

Nama : Alvin Zanua Putra

NRP : 5025231064

Kelas : Pemrograman Jaringan – D

TUGAS 2

Link Github File server_thread.py :

https://github.com/alvinzanuaputra/PROGJAR24/blob/main/assets/src/TUGAS2/server_thread.py

A. Port 45000 dengan transport TCP

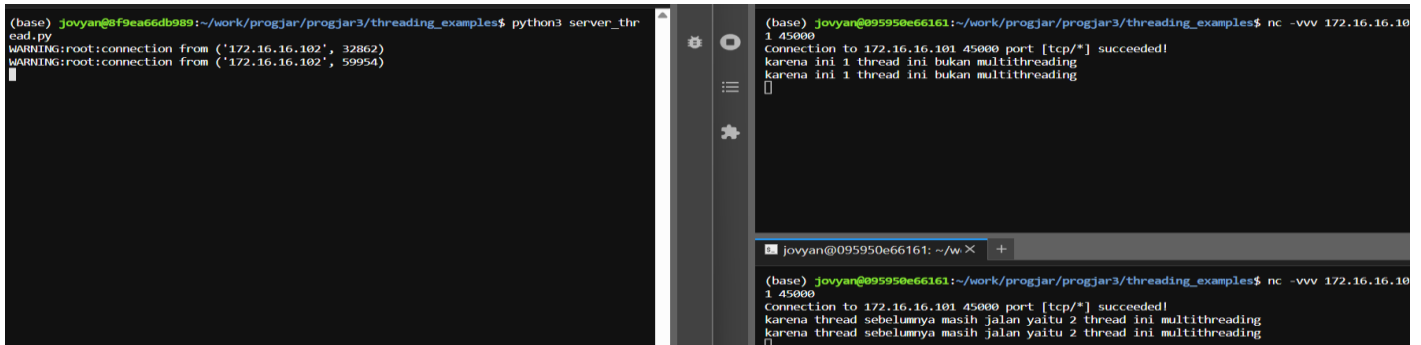
Jalankan mesin 1, kemudian ubah port yang ada di dalam file server_thread.py dari '8889' menjadi '45000' dalam self.my_socket.bind dalam class Server di bagian function def run (self) menjadi self.my_socket.bind(('0.0.0.0', 45000)). Angka '0.0.0.0' yang artinya server akan bind di semua interface server dan bisa diakses dari manapun tanpa mencantumkan port spesifik

```
class Server(threading.Thread):
    def __init__(self):
        self.the_clients = []
        self.my_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        threading.Thread.__init__(self)

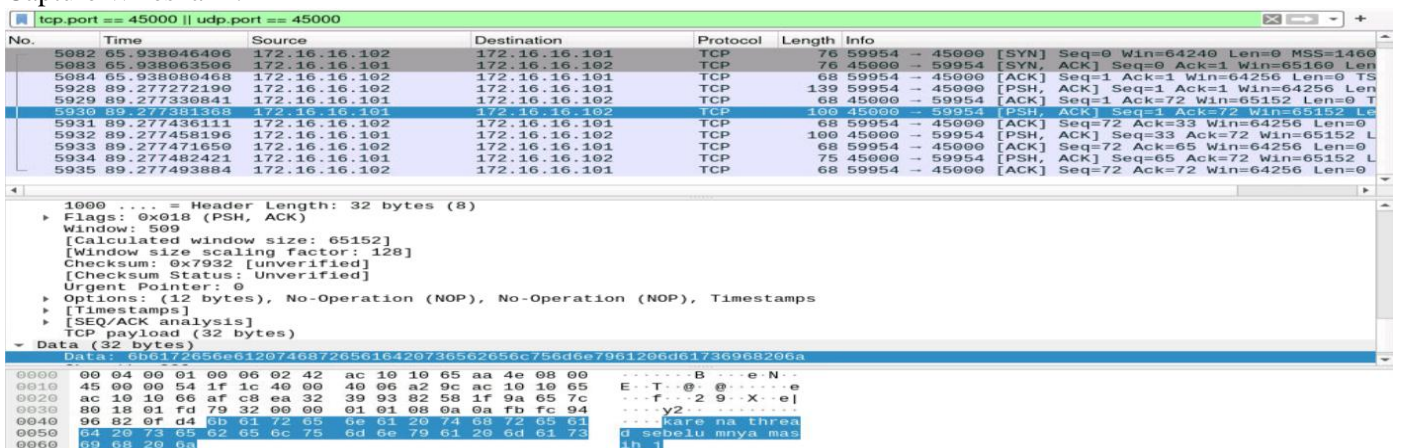
    def run(self):
        self.my_socket.bind(('0.0.0.0', 45000))
        self.my_socket.listen(1)
```

