Writing Task 5

$1.$

2 sessions.

$12, 1652$ segments.

$2.$

$(10.0.0.74, 43120, 115.27.207.221, 80)$

$(10.0.0.74, 43122, 115.27.207.221, 80)$

$3.$

43520 . The "window" bytes are $0\text{x}55 = 85$, $85 \times 512 = 43520$.

Programming Task 4

I divided the whole TCP stack into 4 header files. `socket.h` is for sockets, `netstack.h` is to set up the network stack, `connection.h` is for maintaining TCP states, `tcp.h` is for reliable byte transmission.

Run Tasks

Run `sudo make` .

`cd vnetUtils/examples`, `sudo bash ./makeVNet < test.txt`

```
ns1 --- ns2 --- ns3 --- ns4
```

Then open a terminal with every ns hosts.

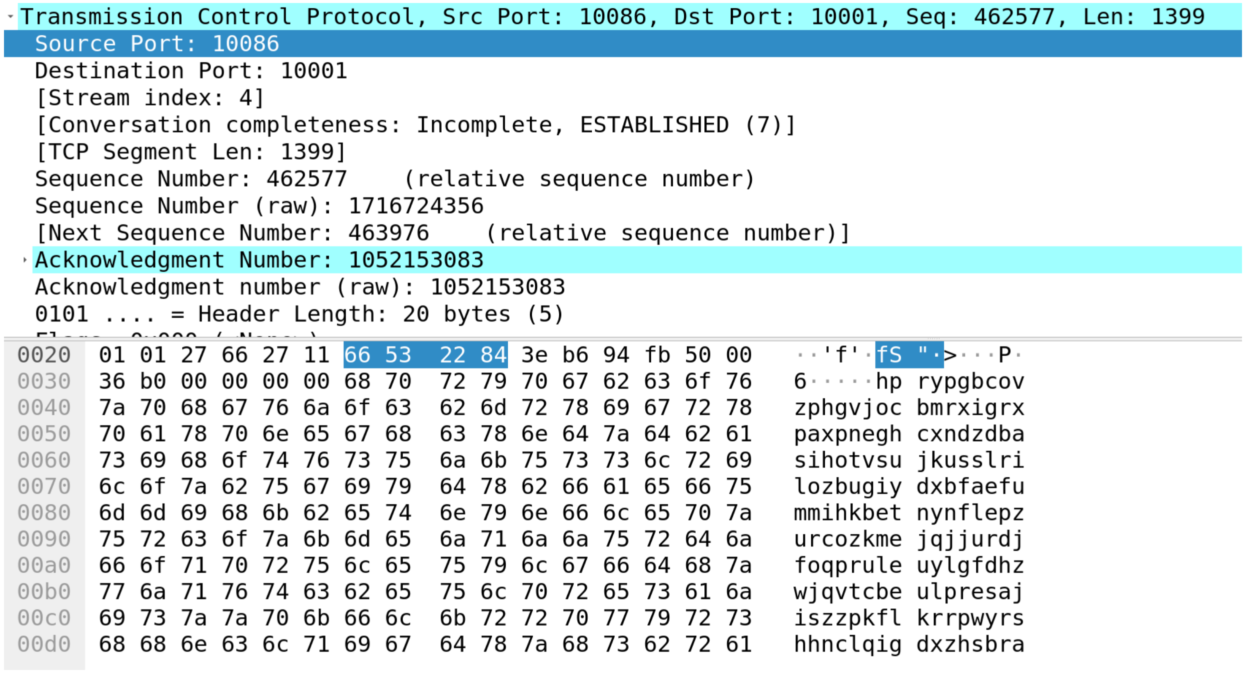
```
cd vnetUtils/helper;
sudo ./execNS ns* bash
cd ../../build
```

Then `sudo ./router` on ns2 and ns3.

In checkpoint 9, `sudo ./echo_server` on ns4, `sudo ./echo_client 10.100.3.2` on ns1.

In checkpoint 10, `sudo ./perf_server` on ns4, `sudo ./perf_client 10.100.3.2` on ns1.

Checkpoint 7



```
Transmission Control Protocol, Src Port: 10086, Dst Port: 10001, Seq: 462577, Len: 1399
  Source Port: 10086
  Destination Port: 10001
  [Stream index: 4]
  [Conversation completeness: Incomplete, ESTABLISHED (7)]
  [TCP Segment Len: 1399]
  Sequence Number: 462577    (relative sequence number)
  Sequence Number (raw): 1716724356
  [Next Sequence Number: 463976    (relative sequence number)]
  Acknowledgment Number: 1052153083
  Acknowledgment number (raw): 1052153083
  0101 .... = Header Length: 20 bytes (5)
```

