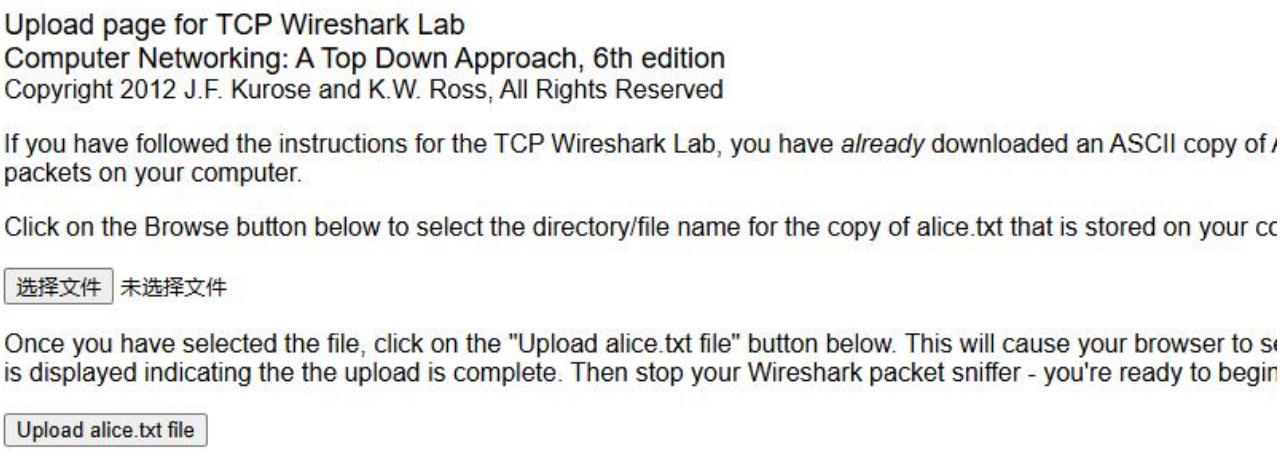
山东大学 计算机 学院

计算机网络 课程实验报告

学号：202400130240	姓名： 贾宗翰	班级： 24.6
实验题目： TCP 实验		
实验学时：2h	实验日期： 2025.9.30	
实验目的： 通过 WireShark 结合了实际的数据包捕获和分析，以及理论知识，从而更深入地理解 TCP 协议的原理和实践		
硬件环境： 联想拯救者		
软件环境： Wireshark，edge 浏览器		

实验步骤与内容：

首先保存 txt 文件，随后打开网站：https://gaia.cs.umass.edu/wireshark-labs/TCP-wireshark-file1.html



随后打开 wireshark，之后发送 txt，出现：

随后停止抓包，开始分析。

1. 将 alice.txt 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？要回答这个问题，最简单的方法是选择一条 HTTP 消息，并使用“所选数据包标头窗口的详细信息”（如果您不确定 Wireshark 窗口，请参阅“Wireshark 入门”实验室中的图 2）。
2. gaia.cs.umass.edu 的 IP 地址是什么？它在哪个端口号上发送和接收此连接的 TCP 段？
3. 用于在客户端计算机和 gaia.cs.umass.edu 之间启动 TCP 连接的 TCP SYN 段的序

列号是什么？（注意：这是 TCP 网段本身携带的“原始”序列号;它不是 Wireshark 窗口中“No.”列中的数据包 #。请记住，TCP 或 UDP 中没有“数据包号”这样的东西;如您所知，TCP 中有序列号，这就是我们在这里所追求的。另请注意，这不是相对于此 TCPsession 的起始序列号的相对序列号。）此 TCP 段中将该段标识为 SYNsegment 的是什么？此会话中的 TCP 接收器是否能够使用选择性确认（允许 TCP 功能更像“选择性重复”接收器，请参阅文本中的第 3.4.5 节）？

