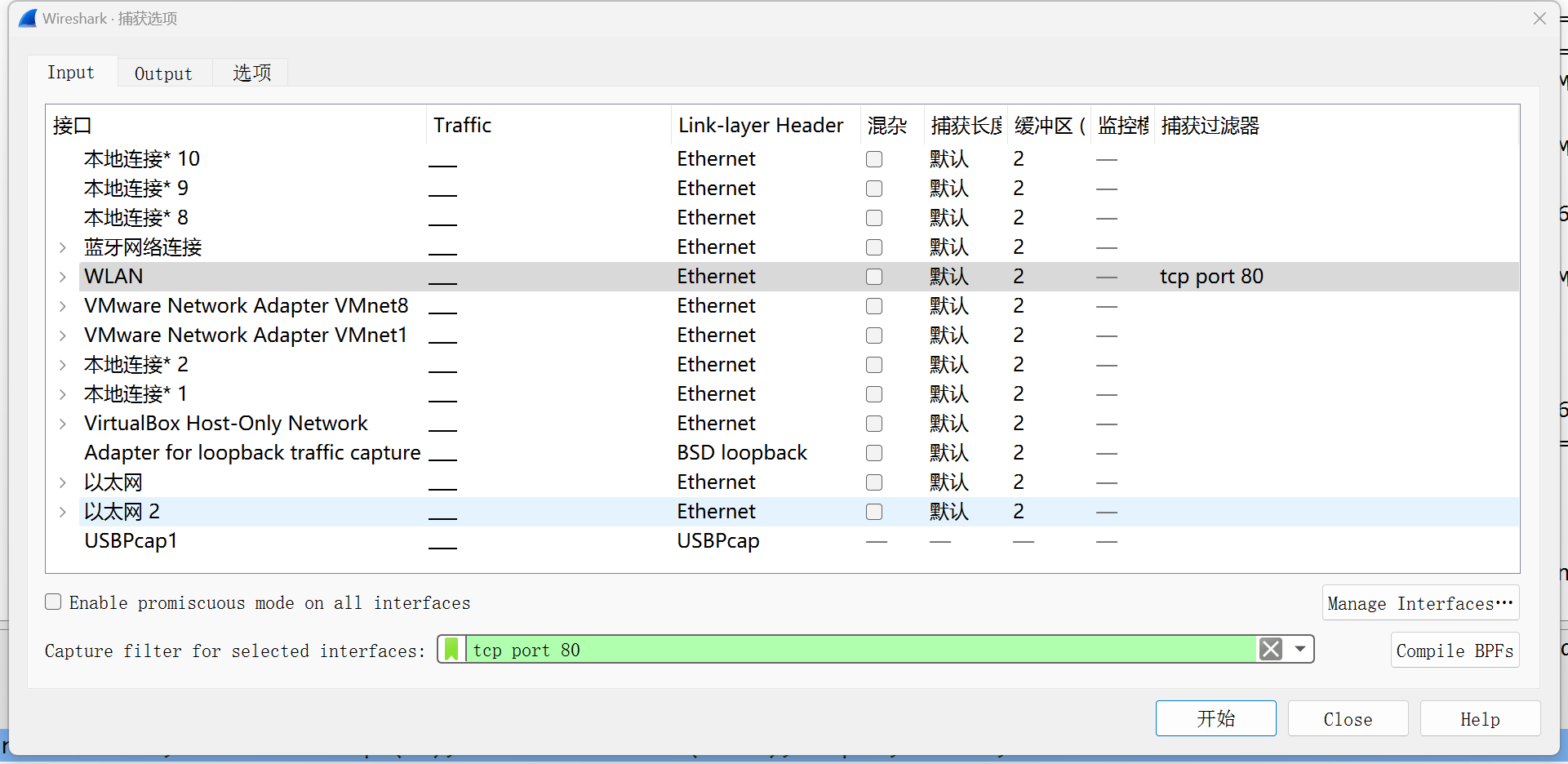
**四、实验过程与分析**

**4.1 获取TCP消息**

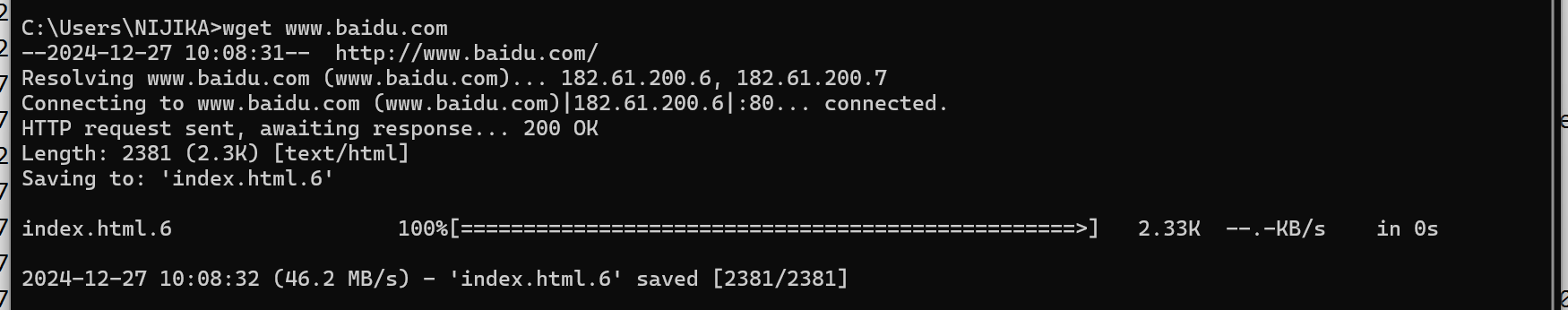
（1）用wget确认该链接有效。在命令行输入wget [www.baidu.com](http://www.baidu.com)



