

Wireshark Lab: TCP

Copyright (c) 2020 Minaduki Shigure.

南京大学 电子科学与工程学院 吴康正 171180571

实验环境

macOS "Mojave" 10.14.5

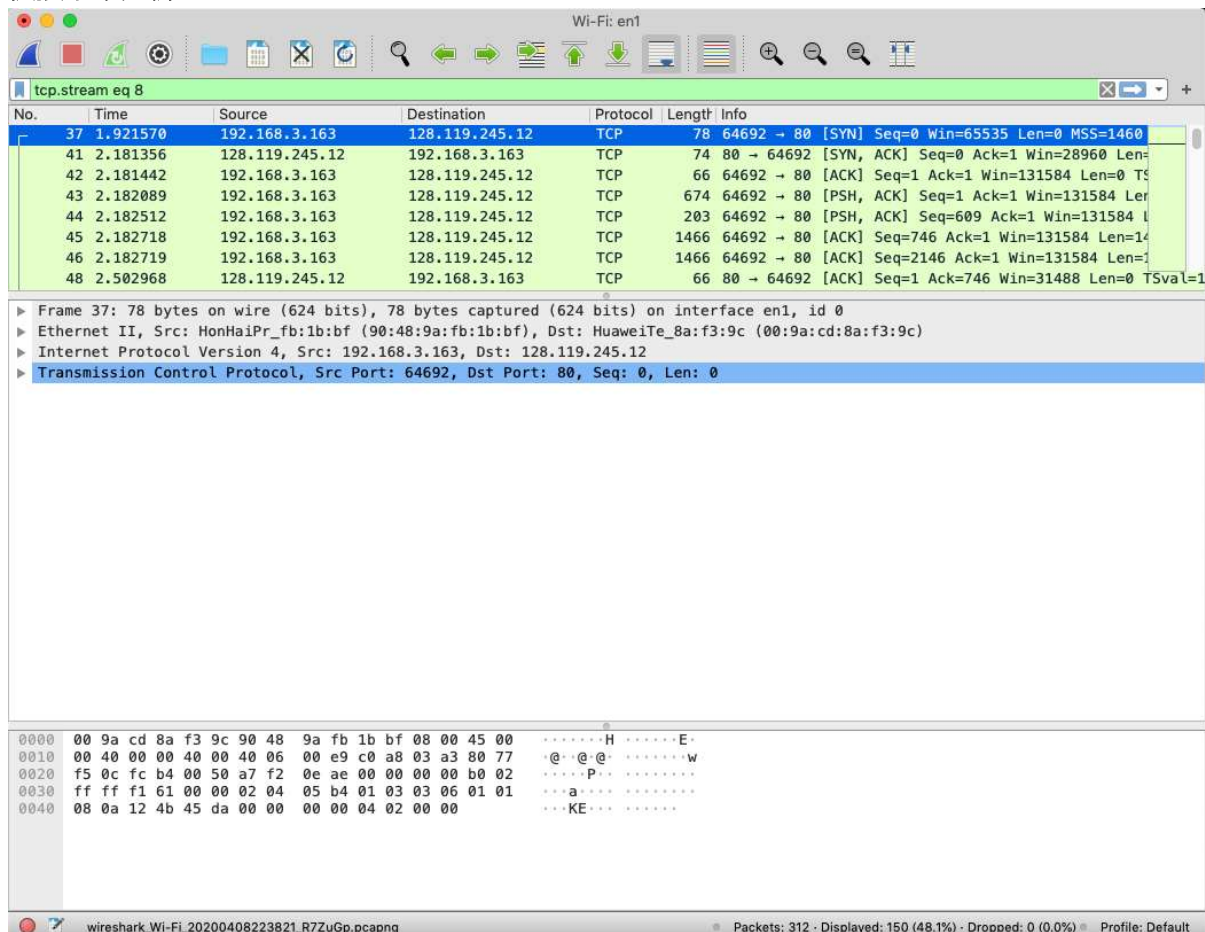
Broadcom BCM4352 Wireless Network Adapter

