TCP 协议攻击

N6.1. 本题基于以下 TCP 客户端程序。（1）客户端需要一个源端口才能接收服务器响应，但程序中没有显式绑定端口。它如何收到响应？（2）哪一行代码触发 TCP 三次握手？（3）程序调用了两次 `sendall()`，每次调用是否一定产生一个独立的 TCP 报文？

```
#!/bin/env python3
import socket

tcp = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
tcp.connect(('10.0.2.69', 9090))

tcp.sendall(b"Hello Server!\n")
tcp.sendall(b"Hello Again!\n")

tcp.close()
```

N6.2. 本题基于以下 TCP 服务器程序。（1）`listen()` 是否会阻塞，直到连接到来？（2）`accept()` 有什么作用？（3）为什么 `accept()` 要返回一个新套接字，不能继续使用监听套接字？

```
#!/bin/env python3
import socket

tcp = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
tcp.bind(("0.0.0.0", 9090))
tcp.listen()

conn, addr = tcp.accept()
with conn:
    print('Connected by', addr)
    while True:
        data = conn.recv(1024)
        if not data:
            break
        print(data)
```

```
        conn.sendall(b"Got the data!\n")
```

N6.3. 我们有两台机器，A 和 B。(1) 机器 A 上的两个 TCP 客户端程序将它们的数据发送到正在监听的 TCP 服务器 B 机的 8023 端口。这两个客户端程序的数据会在 B 机上混合在一起吗？服务器端？请解释一下。(2) 机器 A 上的两个 UDP 客户端程序将它们的数据发送到正在监听的 UDP 服务器 B 机的 8023 端口。这两个客户端程序的数据会在 B 机上混合在一起吗？服务器端？请解释一下。

N6.4. 一个程序要向服务器发送多段数据，每段数据通过一次独立的发送调用传输。服务器需要识别各段数据的边界。(1) 使用 UDP 时，服务器如何识别边界？(2) 使用 TCP 时又该怎么办？

N6.5. SYN 泛洪攻击是否会让受害服务器完全停止运行？

N6.6. SYN 泛洪攻击为什么要随机伪造源 IP，而不始终使用同一个地址？

N6.7. SYN 泛洪中伪造的源 IP 若恰好属于一台正在运行的主机，会发生什么？

N6.8. 某 Telnet 服务同时监听端口 `23` 和 `8023`。攻击者只对默认端口 `23` 实施 SYN 泛洪。攻击期间，用户还能通过端口 `8023` 登录吗？

N6.9. 能否在没有 root 权限的情况下，从一台主机发起经典的伪造源地址 SYN 泛洪攻击？

N6.10. 为什么 SYN 洪泛攻击选择耗尽保存半开连接的内存，而不直接攻击保存完整连接的内存？完整连接需要更多内存，看起来似乎更容易耗尽资源。

N6.11. 如果 TCP 在三次握手的 `SYN+ACK` 中总使用固定初始序列号（例如 0），如何让服务器与大量不存在的主机建立“完整连接”，从而耗尽内存？

N6.12. 服务器需要知道的所有信息关于连接的信息不仅包含在 `SYN` 数据包中，还包含在来自客户端的最终 `ACK` 数据包中。因此，就信息而言，没有必要分配缓冲区保存半开连接的信息。如果去掉这个缓冲区，SYN 洪泛攻击就不再有效。你是否同意这一说法？请说明理由。

N6.13. 攻击者看不到两台远程主机之间的报文。要伪造 RST 重置它们的 TCP 连接，主要困难是什么？

N6.14. TCP 重置攻击对 SSH 等加密连接是否仍然有效？

N6.15. UDP 通信是否会受到重置攻击？

N6.16. Telnet 客户端（10.0.2.5）与服务器（10.0.2.9）之间有一条活动连接。服务器刚确认到序列号 1000，客户端刚确认到序列号 3000。攻击者与两台主机位于同一局域网，希望劫持会话并在服务器上执行命令。请构造攻击报文并填写下列字段：

- 源 IP 和目的 IP；
- 源端口和目的端口；
- 序列号；
- TCP 数据字段。

N6.17. TCP 会话劫持时，服务器期望从序列号 X 开始接收数据，但攻击报文使用 X+100。攻击会成功还是失败？

N6.18. 我们可以对 SSH 连接发起 TCP 会话劫持攻击吗？

N6.19. 两台主机之间有一条活动 TCP 连接（Telnet 会话），下面是刚捕获的最新报文。请构造报文劫持该会话，在服务器端执行 "/bin/rm -f /"。由于嗅探器故障，确认号最后两位损坏，以 ** 表示。