4. gaia.cs.umass.edu 发送到客户端计算机以响应 SYN 的 SYNACK 段的序列号是什么？区段中将区段标识为 SYNACK 区段的什么？SYNACK 区段中 Acknowledgement 字段的值是什么？gaia.cs.umass.edu 是如何确定该值的？
5. 包含 HTTP POST 命令标头的 TCP 段的序列号是多少？请注意，为了找到 POST 消息报头，您需要深入研究 Wireshark 窗口底部的数据包内容字段，查找其 DATA 字段中带有 ASCII 文本“POST”的段 4,5。此 TCPsegment 的 payload （data） 字段中包含多少字节的数据？传输文件中的所有数据是否 alice.txt 适合这个段？
6. 将包含 HTTP “POST” 的 TCP 段视为 TCP 连接的数据传输部分中的第一个段。
   - TCP 连接的数据传输部分的第一个段（包含 HTTP POST 的段）是在什么时候发送的？
   - 何时收到第一个包含数据的区段的 ACK？
   - 第一个包含数据的区段的 RTT 是多少？
   - 第二个数据承载 TCP 段的 RTT 值及其 ACK 是多少？
   - 收到第二个数据承载段的 ACK 后，估计的 RTT 值（参见正文中的第 3.5.3 节）是多少？假设在收到第二个 Segment 的 ACK 后进行此计算时，EstimatedRTT 的初始值等于第一段的测量 RTT,然后使用第 242 页的 EstimatedRTT 公式计算,值 $\alpha = 0.125$。注意：Wireshark 有一个很好的功能，允许您为发送的每个 TCP 段绘制 RTT。在 “list of captured packets” 窗口中选择从客户端发送到 gaia.cs.umass.edu 服务器的 TCP 段。然后选择：Statistics->TCP Stream Graph >Round Trip Time Graph
7. 前四个数据承载 TCP 段的长度（标头加有效负载）是多少？
8. 在这前 4 个数据承载 TCP 分段中，gaia.cs.umass.edu 向客户端通告的最小可用缓冲区空间量是多少？缺少接收方缓冲区空间是否会限制这前四个数据承载段的发送方？
9. 跟踪文件中是否有任何重新传输的段？为了回答这个问题，您检查了什么（在跟踪中）？
10. 在从客户端发送到 gaia.cs.umass.edu 的前 10 个数据承载段中，接收方通常在 ACK 中确认多少数据？您能否确定接收方在这前 10 个数据承载段中每隔一个接收到的段（参见文本中的表 3.2）确认一次的情况？
11. TCP 连接的吞吐量（每单位时间传输的字节数）是多少？说明您是如何计算此值的。
12. 使用 Time-Sequence-Graph （Stevens） 绘图工具查看从客户端发送到 gaia.cs.umass.edu 服务器的 Segment 的序列号与时间图。考虑在 $t = 0.025$、$t = 0.053$、$t = 0.082$ 和 $t = 0.1$ 附近发送的数据包的“队列”。评论这是否看起来 TCP 处于缓慢启动阶段、拥塞避免阶段或其他阶段。图 6 显示了此数据的略有不同的视图。
13. 这些分段的 “队列” 似乎具有一定的周期性。您对这段时间有什么看法？

回答上述两个问题，了解在将文件从计算机传输到 gaia.cs.umass.edu 时收集的跟踪

多次实验之后，并未能找到 http，于是用作者发的数据包进行分析。

```
Ethernet II, Src: Apple_98:d9:27 (78:4f:43:98:d9:27), Dst: Google_89:0e:c8 (3c:28:6d:89:0e:c8)
Internet Protocol Version 4, Src: 192.168.86.68, Dst: 128.119.245.12
Transmission Control Protocol, Src Port: 55639, Dst Port: 80, Seq: 152041, Ack: 1, Len: 1385
    Source Port: 55639
    Destination Port: 80
    [Stream index: 0]
    [Stream Packet Number: 153]
  ▸ [Conversation completeness: Incomplete, DATA (15)]
    [TCP Segment Len: 1385]
    Sequence Number: 152041    (relative sequence number)
    Sequence Number (raw): 4236801228
    [Next Sequence Number: 153426    (relative sequence number)]
    Acknowledgment Number: 1    (relative ack number)
    Acknowledgment number (raw): 1068969753
    1000 .... = Header Length: 32 bytes (8)
  ▸ Flags: 0x018 (PSH, ACK)
    Window: 2058
    [Calculated window size: 131712]
```

1. 分析可知作者的 ip 是 192.168.86.68，tcp 源端口号是 55639.
2. gaia.cs.umass.edu 的 IP 地址是 128.119.245.12，TCP 接受端口 80。

