

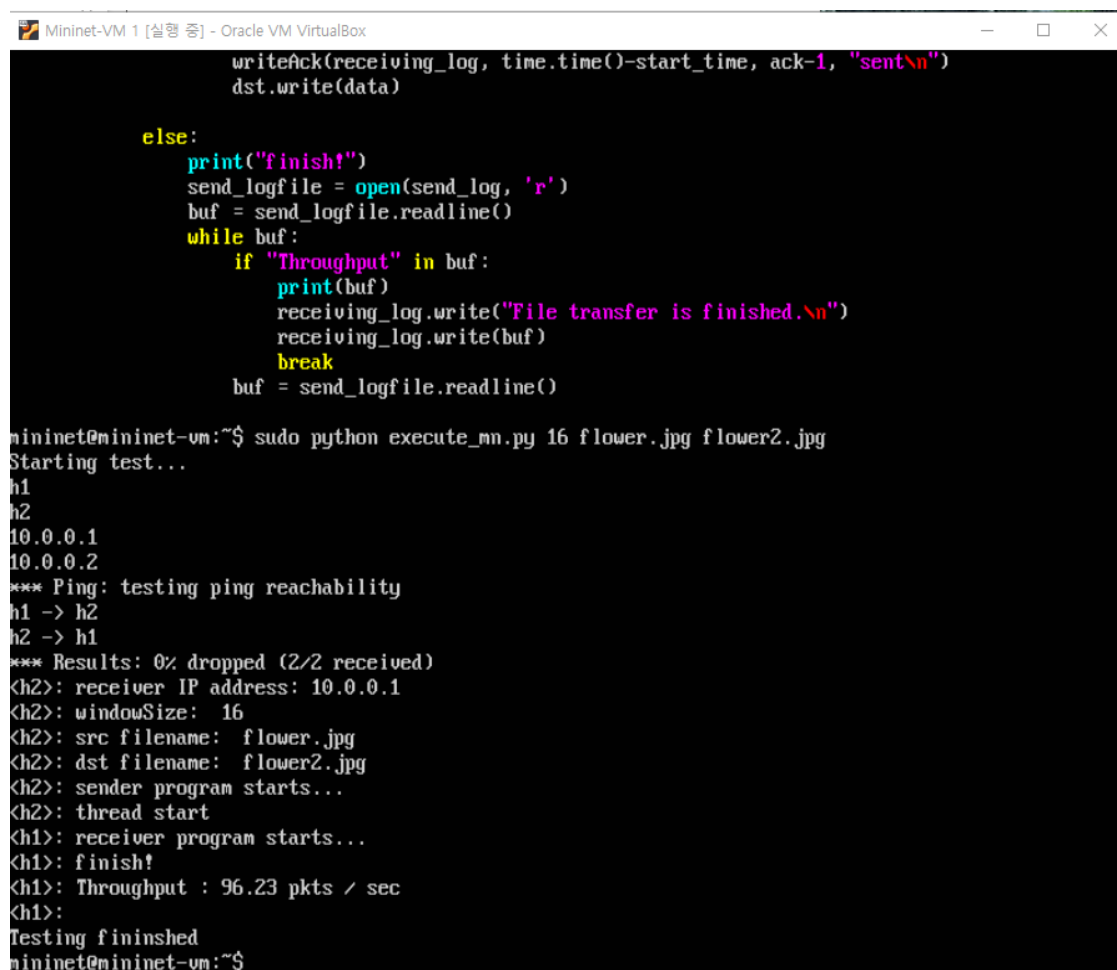
- 1) Present how to design your assignment such as data structures and algorithms.

패킷마다 RTT 값을 저장하기 위해 딕셔너리 자료구조를 사용하였습니다. 패킷 번호가 key가 되고, RTT 값이 value가 됩니다. 나중에 파일 전송이 끝나고 avgRTT를 구하기 위해 사용되었습니다. Sender는 전송해야 하는 파일을 1390바이트씩 읽고, 패킷 번호와 합쳐 하나의 패킷으로 receiver에게 전송하고 사용 중인 window 값을 1 증가시킵니다. Receiver는 패킷을 받아 패킷 번호를 파싱해서 순서에 맞는 패킷이 온 건지 확인하고 ack를 sender에게 보냅니다. 다시 sender의 ack를 받는 스레드에서 ack를 받고 window 값은 1 감소합니다. 패킷은 pack() 함수를 사용하여 구조체 형식으로 보냈습니다. 패킷 번호와 파일 데이터와 같이 보내기 위해 구조체를 사용했습니다.

