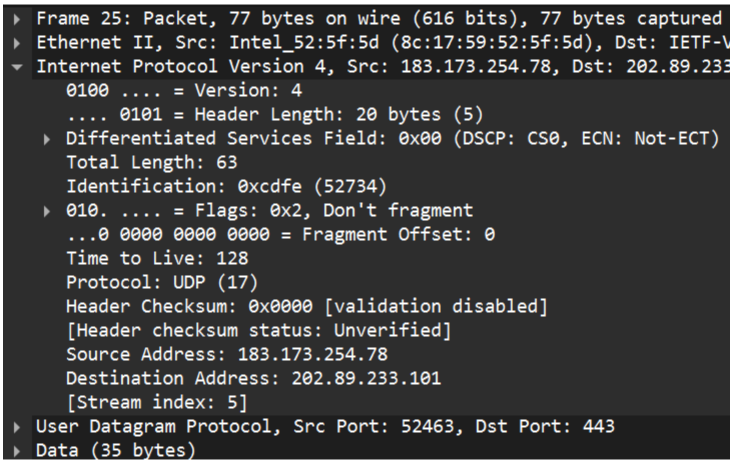
第四次实验报告

张峰源 2023010859

抓包实验1

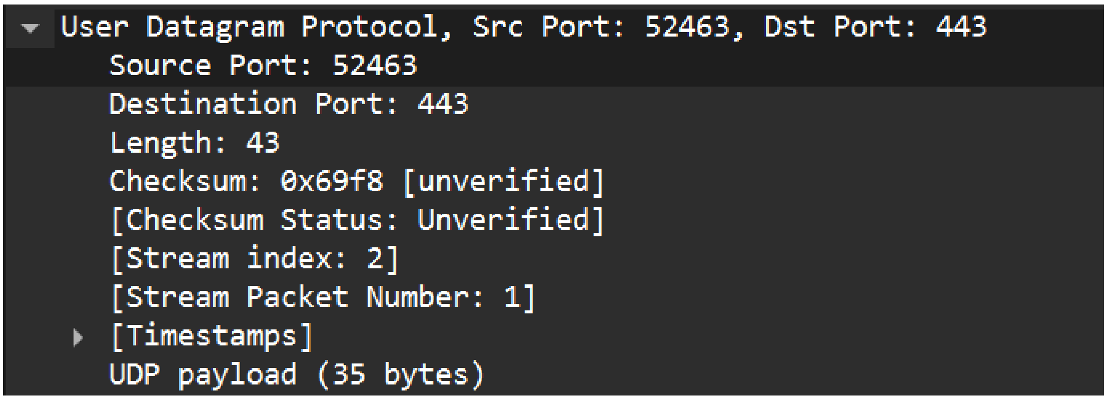
（1）UDP数据包在IP层的类型编号是?



```
▸ Frame 25: Packet, 77 bytes on wire (616 bits), 77 bytes captured
▸ Ethernet II, Src: Intel_52:5f:5d (8c:17:59:52:5f:5d), Dst: IETF-V
▾ Internet Protocol Version 4, Src: 183.173.254.78, Dst: 202.89.233
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  ▸ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 63
    Identification: 0xcdfe (52734)
  ▸ 010. .... = Flags: 0x2, Don't fragment
    ...0 0000 0000 0000 = Fragment Offset: 0
    Time to Live: 128
    Protocol: UDP (17)
    Header Checksum: 0x0000 [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 183.173.254.78
    Destination Address: 202.89.233.101
    [Stream index: 5]
▸ User Datagram Protocol, Src Port: 52463, Dst Port: 443
▸ Data (35 bytes)
```

编号为17.

（2）UDP数据包头字段依次是?