Wireshark · 分组 1 · tcp-wireshark-trace1-1.pcapng

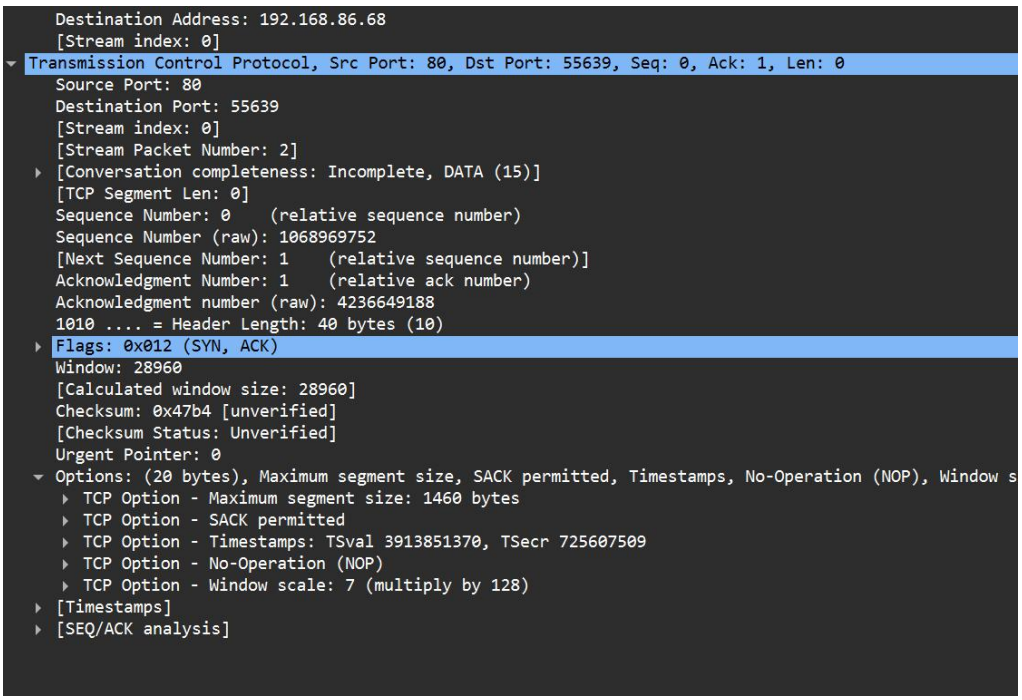
```
Frame 1: 78 bytes on wire (624 bits), 78 bytes captured (624 bits) on interface en0, id 0
Ethernet II, Src: Apple_98:d9:27 (78:4f:43:98:d9:27), Dst: Google_89:0e:c8 (3c:28:6d:89:0e:c8)
Internet Protocol Version 4, Src: 192.168.86.68, Dst: 128.119.245.12
Transmission Control Protocol, Src Port: 55639, Dst Port: 80, Seq: 0, Len: 0
    Source Port: 55639
    Destination Port: 80
    [Stream index: 0]
    [Stream Packet Number: 1]
  ▸ [Conversation completeness: Incomplete, DATA (15)]
    [TCP Segment Len: 0]
    Sequence Number: 0    (relative sequence number)
    Sequence Number (raw): 4236649187
    [Next Sequence Number: 1    (relative sequence number)]
    Acknowledgment Number: 0
    Acknowledgment number (raw): 0
    1011 .... = Header Length: 44 bytes (11)
  ▸ Flags: 0x002 (SYN)
    Window: 65535
    [Calculated window size: 65535]
```

```
0000  3c 28 6d 89 0e c8 78 4f  43 98 d9 27 08 00 45 00   <(m···xO C··'··E·
0010  00 40 00 00 40 00 40 06  ae 47 c0 a8 56 44 80 77   ·@··@·@· ·G··VD·w
0020  f5 0c d9 57 00 50 fc 86  22 e3 00 00 00 00 b0 02   ···W·P·· "·······
0030  ff ff a1 e4 00 00 02 04  05 b4 01 03 03 06 01 01   ···· ···· ····
0040  08 0a 2b 3f e4 55 00 00  00 00 04 02 00 00         ··+?·U·· ····
```

