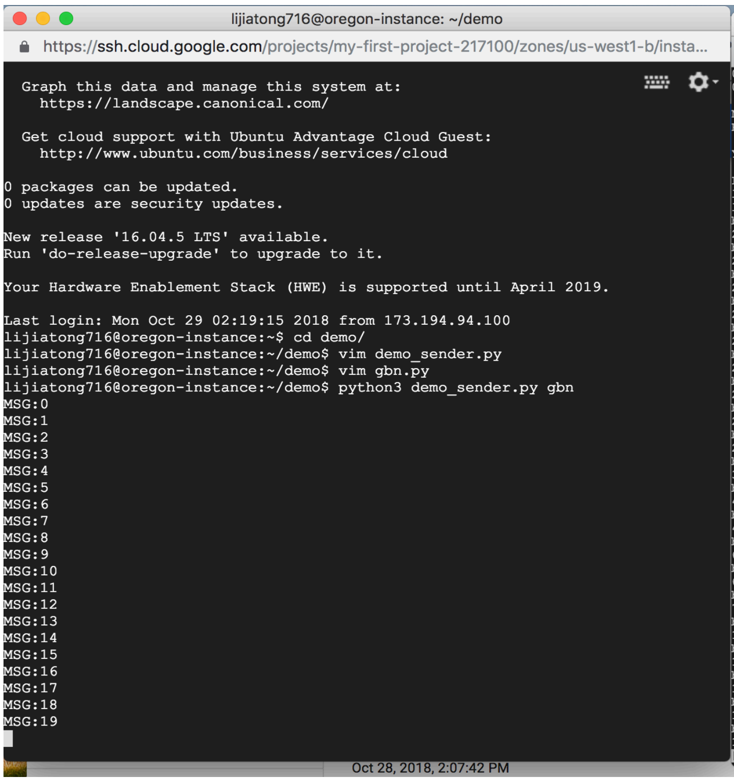
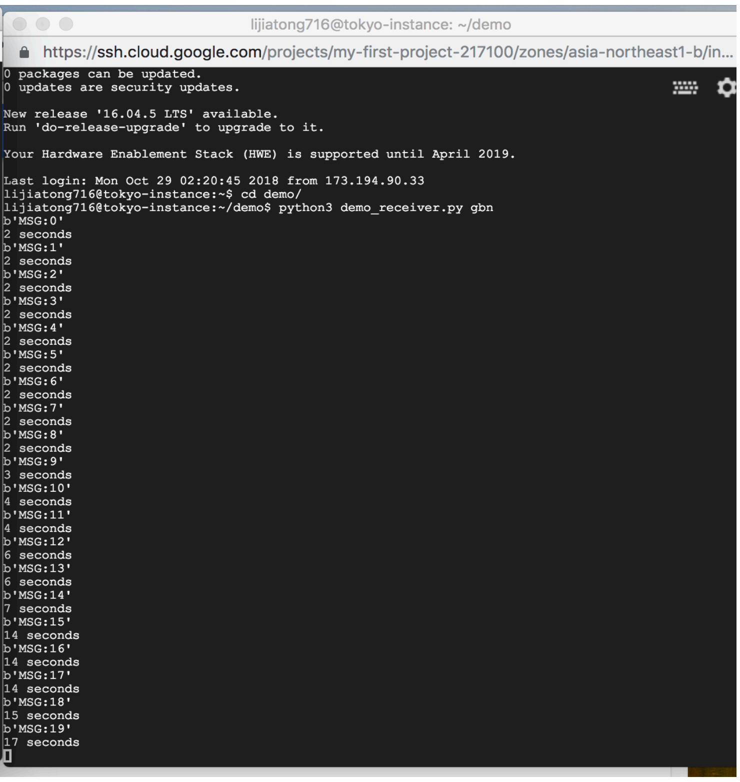