B. Server harus dapat melayani request yang concurrent, gunakan contoh multithreading pada file server_thread.py

menunjukkan proses komunikasi antara server dan client menggunakan socket di Python. Pada koneksi kedua, server mengirimkan pesan ke client berupa **"karena thread sebelumnya masih jalan yaitu 2 thread ini multithreading"**, sebagai contoh output program untuk menunjukkan bahwa server sedang menangani lebih dari satu koneksi secara bersamaan dengan multithreading.



Capture Wireshark :



Pada hasil capture Wireshark, terlihat adanya komunikasi TCP antara dua mesin, yaitu dari IP **172.16.16.102** (client) menuju **172.16.16.101** (server) pada port **45000**. Proses TCP handshake berjalan sesuai prosedur, diawali dengan

SYN, dilanjutkan SYN-ACK, lalu ACK, yang menandakan koneksi berhasil dibangun. Setelah itu, terdapat paket data yang dikirim dari server ke client dengan flag PSH, ACK, yang artinya server sedang mengirimkan data payload ke client sambil tetap menjaga koneksi terbuka. Hasil request terlihat Sebagian yaitu **“karena thread sebelumnya masih j”** menandakan bahwa multithreading berhasil dilakukan antar server.

C. Ketentuan request yang dilayani : Diawali dengan string **“TIME** dan diakhiri dengan karakter 13 dan karakter 10”. Setiap request dapat diakhiri dengan string **“QUIT”** yang diakhiri dengan karakter 13 dan 10

Disini saya menambahkan pada bagian def run (self) : logging agar server mencetak isi request **dalam bentuk byte** (untuk membuktikan `\r\n = \x0d\x0a` atau 13 10).

```
def run(self):
    while True:
        data = self.connection.recv(1024)
        if not data:
            break
        logging.warning(f"Raw data dari client: {data} ({list(data)})") # log byte
        message = data.decode()
        # Respons hanya jika diakhiri \r\n
        if message.endswith("\r\n"):
            self.connection.sendall(b"DITERIMA\r\n")
        else:
            self.connection.sendall(b"DITOLAK: Tidak diakhiri CRLF\r\n")
        if message.strip() == "QUIT":
            break
```

Apa pun isi pesannya, selama diakhiri `\r\n`, server membalas "DITERIMA\r\n". Jika tidak, balas "DITOLAK: Tidak diakhiri CRLF\r\n". Jika isi "QUIT", server keluar.

```
(base) jovyann@9343b9693c1c:~/work/progjar/progjar3/threading_examples$ python3 server_thread.py
WARNING:root:connection from ('172.16.16.102', 50778)
WARNING:root:Raw data dari client: b'TIME\r\n' ([84, 73, 77, 69, 13, 10])
WARNING:root:Raw data dari client: b'QUIT\r\n' ([81, 85, 73, 84, 13, 10])
[]
```

```
(base) jovyann@2c56b131a378:~/work/progjar/progjar3/threading_examples$ telnet 172.16.16.101 45000
Trying 172.16.16.101...
Connected to 172.16.16.101.
Escape character is '^'.
TIME
DITERIMA
QUIT
DITERIMA
```

tcp.port == 45000 udp.port == 45000						
No.	Time	Source	Destination	Protocol	Length	Info
5479	172.172769298	172.16.16.101	172.16.16.102	TCP	78	45000 → 50778 [PSH, ACK] Seq=7
5480	172.172842144	172.16.16.102	172.16.16.101	TCP	68	50778 → 45000 [ACK] Seq=7 Ack=
5655	176.911142762	172.16.16.102	172.16.16.101	TCP	74	50778 → 45000 [PSH, ACK] Seq=
5656	176.911405639	172.16.16.101	172.16.16.102	TCP	78	45000 → 50778 [PSH, ACK] Seq=
5657	176.911456309	172.16.16.102	172.16.16.101	TCP	68	50778 → 45000 [ACK] Seq=13 Ack=
5814	179.974185893	172.16.16.102	172.16.16.101	TCP	73	50778 → 45000 [PSH, ACK] Seq=
5823	180.016200260	172.16.16.101	172.16.16.102	TCP	68	45000 → 50778 [ACK] Seq=21 Ack=
6045	183.689854714	172.16.16.102	172.16.16.101	TCP	74	50778 → 45000 [PSH, ACK] Seq=
6046	183.689870755	172.16.16.101	172.16.16.102	TCP	68	45000 → 50778 [ACK] Seq=21 Ack=
6219	186.021000250	172.16.16.102	172.16.16.101	TCP	74	50778 → 45000 [PSH, ACK] Seq=
6220	186.021029769	172.16.16.101	172.16.16.102	TCP	68	45000 → 50778 [ACK] Seq=21 Ack=

