

# tcpiplab

221501029潘泓旭

221501029@smail.nju.edu.cn

1.我的ID:172.24.38.17,TCP端口:2952

2.服务器端口:8080

```
▶ Internet Protocol Version 4, Src: 172.24.38.17, Dst: 114.212.10.193
▼ Transmission Control Protocol, Src Port: 2952, Dst Port: 8080, Seq: 151837, Ack: 1, Len: 1321
    Source Port: 2952
    Destination Port: 8080
```

3.TCP(6)

```
...0 0000 0000 0000 = Fragment Offset: 0
Time to Live: 128
Protocol: TCP (6)
Header Checksum: 0xc11b [validation disabled]
[Header checksum status: Unverified]
```

4.20bytes,有效负载为总长度-报头长度=1341

```
▼ Internet Protocol Version 4, Src: 172.24.38.17, Dst: 114.212.10.193
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
    ▶ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 1361
```

- 5.没分段,因为Don't fragment是Set,More fragments是Not set,  
fragment offset也为0
6. 59195,128

```
Identification: 0xe73b (59195)
▼ 010. .... = Flags: 0x2, Don't fragment
    0... .... = Reserved bit: Not set
    .1... .... = Don't fragment: Set
    ..0. .... = More fragments: Not set
    ...0 0000 0000 0000 = Fragment Offset: 0
Time to Live: 128
```

7.相对序列号为151837,绝对序列号为2818240797;根据三次握手,客户端发送SYN请求来建立连接,我找到发送的第一个请求并且发现我电脑把SYN标为151837从而建立连接,sequence number:2818240797是个随机值,以上是三次握手的第一步

```
Sequence Number: 151837 (relative sequence number)
Sequence Number (raw): 2818240797
[Next Sequence Number: 153158 (relative sequence number)]
Acknowledgment Number: 1 (relative ack number)
Acknowledgment number (raw): 973546000
0101 .... = Header Length: 20 bytes (5)
► Flags: 0x018 (PSH, ACK)
```

8.381,153158(2818242118)Acknowledge number是前面那个的Next Sequence Number;意思是服务器收到连接请求并且发送SYN-ACK确认,这是三次握手第二步骤

```
Sequence Number: 381 (relative sequence number)
Sequence Number (raw): 973546380
[Next Sequence Number: 386 (relative sequence number)]
Acknowledgment Number: 153158 (relative ack number)
Acknowledgment number (raw): 2818242118
```

9. 1

```

Transmission Control Protocol, Src Port: 2952, Dst Port: 8080, Seq: 1, Ack: 1, Len: 728
  Source Port: 2952
  Destination Port: 8080
  [Stream index: 3]
  [Conversation completeness: Incomplete, DATA (15)]
  [TCP Segment Len: 728]
  Sequence Number: 1 (relative sequence number)

```

10.如表所示;

序号号	发送时间	ACK序号	接收时间	RTT	EstimatedRTT
14	0.195109	24	0.196977	0.00187	0.001868
28	0.197639	48	0.199928	0.00229	0.001920625
41	0.198829	64	0.200847	0.00202	0.001932797
55	0.199973	85	0.201926	0.00195	0.001935322
69	0.200937	118	0.204686	0.00375	0.002162032
81	0.200937	121	0.204686	0.00375	0.002360403