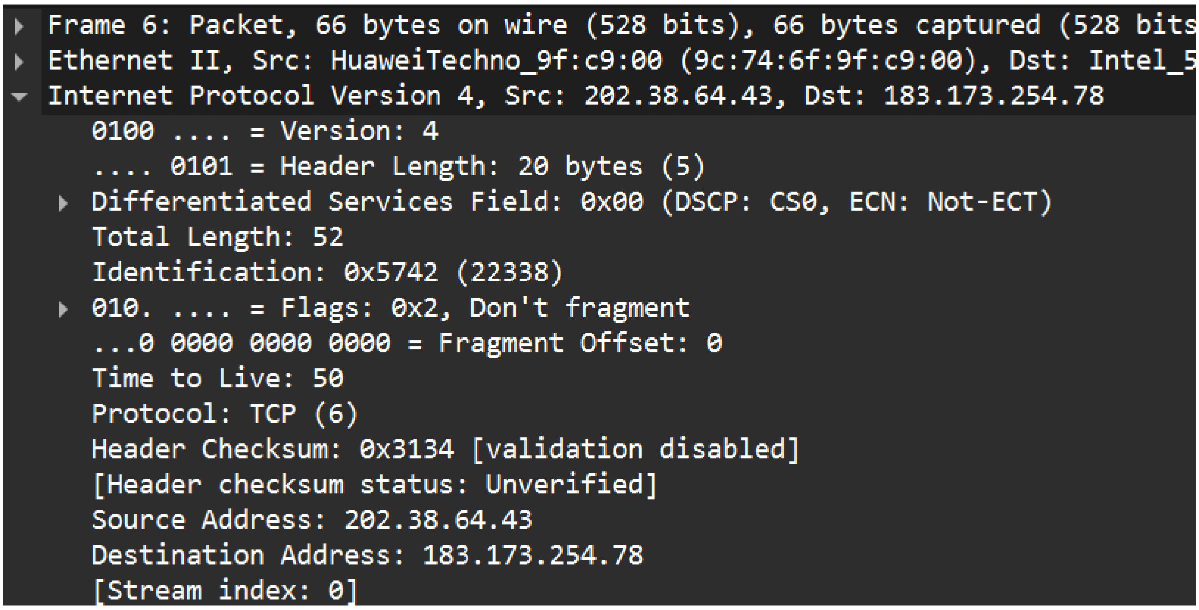


```
▾ User Datagram Protocol, Src Port: 52463, Dst Port: 443
    Source Port: 52463
    Destination Port: 443
    Length: 43
    Checksum: 0x69f8 [unverified]
    [Checksum Status: Unverified]
    [Stream index: 2]
    [Stream Packet Number: 1]
  ▸ [Timestamps]
    UDP payload (35 bytes)
```

源端口、目的端口、UDP长度、校验和

抓包实验2

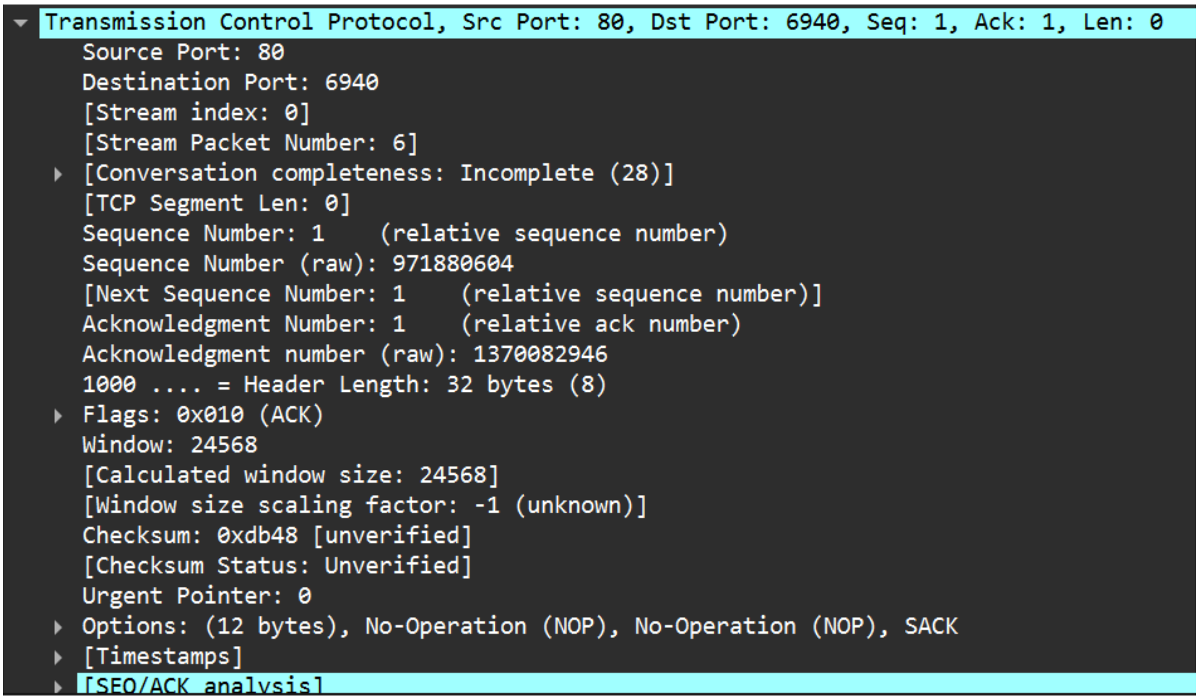
（1）TCP数据包在IP层的类型编号是?



```
▸ Frame 6: Packet, 66 bytes on wire (528 bits), 66 bytes captured (528 bits
▸ Ethernet II, Src: HuaweiTechno_9f:c9:00 (9c:74:6f:9f:c9:00), Dst: Intel_5
▾ Internet Protocol Version 4, Src: 202.38.64.43, Dst: 183.173.254.78
     0100 .... = Version: 4
     .... 0101 = Header Length: 20 bytes (5)
   ▸ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
     Total Length: 52
     Identification: 0x5742 (22338)
   ▸ 010. .... = Flags: 0x2, Don't fragment
     ...0 0000 0000 0000 = Fragment Offset: 0
     Time to Live: 50
     Protocol: TCP (6)
     Header Checksum: 0x3134 [validation disabled]
     [Header checksum status: Unverified]
     Source Address: 202.38.64.43
     Destination Address: 183.173.254.78
     [Stream index: 0]
```

