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：

学院	**工程管理院系**	专业（方向）	**金融工程**
学号	**211275041**	姓名	**杨晨毅**
Email		开始/完成日期	**5.27-5.29**

1. 实验名称

Lab-6 实现可靠通信

2. 实验目的

实现 blaster、middlebox、blastee 功能

3. 实验内容与核心代码

1. 实现 middlebox 的转发功能

若从 blaster 发来，则概率丢包

```
        print(intf.ethaddr)
if fromIface == "middlebox-eth0":
    print("recev from blaster")
    log_debug("Received from blaster")
    drop = self.dropRate
    if random.random()<=drop:
        print("A packet is dropped")
        return
    '''
    Received data packet
    Should I drop it?
    If not, modify headers & send to blastee
    '''
    packet[0].src = "40:00:00:00:00:02"
    packet[0].dst = "20:00:00:00:00:01"
    self.net.send_packet("middlebox-eth1", packet)
```

若从 blastee 发来，则直接转发

```
elif fromIface == "middlebox-eth1":
    print("receiv from blastee")
    log_debug("Received from blastee")
    packet[0].src = "40:00:00:00:00:01"
    packet[0].dst = "10:00:00:00:00:01"
    self.net.send_packet("middlebox-eth0", packet)
```

2. 实现 blastee 的回复功能

首先复制序列号在前 32 比特，之后判断 payload 长度是否足够 8byte，如果不够就补 0；

```
def handle_packet(self, recv: switchyard.llnetbase.ReceivedPacket):
    _, fromIface, packet = recv
    print("we get a pkt")
    log_debug(f"I got a packet from {fromIface}")
    log_debug(f"Pkt: {packet}")
    ack_pkt = Ethernet()+IPv4()+UDP()
    ack_pkt[0].src = "20:00:00:00:00:01"
    ack_pkt[0].dst = "40:00:00:00:00:01"
    ack_pkt[1].src = "192.168.200.1"
    ack_pkt[1].dst = self.blasterIp
    ack_pkt[1].protocol = IPProtocol.UDP
    ack_pkt[1].ttl = 64
    print(packet[3].to_bytes()[:4])
    ack_pkt += packet[3].to_bytes()[:4]
    lenth = int.from_bytes(packet[3].to_bytes()[4:6],byteorder='big')
    if lenth >= 8:
        ack_pkt += packet[3].to_bytes()[6:14]
    else:
        ack_pkt += packet[3].to_bytes()[6:]
        ack_pkt += (0).to_bytes(8-lenth,byteorder='big')
    print("we resend the pkt")
    self.net.send_packet(fromIface,ack_pkt)
```

3. 实现 blaster 的发包、sendwind、重发包功能

（1）handle_packet

首先在 handle 下取出包的序列号 sqc 然后对相应数据赋 1 表示已接到相应序列的包

```
def handle_packet(self, recv: switchyard.llnetbase.ReceivedPacket):
    _, fromIface, packet = recv
    log_debug("I got a packet")
    print("we get a ack pkt")
    print(packet)
    LHS = self.LHS
    RHS = self.RHS
    byt = packet[3].to_bytes()[:4]# get the sqc of ack pkt
    sqc = int.from_bytes(byt,byteorder='big')
    print("the sqc is :",sqc)
    if sqc == self.num:
        self.end = time.time()
    self.getpkt[sqc] = 1
```

接着更新 **LHS** 的指向，若成功则重设重传时间

```
oldlhs = LHS
while True: #move LHS to newest
    if self.getpkt[LHS]==1 and self.getpkt[LHS+1] == 1:
        LHS += 1
    else:
        break
if LHS != oldlhs:
    self.check_time = time.time()
```

之后对是否超时进行判断，若超时则跳过让 **handle_no** 做，若不超时则进行发包

```
oldlhs = LHS
while True: #move LHS to newest
    if self.getpkt[LHS]==1 and self.getpkt[LHS+1] == 1:
        LHS += 1
    else:
        break
if LHS != oldlhs:
    self.check_time = time.time()
if time.time()-self.check_time>=self.timeout/1000:
    print("timeout and send pkt again")
else:
    while RHS-LHS+2<=self.senderWindow and RHS<self.num:
        RHS+=1
        pkt = self.new_pkt(RHS)
        self.all_byte += int.from_bytes(pkt[3].to_bytes()[6:],byteorder='big')
        if self.have_sent[RHS] == 1:
            self.resend_count += 1
        else:
            self.have_sent[RHS] = 1
        self.net.send_packet('blaster-eth0',pkt)
```

最后进行结束条件的判断

```
    print("now RHS is : ",RHS)
    self.LHS = LHS
    self.RHS = RHS
    if(self.LHS == self.num and self.getpkt[self.num]==1):
        self.end = time.time()
        self.net.shutdown()

def new_pkt(self,seq):
```