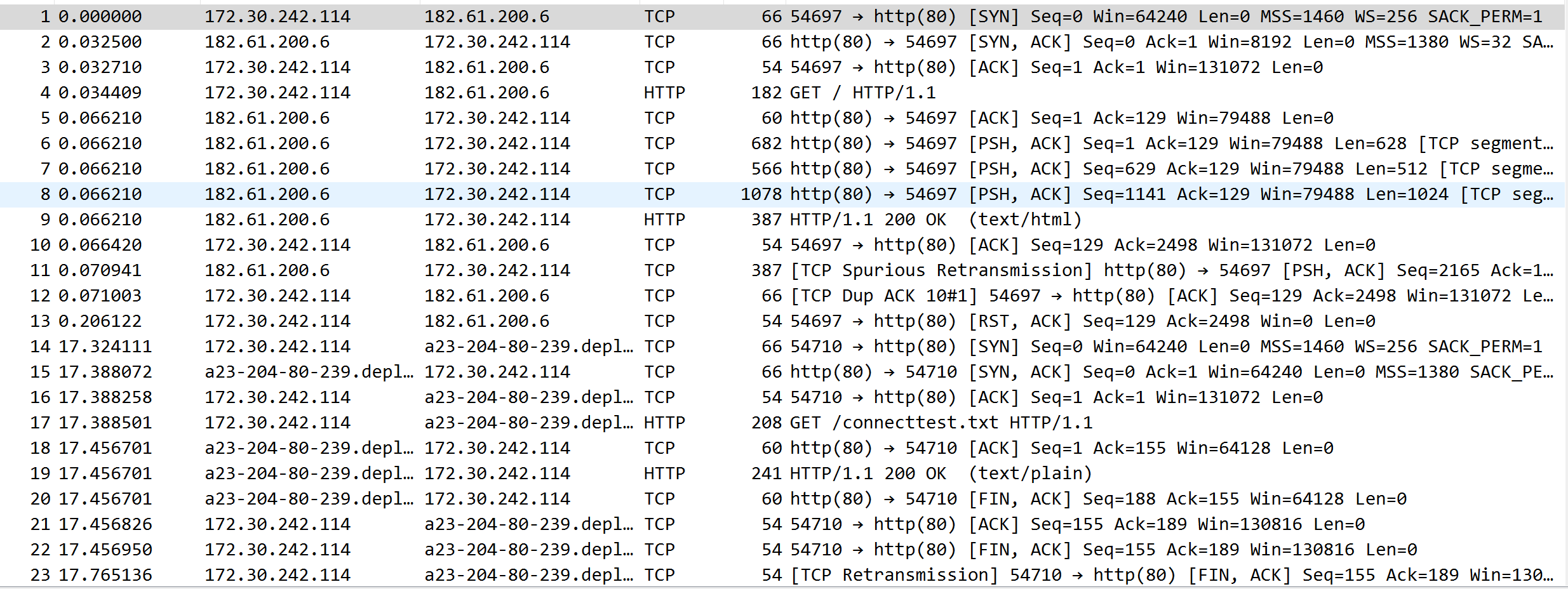
（2）启动Wireshark，在菜单栏的捕获->选项中进行设置，选择已连接的以太网，设置捕获过滤器为tcp port 80，我们主要观察客户端与服务器之间的tcp流