编号为6

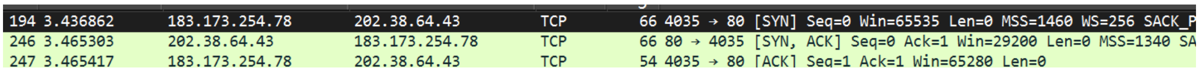
（2）TCP数据包头字段依次是?



```
▾ Transmission Control Protocol, Src Port: 80, Dst Port: 6940, Seq: 1, Ack: 1, Len: 0
     Source Port: 80
     Destination Port: 6940
     [Stream index: 0]
     [Stream Packet Number: 6]
   ▸ [Conversation completeness: Incomplete (28)]
     [TCP Segment Len: 0]
     Sequence Number: 1    (relative sequence number)
     Sequence Number (raw): 971880604
     [Next Sequence Number: 1    (relative sequence number)]
     Acknowledgment Number: 1    (relative ack number)
     Acknowledgment number (raw): 1370082946
     1000 .... = Header Length: 32 bytes (8)
   ▸ Flags: 0x010 (ACK)
     Window: 24568
     [Calculated window size: 24568]
     [Window size scaling factor: -1 (unknown)]
     Checksum: 0xdb48 [unverified]
     [Checksum Status: Unverified]
     Urgent Pointer: 0
   ▸ Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), SACK
   ▸ [Timestamps]
   ▸ [SEQ/ACK analysis]
```

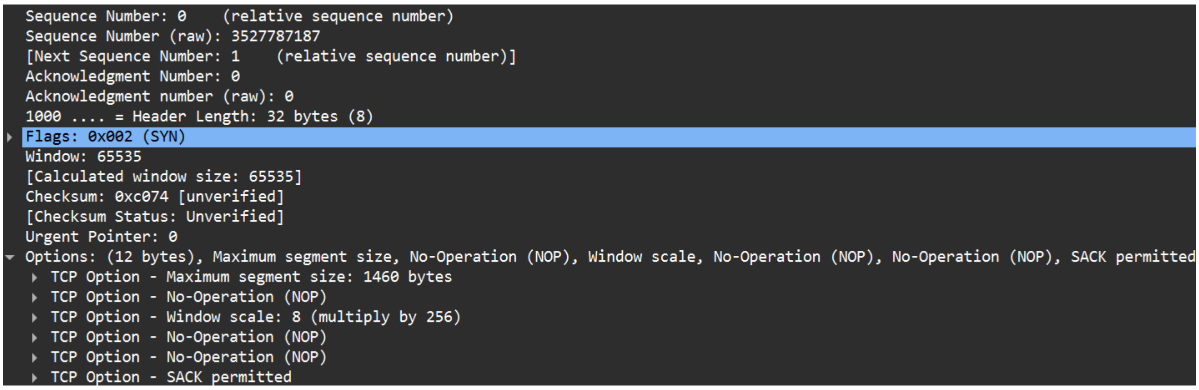
源端口、目的端口、序列号、确认序列号、TCP头部长度、标志位（flags）、窗口大小、校验和、紧急指针

