
TCP/IP 소켓 프로그래밍

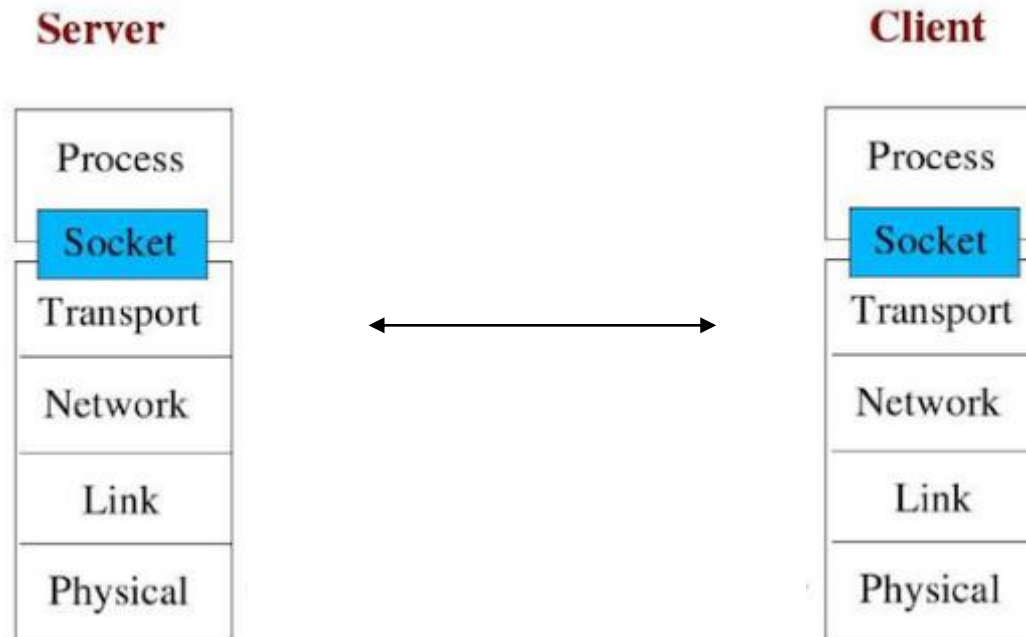
충북대학교
2020. 12. 03.



TCP/IP 소켓 프로그래밍

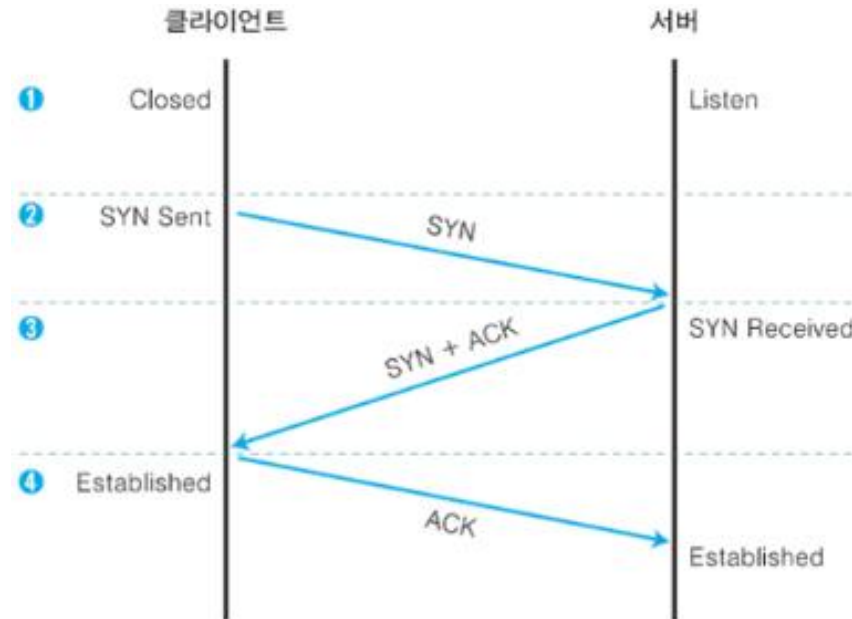
□ 소켓이란?

