Project #1 Report – Part 2: Proxyserver

Explanation of intial attempts

1. **Bind and Connect**
   At first I was pretty confused about what is bind and connect. I was using connect directly without using bind. Then I realize, how can a server/proxy know what's their own port. Then I realize what I need to do it bind. I looked this website for some tutorial: https://realpython.com/python-sockets/

2. **When to end server and proxy?**
   At first, I break the server and proxy after one transmission. But later, I think this is not too logical. So instead, I decided to let server and proxy always running with `While True` loop and use KeyboardInterrupt to shut it down. And client can run multiple times to send different message to proxy and server. (Before I was trying to do close proxy and server when type exit in input. But input statement can block the while statement so it doesn't work. So I go with KeyboardInterrupt in the end.)

3. **Send message multiple times**
   When we decided to keep server and proxy always running and client can run multiple times, there are following issues: *Only one usage of each socket address (protocol/network address/port) is normally permitted*.

I did some research and figure out it's a resue problem. I first try to allow resue with `client_socket.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)`. But this doesn't work and the issue still exit. Then I figure out this is due to the TIME_WAIT. So the recycled port are stuck in TIME_WAIT and might take OS a few mintues to recycle the port. But even with this, it still doesn't allow resue of exact same local-to-remote connections.

Then, I try to dynamically select port number for client as client doesn't need a specific port to make connection. Instead, I use `client_socket.bind((client_ip, 0))` and let the OS choose a port. This can allow the client to run multiple times and send message smoothly to proxy.

I used this article on bind to figure out this issue: https://hea-www.harvard.edu/~fine/Tech/addrinuse.html

4. **Proxy and server in same file?**
   Okay I was being stupid with this one. In first instruction, I thought we can only use two files, proxy_server and client instead of three. So I was trying to put server and proxy in the same file. And this doesn't really work obviously. But I realize we can use three file in the end.

Description of expected result

So this server works by adding a extra layer between the client and server. The proxy take the data from client and forward it to. Then it take the information from server and send back to client.
In this code, the client send 4 random generated letter to proxy. The proxy check for block list and decide whether forward to server. The server send back to proxy DONE if received the message. The proxy forward this message to client

For this project, details for each parts are as below:

Client

1. The client generate a random 4 letter words.
2. This words is packed with server_ip, server_port to a json format. This package is send to proxy.
3. The client establish the connection with proxy server and forward the message.
4. The client stop after it receive feedback.

Proxy

1. The proxy server bind and listen to it's port defined.
2. When the data from client arrives, it decompose the package and extract server_ip, server_port and message from json data.
3. Then, it check whether the ip port is on the black list. *Notes on blocklist*: since we are using localhost for all ip, the blocklist is implemented on the port number. Currently, port 7777 on ip '127.0.0.1' (localhost) is on the blocklist*

- If on blocklist, the proxy send message back to client.
- If not on blocklist, the proxy forward the message to server.

4. The proxy wait for the feedback from server.

Server

1. The server bind and listen to indicated port.
2. It wait for the connection from proxy and read the message. It send "DONE" if the message it received.

Issues faced or what didn’t work with your solution

I guess I am still a little confused about resue port thing. So when after the client done with transmission and terminated. The port it was using can't not be reused until few minutes after? For this code, before the client program terminated, it can send multiple information over. But once it terminated, it seems diffcult to reuse the same port right away.

Also, when closing, I think (at least on my machine), it must close the server first and then proxy to make it exit elegantly.

How are we supposed to run your code

1. Run following in terminal to start the *proxy server*

```
python proxy_servervictorialam_919626106_jinsiguo_918709406.py
```

The proxy should print '*Proxy Server listening on 127.0.0.1:6000*'

2. In another terminal, run following to start the *server*

```
python servervictorialam_919626106_jinsiguo_918709406.py
```

The server should shown '*Server listening on 127.0.0.1:8888*'

3. In another terminal, run following to start *client*

```
python clientvictorialam_919626106_jinsiguo_918709406.py
```

Then, it will prompt you *"Enter 'block' to test block mode, anything else to send normally:*"

- If you enter **block**, the program will try to send a blocked port '7777'. The proxy will feedback "Error, not allowed address"
- If **press enter or anything else**, the program will send a random generated 4 letter word to proxy and expecting feedback "DONE" from server(forward by proxy).

4. In the proxy server, it will print the json and extracted data. For example,

```
Proxy received: {"server_ip": "127.0.0.1", "server_port": 8888, "message": "tgmr"}
Server IP: 127.0.0.1
Server Port: 8888
Message: tgmr
response:  DONE
response send to Client
```

5. In server, it will print the data it received. For example,

```
Server listening on 127.0.0.1:8888
Server received: tgmr
```

6. In client, it will print whether it been transmit or blocked by server If server recieved the data

```
Enter 'block' to test block mode, anything else to send normally:
Random 4 letter string:  ebvr
Json data {'server_ip': '127.0.0.1', 'server_port': 8888, 'message': 'ebvr'}
Response from server(forwarded by Proxy):  DONE
```

If the port is on blocklist

```
Enter 'block' to test block mode, anything else to send normally: block
Random 4 letter string:  cykb
Json data {'server_ip': '127.0.0.1', 'server_port': 7777, 'message': 'cykb'}
Response from server(forwarded by Proxy):  Error, not allowed address
```

7. Run the client again and try different mode.

8. Use `ctrl+C` to stop the server and proxy.