（3）TCP三次握手过程使用三个数据包，他们的标记位，序列号，确认序列号有什么特点？TCP握手时使用选项协商链接参数，举出一个例子?



```
194 3.436862    183.173.254.78    202.38.64.43      TCP     66 4035 → 80 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=256 SACK_P
246 3.465303    202.38.64.43      183.173.254.78    TCP     66 80 → 4035 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1340 SA
247 3.465417    183.173.254.78    202.38.64.43      TCP     54 4035 → 80 [ACK] Seq=1 Ack=1 Win=65280 Len=0
```

如图，前三个TCP包的标记为分别为：SYN　SYN,ACK　ACK

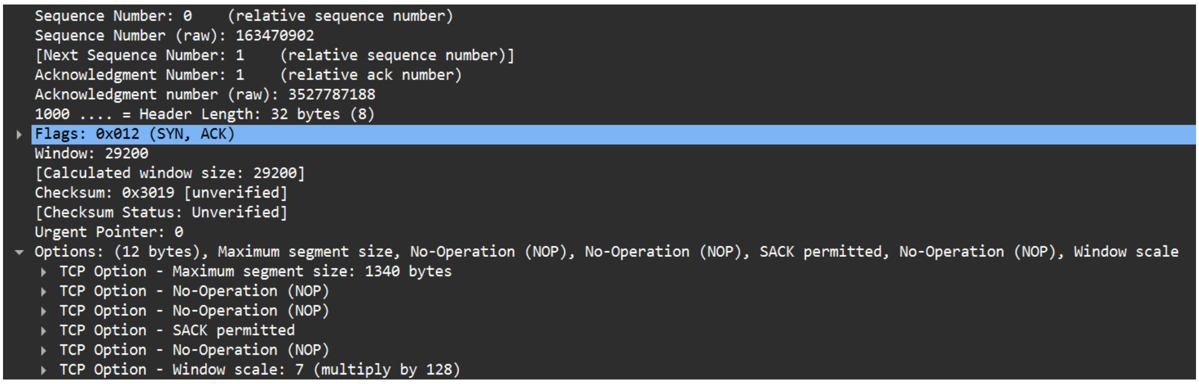


```
  Sequence Number: 0    (relative sequence number)
  Sequence Number (raw): 3527787187
  [Next Sequence Number: 1    (relative sequence number)]
  Acknowledgment Number: 0
  Acknowledgment number (raw): 0
  1000 .... = Header Length: 32 bytes (8)
▸ Flags: 0x002 (SYN)
  Window: 65535
  [Calculated window size: 65535]
  Checksum: 0xc074 [unverified]
  [Checksum Status: Unverified]
  Urgent Pointer: 0
▾ Options: (12 bytes), Maximum segment size, No-Operation (NOP), Window scale, No-Operation (NOP), No-Operation (NOP), SACK permitted
  ▸ TCP Option - Maximum segment size: 1460 bytes
  ▸ TCP Option - No-Operation (NOP)
  ▸ TCP Option - Window scale: 8 (multiply by 256)
  ▸ TCP Option - No-Operation (NOP)
  ▸ TCP Option - No-Operation (NOP)
  ▸ TCP Option - SACK permitted
```

