重庆大学大数据与软件学院

上 机 报 告

上机名称　　利用 wireshark 分析 TCP 协议

课程名称　　计算机网络

开课实验室：DS-1501　　**2025 年 3 月 30 日**

姓　名		学号		成　　绩	
上机（项目）名称		利用 wireshark 分析 TCP 协议		指导教师	高旻
教师评语					

教师签名：高旻

年　月　日

一、实验目的

1、深入理解 TCP 的工作原理，了解 TCP 的连接、数据传送、数据确认、拥塞控制等机制

2、进一步熟悉 Wireshark 的操作，学会利用 Wireshark 进行数据的截取和分析

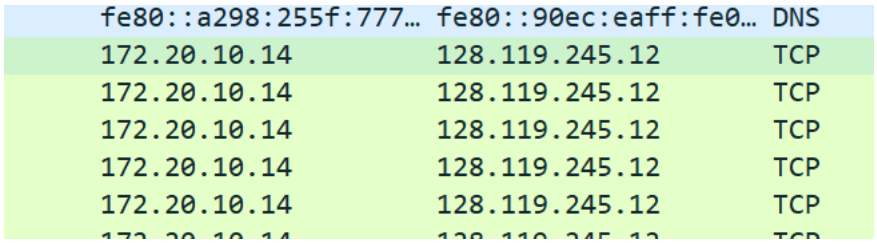
二、实验内容

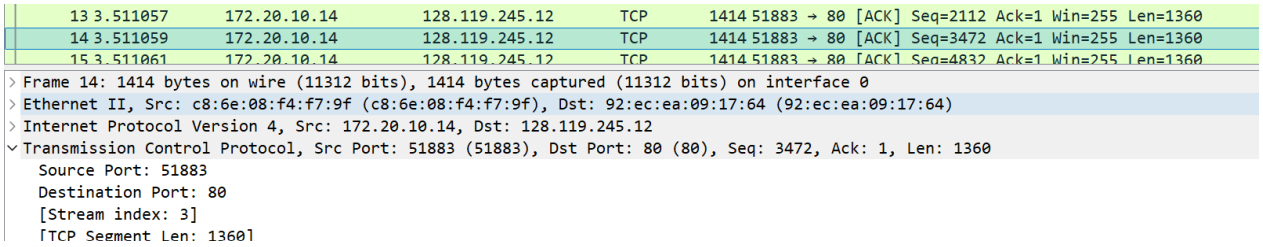
1、在浏览器输入：**http://gaia.cs.umass.edu/ethereal-labs/TCP-ethereal-file1.html**
2、用“选择文件”按钮选取实验文件包里的Alice.txt文件，先不要按“Upload alice.txt file”按钮；
3、打开 Wireshark，开始抓包；
4、再回到浏览器，按下“Upload alice.txt file”按钮向**gaia.cs.umass.edu**服务器来上载文件；
5、停止 Wireshark 的捕获；
6、**抓包后点菜单中的“文件”-“保存”，把抓到的包保存成一个文件。**

三、使用的软件、硬件

四、实验步骤及实验过程原始记录(回答实验指导书问题，记录数据、图表、计算等，需要有截图和简要说明)

1. 根据下面两个截图，我们可以看出客户端主机的 IP 地址为 172.20.10.14，使用的端口号为 51883





2. 服务器的IP地址为128.119.245.12。此次使用的端口号是80。

3.客户服务器其之间初始化 TCP 连接的 SYN 报文段序号是 0。在头部信息中，用 Syn 状态设为 1，表明是 SYN 报文段

```
Sequence number: 0    (relative sequence number)
Acknowledgment number: 1    (relative ack number)
Header Length: 32 bytes
Flags: 0x012 (SYN, ACK)
  000. .... .... = Reserved: Not set
  ...0 .... .... = Nonce: Not set
  .... 0... .... = Congestion Window Reduced (CWR): Not set
  .... .0.. .... = ECN-Echo: Not set
  .... ..0. .... = Urgent: Not set
  .... ...1 .... = Acknowledgment: Set
  .... .... 0... = Push: Not set
  .... .... .0.. = Reset: Not set
  .... .... ..1. = Syn: Set
  .... .... ...0 = Fin: Not set
```

4.