不需要编写程序，只需展示数据包中的关键字段，并附上简要说明即可。

```
▶ Internet Protocol Version 4, Src: 10.9.0.7, Dst: 10.9.0.6
▼ Transmission Control Protocol, Src Port: 23, Dst Port: 45634 ...
    Source Port: 23
    Destination Port: 45634
    Sequence number: 2737422009
    Acknowledgment number: 7185323**
    ...
▶ TCP Data, length: 20 bytes
```

N6.20. Mitnick 攻击是 TCP 会话劫持的一种变体，涉及圣地亚哥超级计算机中心的两台主机 A 和 B。B 信任 A，从 A 登录 B 时无需密码。Kevin Mitnick 既不知道 B 的密码，也不能控制 A，只能从远程冒充 A。登录前必须先建立 TCP 连接。

为简化场景，先假设 A 离线、只有 B 在线。请说明 Mitnick 如何让 B 误以为已经与 A 建立连接。当时 TCP 初始序列号尚未充分随机化，具有可预测性。

N6.21. UDP 服务可用于反射放大攻击，为什么 TCP 服务通常难以用同样方式攻击？

N6.22. 用 Python 编写的 SYN 泛洪程序逻辑正确，却始终不能奏效。观察发现它每秒只能发送少量伪造报文，同时受害服务器收到大量 RST。请解释攻击失败的原因。

N6.23. 组织内部的机器 A（10.9.0.5）运行 "nc -l 9090"。防火墙只允许源 IP 为机器 B （192.168.60.5）的外部报文访问该服务。已知：（1）A 在 TCP 三次握手中始终使用固定初始序列号 0x1000；（2）B 当前离线。

你是远程攻击者，既不能控制 B，也无法嗅探 A 的报文，但希望向该 TCP 服务发送消息。请参考下面的 Scapy 示例编写 Python 攻击程序。

```
# Create an IP object
# src: source IP, dst: destination IP
ip = IP(src="1.1.1.1", dst="2.2.2.2")

# Create a TCP object
# sport: source port,      dport: destinatin port
# seq:   sequence number,  ack: acknowledge number
# flags: see the notes below
tcp = TCP(sport="33333", dport="23",
          seq=8000, ack=5000, flags="SA")

# Here are a few examples of the TCP flags
# flags = "A" : Set the ACK bit only
# flags = "S" : Set the SYN bit only
# flags = "SA": Set the SYN and ACK bits only

# Construct and send the packet
pkt = ip/tcp
send(pkt, verbose=0)
```

N6.24. Alice 使用 telnet 登录到一台服务器。在其会话中，攻击者捕获了如下 TCP 数据包。抓包完成后，Alice 暂时停止了操作，服务器也没有继续发送数据。

包编号	方向	Sequence #	ACK #	TCP 数据长度	Flags
1	Alice → Server	825400100	1642209000	18	PSH, ACK
2	Server → Alice	1642209000	825400118	36	PSH, ACK
3	Server → Alice	1642209000	825400118	36	PSH, ACK
4	Alice → Server	825400118	1642209036	0	ACK

相关地址和端口如下：

主机	IP 地址	TCP 端口
Alice	10.20.0.11	49860
Server	10.20.0.8	23

攻击者希望冒充 Alice，向服务器发送以下命令：`/bin/echo SEED > /tmp/x`

假设攻击者发送的 TCP payload 是该命令对应的 ASCII 字节，并在末尾添加一个换行符 `\n`。

请完成以下任务：

1. 判断上述数据包中是否存在 TCP 重传包，并说明判断依据。
2. 构造用于实施会话劫持的 TCP 数据包，给出以下关键字段：
   - 源 IP 地址和目的 IP 地址；
   - 源端口和目的端口；
   - Sequence Number；
   - Acknowledgment Number；
   - TCP flags；
   - TCP payload。
3. 计算该伪造数据包的 TCP payload 长度，并给出服务器成功接收该数据包后所期望的下一个 Sequence Number。

不需要编写程序，也不需要计算 IP 和 TCP 校验和。请简要说明各字段数值的推导过程。