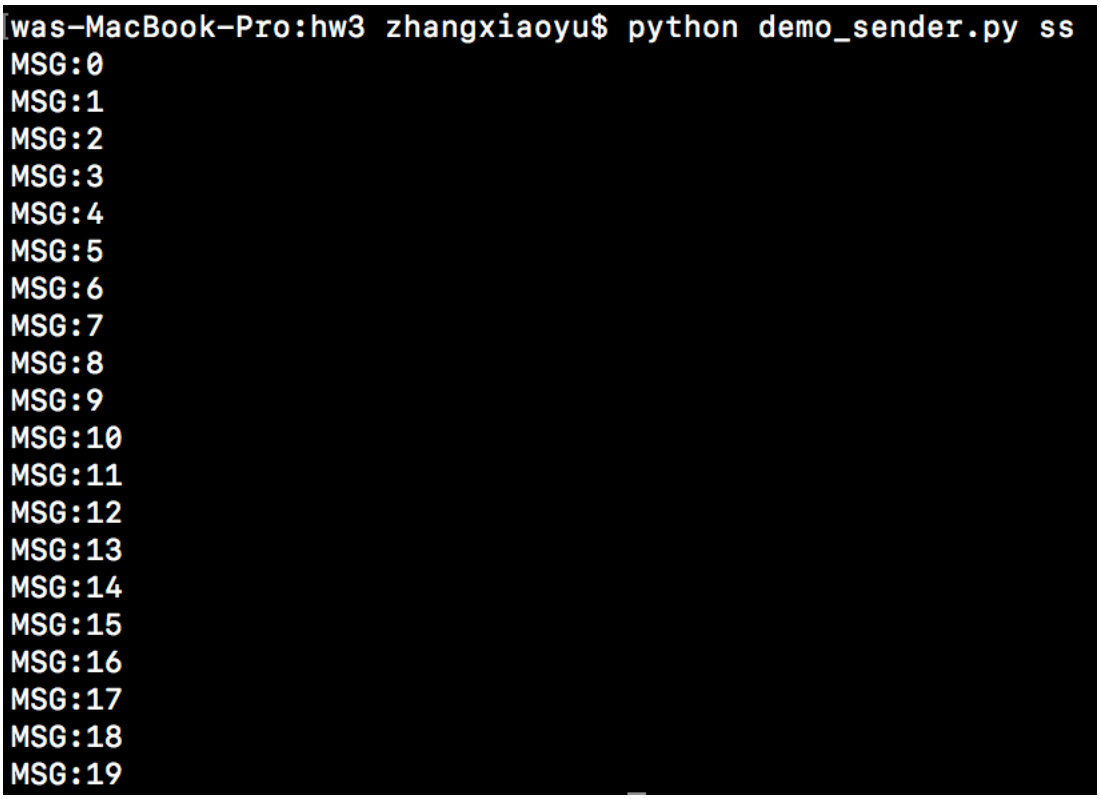
Assignment3-Report

Xiaoyu Zhang

1. Screenshot of your sender and server running successfully

a) The Successful run of the demo sender and receiver

(1) Stop-And-Wait:



```
was-MacBook-Pro:hw3 zhangxiaoyu$ python demo_receiver.py ss
'MSG:0'
'MSG:1'
'MSG:2'
'MSG:3'
'MSG:4'
'MSG:5'
'MSG:6'
'MSG:7'
'MSG:8'
'MSG:9'
'MSG:10'
'MSG:11'
'MSG:12'
'MSG:13'
'MSG:14'
'MSG:15'
'MSG:16'
'MSG:17'
'MSG:18'
'MSG:19'
```

(2) Go-Back-N:

```
was-MacBook-Pro:hw3 zhangxiaoyu$ python demo_sender.py gbn
MSG:0
MSG:1
MSG:2
MSG:3
MSG:4
MSG:5
MSG:6
MSG:7
MSG:8
MSG:9
MSG:10
MSG:11
MSG:12
MSG:13
MSG:14
MSG:15
MSG:16
MSG:17
MSG:18
MSG:19
```

```
was-MacBook-Pro:hw3 zhangxiaoyu$ python demo_receiver.py gbn
'MSG:0'
'MSG:1'
'MSG:2'
'MSG:3'
'MSG:4'
'MSG:5'
'MSG:6'
'MSG:7'
'MSG:8'
'MSG:9'
'MSG:10'
'MSG:11'
'MSG:12'
'MSG:13'
'MSG:14'
'MSG:15'
'MSG:16'
'MSG:17'
'MSG:18'
'MSG:19'
```