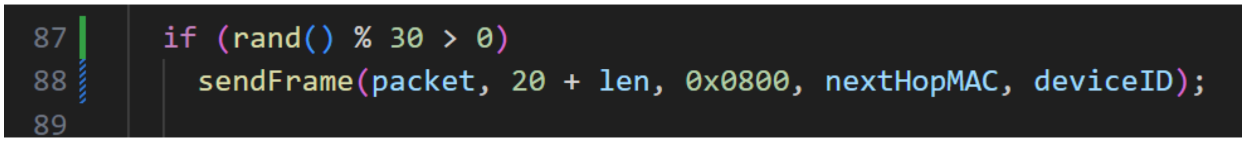
```
0020  01 01 27 66 27 11 66 53  22 84 3e b6 94 fb 50 00   ··'f'·fS "·>···P·
0030  36 b0 00 00 00 00 68 70  72 79 70 67 62 63 6f 76   6·····hp rypgbcov
0040  7a 70 68 67 76 6a 6f 63  62 6d 72 78 69 67 72 78   zphgvjoc bmrxigrx
0050  70 61 78 70 6e 65 67 68  63 78 6e 64 7a 64 62 61   paxpnegh cxndzdba
0060  73 69 68 6f 74 76 73 75  6a 6b 75 73 73 6c 72 69   sihotvsu jkusslri
0070  6c 6f 7a 62 75 67 69 79  64 78 62 66 61 65 66 75   lozbugiy dxbfaefu
0080  6d 6d 69 68 6b 62 65 74  6e 79 6e 66 6c 65 70 7a   mmihkbet nynflepz
0090  75 72 63 6f 7a 6b 6d 65  6a 71 6a 6a 75 72 64 6a   urcozkme jqjjurdj
00a0  66 6f 71 70 72 75 6c 65  75 79 6c 67 66 64 68 7a   foqprule uylgfdhz
00b0  77 6a 71 76 74 63 62 65  75 6c 70 72 65 73 61 6a   wjqvtcbe ulpresaj
00c0  69 73 7a 7a 70 6b 66 6c  6b 72 72 70 77 79 72 73   iszzpkfl krrpwyrs
00d0  68 68 6e 63 6c 71 69 67  64 78 7a 68 73 62 72 61   hhnclqig dxzhsbra
```

These bytes stand for the source port, destination port, sequence number, acknowledgment number, header's length, TCP flags, window size, TCP checksum, urgent pointer.

Checkpoint 8

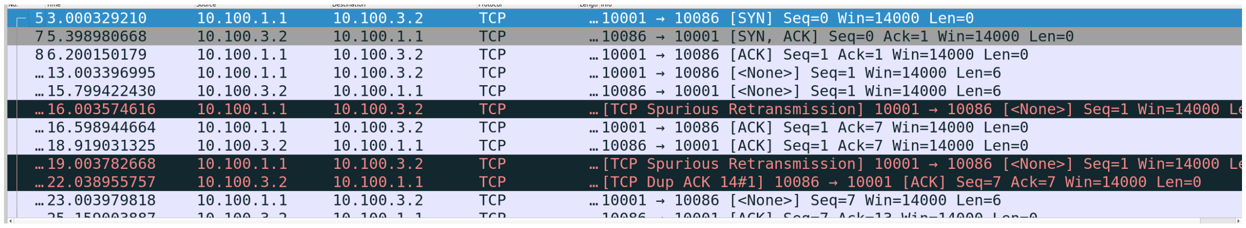
I used a very brutal emulation for bad links.

Whenever sending an IP packet, there's a chance of simply dropping it.



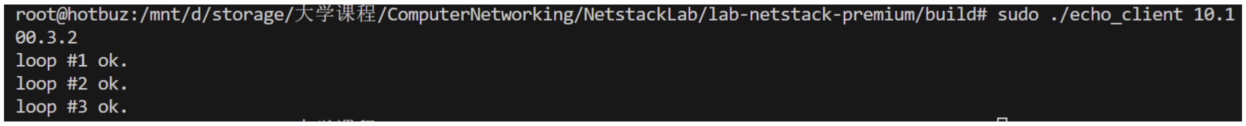
```
    if (rand() % 30 > 0)
      sendFrame(packet, 20 + len, 0x0800, nextHopMAC, deviceID);
```

This is checkpoint 9 running. The wireshark is monitoring vnet1-2 on ns1. We can see there are retransmissions and dup ACKs.