第一次握手：序列号为0（raw为客户端随机初始值），确认序列号0

options为

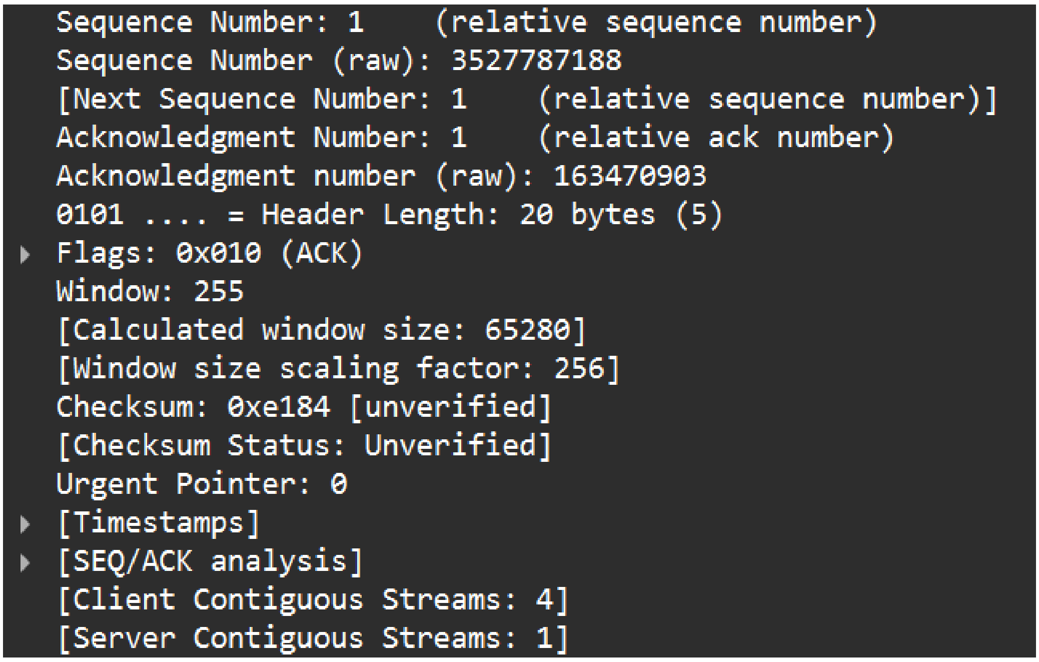
- 最大分段大小，在 SYN 包中告知对方自己能接收的最大 TCP 段长度（避免 IP 分片）
- 窗口缩放，扩展 TCP 窗口的最大上限
- SACK，选择性确认。协商是否支持仅重传丢失的数据段



```
  Sequence Number: 0    (relative sequence number)
  Sequence Number (raw): 163470902
  [Next Sequence Number: 1    (relative sequence number)]
  Acknowledgment Number: 1    (relative ack number)
  Acknowledgment number (raw): 3527787188
  1000 .... = Header Length: 32 bytes (8)
▸ Flags: 0x012 (SYN, ACK)
  Window: 29200
  [Calculated window size: 29200]
  Checksum: 0x3019 [unverified]
  [Checksum Status: Unverified]
  Urgent Pointer: 0
▾ Options: (12 bytes), Maximum segment size, No-Operation (NOP), No-Operation (NOP), SACK permitted, No-Operation (NOP), Window scale
  ▸ TCP Option - Maximum segment size: 1340 bytes
  ▸ TCP Option - No-Operation (NOP)
  ▸ TCP Option - No-Operation (NOP)
  ▸ TCP Option - SACK permitted
  ▸ TCP Option - No-Operation (NOP)
  ▸ TCP Option - Window scale: 7 (multiply by 128)
```