Wireshark Version 3.2.2 (v3.2.2-0-ga3efece3d640)

实验内容：握手言欢

按照课本要求，下载文件 <http://gaia.cs.umass.edu/wireshark-labs/alice.txt> 到本地，然后上传文件并抓包。

找到起始的SYN包，然后右键选择“跟踪->TCP流”，就能快速列出与这次上传所有有关的数据包，供接下来分析。



将文件传输到 gaia.cs.umass.edu 的客户端计算机(源)使用的 IP 地址和 TCP 端口号是什么？

IP 地址为 192.168.3.163，端口号为 64692。

gaia.cs.umass.edu 的 IP 地址是什么？在哪个端口号上发送和接收此连接的 TCP 段？

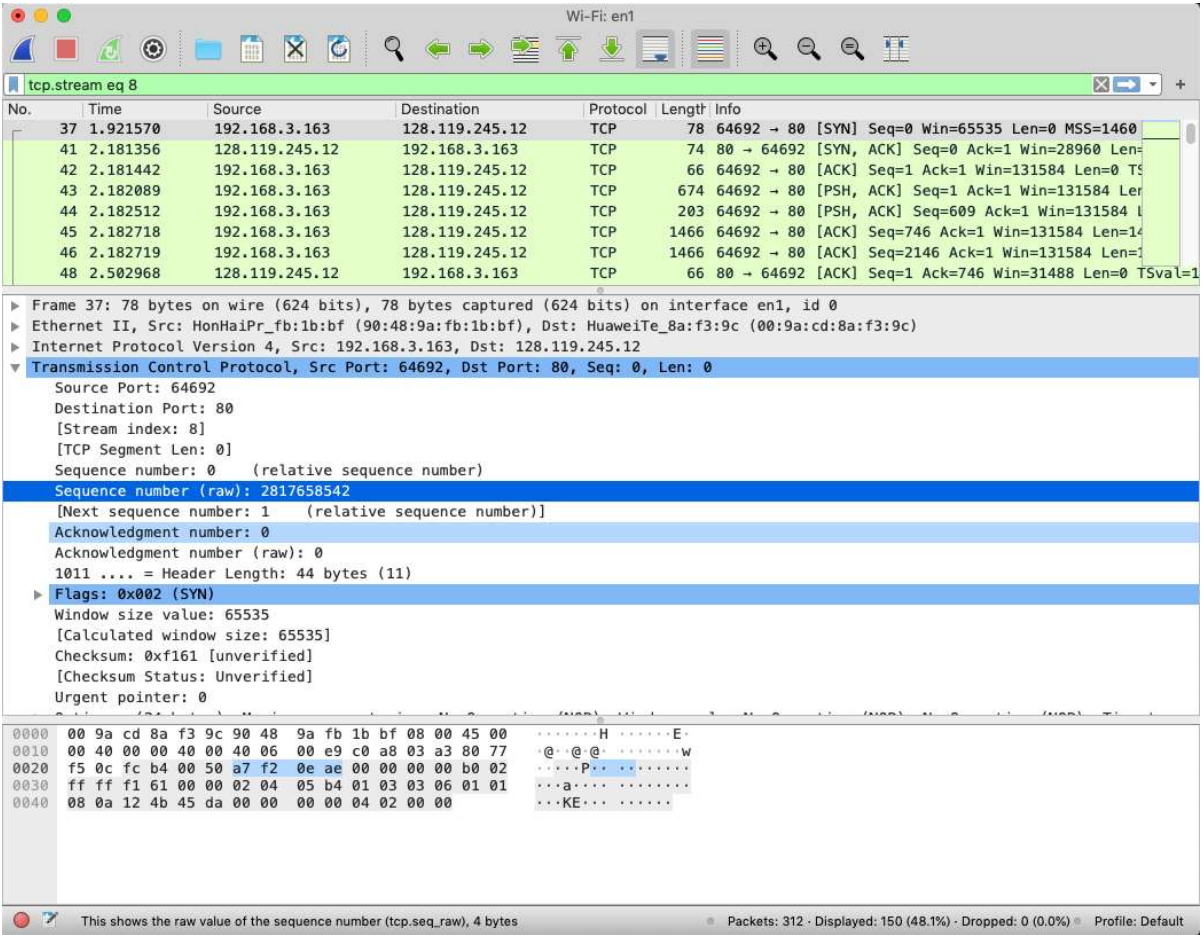
gaia.cs.umass.edu 的 IP 地址为 128.119.245.12，端口为 80。