- 두 프로그램이 네트워크를 통해 서로 통신을 수행할 수 있도록 양쪽에 생성되는 링크의 단자



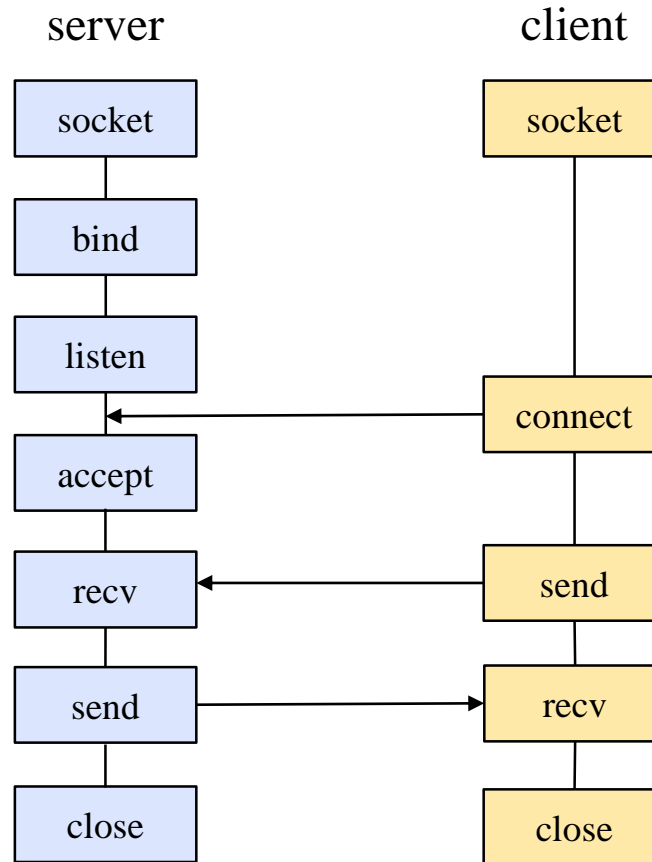
TCP/IP 소켓 프로그래밍

□ 3way handshaking



TCP/IP 소켓 프로그래밍

□ 소켓의 통신 과정



소스코드

□ tcp_server.py

```
1 import socket
2 import sys
3
4 PORT = 10000
5
6 server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
7 server_socket.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
8 server_socket.bind(('', PORT))
9 server_socket.listen(10)
10
11 client_socket, addr = server_socket.accept()
12
13 print('Connected by', addr)
14
15 while True:
16     data = client_socket.recv(1024)
17     if not data:
18         break
19     print('Received from', addr, data.decode())
20     client_socket.sendall(data)
21
22 client_socket.close()
23 server_socket.close()
```

소스코드

□ tcp_client.py

```
1 import socket
2 import sys
3
4 HOST = '127.0.0.1'
5 PORT = 9999
6
7 client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
8
9 client_socket.connect((HOST, PORT))
10
11 client_socket.sendall('test'.encode())
12
13 data = client_socket.recv(1024)
14 print('Received', repr(data.decode()))
15
16 client_socket.close()
```

실행

□ 터미널 실행



□ 서버 소스 실행

```
pi@raspberrypi:~/test $ sudo python tcp_server.py
```

□ 터미널 추가 실행



□ 클라이언트 소스 실행

```
pi@raspberrypi:~/test $ python tcp_client.py
```

□ 결과

- 서버

```
('Connected by', ('127.0.0.1', 59210))  
( 'Received from', ('127.0.0.1', 59210), u'test' )
```