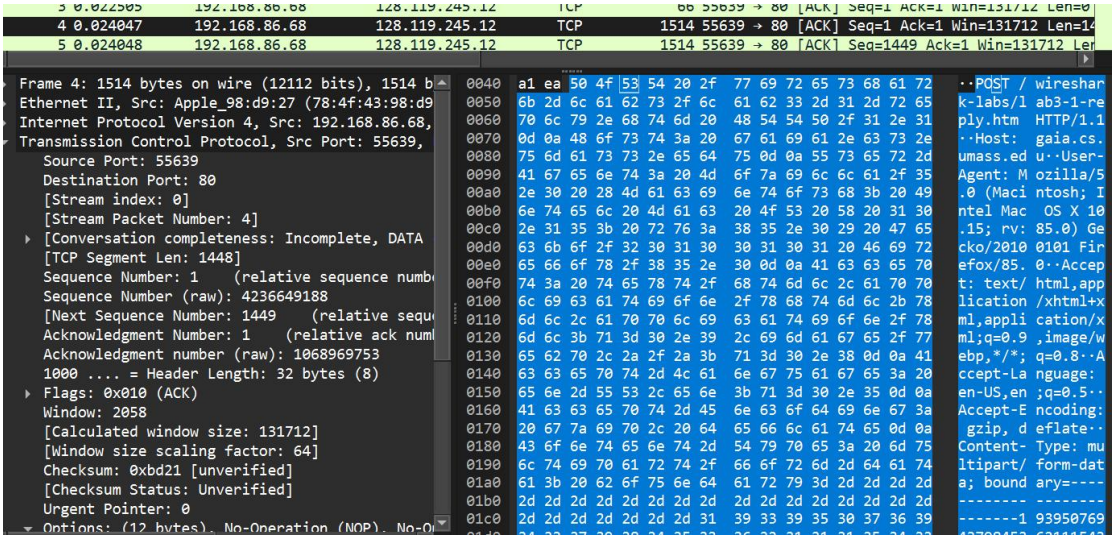
3. SYN 相对序列号为 0， 绝对序列号为 4236649187。该段被识别为 SYN 段，通过“Flags: 0x002 (SYN)”字段。 标志 0x002 (SYN) 设置同步序列号标志。 这就是 TCP 指示连接建立请求的方式。此外，是可以选择性确认（SACK）的：

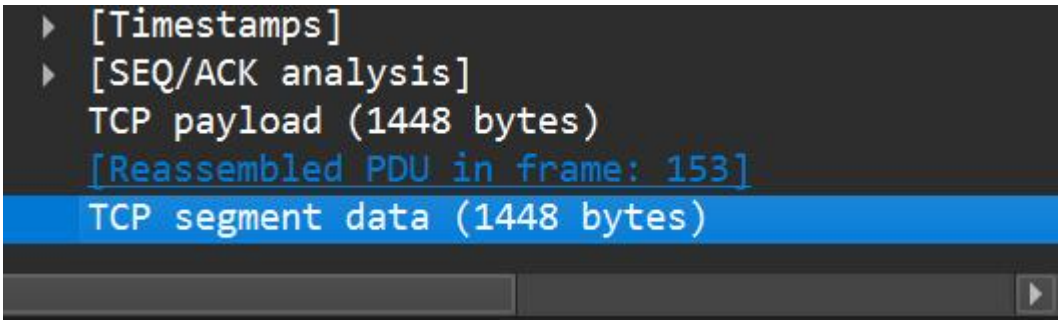
```
  ▸ TCP Option - Window scale: 6 (multiply by 64)
  ▸ TCP Option - No-Operation (NOP)
  ▸ TCP Option - No-Operation (NOP)
  ▸ TCP Option - Timestamps: TSval 725607509, TSecr 0
  ▸ TCP Option - SACK permitted
  ▸ TCP Option - End of Option List (EOL)
  ▸ TCP Option - End of Option List (EOL)
▸ [Timestamps]
```

```
0000  3c 28 6d 89 0e c8 78 4f  43 98 d9 27 08 00 45 00   <(m···xO C··'··E·
```