第二次握手：序列号为0（raw为服务器随机初始值），确认序列号为1（客户端随机初始值+1）

options依旧为之前的三个词条

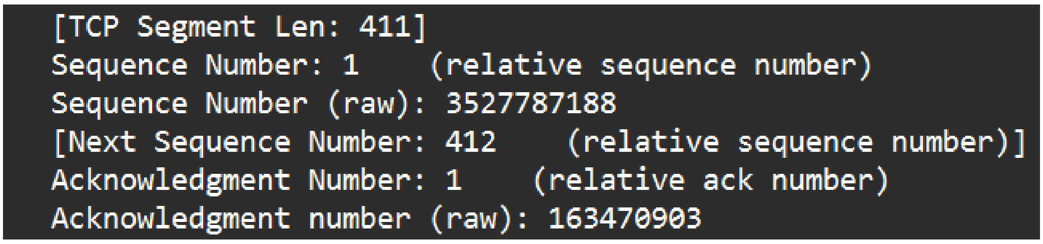


```
  Sequence Number: 1    (relative sequence number)
  Sequence Number (raw): 3527787188
  [Next Sequence Number: 1    (relative sequence number)]
  Acknowledgment Number: 1    (relative ack number)
  Acknowledgment number (raw): 163470903
  0101 .... = Header Length: 20 bytes (5)
  Flags: 0x010 (ACK)
  Window: 255
  [Calculated window size: 65280]
  [Window size scaling factor: 256]
  Checksum: 0xe184 [unverified]
  [Checksum Status: Unverified]
  Urgent Pointer: 0
  [Timestamps]
  [SEQ/ACK analysis]
  [Client Contiguous Streams: 4]
  [Server Contiguous Streams: 1]
```

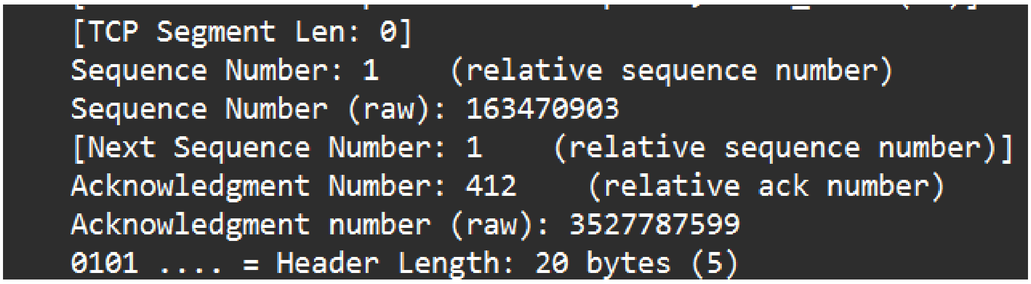
第三次握手：序列号为1（延续了客户端序列号），确认序列号为1（服务器随机初始值+1）

无options（已确认过协商信息）

（4）TCP传输过程中利用序列号和确认序列号实现数据的可靠传输。序列号增长和包长关系是什么？确认包确认序列号和原包序列号的关系是什么？



```
  [TCP Segment Len: 411]
  Sequence Number: 1    (relative sequence number)
  Sequence Number (raw): 3527787188
  [Next Sequence Number: 412    (relative sequence number)]
  Acknowledgment Number: 1    (relative ack number)
  Acknowledgment number (raw): 163470903
```



```
  [TCP Segment Len: 0]
  Sequence Number: 1    (relative sequence number)
  Sequence Number (raw): 163470903
  [Next Sequence Number: 1    (relative sequence number)]
  Acknowledgment Number: 412    (relative ack number)
  Acknowledgment number (raw): 3527787599
  0101 .... = Header Length: 20 bytes (5)
```

序列号增长为包长（TCP Segment Len）

确认包确认序列号为原包序列号加原包长。

简述题

（1）TCP建立连接时使用选项协商MTU信息。上网查资料，TCP选项还支持什么特殊的功能?

答：除 MTU 协商外，还支持：①MSS（最大分段大小）协商（避免 IP 分片）；②窗口缩放（WScale，扩展 TCP 窗口大小上限）；③选择性确认（SACK，仅重传丢失的数据段）；④时间戳（TSopt，计算往返时间 RTT，避免序列号回绕）；⑤TCP 快速打开（TFO，减少连接建立延迟）。

（2）反射DoS攻击中，攻击者将数据包源地址改为受害者IP向公共服务(DNS，NTP等等)发送请求，公共服务回复数据包至受害者IP，使受害者带宽耗尽。为什么此类攻击大多使用基于UDP的公共服务，而不是基于TCP呢?

答：UDP是无连接协议，不需要TCP的三次握手。如果基于TCP进行攻击，那么第一次握手后，公共服务向受害者IP发送SYN,ACK确认，而此时受害者IP收到后由于实际并未发起连接请求，会返回RST，连接终止，就不能开始数据包传输了。