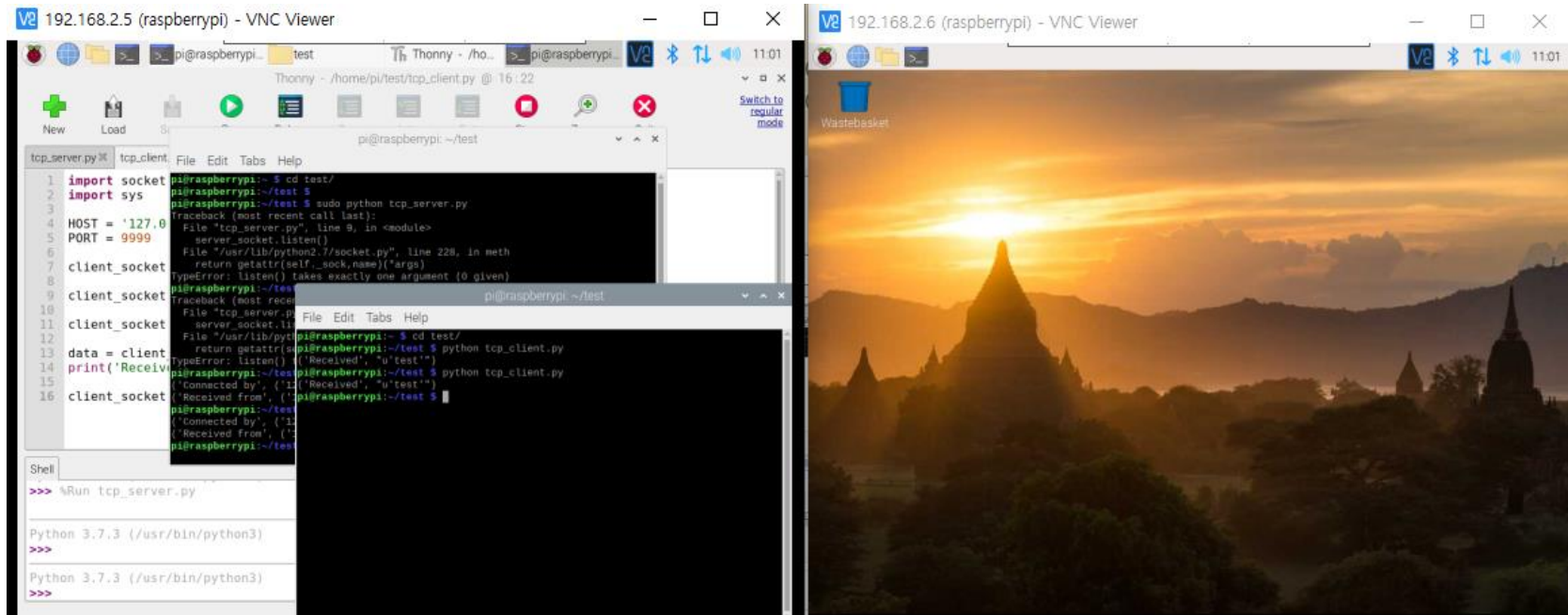
- 클라이언트

```
( 'Received', "u'test'" )
```