用于在客户端计算机和 gaia.cs.umass.edu 之间启动 TCP 连接的 TCP SYN 区段的序列号是什么？

将区段标识为 SYN 段的段有什么功能？

由下面的截图可以看出，原生序列号为 2817658542。SYN 标识代表此段为一个开始 TCP 连接的

请求。

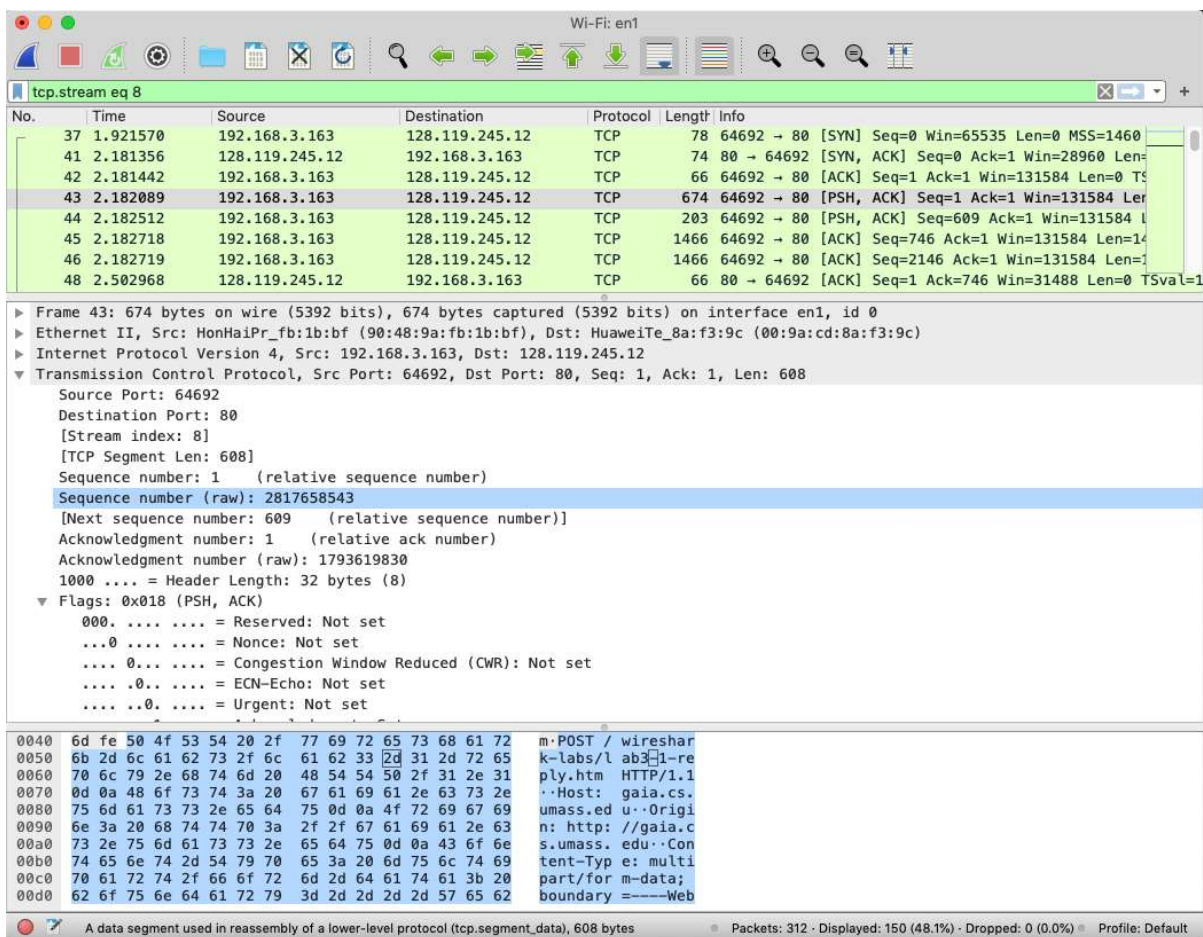


gaia.cs.umass.edu 发送给客户端计算机以回复 SYN 的 SYNACK 区段的序列号是多少? SYNACK 区段中的 Acknowledgment 栏位的值是多少?

同理可以得到, SYNACK 段的序列号为 1793619829。Acknowledgment 位的值为 1。

包含 HTTP POST 命令的 TCP 段的序列号是多少?

根据具体包段内容, 包含 HTTP POST 命令的 TCP 段的序列号是 2817658543。



将包含 HTTP POST 的 TCP 区段视为 TCP 连接中的第一个区段。在这个 TCP 连线中前六个 TCP 区段的序列号是什么(包括包含 HTTP POST 的段)? 每区段发送的时间是什么时候? 收到的每个区段的 ACK 是什么时候? 鉴于发送每个 TCP 区段的时间与收到确认的时间之间的差异, 六个区段中每个区段的 RTT 值是多少? 收到每个 ACK 后, Estimated RTT 值是什么? 假设第一个 Estimated RTT 的值等于第一个区段的测量 RTT, 然后使用课本的 Estimated RTT 公式计算所有后续区段。前六个 TCP 区段的长度是多少?