（3）捕获开始后，重复第一步，重新发送请求

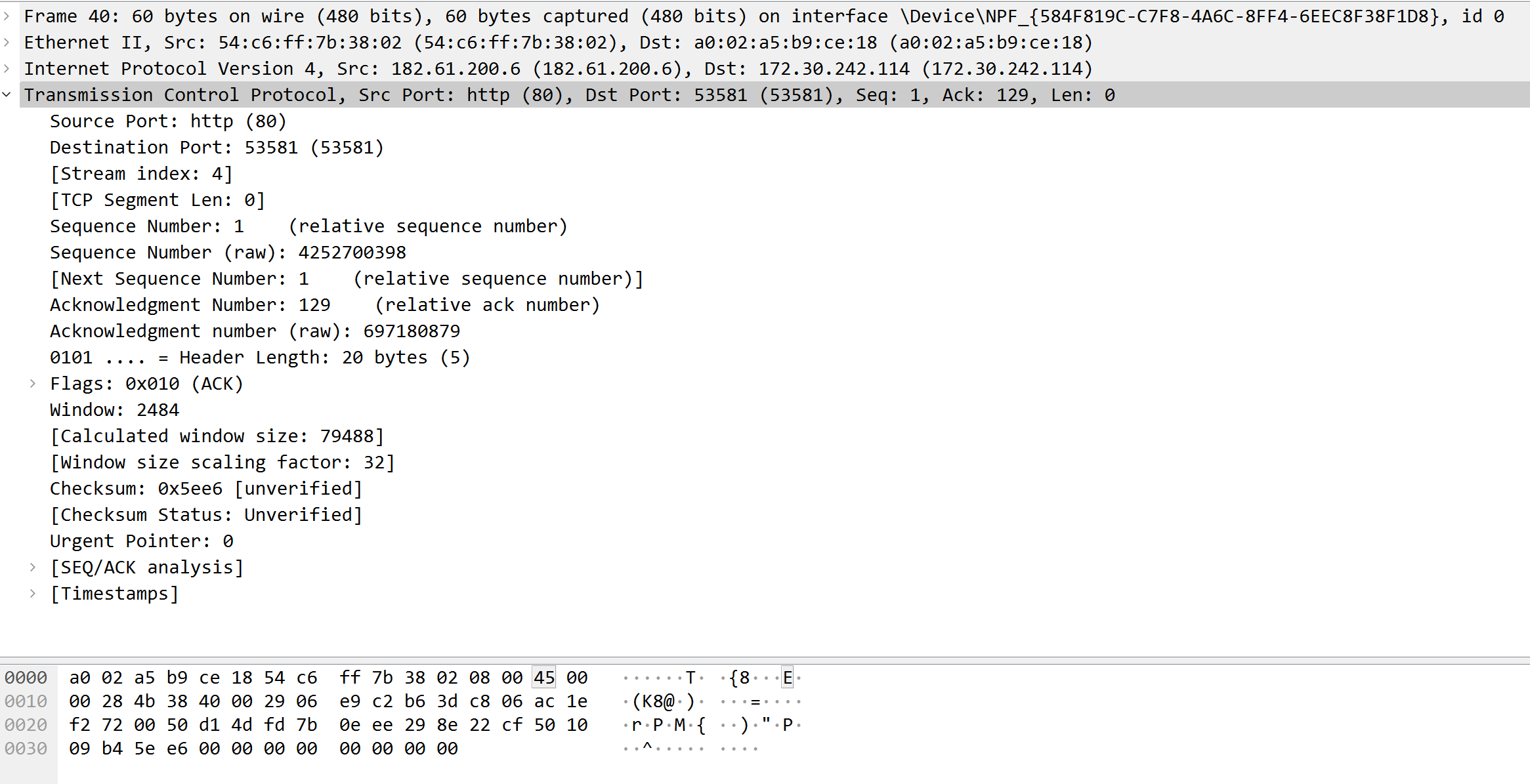


（4）当wget命令结束后，停止wireshark捕获，得到如下结果：



**4.2 分析TCP包**

选择一个TCP数据包，如下所示：



1、源端口：2字节，用来标识源主机的应用程序进程。

目的端口：2字节，用来标识目的主机的应用程序进程。

2、序号：4字节，用来标识从 TCP 源端向目的端发送的字节流，发起方发送数据时对此进行标记，以保证按序交付。

3、确认号：4字节。TCP含有确认机制，接收端要向发送端发送确认号。只有 ACK 标志位为 1 时，确认号字段才有效，确认号等于上次接收到的字节序号加1。

4、数据偏移：占4位。这是首部的偏移，单位是4B。

5、保留字段：占6位，目前置为0。

6、紧急比特URG：当等于1的时候表示紧急，优先级高。紧急指针指向的是最后一个编号。

7、确认比特ACK：为1则有效确认序号有效。为2则无效。

8、推送比特PSH：PSH=1时，接收方需尽快交付接收应用进程，不再等到缓存填满再向上交付。

9、复位比特RST：RST=1时，表明TCP连接中出现严重差错，必须释放连接，然后再重新建立传输链接。

10、同步比特SYN：SYN=1时，表示这是一个连接请求或者连接接受报文。

11、终止比特FIN：FIN=1时,此报文段的发送端数据已经全部发送，要求释放传输连接。

12、窗口字段：2字节。这个窗口用于控制对方最大发送数据量，单位是B

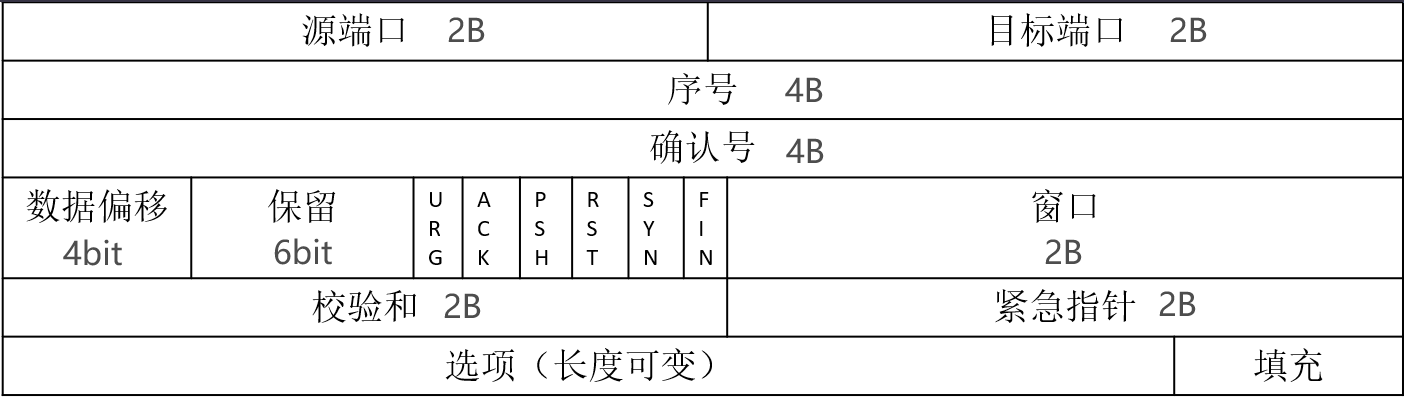
13、校验和：2字节。

14、紧急指针：2字节。URG=1时才有意义，指出本报文段中紧急数据的字节数（紧急数据在报文段数据的最前面）。

15、选项字段：每个选项开始是一个字节的kind字段，说明选项类型。每个类型描述的内容也不同。长度可变。最大报文段长度MSS、窗口扩大、时间戳、选择确认。

16、填充字段：为了使整个首部长度是4B的整数倍。

由此可以画出TCP的结构图：



**4.3 TCP连接的建立**

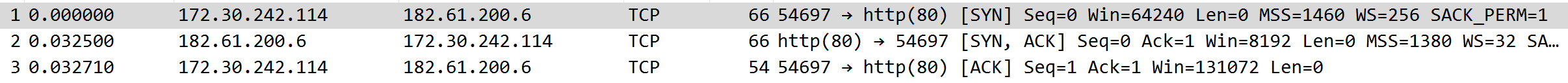
TCP的连接需要经过如下的三次握手：

1.客户机先向服务器发送一个连接请求报文段。它不含应用层数据，其首部中的SYN标志位=1，另外，客户机要随机选择一个起始序号seq=x。TCP客户进程进入SYN-SENT（同步已发送）状态。