통신 실습

□ 좌측 라즈베리파이 실행

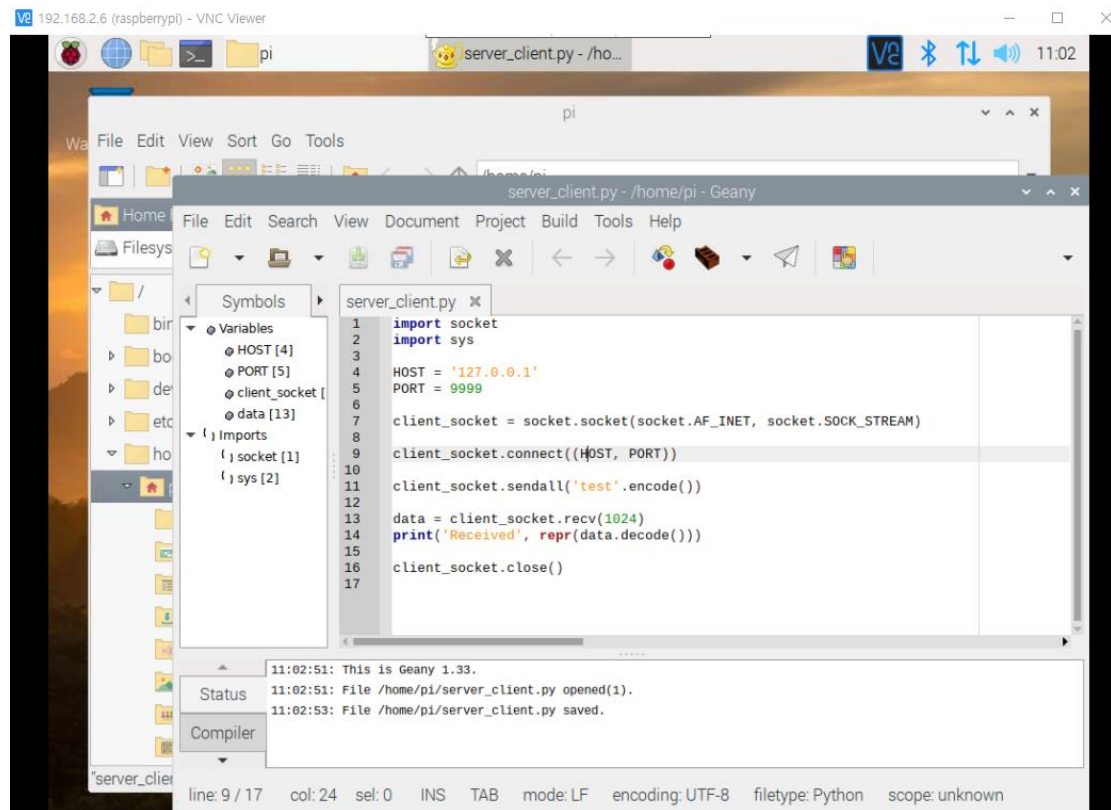
● VNC 접속



통신 실습

□ 클라이언트 소스 생성

- 좌측 라즈베리파이에 tcp_client.py 파일 생성



The screenshot displays a Raspberry Pi desktop environment accessed via VNC Viewer. The desktop background is the standard Raspberry Pi OS desktop. A file manager window shows the contents of the /home/pi directory. Overlaid on this is the Geany text editor, which is editing a file named server_client.py. The code in the file is a Python script for a TCP client. The script imports the socket and sys modules, defines a HOST variable as '127.0.0.1' and a PORT variable as 9999. It then creates a client_socket using socket.socket(socket.AF_INET, socket.SOCK_STREAM), connects it to the host and port, sends the string 'test' encoded in bytes, receives data (buffered to 1024 bytes), prints the received data, and finally closes the socket. The status bar at the bottom of the Geany window indicates the current line is 9 of 17, column 24, selection 0, and the file is a Python script.

```
1 import socket
2 import sys
3
4 HOST = '127.0.0.1'
5 PORT = 9999
6
7 client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
8
9 client_socket.connect((HOST, PORT))
10
11 client_socket.sendall('test'.encode())
12
13 data = client_socket.recv(1024)
14 print('Received', repr(data.decode()))
15
16 client_socket.close()
17
```

통신 실습

□ IP 주소 입력

- IP 주소 (127.0.0.1)에 상대 라즈베리파이 주소 입력

```
server_client.py
File Edit Search View Document Project Build
Symbols
Variables
  HOST [4]
  PORT [5]
  client_socket [
  data [12]
1 import socket
2 import sys
3
4 HOST = '192.168.2.5'
5 PORT = 9999
6
```