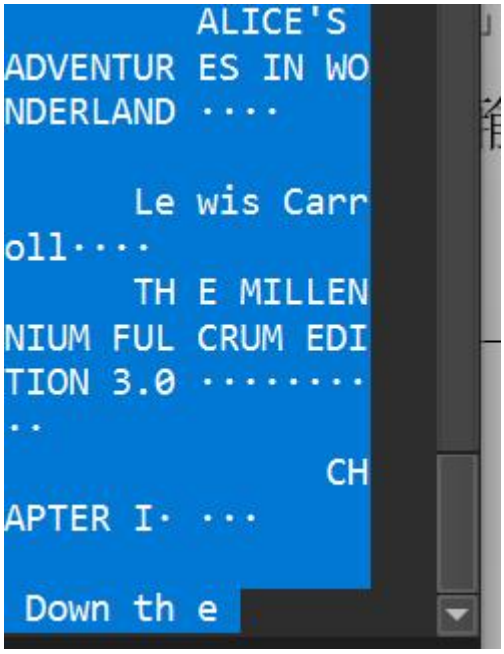


4. SYN 为 0，该段通过“Flags: 0x012 (SYN, ACK)”字段识别为 SYN-ACK 段。这表示 SYN（同步）和 ACK（确认）标志都已设置。Acknowledgement 值是 1，通过 4236649188-4236649187 = 1 确定的。