- 2) Describe how to run and test your programs using snapshots

Mininet console 창에 >sudo python execute_mn.py 16 flower.jpg flower2.jpg

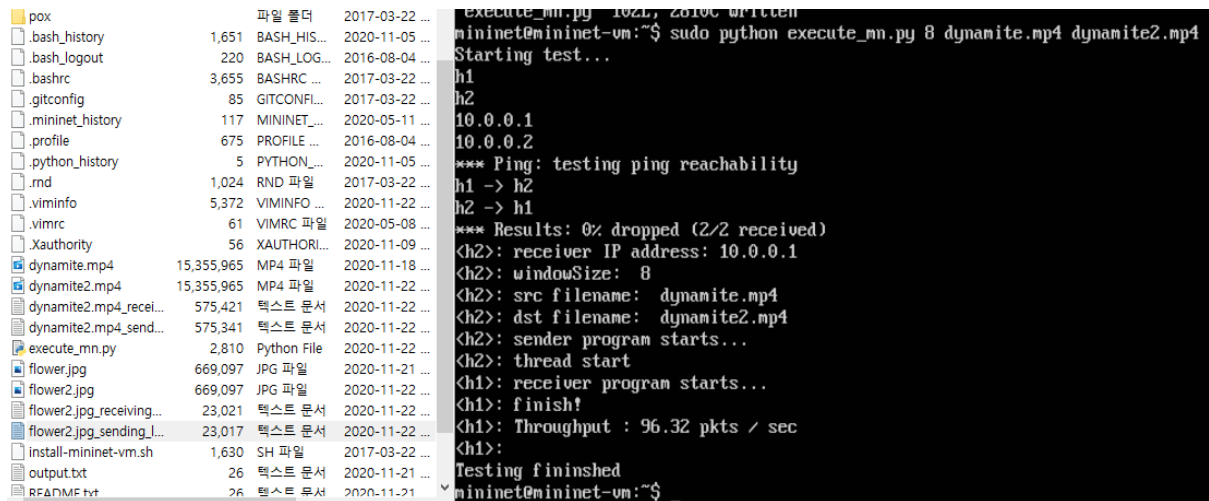
이런 형태의 커맨드를 입력함으로써 프로그램을 실행시킬 수 있습니다.



```
writeack(receiving_log, time.time()-start_time, ack-1, "sent\n")
dst.write(data)

else:
    print("finish!")
    send_logfile = open(send_log, 'r')
    buf = send_logfile.readline()
    while buf:
        if "Throughput" in buf:
            print(buf)
            receiving_log.write("File transfer is finished.\n")
            receiving_log.write(buf)
            break
        buf = send_logfile.readline()

mininet@mininet-vm:~$ sudo python execute_mn.py 16 flower.jpg flower2.jpg
Starting test...
h1
h2
10.0.0.1
10.0.0.2
*** Ping: testing ping reachability
h1 -> h2
h2 -> h1
*** Results: 0% dropped (2/2 received)
<h2>: receiver IP address: 10.0.0.1
<h2>: windowSize: 16
<h2>: src filename: flower.jpg
<h2>: dst filename: flower2.jpg
<h2>: sender program starts...
<h2>: thread start
<h1>: receiver program starts...
<h1>: finish!
<h1>: Throughput : 96.23 pkts / sec
<h1>:
Testing finished
mininet@mininet-vm:~$
```

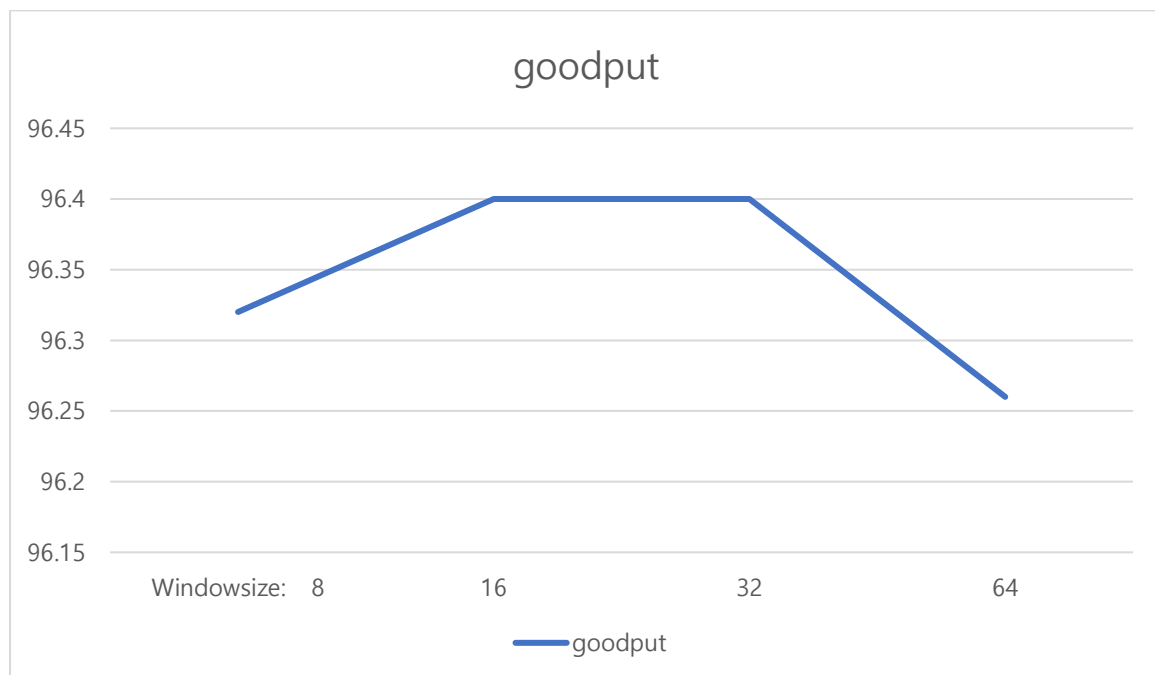


약 15MB의 동영상을 정상적으로 복사한 모습입니다.

3) Show experiment graphs for the experimentation results.

패킷 loss가 일어날 때의 재전송 기능 구현에 실패했습니다. 그래서 Window size만 다르게 했을 때의 goodput graph와 average RTT를 나타내었습니다.

Premise : Bandwidth 10Mbps, 25ms one way delay



Packet loss가 없기 때문에 window 사이즈에 상관없이 같은 속도로 전송이 되었습니다.