b) The Successful run of the file sender and receiver

(1) Stop-And-Wait:

```
was-MacBook-Pro:hw3 zhangxiaoyu$ python3 file_sender.py ss test.txt
MSG of length  500
MSG of length  500
MSG of length  500
MSG of length  500
MSG of length  500
MSG of length  500
MSG of length  500
MSG of length  500
MSG of length  500
MSG of length  500
MSG of length  500
MSG of length  500
MSG of length  500
MSG of length  500
MSG of length  500
MSG of length  500
MSG of length  500
MSG of length  89
Time used [secs]: 31.798991918563843
was-MacBook-Pro:hw3 zhangxiaoyu$
```

```
was-MacBook-Pro:hw3 zhangxiaoyu$ md5 test.txt
MD5 (test.txt) = 174ce37b93a177b4564f90a4fee7292c
was-MacBook-Pro:hw3 zhangxiaoyu$ md5 output.txt
MD5 (output.txt) = 174ce37b93a177b4564f90a4fee7292c
was-MacBook-Pro:hw3 zhangxiaoyu$
```

(2) Go-Back-N:

```
was-MacBook-Pro:hw3 zhangxiaoyu$ python3 file_sender.py gbn test.txt
MSG of length  500
MSG of length  500
MSG of length  500
MSG of length  500
MSG of length  500
MSG of length  500
MSG of length  500
MSG of length  500
MSG of length  500
MSG of length  500
MSG of length  89
Time used [secs]: 30.979515075683594
```

```
was-MacBook-Pro:hw3 zhangxiaoyu$ md5 test.txt
MD5 (test.txt) = 174ce37b93a177b4564f90a4fee7292c
was-MacBook-Pro:hw3 zhangxiaoyu$ md5 output_gbn.txt
MD5 (output_gbn.txt) = 174ce37b93a177b4564f90a4fee7292c
was-MacBook-Pro:hw3 zhangxiaoyu$
```

2. A simple description of how you implement two different protocols

First of all, since both of the two protocols need to make packet, unpacket and checksum, I wrote these helpers in the file function.py

(1) Stop-And-Wait:

Sender: When sender wants to send a message, first it makes a packet, sends to the network layer, and starts a timer. If it receives ack before time out, it will send the next packet, otherwise it will resend the same packet.

Receiver: When the receiver receives a message from the network layer, first it will unpacket the message. If the message is correct, the receiver will send ack to the sender. If the message is not correct, the receiver will do nothing and wait for the time out, then the sender will send the same packet again.

(2) Go-Back-N:

Sender: The sender will send window size number of packets continuously, and waiting for the acks from the receiver. Every time it receives an ack, the base number pluses 1. After the time

out, the sender will start from the current base, since before this base all the packets are correctly received by the receiver.

Receiver: If it receives the correct packet, it sends an ack to the sender, otherwise it sends the previous ack to the sender.

3. Compare the performances of the two protocol under different environment:

a) The high and low error rate

I set the high error rate to 0.2 and low error rate to 0.05 and compare two protocols.

(1) Stop-And-Wait

A. High Error Rate 0.2------44.7s

```
was-MacBook-Pro:hw3 zhangxiaoyu$ python3 file_sender.py ss test.txt
MSG of length  500
MSG of length  500
MSG of length  500
MSG of length  500
MSG of length  500
```

```
MSG of length  500
MSG of length  89
Time used [secs]: 44.69341588020325
was-MacBook-Pro:hw3 zhangxiaoyu$
```

B. Low Error Rate 0.05 ----- 25.6s

```
was-MacBook-Pro:hw3 zhangxiaoyu$ python3 file_sender.py ss test.txt
MSG of length  500
MSG of length  500
MSG of length  500
MSG of length  500
MSG of length  500
```

```
MSG of length  500
MSG of length  500
MSG of length  500
MSG of length  89
Time used [secs]: 25.56318211555481
```