服务器向客户端发送的 SYNACK 报文段序号是 1，确认标识是 1

```
Sequence number: 1    (relative sequence number)
[Next sequence number: 752    (relative sequence number)]
Acknowledgment number: 1    (relative ack number)
    .... .0.. .... = ECN-Echo: Not set
    .... ..0. .... = Urgent: Not set
    .... ...1 .... = Acknowledgment: Set
    .... .0.. .... = ECN-Echo: Not set
    .... ..0. .... = Urgent: Not set
    .... ...1 .... = Acknowledgment: Set
    .... .... 0... = Push: Not set
    .... .... .0.. = Reset: Not set
    .... .... ..1. = Syn: Set
    .... .... ...0 = Fin: Not set
    [TCP Flags: *******A**S*]
```

CK 片段的

5.初始号为 1

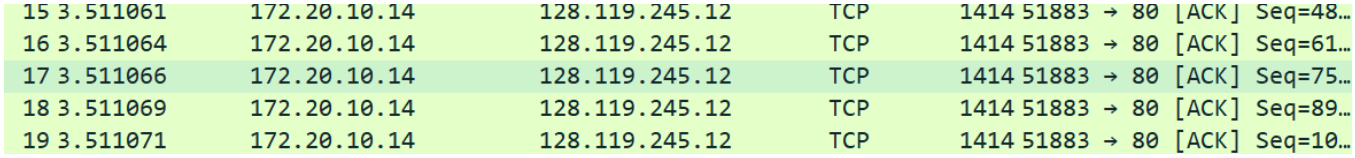
```
Sequence number: 1    (relative sequence number)
[Next sequence number: 752    (relative sequence number)]
Acknowledgment number: 1    (relative ack number)
```

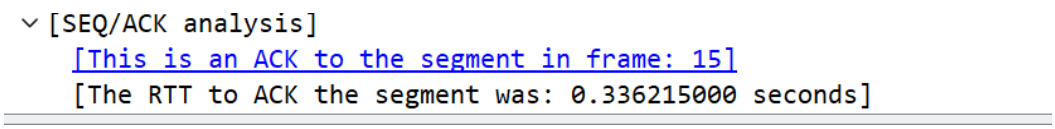
6.

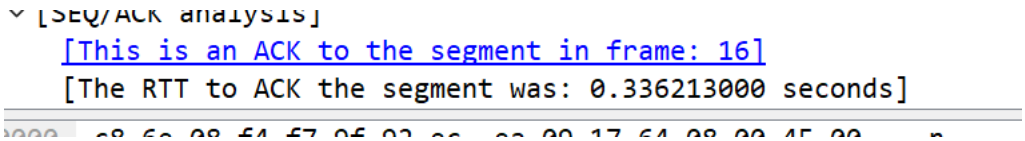
接收方 frame 分别是：27，28，29，30

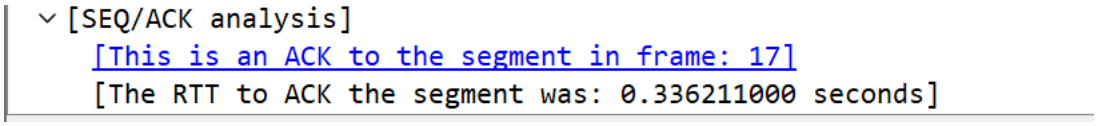
26	3.847276	128.119.245.12	172.20.10.14	TCP	54 80 → 51883 [ACK] Seq=1 Ack=3472 Win=
27	3.847276	128.119.245.12	172.20.10.14	TCP	54 80 → 51883 [ACK] Seq=1 Ack=6192 Win=
28	3.847277	128.119.245.12	172.20.10.14	TCP	54 80 → 51883 [ACK] Seq=1 Ack=7552 Win=
29	3.847277	128.119.245.12	172.20.10.14	TCP	54 80 → 51883 [ACK] Seq=1 Ack=8912 Win=
30	3.847277	128.119.245.12	172.20.10.14	TCP	54 80 → 51883 [ACK] Seq=1 Ack=11632 Win