根据时间顺序进行排序, 以 Wireshark 中提供的“下一个序列号”作为辅助, 可以很轻松的定位出前六个 TCP 段, 另外, 也可以使用追踪 TCP 流的方式完成定位。在时间的问题上, 使用包含 HTTP POST 的 TCP 段的时间作为参考零点。

序列号	发送时间	ACK 时间	RTT	Estimated RTT	长度
2817658543	0	未收到	未知	未知	674
2817659151	0.000423	0.320879	0.320456	0.320456	203
2817659288	0.000629	未收到	未知	0.320456	1466
2817660688	0.000630	未收到	未知	0.320456	1466
2817662088	0.321063	未收到	未知	0.320456	1466
2817663488	1.237397	1.549955	0.312448	0.319455	1466

对于整个过程, 收到的最小可用缓冲区空间量是多少? 缺少接收器缓冲区空间是否会限制发送方传送 TCP 区段?

对于本机, 没有经历最小可用缓冲区空间, 窗口大小均为 131584。

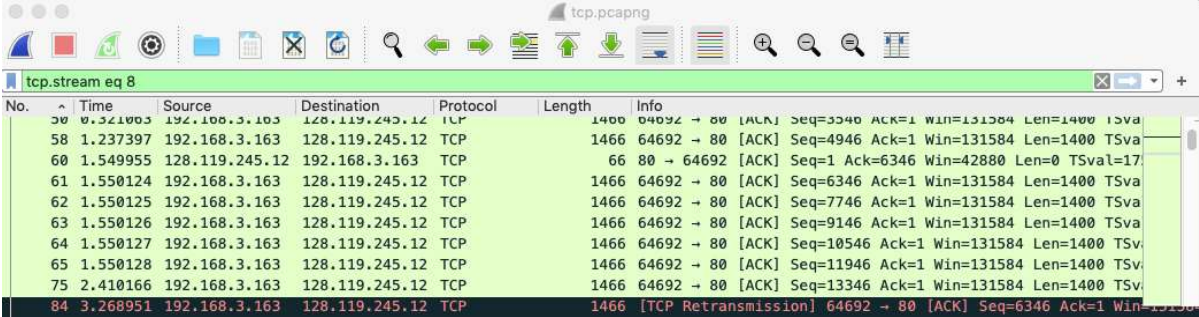
对于远端计算机, 经历过的最小可用缓冲区空间为 28960, 出现在第 41 帧。

理论上说, 如果接收方的缓冲区空间用尽, 则会进入等待询问, 发送方将暂时停止继续发送, 不

过在本次实验中，没有出现这种情况。

在跟踪文件中是否有重传的区段? 为了回答这个问题，您检查了什么?