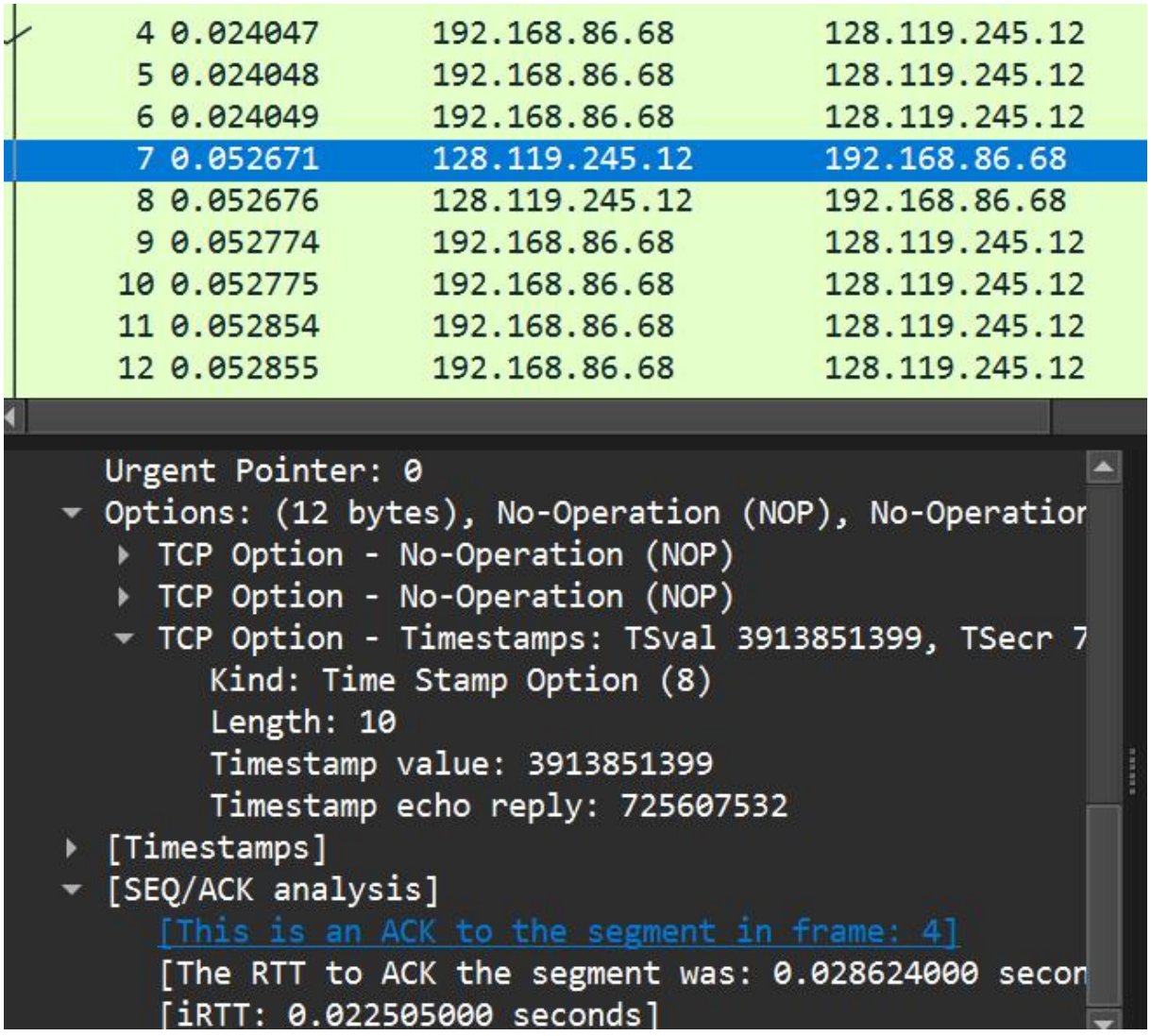




5. 分析可知，这个 tcp 带着 post，它的序列号是 1，有 1448 字节的数据，传输的 txt 文件并不都在这个里面：

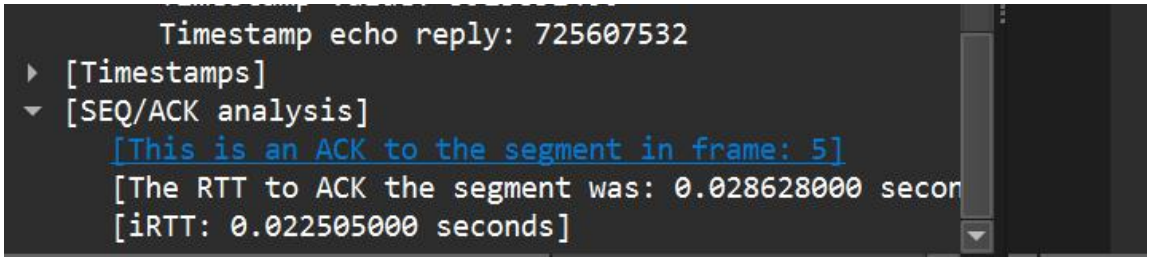


6.发送时间：0.024047s



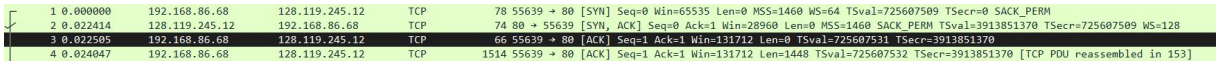
分析得知 7 是 4 的 ACK，RTT 是 0.022505000.