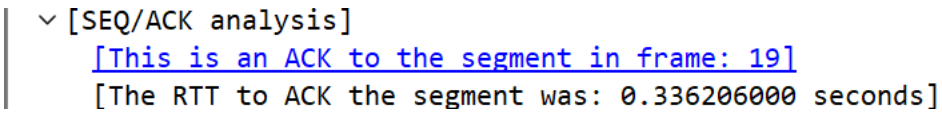
它们所对应的发送方的报文段分别是 15，16，17，19



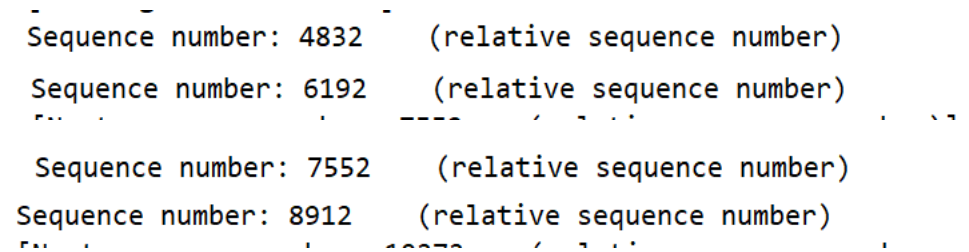








四个发送方对应的序列号：



序号	对应的序列号	发送时间	接受 AVK 时间	RTT
15	4832	3.511061	3.847276	0.33621500
16	6192	3.511064	3.847277	0.33621300
17	7552	3.511066	3.847277	0.33621100
19	8912	3.511069	3.847277	0.33620600

由 EstimatedRTT = (1- α)*EstimatedRTT + α*SampleRTT（假设α=0.125）
接收到第 1 段后的 EstimatedRTT 为：
EstimatedRTT = 3.847276 s
接收到第 2 段后的 EstimatedRTT 为：
EstimatedRTT = 0.875 * 3.847276 + 0.125 * 0.33621300 = 3.408393125 s
接收到第 3 段后的 EstimatedRTT 为：

EstimatedRTT = 0.875 * 3.408393125 + 0.125 * 0.33621100 = 3.02437035 s

接收到第 4 段后的 EstimatedRTT 为：

EstimatedRTT = 0.875 * 3.02437035 + 0.125 * 0.33620600 =3.0243697 s

7. 都是 1360bytes

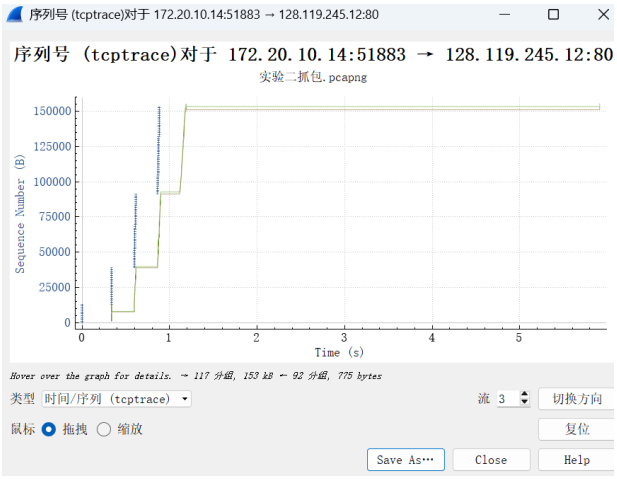
```
> Transmission Control Protocol, Src Port: 51883 (51883), Dst Port: 80 (80), Seq: 4832, Ack: 1, Len: 1360
> Transmission Control Protocol, Src Port: 51883 (51883), Dst Port: 80 (80), Seq: 6192, Ack: 1, Len: 1360
> Transmission Control Protocol, Src Port: 51883 (51883), Dst Port: 80 (80), Seq: 7552, Ack: 1, Len: 1360
> Transmission Control Protocol, Src Port: 51883 (51883), Dst Port: 80 (80), Seq: 10272, Ack: 1, Len: 1360
```

8.最大为 2301/最小为 325


```
Sequence number: 776    (relative sequence number)
Acknowledgment number: 153074    (relative ack number)
Header Length: 20 bytes
> Flags: 0x011 (FIN, ACK)
Window size value: 2301
[Calculated window size: 2301]
[Window size scaling factor: -1 (unknown)]
```

```
Header Length: 20 bytes
> Flags: 0x010 (ACK)
Window size value: 325
[Calculated window size: 325]
[Window size scaling factor: -1 (unknown)]
> Checksum: 0x4e82 [validation disabled]
Urgent pointer: 0
v [SEQ/ACK analysis]
  [This is an ACK to the segment in frame: 15]
```

9. 图可以看到，发送方每隔一定的时间发送一连串的分组，从图可以看到这些分组的序列号 Sequence number 出现了重复，所以存在出现重传分组的行为。



10.