由于 Wireshark 的高亮机制，可以看出重传的帧被使用黑红色标记了出来，此TCP流中存在着重传，第 84 帧为第 61 帧（对应序列号 2817664888）的重传。



No.	Time	Source	Destination	Protocol	Length	Info
58	1.237397	192.168.3.163	128.119.245.12	TCP	1466	64692 → 80 [ACK] Seq=3340 Ack=1 Win=131584 Len=1400 TSva
60	1.549955	128.119.245.12	192.168.3.163	TCP	66	80 → 64692 [ACK] Seq=1 Ack=6346 Win=42880 Len=0 TSval=17
61	1.550124	192.168.3.163	128.119.245.12	TCP	1466	64692 → 80 [ACK] Seq=6346 Ack=1 Win=131584 Len=1400 TSva
62	1.550125	192.168.3.163	128.119.245.12	TCP	1466	64692 → 80 [ACK] Seq=7746 Ack=1 Win=131584 Len=1400 TSva
63	1.550126	192.168.3.163	128.119.245.12	TCP	1466	64692 → 80 [ACK] Seq=9146 Ack=1 Win=131584 Len=1400 TSva
64	1.550127	192.168.3.163	128.119.245.12	TCP	1466	64692 → 80 [ACK] Seq=10546 Ack=1 Win=131584 Len=1400 TSva
65	1.550128	192.168.3.163	128.119.245.12	TCP	1466	64692 → 80 [ACK] Seq=11946 Ack=1 Win=131584 Len=1400 TSva
75	2.410166	192.168.3.163	128.119.245.12	TCP	1466	64692 → 80 [ACK] Seq=13346 Ack=1 Win=131584 Len=1400 TSva
84	3.268951	192.168.3.163	128.119.245.12	TCP	1466	[TCP Retransmission] 64692 → 80 [ACK] Seq=6346 Ack=1 Win=131584 Len=1400 TSva