통신 실습

▣ 각각 서버, 클라이언트 실행

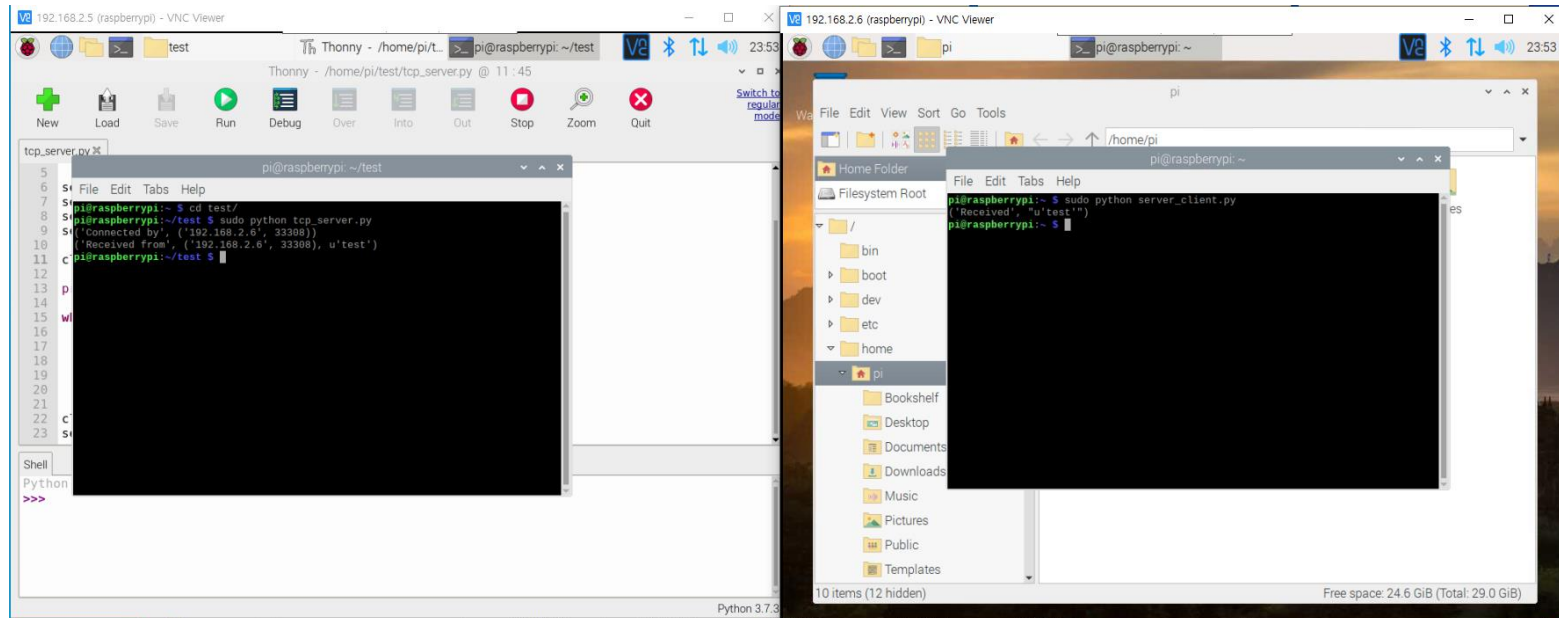
- 우측 라즈베리파이 (서버)

```
pi@raspberrypi:~/test $ sudo python tcp_server.py
```

- 좌측 라즈베리파이 (클라이언트)

```
pi@raspberrypi:~ $ sudo python server_client.py
```

- 결과



과제

□ 과제 1

- TCP/IP 에코 서버
 - 연결이 끊기지 않고 지속적으로 통신 (1초 간격)

□ 과제 2

- 서버에서 1 – 100까지 증가하는 데이터를 전송
- 클라이언트에서 수신한 데이터를 조도 센서로 출력 (% 단위)