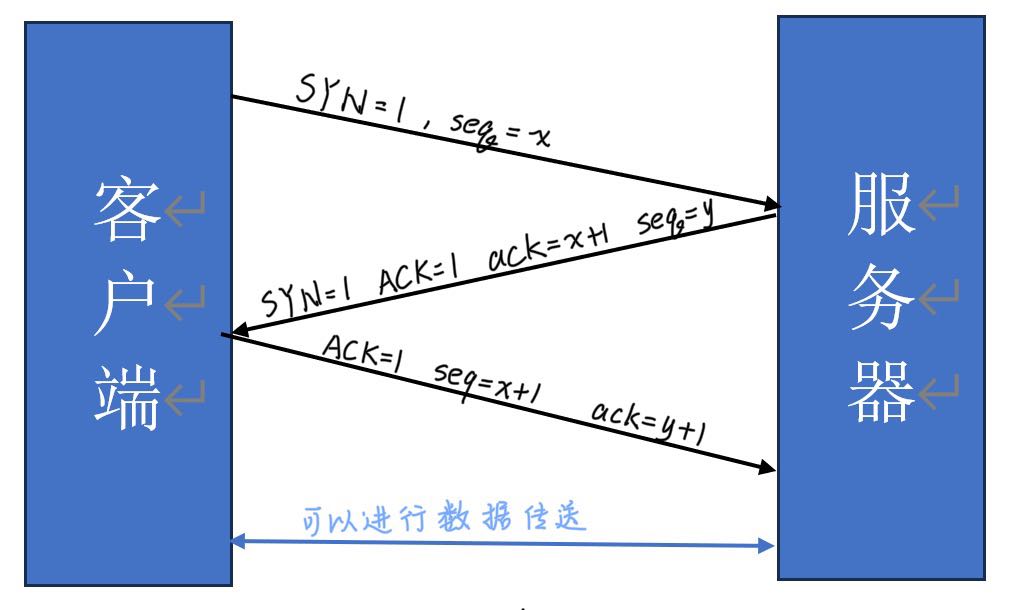
2.服务器收到连接请求报文段后，如同意就向客户机发确认，并为该连接分配缓存和变量。其中，SYN和ACK=1，确认号字段的值ack=x+1，并且服务器随机产生起始序号seq=y，确认报文段同样不包含应用层数据。TCP服务器进程进入SYN-RCVD（同步收到）状态。

3.当客户机收到确认报文段后，还要向服务器给出确认，并且也要给该连接分配缓存和变量。其中ACK标志位=1，序号字段seq=x+1，确认号字段ack=y+1，该报文段可以携带数据，如果不携带数据则不消耗序号。TCP客户进程进入ESTABLISHED（已建立连接）状态。

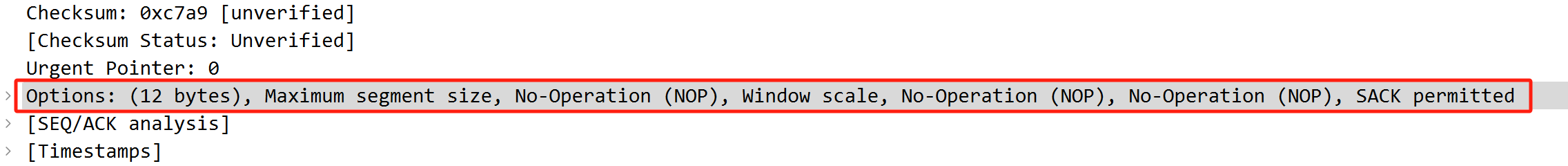
wireshark中，我们可以看到这样一次TCP连接的建立过程：



（1）观察客户端与服务器的连接建立过程，画出三次握手协议的步骤图



（2）观察TCP数据段中的option字段：



在这里，option字段指示出：1.最大报文段长度MSS、2.窗口扩大、3.选择确认。

**4.4 TCP连接的释放**

TCP连接的释放要经历如下的四次握手过程：

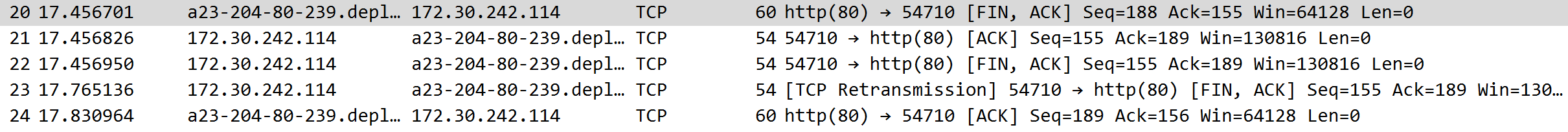
1.客户端发送连接释放报文段，停止发送数据，主动关闭TCP连接。FIN=1，seq=u，它等于前面已传送过的数据的最后一个字节的序号加1。TCP客户进程进入FIN-WAIT-1（终止等待1）状态。

2.服务器端回送一个确认报文段，ACK=1，seq=v，ack=u+1，客户到服务器这个方向的连接就释放了，TCP连接处于半关闭状态。服务器进入CLOSE-WAIT（关闭等待）状态。客户收到后进入FIN-WAIT-2（终止等待2）状态。