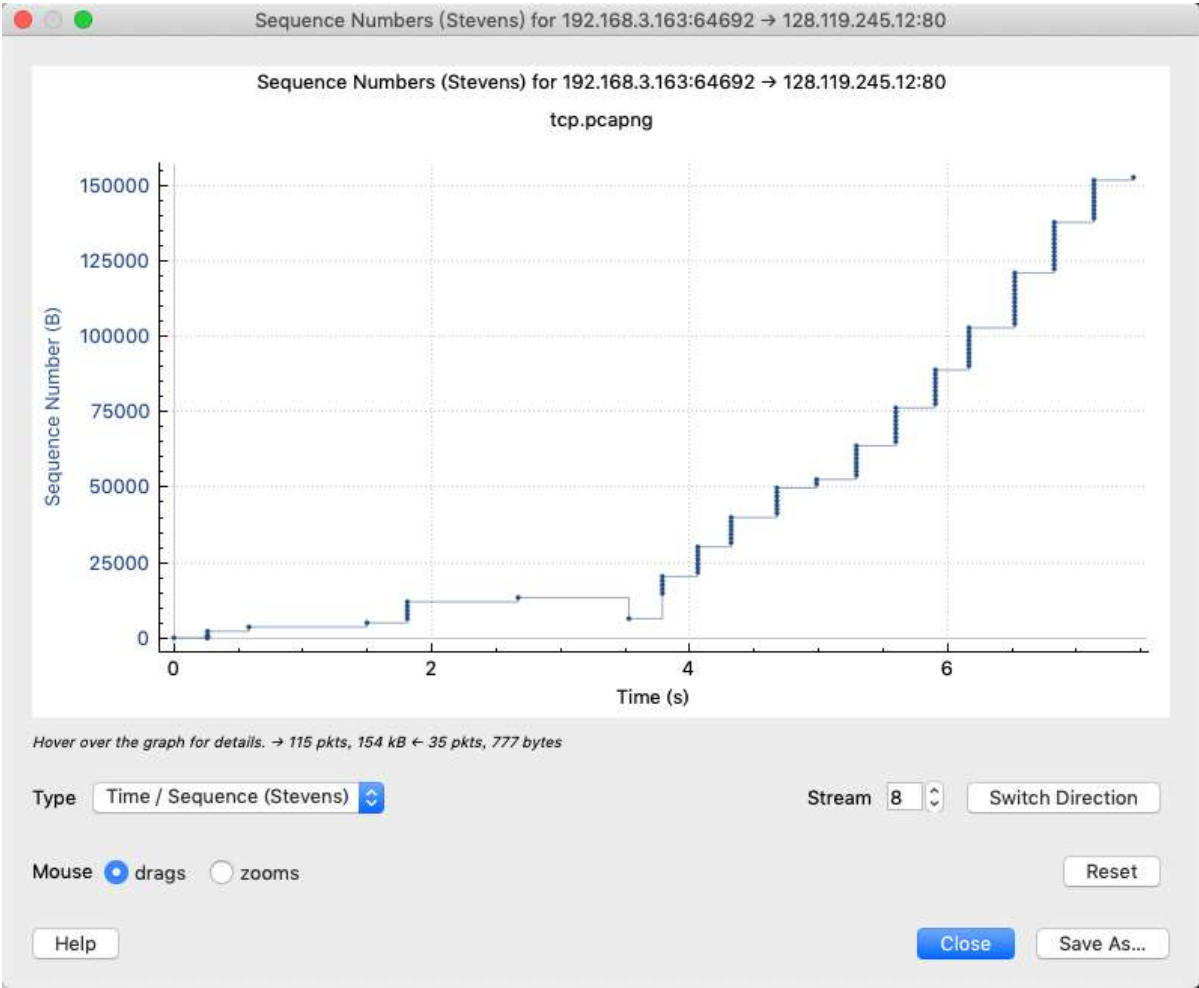
接收方通常在 ACK 中确认多少数据? 您是否可以识别接收方每隔一个接收到的区段才发送确认的情况?