$$\text{EstimatedRTT} = 0.875 * \text{EstimatedRTT} + 0.125 * \text{SampleRTT}$$

14	2023-12-17 03:29:43.195109	172.24.38.17	114.212.10.193	TCP	782	2952 → 8080	[PSH, ACK]	Seq=1 Ack=1
28	2023-12-17 03:29:43.197639	172.24.38.17	114.212.10.193	TCP	1514	2952 → 8080	[PSH, ACK]	Seq=16789 Ack=1
41	2023-12-17 03:29:43.198829	172.24.38.17	114.212.10.193	TCP	1514	2952 → 8080	[PSH, ACK]	Seq=32849 Ack=1
55	2023-12-17 03:29:43.199973	172.24.38.17	114.212.10.193	TCP	1514	2952 → 8080	[PSH, ACK]	Seq=48909 Ack=1
69	2023-12-17 03:29:43.200937	172.24.38.17	114.212.10.193	TCP	1514	2952 → 8080	[PSH, ACK]	Seq=64969 Ack=1
81	2023-12-17 03:29:43.200937	172.24.38.17	114.212.10.193	TCP	782	2952 → 8080	[PSH, ACK]	Seq=82489 Ack=1

14	2023-12-17 03:29:43.195109
28	2023-12-17 03:29:43.197639
41	2023-12-17 03:29:43.198829
55	2023-12-17 03:29:43.199973
69	2023-12-17 03:29:43.200937
81	2023-12-17 03:29:43.200937

24	2023-12-17 03:29:43.196977
48	2023-12-17 03:29:43.199928
64	2023-12-17 03:29:43.200847
85	2023-12-17 03:29:43.201926
118	2023-12-17 03:29:43.204686
121	2023-12-17 03:29:43.204686

(a) 发送时间

(b) 接收时间

11.由ACK号,可知长度各为

1)  $729-1=728$

2)  $18249-729=17520$

3)  $34309-18429=15880$

4)  $50369-34309=16060$

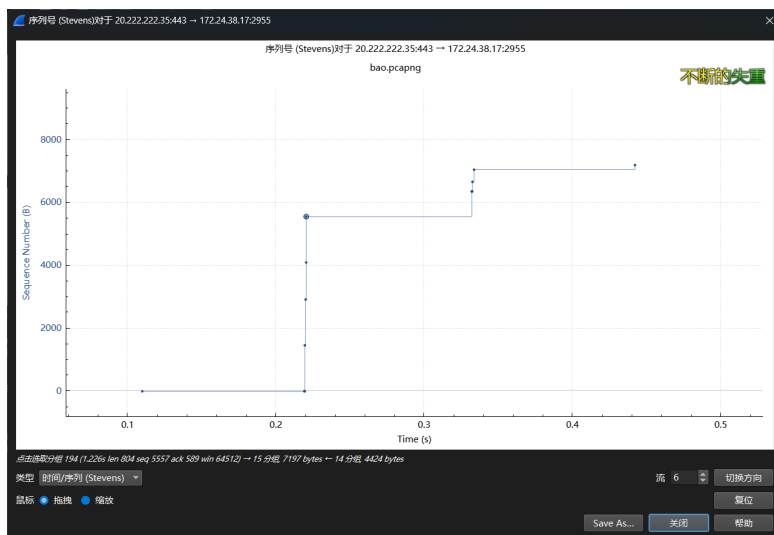
5)  $66429-50369=16060$

6)  $83217-66429=16788$

```
60 8080 → 2952 [ACK] Seq=1 Ack=729 W
60 8080 → 2952 [ACK] Seq=1 Ack=18249
60 8080 → 2952 [ACK] Seq=1 Ack=34309
60 8080 → 2952 [ACK] Seq=1 Ack=50369 W
60 8080 → 2952 [ACK] Seq=1 Ack=66429
60 8080 → 2952 [ACK] Seq=1 Ack=83217 W
```

12.30720字节(只需看第一个确认(窗口递增)),发送器永远不会因为接收器缓冲区空间不足而被抑制

13.没有,为此我检查了TCP数据段的序列号,发现是单调递增的



14.1460,注意到大致都是1460的倍数( $16060=1460*11$ , $17520=1460*12$ )

比1460大就可以识别发晚了的情况,我这里就都是这样的(汗颜)

15.可以通过第一个TCP数据的序列号和最后一个ACK的序列号计算,此处取六个好了

$(121-14)/(0.204686-0.195109)=11172.6$ (字节/秒)