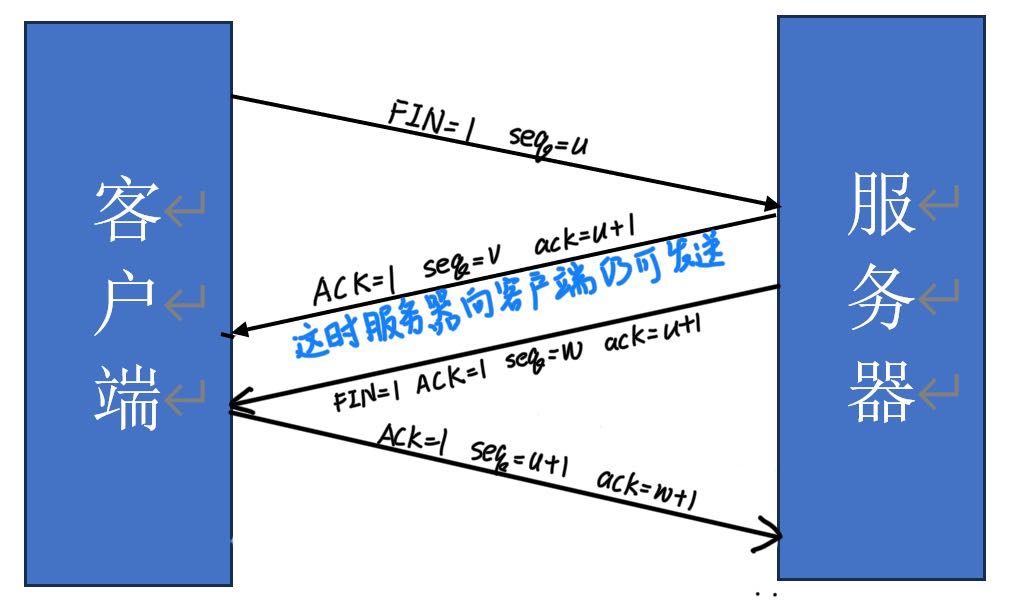
3.服务器端发完数据，就发出连接释放报文段，主动关闭TCP连接。FIN=1，ACK=1，seq=w，ack=u+1。服务器进入LAST-ACK（最后确认）状态。

4.客户端回送一个确认报文段，ACK=1, seq=u+1, ack=w+1，客户进入进入到TIME-WAIT（时间等待）状态。再等到时间等待计时器设置的2MSL（最长报文段寿命）后，连接彻底关闭。客户机和服务器进入CLOSED（连接关闭）**状态。**

**wireshark中，我们可以看到这样一次TCP连接的建立过程：**



观察客户端与服务器连接的释放过程，画出释放连接的步骤图

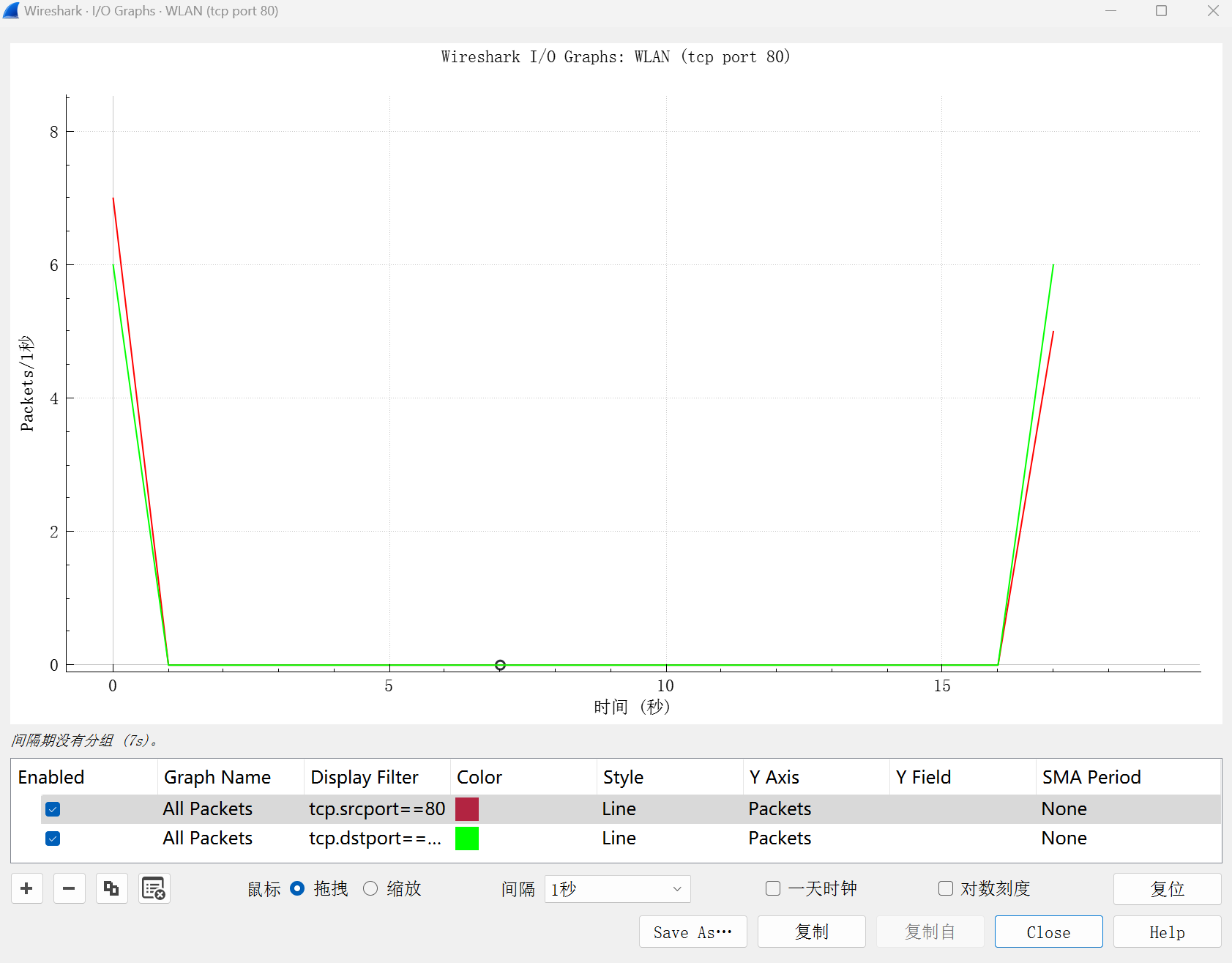


思考：为什么连接的时候是三次握手，关闭的时候却是四次握手？

当服务器端回送一个确认报文段，ACK=1，seq=v，ack=u+1，客户到服务器这个方向的连接就释放了，TCP连接处于半关闭状态。但服务器若发送数据，客户机仍要接收，即从服务器到客户机这个方向的连接并未关闭。服务器端发完数据，就发出**连接释放报文段**，主动同意关闭TCP连接。最后客户端收到 FIN ACK 包，向它回复 ACK 包，于是就需要四次握手。而建立连接时，不会出现这个问题，所以只需要三次握手即可建立连接。