(2) Go-Back-N

A. High Error Rate 0.2------ 43.3s

```
was-MacBook-Pro:hw3 zhangxiaoyu$ python3 file_sender.py gbn test.txt
MSG of length  500
MSG of length  500
MSG of length  500
MSG of length  500
MSG of length  500
```

```
MSG of length  500
MSG of length  500
MSG of length  500
MSG of length  89
Time used [secs]: 43.286295652389526
```

B. Low Error Rate 0.05 ----- 22.8s

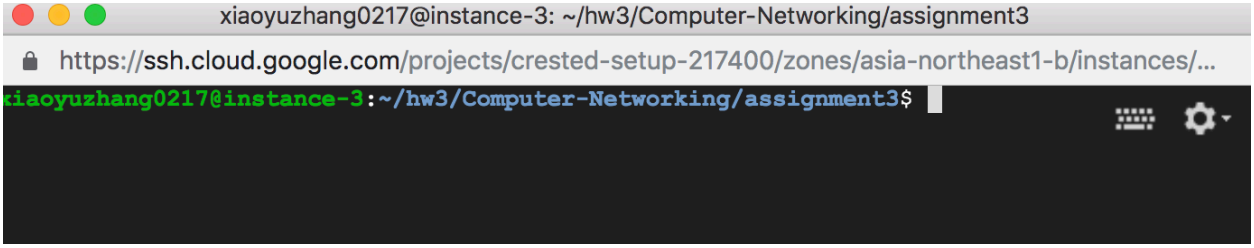
```
was-MacBook-Pro:hw3 zhangxiaoyu$ python3 file_sender.py gbn test.txt
MSG of length  500
MSG of length  500
MSG of length  500
MSG of length  500
MSG of length  500
```

```
MSG of length  500
MSG of length  500
MSG of length  89
Time used [secs]: 22.78952407836914
```

Conclusion: The higher error rate is, the longer transmission time it takes.

b) Long and short RTT (you may use VMs located in different places to simulate the RTT, e.g. connections between Oregon and California should have longer RTT than that between Oregon and Beijing)

I run my receiver in East Asia and receiver in West US.



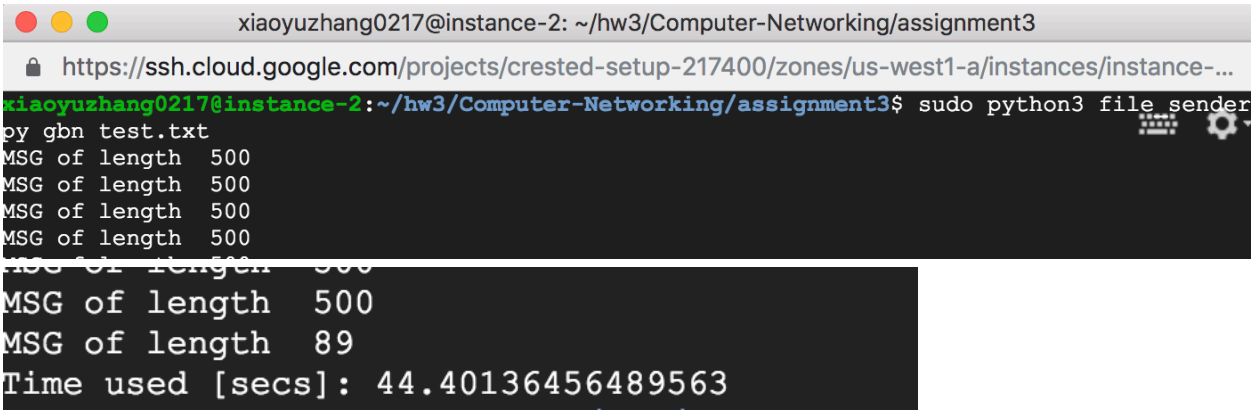
The error rate is 0.1 as in the first step so I can compare it.

(1) Stop-And-Wait

```
MSG of length  500
MSG of length  500
MSG of length  89
Time used [secs]: 41.121864557266235
```

The former time is 31.8s, and this time is 41.1s.

(2) Go-Back-N:



The former time is 30.98s, and this time is 44.4s.

Conclusion: The higher RTT is, the longer transmission time it takes.

c) Write a few simple sentences to describe your conclusion of the comparison

The higher error rate is, the longer transmission time it takes.

The higher RTT is, the longer transmission time it takes.