lijiatong716@oregon-instance: ~/demo

https://ssh.cloud.google.com/projects/my-first-project-217100/zones/us-west1-b/insta...

```
  Graph this data and manage this system at:
    https://landscape.canonical.com/

  Get cloud support with Ubuntu Advantage Cloud Guest:
    http://www.ubuntu.com/business/services/cloud

0 packages can be updated.
0 updates are security updates.

New release '16.04.5 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Your Hardware Enablement Stack (HWE) is supported until April 2019.

Last login: Mon Oct 29 02:19:15 2018 from 173.194.94.100
lijiatong716@oregon-instance:~$ cd demo/
lijiatong716@oregon-instance:~/demo$ vim demo_sender.py
lijiatong716@oregon-instance:~/demo$ vim gbn.py
lijiatong716@oregon-instance:~/demo$ python3 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
```

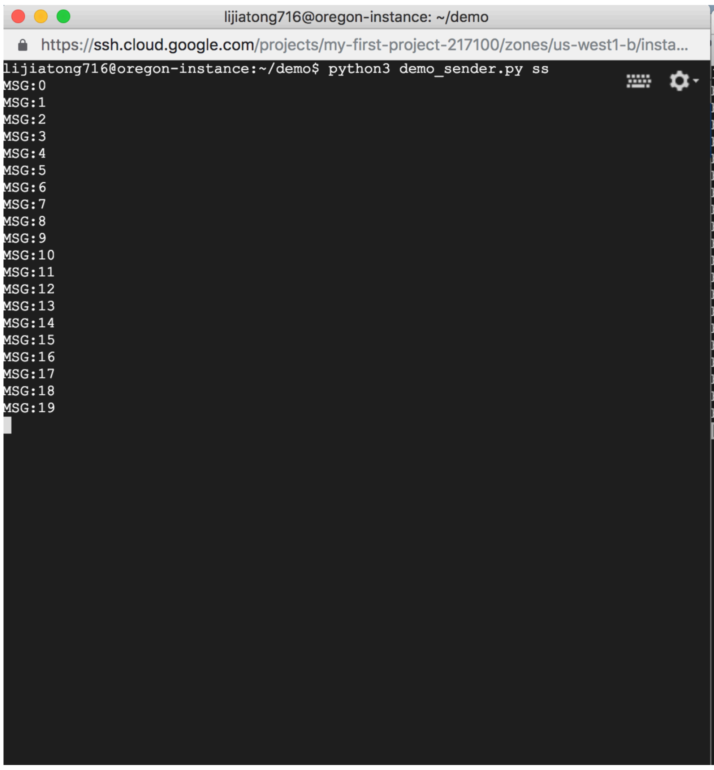
Oct 28, 2018, 2:07:42 PM



lijiatong716@tokyo-instance: ~/demo

https://ssh.cloud.google.com/projects/my-first-project-217100/zones/asia-northeast1-b/in...