（2）handle_no_packet

进行开始时间的设置

```
if self.check_time == 0:
    self.check_time = time.time()
```

之后进行超时重传从 LHS 开始，若 getpkt[i]不为 1（没收到包）则进行记录后重传。

```
if time.time() - self.check_time>=self.timeout/1000:
    self.to_count += 1
    print("timeout task")
    i = LHS
    self.check_time = time.time()
    while i <= RHS:
        if self.getpkt[i] == 0:
            pkt = self.new_pkt(i)
            print("we send packet")
            print(pkt)
            if self.have_sent[i] == 1:
                self.resend_count += 1
            else:
                self.have_sent[i] = 1
            self.net.send_packet('blaster-eth0',pkt)
        i+=1
    print("after check the LHS is :",LHS)
```

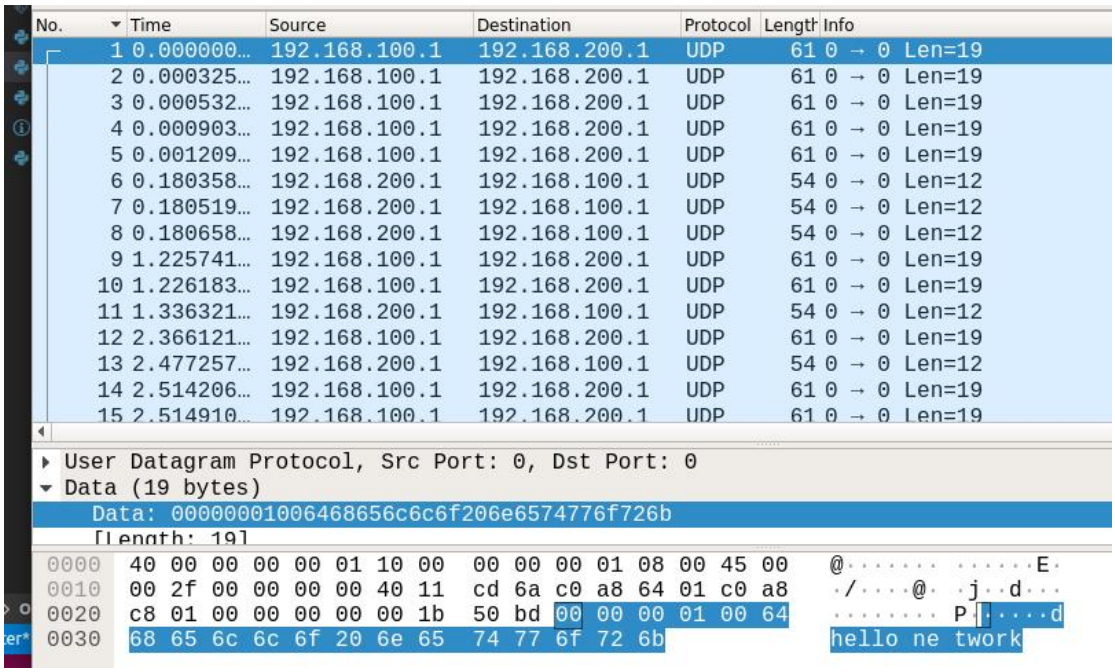
之后进行未超时的处理以 sendwindow 为标准判断是否应接下去发包。

```
else:
    if RHS == 1:
        self.begin = time.time() # get the first pkt time
        pkt = self.new_pkt(RHS)
        if self.have_sent[RHS] == 1:
            self.resend_count += 1
        else:
            self.have_sent[RHS] = 1
        self.net.send_packet('blaster-eth0',pkt)
    while RHS-LHS+2<=self.senderWindow:
        RHS+=1
        pkt = self.new_pkt(RHS)
        if self.have_sent[RHS] == 1:
            self.resend_count += 1
        else:
            self.have_sent[RHS] = 1
        self.net.send_packet('blaster-eth0',pkt)
```