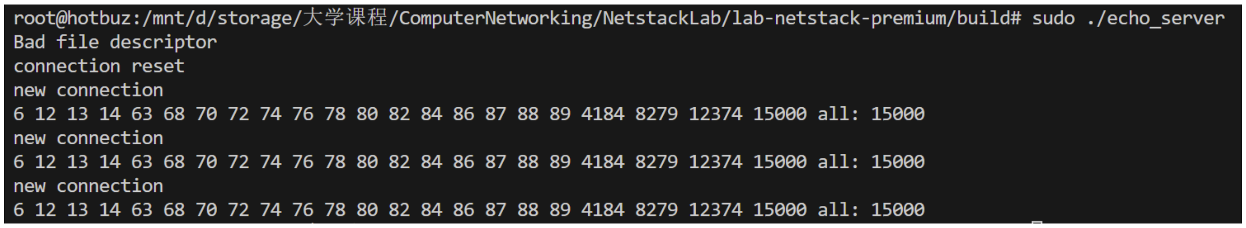


```
 5 3.000329210   10.100.1.1   10.100.3.2   TCP   …10001 → 10086 [SYN] Seq=0 Win=14000 Len=0
 7 5.398980668   10.100.3.2   10.100.1.1   TCP   …10086 → 10001 [SYN, ACK] Seq=0 Ack=1 Win=14000 Len=0
 8 6.200150179   10.100.1.1   10.100.3.2   TCP   …10001 → 10086 [ACK] Seq=1 Ack=1 Win=14000 Len=0
 …13.003396995   10.100.1.1   10.100.3.2   TCP   …10001 → 10086 [<None>] Seq=1 Win=14000 Len=6
 …15.799422430   10.100.3.2   10.100.1.1   TCP   …10086 → 10001 [<None>] Seq=1 Win=14000 Len=6
 …16.003574616   10.100.1.1   10.100.3.2   TCP   …[TCP Spurious Retransmission] 10001 → 10086 [<None>] Seq=1 Win=14000 Le
 …16.598944664   10.100.1.1   10.100.3.2   TCP   …10001 → 10086 [ACK] Seq=1 Ack=7 Win=14000 Len=0
 …18.919031325   10.100.3.2   10.100.1.1   TCP   …10086 → 10001 [ACK] Seq=1 Ack=7 Win=14000 Len=0
 …19.003782668   10.100.1.1   10.100.3.2   TCP   …[TCP Spurious Retransmission] 10001 → 10086 [<None>] Seq=1 Win=14000 Le
 …22.038955757   10.100.3.2   10.100.1.1   TCP   …[TCP Dup ACK 14#1] 10086 → 10001 [ACK] Seq=7 Ack=7 Win=14000 Len=0
 …23.003979818   10.100.1.1   10.100.3.2   TCP   …10001 → 10086 [<None>] Seq=7 Win=14000 Len=6
```

Checkpoint 9

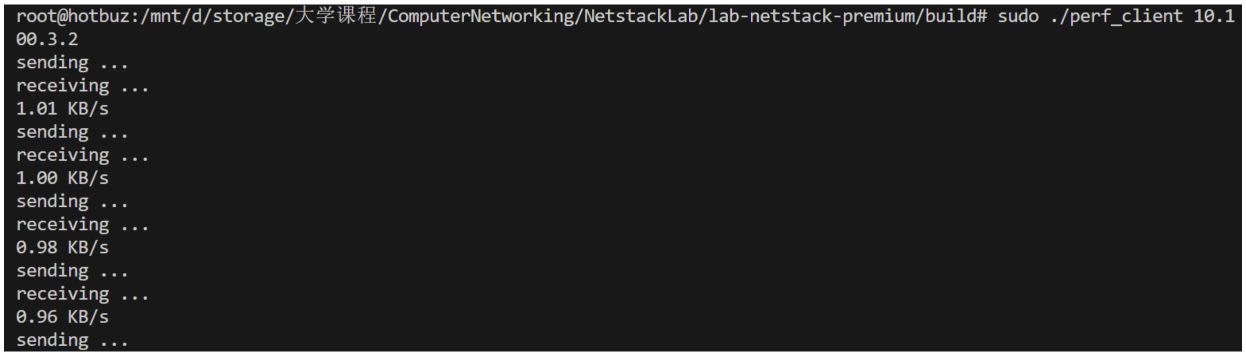


```
root@hotbuz:/mnt/d/storage/大学课程/ComputerNetworking/NetstackLab/lab-netstack-premium/build# sudo ./echo_client 10.1
00.3.2
loop #1 ok.
loop #2 ok.
loop #3 ok.
```



```
root@hotbuz:/mnt/d/storage/大学课程/ComputerNetworking/NetstackLab/lab-netstack-premium/build# sudo ./echo_server
Bad file descriptor
connection reset
new connection
6 12 13 14 63 68 70 72 74 76 78 80 82 84 86 87 88 89 4184 8279 12374 15000 all: 15000
new connection
6 12 13 14 63 68 70 72 74 76 78 80 82 84 86 87 88 89 4184 8279 12374 15000 all: 15000
new connection
6 12 13 14 63 68 70 72 74 76 78 80 82 84 86 87 88 89 4184 8279 12374 15000 all: 15000
```

Checkpoint 10



```
root@hotbuz:/mnt/d/storage/大学课程/ComputerNetworking/NetstackLab/lab-netstack-premium/build# sudo ./perf_client 10.1
00.3.2
sending ...
receiving ...
1.01 KB/s
sending ...
receiving ...
1.00 KB/s
sending ...
receiving ...
0.98 KB/s
sending ...
receiving ...
0.96 KB/s
sending ...
```