

# 2019년 데이터통신

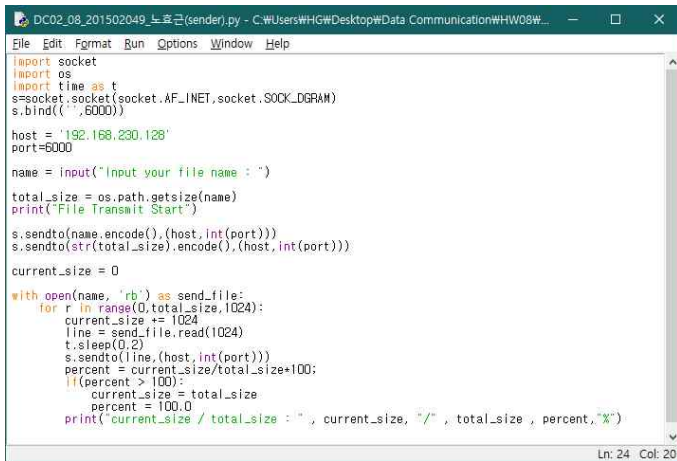
- HW 08 -

이름	노효근
학번	201502049
분반	02

## \* 과제 목표

- UDP로 파일, 전송 프로그램을 구현한다.
- 송,수신 후로 에러율을 구현한다.

## \* sender.py



```
import socket
import os
import time as t
s=socket.socket(socket.AF_INET,socket.SOCK_DGRAM)
s.bind(('',6000))

host = '192.168.230.128'
port=6000

name = input('Input your file name : ')
total_size = os.path.getsize(name)
print("File Transmit Start")


s.sendto(name.encode(),(host,int(port)))
s.sendto(str(total_size).encode(),(host,int(port)))

current_size = 0

with open(name, 'rb') as send_file:
    for r in range(0,total_size,1024):
        current_size += 1024
        line = send_file.read(1024)
        t.sleep(0.2)
        s.sendto(line,(host,int(port)))
        percent = current_size/total_size*100;
        if(percent > 100):
            current_size = total_size
            percent = 100.0
        print("current_size / total_size : ", current_size, "/", total_size, percent,"%")
```

: socket 라이브러리를 import한다. 이후, socket객체를 생성하여 통신에 필요한 배경을 만들어준다. 내가 보내줄 파일을 입력할 창을 만들고, 내가 보낼 파일의 총 size를 os.path.getsize() 메소드를 이용하여 구한다. 내가 보낼 장치의 ip주소와 port 번호를 이용하여 sendto() 메소드로 파일의 이름과 총 size를 전송한다. 이후 with open 메소드를 이용하여, 파일 읽기 전용으로 파일을 바이트로 연다. for문을 이용하여 0부터 총 size까지 1024byte씩 건너뛰면서 파일을 읽는다. 읽은 파일을 1024byte씩 reciver에게 보내도록 한다. 이때, sleep을 이용하여, 0.2초의 텀을 주고 보내도록 한다. 보낸 byte와 총 파일의 size를 이용하여, 보낸 %를 계산하여 출력하도록한다.

## \* reciver.py



```
import socket
import os

s=socket.socket(socket.AF_INET,socket.SOCK_DGRAM)
s.bind(('',6000))

name, senderaddr = s.recvfrom(2000)
total_size, _ = s.recvfrom(2000)

print("File recv start from ", senderaddr)
print("File Name : ", name.decode())
print("File Size : ", total_size.decode())

current_size = 0

with open(name, 'wb') as receive_file :
    for r in range(0,int(total_size),1024):
        current_size += 1024
        data, _ = s.recvfrom(1024)
        receive_file.write(data)
        percent = current_size/int(total_size)*100;
        if(percent > 100):
            #current_size = total_size.decode()
            #percent = 100.0
            print("current_size / total_size : ", current_size, "/", total_size.decode(), percent,"%")
```