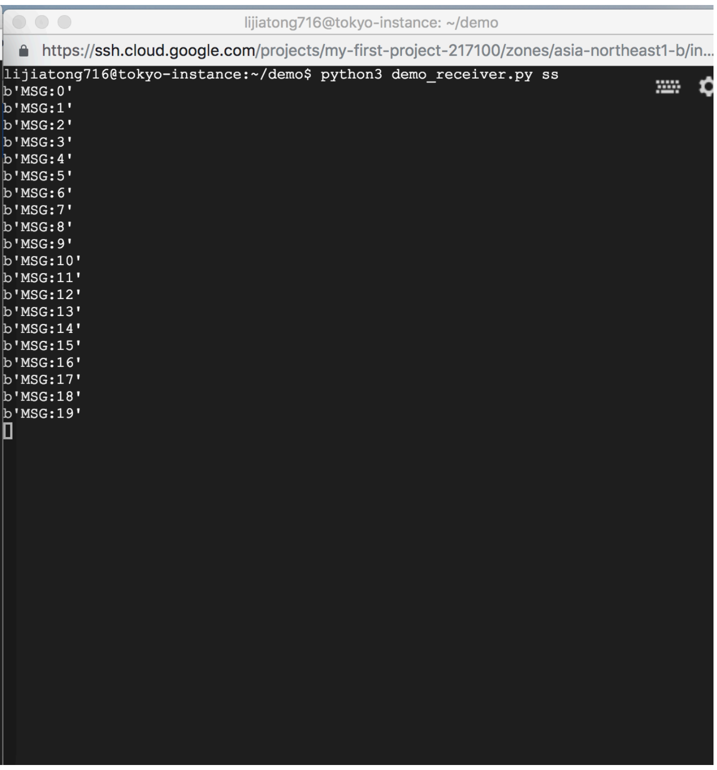
```
0 packages can be updated.
0 updates are security updates.

New release '16.04.5 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Your Hardware Enablement Stack (HWE) is supported until April 2019.

Last login: Mon Oct 29 02:20:45 2018 from 173.194.90.33
lijiatong716@tokyo-instance:~$ cd demo/
lijiatong716@tokyo-instance:~/demo$ python3 demo_receiver.py gbn
b'MSG:0'
2 seconds
b'MSG:1'
2 seconds
b'MSG:2'
2 seconds
b'MSG:3'
2 seconds
b'MSG:4'
2 seconds
b'MSG:5'
2 seconds
b'MSG:6'
2 seconds
b'MSG:7'
2 seconds
b'MSG:8'
2 seconds
b'MSG:9'
3 seconds
b'MSG:10'
4 seconds
b'MSG:11'
4 seconds
b'MSG:12'
6 seconds
b'MSG:13'
6 seconds
b'MSG:14'
7 seconds
b'MSG:15'
14 seconds
b'MSG:16'
14 seconds
b'MSG:17'
14 seconds
b'MSG:18'
15 seconds
b'MSG:19'
17 seconds
```



lijiatong716@oregon-instance: ~/demo

https://ssh.cloud.google.com/projects/my-first-project-217100/zones/us-west1-b/insta...

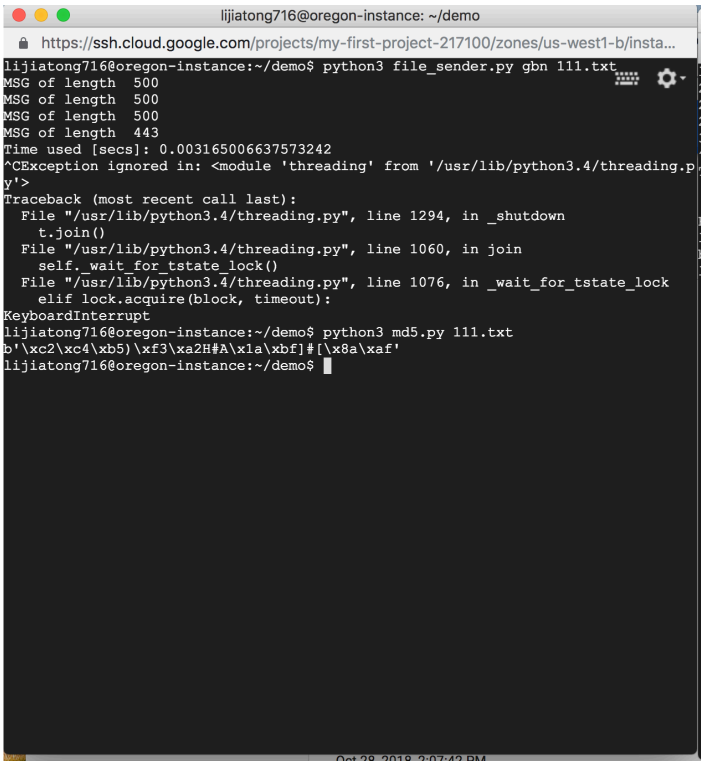
```
lijiatong716@oregon-instance:~/demo$ python3 demo_sender.py ss
MSG:0
MSG:1
MSG:2
MSG:3
MSG:4
MSG:5
MSG:6
MSG:7
MSG:8
MSG:9
MSG:10
MSG:11
MSG:12
MSG:13
MSG:14
MSG:15
MSG:16
MSG:17
MSG:18
MSG:19
```