最后抓包：

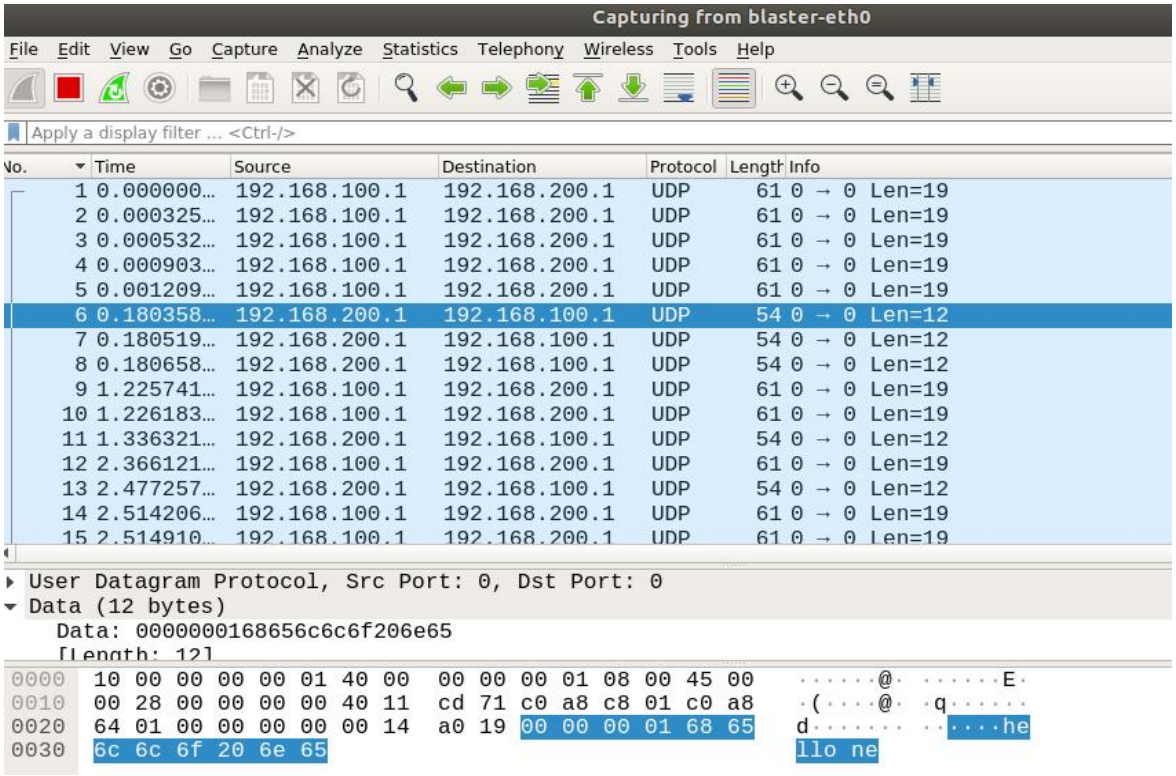
对第一个 blaster 发出的包，16 进制下，蓝色前 4 个为序列号，后两

个为 length，最后为 payload



对从 blastee 的回复包分析：

16 进制下，前四个为序列号，后 8 个为 payload 的前部截取校验



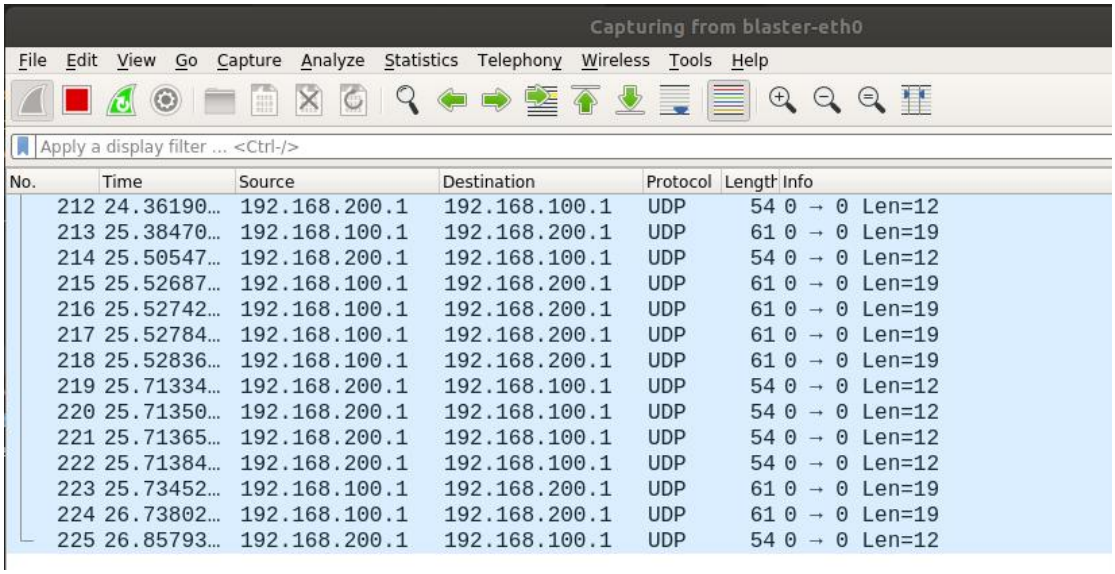
包的结构正确

4. 实验结果

以下是输出截图

```
Time is : 26.881626844406128
Number of reTX is : 25
TOs is : 19
Throughout puts is : 3.8153145768076096e+31
Goodput is : 3.046097928257688e+31
```

与 **blaster** 最后一个收包对比可知时间记录、重传次数正确



同时 **Goodput/Throughput =0.798 ≈ 100/125**（后发现是第一个包发出是未记录）改正代码后数据正确。

5. 总结与感想

进一步加深理解可靠通信