可见 5 是第二段的，ACK 是 1.
第二段的 RTT：

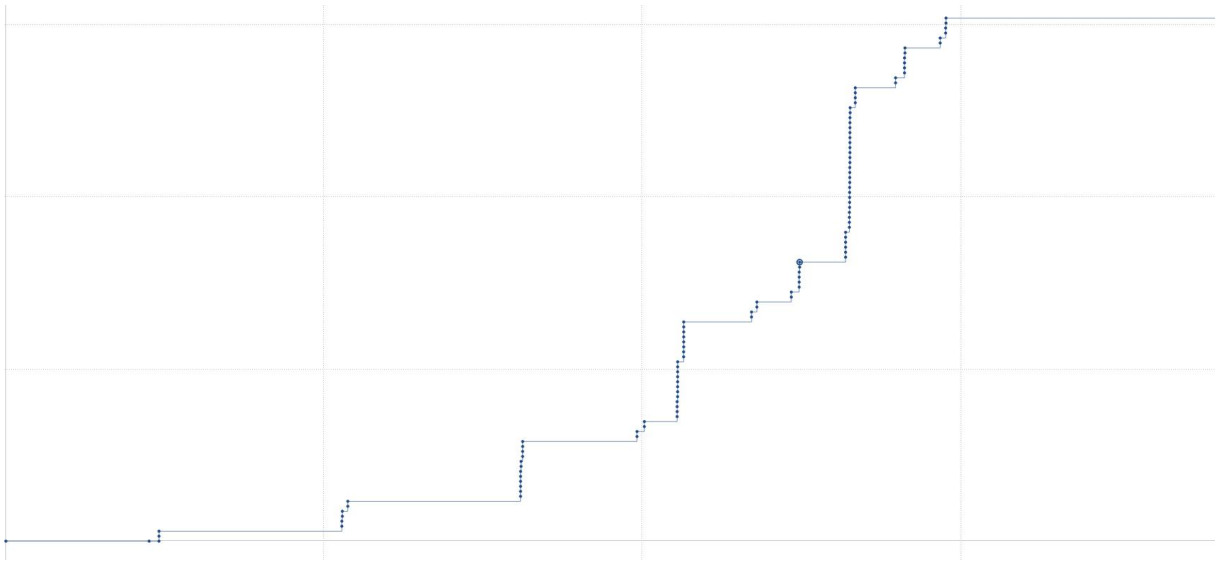


$$\text{EstimatedRTT} = (1 - 0.125) * 0.022505 + 0.125 * 0.028628$$
$$\text{EstimatedRTT} = (0.875 * 0.022505) + (0.125 * 0.028628)$$
$$\text{EstimatedRTT} = 0.019691875 + 0.0035785$$
$$\text{EstimatedRTT} \approx 0.023270375 \text{ 秒} = 23.270375 \text{ 毫秒}$$

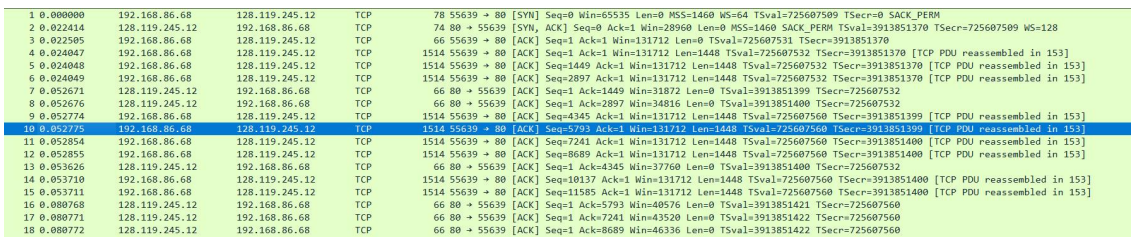
7. 长度是 1448 + 32 = 1480 字节
8. 观察发现最小的窗口大小是 28960，显然不会超出，不会造成限制。

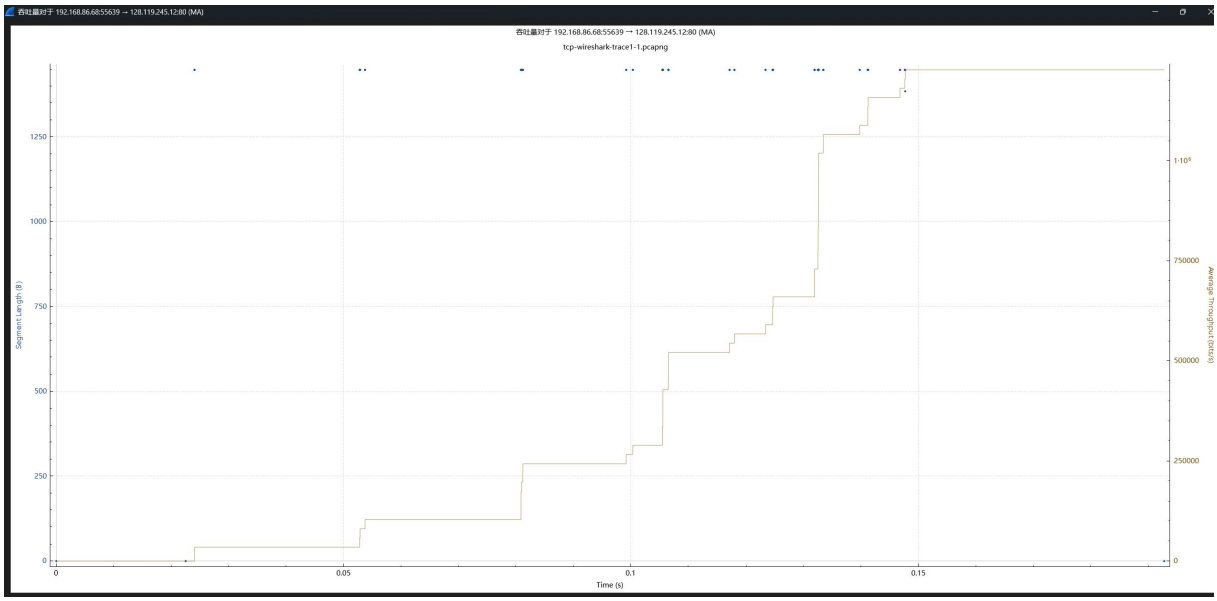


9. 跟踪文件中没有重新传输的段。我们可以通过检查跟踪文件中 TCP 数据段的序列号来验证这一点。在该轨迹的时间序列图(Stevens)中，从源(192.168.86.68)到目的地(128.119.245.12)的所有序列号都随着时间单调递增。如果存在重传数据段，则该重传数据段的序列号应小于其相邻数据段的序列号。