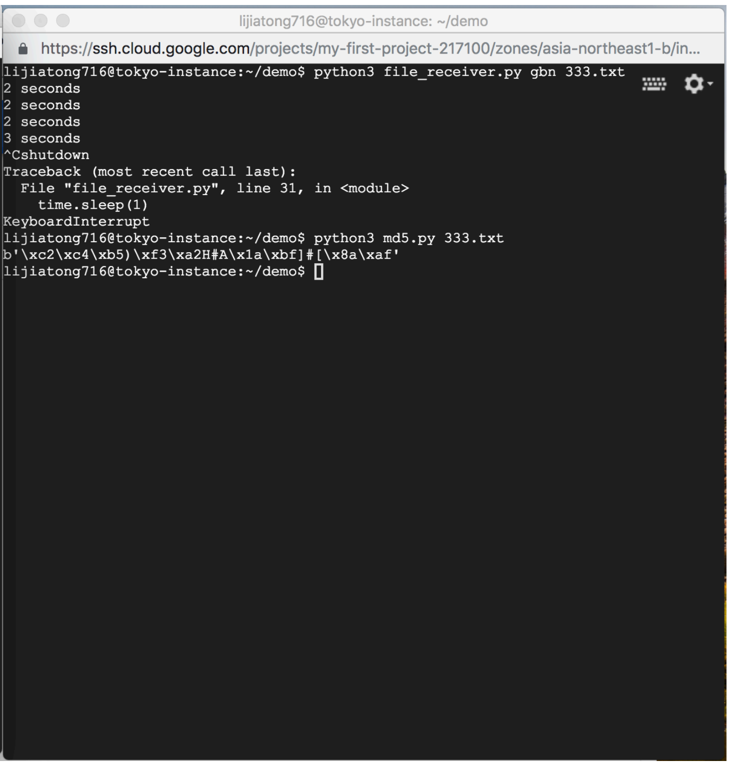
lijiatong716@tokyo-instance: ~/demo

https://ssh.cloud.google.com/projects/my-first-project-217100/zones/asia-northeast1-b/in...

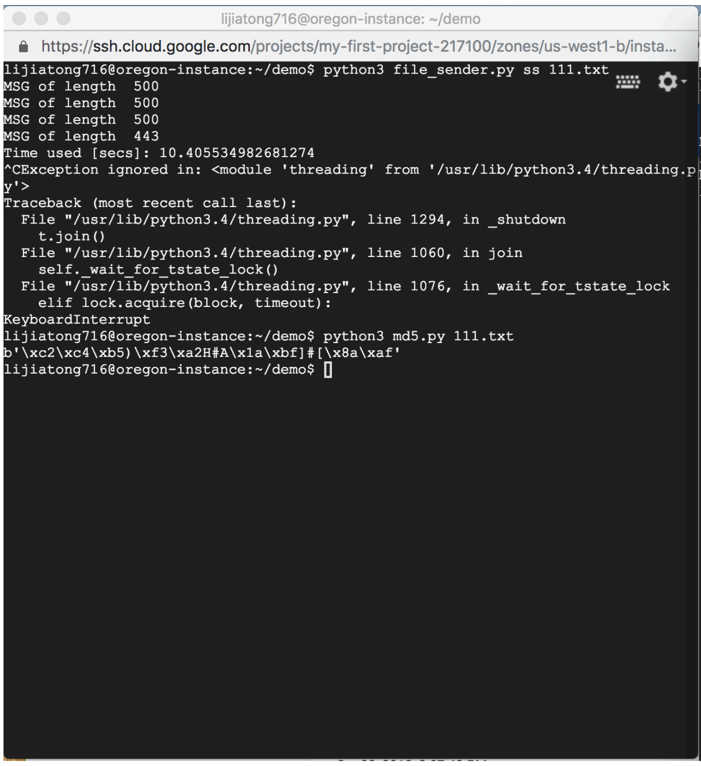
```
lijiatong716@tokyo-instance:~/demo$ python3 demo_receiver.py ss
b'MSG:0'
b'MSG:1'
b'MSG:2'
b'MSG:3'
b'MSG:4'
b'MSG:5'
b'MSG:6'
b'MSG:7'
b'MSG:8'
b'MSG:9'
b'MSG:10'
b'MSG:11'
b'MSG:12'
b'MSG:13'
b'MSG:14'
b'MSG:15'
b'MSG:16'
b'MSG:17'
b'MSG:18'
b'MSG:19'
```