**4.5 TCP数据传输**

在wireshark中，我们可以看到生成的IO图表：



问题：

1）实验中的下载速率是多少？

由图像可知，下载速率为：6packets/s

2）实验中的上传速率，即ACK消息的发送速率是多少？

由图像可知，上传速率为：7packets/s

3）观察数据传输过程中Acknowledgment number，sequence number以及 Segment Len之间的变化。如果最近从服务器收到的TCP数据段的序列号 是X，那么下一个发送的ACK是多少？

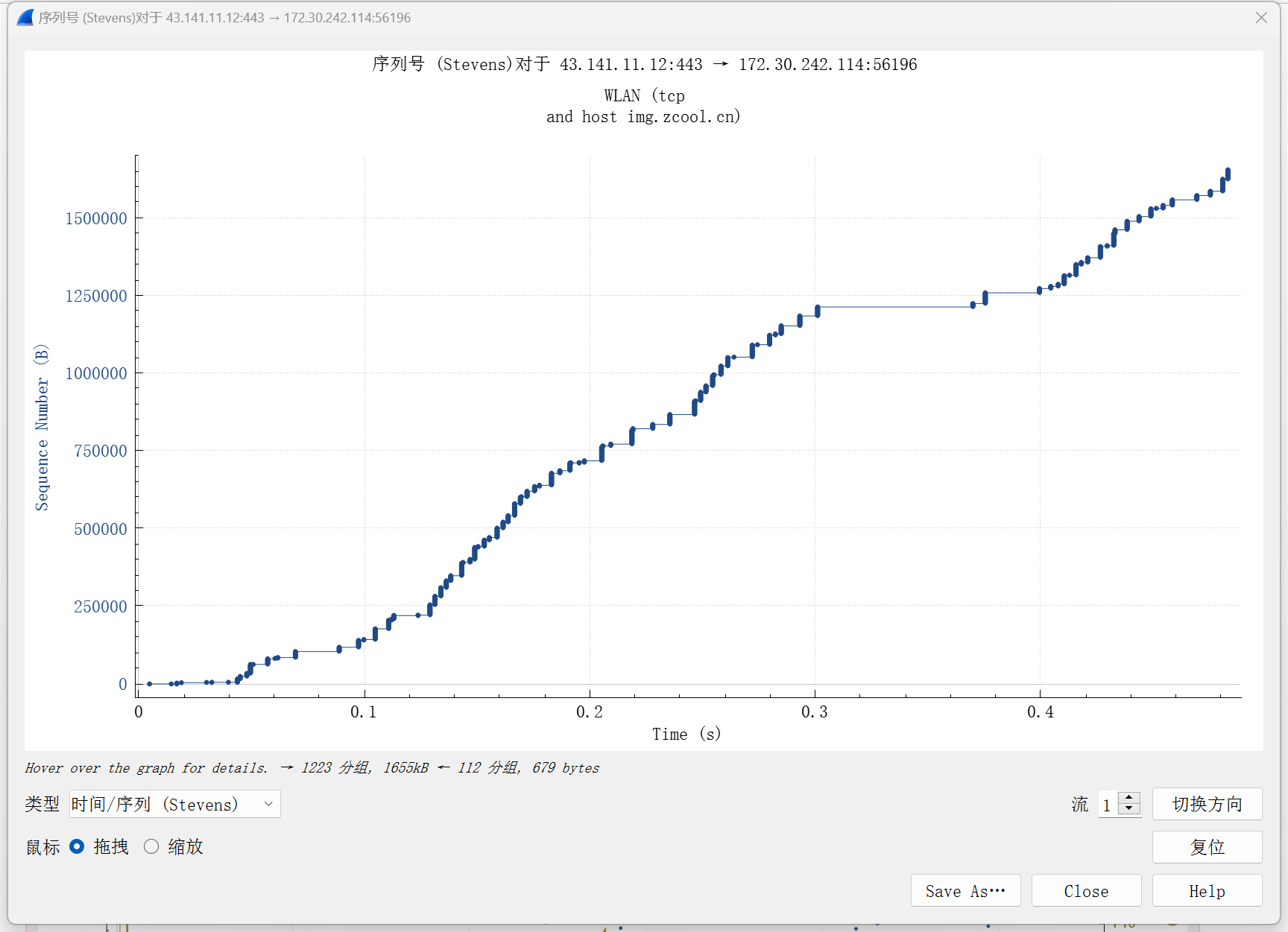
根据观察可知，下一个发送的 ACK 是 X + Segment Len

**4.6 问题讨论**

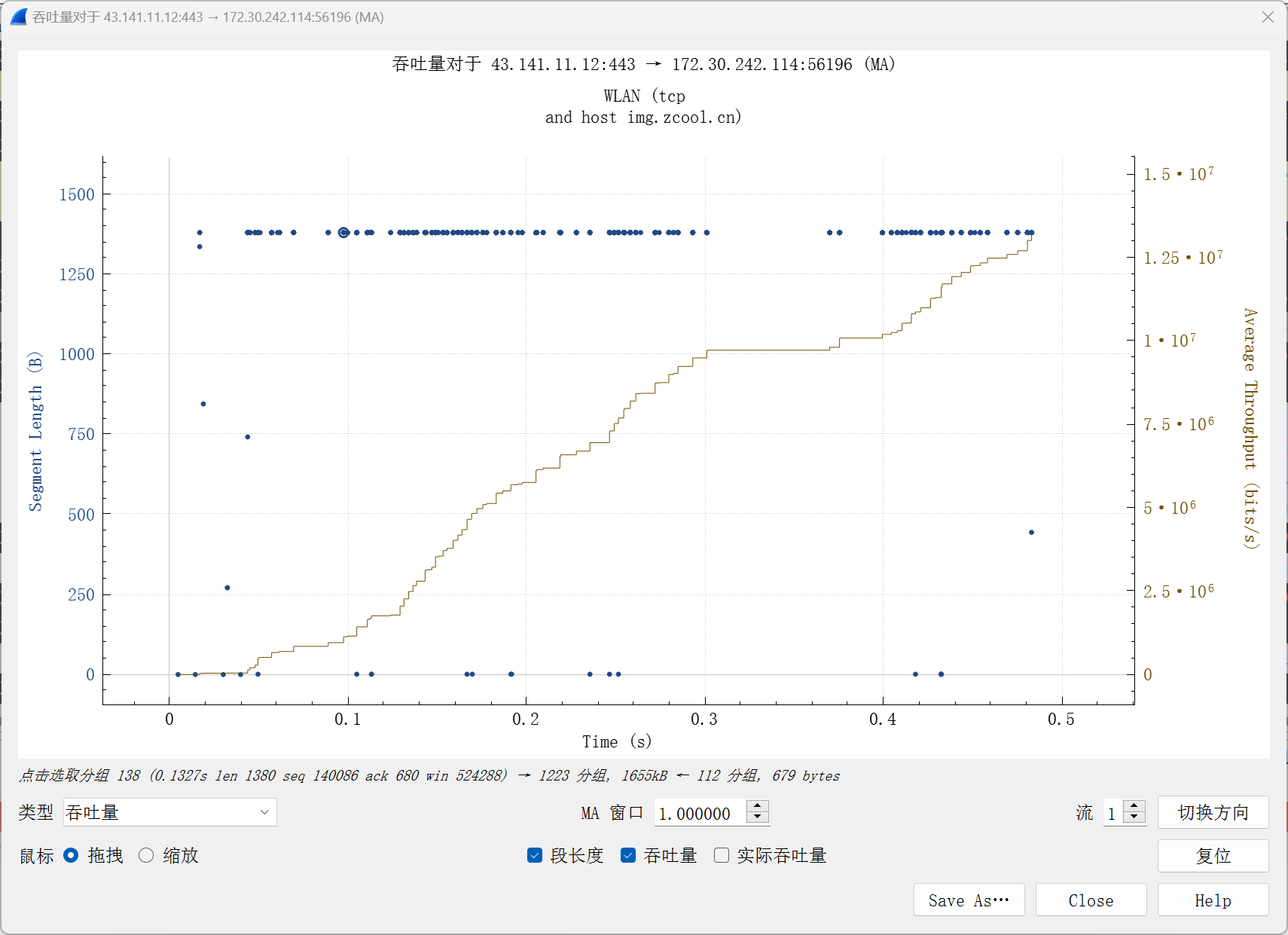
• Explore the congestion control and the classic AIMD behavior of TCP. To do this, you will likely want to capture a trace while you are sending (not receiving) a moderate amount of data on a TCP connection. You can then use the “TCP Stream Graph” tools as well as other analysis to observe how the congestion window changes over time.