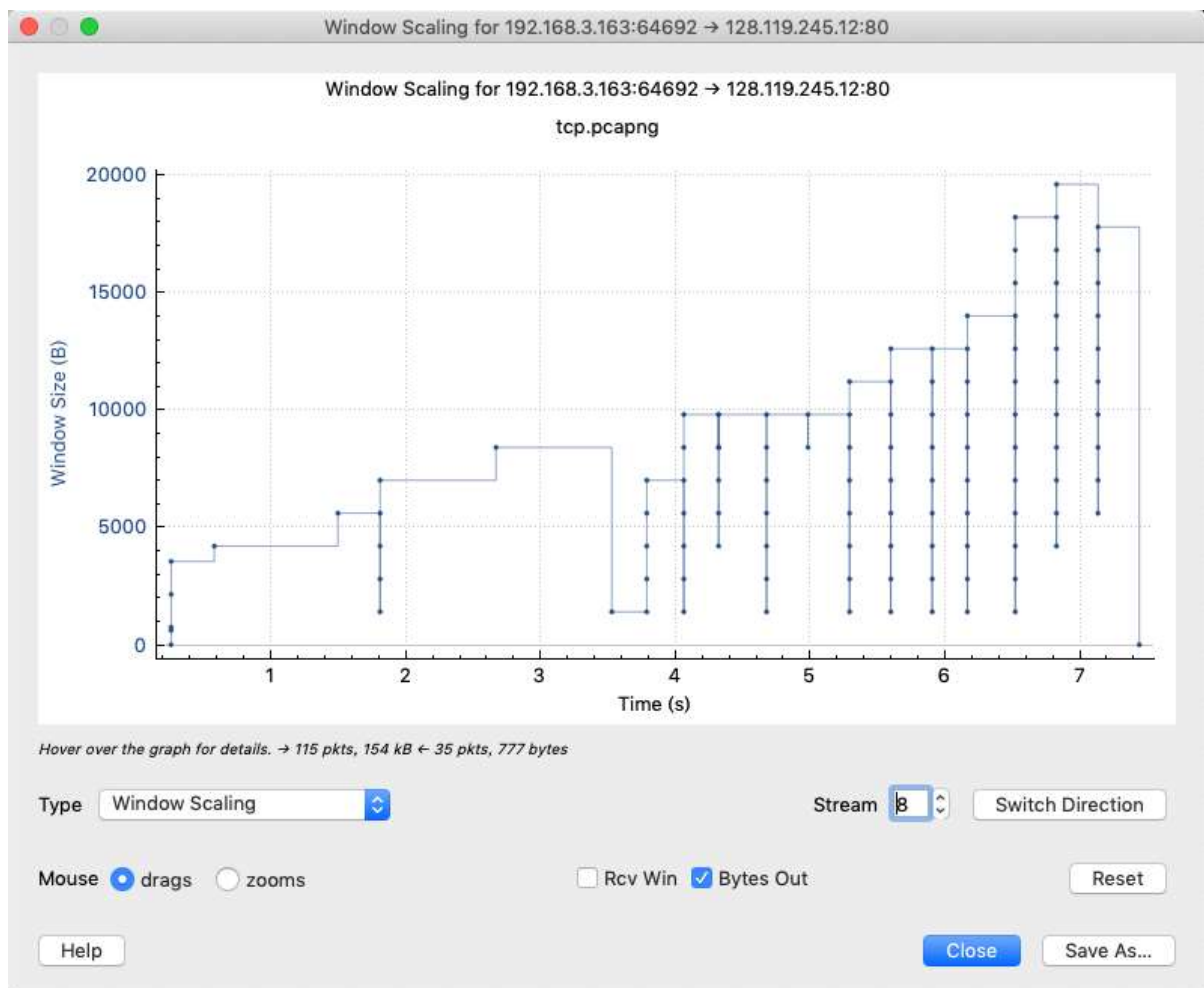
显然从上面的表格中可以看出，对于这六个包，接收端只返回了两次 ACK，也就是说存在接收方合并 ACK 的情况。

TCP 连接的吞吐量(每单位时间传输的字节数)是多少? 解释你如何计算这个值。

借助 Wireshark 的辅助分析，可以直接查看被标记为 HTTP 协议的帧，Wireshark 会自动统计涉及到的 TCP 流的信息。可以看出，在整个连接过程中，总共传输了 152929 字节数据，耗时 7.136142 秒，因此吞吐量为 21.430KiB/s。

使用时序图(Stevens)绘图工具查看从客户端发送到 gaia.cs.umass.edu 服务器的区段的序列号与时间关系图。您能否确定 TCP 的慢启动阶段的开始和结束位置，以及拥塞避免接管的位置? 评论测量数据与我们在文本中研究的 TCP 的理想化行为的不同之处。





利用窗口大小的时序图可以更加直观的判断。建立连接时，TCP慢启动开始，然后在约0.1秒的时候慢启动结束，此后为线性增长阶段。在1.8秒左右触发了拥塞避免，速度跌落，然后快速恢复，继续线性增长，如此往复，直到传输结束。

测量数据的规律并不完全符合理想化的数字，但是大致可以看出符合课本上的规则，关于这一点在小结部分有一些后续的分析。

如果使用教材提供的抓包记录文件进行分析，可以很显著地看出慢启动、拥塞控制和快速恢复的

各个节点。



小结

对于 TCP 窗口大小变化的规律，在一些方面上不严格符合课本上的理论，比如说一些诡异窗口大小变化，这可能是由于一些骨干网的额外控制机制，导致可用的带宽是逐渐增加（而非固定值）而产生的。另外，一开始的窗口大小变化看起来更像是快速恢复而非慢启动，这可能是因为在此次抓包之前就已经尝试过进行传输导致的，一些资料中显示，发送端可能会将这次新的连接以中断恢复的方式处理，使用之前一次传输时的窗口大小作为参考进行快速恢复，以最大程度利用带宽。