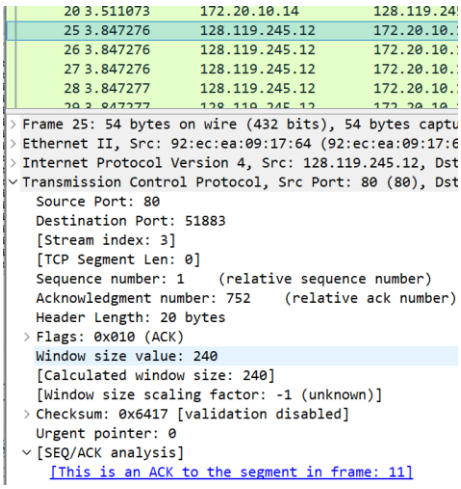
本次实验关于 ACK 确认规律不太明显

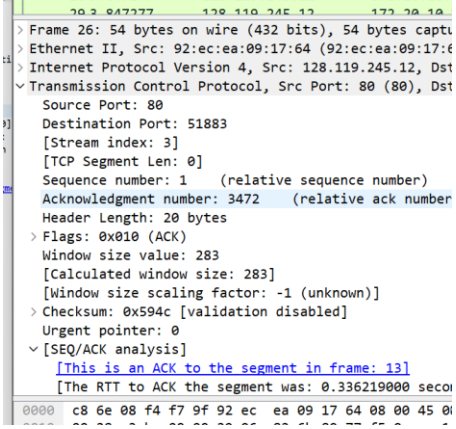
但 28 对于 27，7552-6192=1306（可能原因可以是 **MTU（最大传输单元）**：若网络中某设备的 MTU 小于 1500 字节（如 PPPoE、VPN 隧道等），TCP 会通过**路径 MTU 发现**自动调整 MSS（最大报文段大小））

25	3.847276	128.119.245.12	172.20.10.14	TCP	54 80 → 51883 [ACK] Seq=1 Ack=752 Win=240 Len=0
26	3.847276	128.119.245.12	172.20.10.14	TCP	54 80 → 51883 [ACK] Seq=1 Ack=3472 Win=283 Len=0
27	3.847276	128.119.245.12	172.20.10.14	TCP	54 80 → 51883 [ACK] Seq=1 Ack=6192 Win=325 Len=0
28	3.847277	128.119.245.12	172.20.10.14	TCP	54 80 → 51883 [ACK] Seq=1 Ack=7552 Win=348 Len=0
29	3.847277	128.119.245.12	172.20.10.14	TCP	54 80 → 51883 [ACK] Seq=1 Ack=8912 Win=371 Len=0
30	3.847277	128.119.245.12	172.20.10.14	TCP	54 80 → 51883 [ACK] Seq=1 Ack=11632 Win=413 Len=0
31	3.847277	128.119.245.12	172.20.10.14	TCP	54 80 → 51883 [ACK] Seq=1 Ack=12992 Win=436 Len=0

比如 26 对 25，3472-752=2720（可能是上述基本大小的两倍，也就是累计确认了两个包）

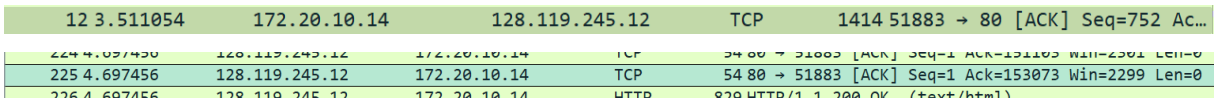
（26 对 12，13 累积确认）





11

。因此总的数据量为 153073 - 1 = 153072 bytes，整个传输的时间为 4.697456 - 3.51154=1.185916 秒，因此 TCP 连接的吞吐量为 153072/1.185916= 126.05 Kbyte/s。



12.