lijiatong716@oregon-instance: ~/demo
https://ssh.cloud.google.com/projects/my-first-project-217100/zones/us-west1-b/insta...

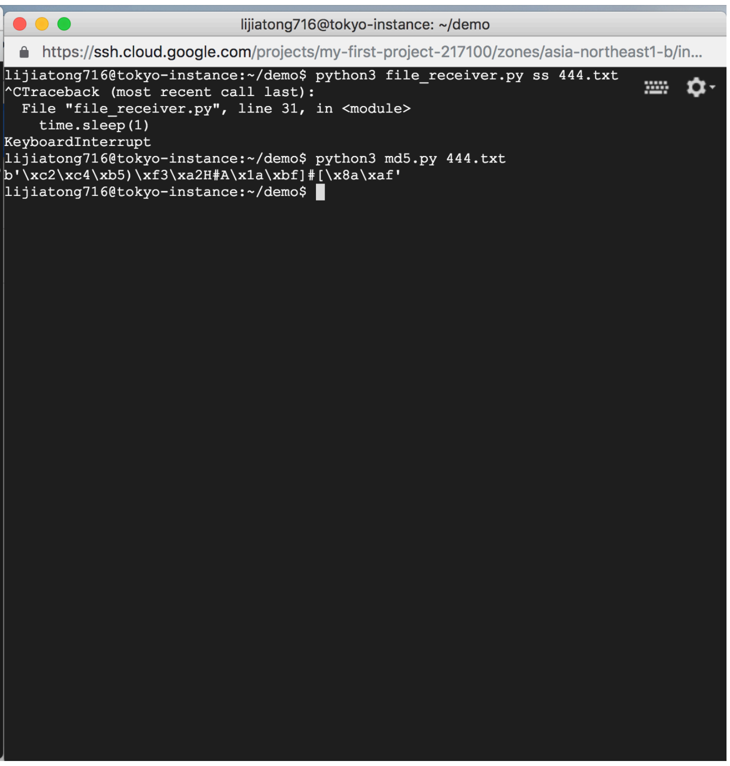
```
lijiatong716@oregon-instance:~/demo$ python3 file_sender.py gbn 111.txt
MSG of length  500
MSG of length  500
MSG of length  500
MSG of length  443
Time used [secs]: 0.003165006637573242
^CException ignored in: <module 'threading' from '/usr/lib/python3.4/threading.p
y'>
Traceback (most recent call last):
  File "/usr/lib/python3.4/threading.py", line 1294, in _shutdown
    t.join()
  File "/usr/lib/python3.4/threading.py", line 1060, in join
    self._wait_for_tstate_lock()
  File "/usr/lib/python3.4/threading.py", line 1076, in _wait_for_tstate_lock
    elif lock.acquire(block, timeout):
KeyboardInterrupt
lijiatong716@oregon-instance:~/demo$ python3 md5.py 111.txt
b'\xc2\xc4\xb5)\xf3\xa2H#A\x1a\xbf]#[\x8a\xaf'
lijiatong716@oregon-instance:~/demo$ 
```