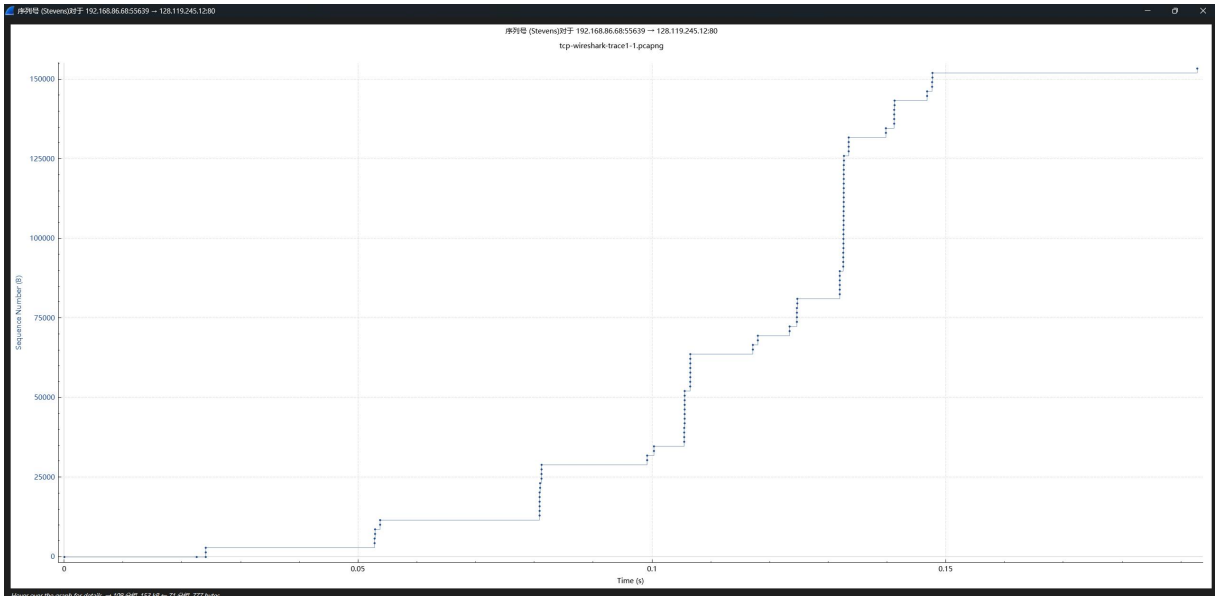


10. 数据承载段是 3、4、5、6, 10, 11, 12, 14, 15, 19。分析得知，客户端发送到 gaia.cs.umass.edu 的前 10 个数据承载段中，gaia.cs.umass.edu 通常在 ACK 中确认 1448, 5792，7240， 8688， 10136, 11584, 18824 的数据。没有出现每隔一个段确认一次的情况。





11. 通过此图可以分析，计算就是吞吐量/时间。



12. 这张图表明，TCP 连接可能经历了一个从慢启动到拥塞避免的过渡。 在连接的早期阶段，数据包的发送比较稀疏，序列号的增长也比较缓慢，这可能表明 TCP 处于慢启动阶段。 随着时间的推移，数据包的发送变得更加密集，序列号的增长也比较稳定，这可能表明 TCP 已经进入拥塞避免阶段。

13. 可能是 TCP 使用延迟确认机制。接收方不会立即发送 ACK，而是等待一段时间，如果在这段时间内收到了其他数据包，则将多个数据包一起确认。这种机制也会导致数据包以 “队列” 的形式到达，然后再一起被确认。此外，如果应用层的数据传输具有周期性，例如以固定的时间间隔发送日志数据或监控数据，那么这种周期性也会反映在 TCP 数据包的发送上。

结论分析与体会：