南京大学本科生实验报告

课程名称：计算机网络　　　任课教师：田臣/李文中　　　助教：方毓楚、郑浩、陈伟等（排名不分先后）

学院	地理与海洋科学学院	专业	地理信息科学
学号	191830173	姓名	徐嵩
Email	song.xv@outlook.com	开始/完成日期	2022/3/24

1. 实验名称

Lab 6: Reliable Communication

2. 实验目的

构建一个在 IP 协议基础上的可靠传输网络，包含服务端、客户端和中间盒。

3. 实验内容

实现中间盒转发和随机丢弃

实现客户端回复 ACK

实现服务端构造可靠传输报文

4. 实验结果

Show how you implement the features of middlebox.

对于来自服务器端的报文选择性丢弃，通过生成的随机数与设定的 dropRate 比较，具体如下：

```
if random.random() < self.dropRate:
    log_info(f"Drop packet: {packet}")
    return
```

然后将收到的报文 ip 包头的 ttl 减 1，修改以太网包头的源地址和目的地址，从对应的端口转发。

Show how you implement the features of blastee.

从收到的报文中大端方式读取 RawPacketContents 的前 4 个字节作为 ACK 的序列号，第 5~6 字节作为有效载荷的长度，然后从有效载荷中截取前 8 个字节，作为 ACK 的有效载荷，不足补 0，发送 ACK。

```
# TODO: 获取 seqnum（4 字节）将其添加到 ACK 中
seqnum = int.from_bytes(packet[3].data[:4], 'big')
# TODO: 获取 length（2 字节）
length = int.from_bytes(packet[3].data[4:6], 'big')
```

Show how you implement the features of blaster.

为收到 ACK 时，检查 LHS 对应的 ACK 是否超时，若超时则将窗口中未收到 ACK 的序号加入 resend list 中。若 resend list 不为空，则发送队列中的第一个序号对应的报文，并删除该序号后返回。若已发送的包小于窗口大小，则将 RHS 右移一位，并发送对应的报文。

```python
# 判断是否符合发包条件
if self.checktime():
    log_info("Timeout, generate resend list...")
    self.acktimer = time.time()
    self.resend()

if self.resendlist:
    self.sendseq(self.resendlist[0])
    self.resendlist.pop(0)
    return

if self.RHS - self.LHS + 1 >= self.SW:
    log_info("Sender Window full")
    return

# 调用发包
if self.RHS < self.num:
    self.RHS += 1
    self.sendseq(self.RHS)
    log_info(f"Send a packet {self.RHS}")
```

收到报文时，截取 RawPacketContents 前 4 字节作为 ACK 序列号，并在 ACK list 中记录。然后遍历 ACK list，从 LHS 开始一直到 RHS+1 的位置，遇到 0 则停下，将对应的序号赋给 LHS，如果 LHS 与之前不同，则更新计时器。

```python
# TODO: 获取 ACK 的序号
seqnum = int.from_bytes(packet[3].data[:4], 'big')
self.acklist[seqnum - 1] = 1

nextLHS = self.LHS
for i in range(self.LHS, self.RHS + 2):
    nextLHS = i
    if self.acklist[i - 1] == 0:
        break

if nextLHS != self.LHS:
    self.LHS = nextLHS
    self.acktimer = time.time()
```
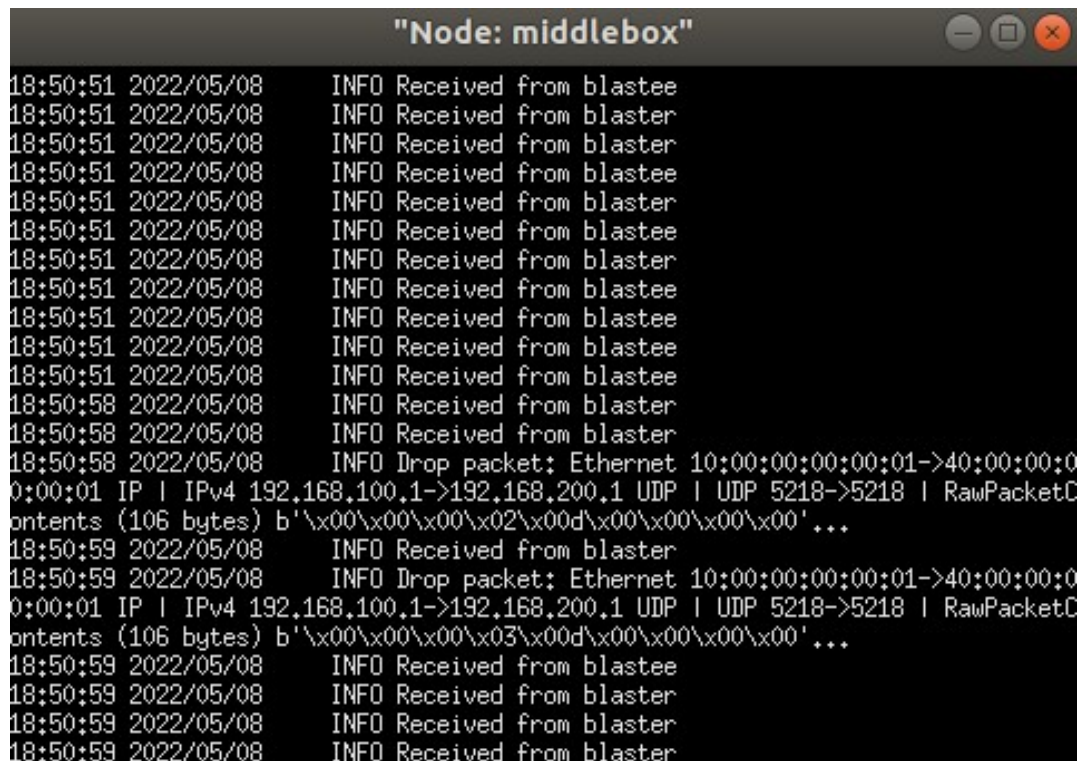
Write the procedure and analysis with screenshots.

先观察 middlebox 的输出日志，在截图中，非常巧合地，middlebox 将序列号为 2 和为 3 的包都丢弃了。

观察 blaster 的输出日志，发现由于序列号为 1 的包正常发送和收到 ACK，因此当 blaster 发送序列号为 6 的包后便不再发送。一直到超时事件发生，blaster 生成重发表，分别在两次 recvTimeout 事件中重发 2 和 3，在收到 2 和 3 的 ACK 后，blaster 的 LHS 直接变为 7，继续正常发包。在序列号 100 的包正常收到 ACK 后不再发送新的数据包。

传输完成后，输出结果本次传输共花费 11s，超时次数 16 次，重传分组 36 个，总吞吐量 1139Bps，有效吞吐量 837.9Bps 截图如下：

修改丢包率为 0.1 后，超时次数和重传次数接近原来的 1/2，截图如下：

修改重传时间为 150ms 后，总吞吐量、超时次数和重传次数接近原来的 2 倍，有效吞吐量变化不大，截图如下：