重新捕获一次TCP，观察 TCP 流图表，如下所示：

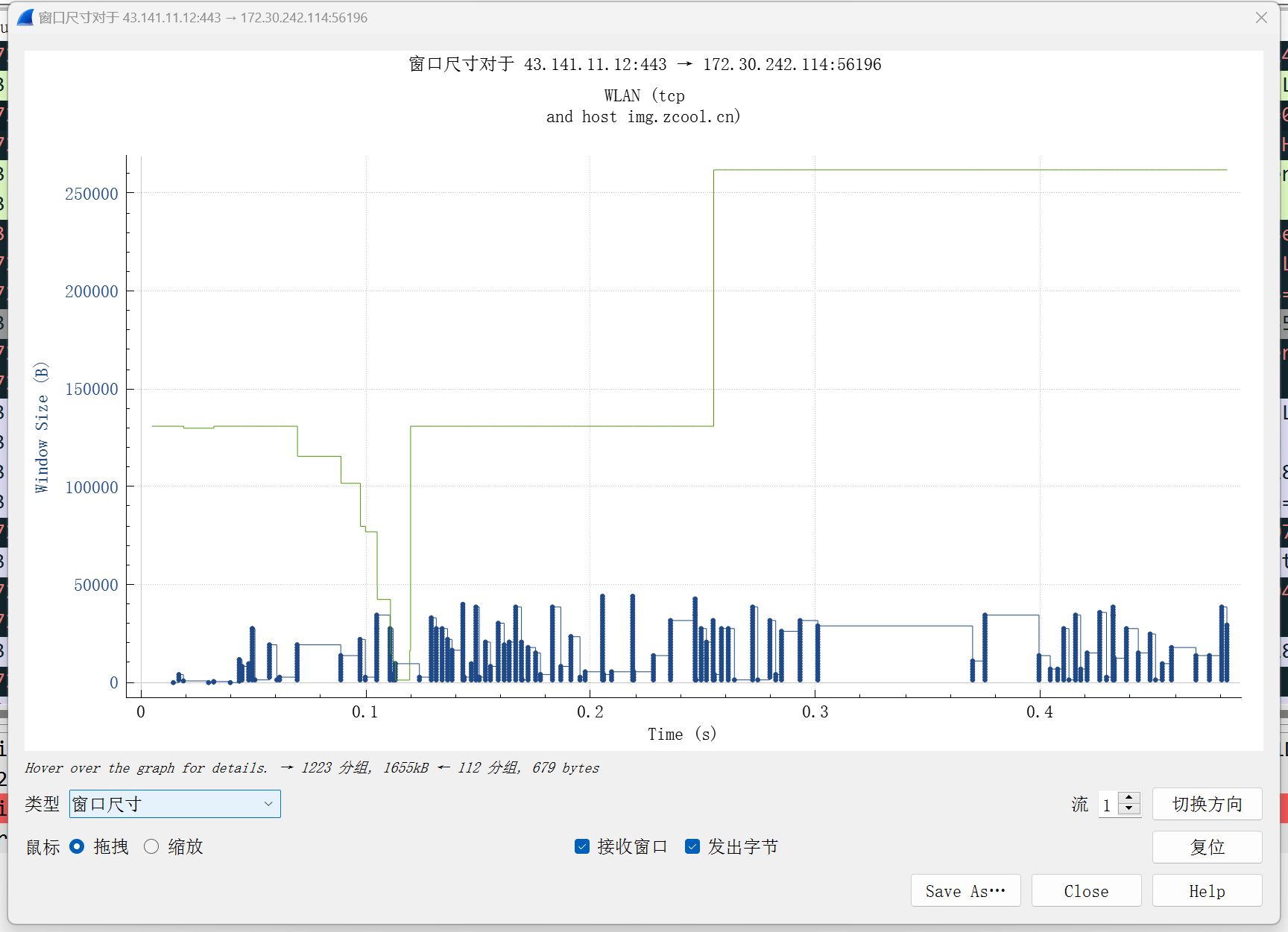
序列号：



吞吐量：



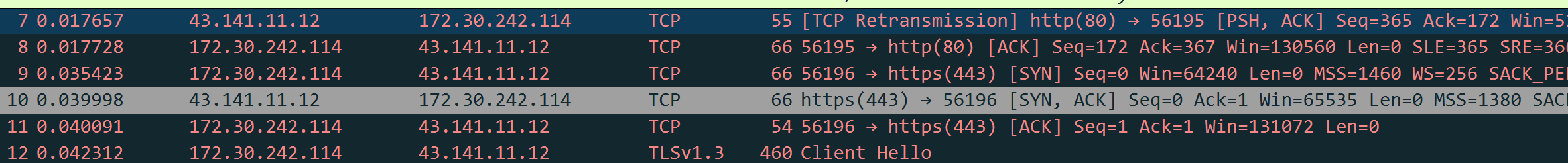
窗口尺寸：



在这里，吞吐量随着时间不断上升。观察到拥塞窗口的增长和减少模式。在网络状况良好时，cwnd 会按照加法增大的方式逐渐增长，即每次成功传输一个窗口的数据后，cwnd 增加一个 MSS大小。当检测到拥塞（例如通过超时或收到三个重复的 ACK）时，cwnd 会按照乘法减小的方式大幅减少，通常减半。

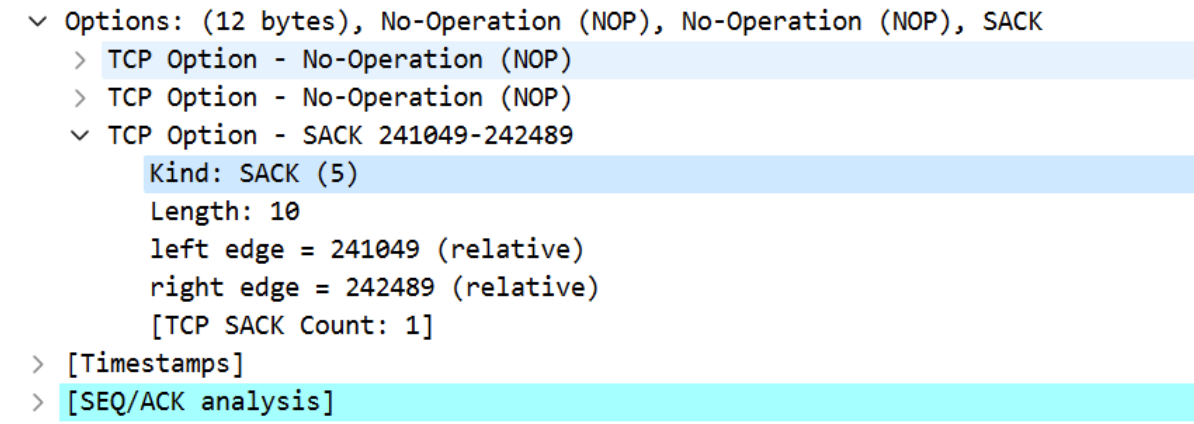
这就是AIMD行为

• Explore the reliability mechanisms of TCP more deeply. Capture a trace of a connection that includes segment loss. See what triggers the retransmissions and when. Also look at the round-trip time estimator.



可以找到TCP的重传。分析捕获的跟踪数据，查看重传（retransmissions）的触发条件和时间。在 Wireshark 中，可以通过查看数据包的序列号和 ACK 号来识别重传的数据包。重传通常在超时或收到三个重复的 ACK 后触发。

• Look at the use of options including SACK to work through the details. You should see information about ranges of received bytes during times of segment loss.



查看一个option字段含有SACK的数据包。不难发现，SACK中包含了丢失的数据包的序列号范围。

• TCP is the transport layer underlying the web. You can see how your browser makes use of TCP by setting up concurrent connections.

由于 TCP 是网络底层的传输层协议，可通过设置并发连接，观察浏览器如何利用 TCP，了解 TCP 在实际网络应用中的运用方式。浏览器处理用户请求时进行并发处理，会建立多个 TCP 连接。

**五、实验结果总结**

本次实验对TCP进行研究分析，在本次实验中，我了解了TCP协议各个字段的含义和功能，并且对TCP数据包的结构有了了解，并通过wireshark对TCP 连接建立和释放的过程有了直观的了解并且学习了 TCP 数据传输的过程，通过画图对这一过程有了更加深入的理解。在最后的问题思考环节，对 TCP 的拥塞控制机制有了了解，看到是如何通过窗口大小来控制吞吐量的，对 TCP 的可靠性机制和其在浏览器的并发处理有了一定的了解。

通过本次实验，我对TCP协议和计算机网络的认识更加深入，受益匪浅。

**六、附录**

无