lijiatong716@tokyo-instance: ~/demo
https://ssh.cloud.google.com/projects/my-first-project-217100/zones/asia-northeast1-b/in...

```
lijiatong716@tokyo-instance:~/demo$ python3 file_receiver.py gbn 333.txt
2 seconds
2 seconds
2 seconds
3 seconds
^Cshutdown
Traceback (most recent call last):
  File "file_receiver.py", line 31, in <module>
    time.sleep(1)
KeyboardInterrupt
lijiatong716@tokyo-instance:~/demo$ python3 md5.py 333.txt
b'\xc2\xc4\xb5)\xf3\xa2H#A\x1a\xbf]#[\x8a\xaf'
lijiatong716@tokyo-instance:~/demo$ 
```


lijiatong716@oregon-instance: ~/demo
https://ssh.cloud.google.com/projects/my-first-project-217100/zones/us-west1-b/insta...

```
lijiatong716@oregon-instance:~/demo$ python3 file_sender.py ss 111.txt
MSG of length  500
MSG of length  500
MSG of length  500
MSG of length  443
Time used [secs]: 10.405534982681274
^CException ignored in: <module 'threading' from '/usr/lib/python3.4/threading.p
y'>
Traceback (most recent call last):
  File "/usr/lib/python3.4/threading.py", line 1294, in _shutdown
    t.join()
  File "/usr/lib/python3.4/threading.py", line 1060, in join
    self._wait_for_tstate_lock()
  File "/usr/lib/python3.4/threading.py", line 1076, in _wait_for_tstate_lock
    elif lock.acquire(block, timeout):
KeyboardInterrupt
lijiatong716@oregon-instance:~/demo$ python3 md5.py 111.txt
b'\xc2\xc4\xb5)\xf3\xa2H#A\x1a\xbf]#[\x8a\xaf'
lijiatong716@oregon-instance:~/demo$ 
```


lijiatong716@tokyo-instance: ~/demo
https://ssh.cloud.google.com/projects/my-first-project-217100/zones/asia-northeast1-b/in...

```
lijiatong716@tokyo-instance:~/demo$ python3 file_receiver.py ss 444.txt
^CTraceback (most recent call last):
  File "file_receiver.py", line 31, in <module>
    time.sleep(1)
KeyboardInterrupt
lijiatong716@tokyo-instance:~/demo$ python3 md5.py 444.txt
b'\xc2\xc4\xb5)\xf3\xa2H#A\x1a\xbf]#[\x8a\xaf'
lijiatong716@tokyo-instance:~/demo$ 
```

2.Description

GBN: implement based on the pipeline mechanism. Implement two main functions, "send" which is called by application and "handler" to be called by network layer when packet is ready. On the sender side, have base, nextSeqnum and a packet array in initiation. Only if nextSeqnum < base +

WINDOW_SIZE, I will send the new packet carrying with new data. And when the sender receives the acknowledgement and packet is not corrupted, I will update the base. On the receiver side, when it receives the data not corrupted, it will deliver the data to the application. Otherwise will send the same previous acknowledgement back to the sender. There is another time_out implemented, when time_out being triggerd, I will send all packets from base to nextSeqnum in packet array again.
SS: This is stop and wait protocol. The two function to be implemented are also “send” and “handler_rec”. However, this time, on the sender side, I will send the new data as long as I receive the right acknowledgement. In ss, I have an old packet to keep track the last packet, when time_out being triggered, I will send the old packet again.

3.Comparison

1) High rate error(0.5) : the performance when using gbn shows it takes about 13 seconds while ss takes about 12 seconds. Both performance are pretty close.
2) Low rate error(0.1): the performance when using gbn shows it takes about 3 seconds, while ss takes about 6 seconds. Therefore, gbn is much faster than ss under low rate error.
3) Short RTT(Oregon-California): both performance are close to 3 seconds. Therefore, gbn and ss are pretty similar.
4) Long RTT(Oregon-Tokyo): the performance when using gbn shows it takes about 5 seconds, while ss takes about 40 seconds. Therefore gbn is much faster than ss when long trip.