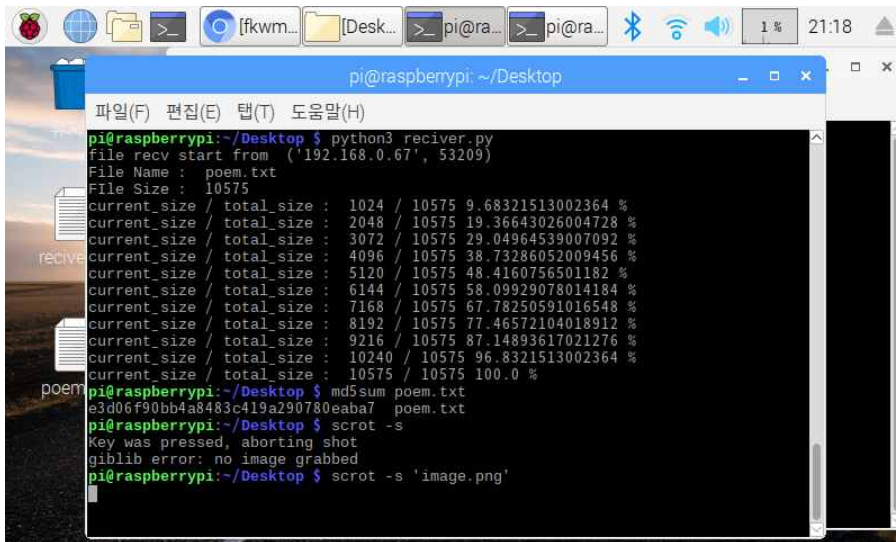
: socket 라이브러리를 import한다. 이후, socket객체를 생성하여 통신에 필요한 배경을 만들어준다. sender에서 보낸 파일이름과 ip주소, 파일의 총 size를 revfrom 메소드를 통해 받는다. 이후 받은 파일에 대한 정보를 출력한다. with open 메소드를 통해, 파일을 쓰기 전용으로 파일을 바이트로 만든다.revfrom으로 1024byte 씩 파일을 받아와 만든 파일에 쓰도록한다. 이때, 받은 byte와 총 파일의 size를 이용하여, 보낸 %를 계산하여 출력하도록한다.

## \* 실행 결과

: 파일 전송을 하는 sender - 개인 PC

```
u201502049@u201502049: ~/Desktop/DC
current_size / total_size : 6144 / 10575 58.09929078014184 %
current_size / total_size : 7168 / 10575 67.78250591016548 %
current_size / total_size : 8192 / 10575 77.46572104018912 %
current_size / total_size : 9216 / 10575 87.14893617021276 %
current_size / total_size : 10240 / 10575 96.8321513002364 %
current_size / total_size : 10575 / 10575 100.0 %
u201502049@u201502049:~/Desktop/DC$ python3 sender.py
Input your file name : poem.txt
File Transmit Start
current_size / total_size : 1024 / 10575 9.68321513002364 %
current_size / total_size : 2048 / 10575 19.36643026004728 %
current_size / total_size : 3072 / 10575 29.04964539007092 %
current_size / total_size : 4096 / 10575 38.73286052009456 %
current_size / total_size : 5120 / 10575 48.4160756501182 %
current_size / total_size : 6144 / 10575 58.09929078014184 %
current_size / total_size : 7168 / 10575 67.78250591016548 %
current_size / total_size : 8192 / 10575 77.46572104018912 %
current_size / total_size : 9216 / 10575 87.14893617021276 %
current_size / total_size : 10240 / 10575 96.8321513002364 %
current_size / total_size : 10575 / 10575 100.0 %
u201502049@u201502049:~/Desktop/DC$ md5sum poem.txt
f1989df7c98ff04863ed4afdb1eb5d1b poem.txt
u201502049@u201502049:~/Desktop/DC$
u201502049@u201502049:~/Desktop/DC$
```

: 파일을 받는 reciver - 지급받은 라즈베리파이



```
pi@raspberrypi: ~/Desktop
파일(F) 편집(E) 탭(T) 도움말(H)
pi@raspberrypi:~/Desktop $ python3 reciver.py
file rcv start from ('192.168.0.67', 53209)
File Name : poem.txt
File Size : 10575
current_size / total_size : 1024 / 10575 9.68321513002364 %
current_size / total_size : 2048 / 10575 19.36643026004728 %
current_size / total_size : 3072 / 10575 29.04964539007092 %
current_size / total_size : 4096 / 10575 38.73286052009456 %
current_size / total_size : 5120 / 10575 48.4160756501182 %
current_size / total_size : 6144 / 10575 58.09929078014184 %
current_size / total_size : 7168 / 10575 67.78250591016548 %
current_size / total_size : 8192 / 10575 77.46572104018912 %
current_size / total_size : 9216 / 10575 87.14893617021276 %
current_size / total_size : 10240 / 10575 96.8321513002364 %
current_size / total_size : 10575 / 10575 100.0 %
pi@raspberrypi:~/Desktop $ md5sum poem.txt
e3d06f90bb4a8483c419a290780eaba7 poem.txt
pi@raspberrypi:~/Desktop $ scrot -s
Key was pressed, aborting shot
glib error: no image grabbed
pi@raspberrypi:~/Desktop $ scrot -s 'image.png'
```

- sudo tc qdisc change dev ens33 root netem loss 0.8% 명령어를 통해 손실율을 80%로 설정하고 파일을 보냈다. (50%을 주었을 때, hash값 변화가 잘 안보였다.)
- md5sum을 통해 파일에 대한 hash값을 계산한 결과, 바뀐 것을 확인 할 수 있다.

\* GitHub ID : Nroot33