[Time since reference or first frame: 183.689854714 seconds]

Frame Number: 6045

Frame Length: 74 bytes (592 bits)

Capture Length: 74 bytes (592 bits)

[Frame is marked: False]

[Frame is ignored: False]

[Protocols in frame: sll:ethertype:ip:tcp:data]

[Coloring Rule Name: TCP]

[Coloring Rule String: tcp]

Linux cooked capture v1

Internet Protocol Version 4, Src: 172.16.16.102, Dst: 172.16.16.101

Transmission Control Protocol, Src Port: 50778, Dst Port: 45000, Seq: 18, Ack: 21, Len: 6

Data (6 bytes)

Data: 54494d450d0a

0000	00 00 00 01 00 06 02 42	ac 10 10 66 60 1b 08 00B...f...
0010	45 10 00 3a f4 bb 40 00	40 06 cd 06 ac 10 10 66	E...@...@...f...
0020	ac 10 10 65 c6 5a af c8	ff 7e e7 c4 41 6e 3e eb	...e-Z...An>...
0030	80 18 01 f6 79 18 00 00	01 01 08 0a 96 df 74 0d	...y.....t...
0040	0b 59 52 73 54 49 4d 45	0d 0a	..YRS TIME ..

Server ini hanya menerima permintaan (request) dari client jika pesan dimulai dengan kata **“TIME** dan diakhiri dengan dua karakter khusus: `\r\n` yang artinya "carriage return" dan "line feed" (kode ASCII 13 dan 10). Ini adalah standar akhir baris dalam banyak protokol jaringan. Jika format tidak persis seperti itu, server akan menolak dengan pesan error. Client juga bisa mengakhiri sesi dengan mengirim pesan **“QUIT\r\n**”, yang memberitahu server untuk mengirim "Goodbye" lalu menutup koneksi. Dibuktikan dengan adanya format request **“TIME** diikuti . . yang berarti ada respons 2 digit yaitu `\r\n`.

- D. Server akan merespon dengan jam dengan ketentuan, Dalam bentuk string (UTF-8) Diawali dengan "JAM<spasi><jam>"<jam> berisikan info jam dalam format "hh:mm:ss" dan diakhiri dengan karakter 13 dan karakter 10

```
if not data:
    break

    logging.warning(f"Raw data dari client: {data} ({list(data)})") # Tambahan ini

    message = data.decode()

    if message.startswith("TIME") and message.endswith("\r\n"):

        now = datetime.now()

        response = now.strftime("JAM %d %m %y %H:%M:%S\r\n")

        self.connection.sendall(response.encode())

    elif message.startswith("QUIT") and message.endswith("\r\n"):

        self.connection.sendall(b"Goodbye\r\n")

        break

    else:

        self.connection.sendall(b"ERROR: Format tidak dikenali\r\n")
```

Server hanya menerima request dengan format `\r\n` (kode 13 dan 10). Salah satu karakter saja (`\n` atau `\r` saja) akan ditolak.

```
(base) jovyang@7236b8d02:~/work/progjar/progjar3/threading_examples$ python3 server_thread.py
WARNING:root:connection from ('172.16.16.101', 47546)
WARNING:root:Raw data dari client: b'TIME\r\n' ([84, 73, 77, 69, 13, 10])
WARNING:root:Raw data dari client: b'QUIT\r\n' ([81, 85, 73, 84, 13, 10])
```

```
(base) jovyang@cbf41dd42f94:~/work/progjar/progjar3/threading_examples$ telnet 172.16.16.101 45000
Trying 172.16.16.101...
Connected to 172.16.16.101.
Escape character is '^['.
TIME
JAM 23 05 25 12:28:29
QUIT
Goodbye
Connection closed by foreign host.
```

Pada Capture Wireshark :

	Source	Destination	Protocol	Length	Info
333791652	172.16.16.102	172.16.16.101	TCP	68	47546 → 45000 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=2529143...
943393496	172.16.16.102	172.16.16.101	TCP	74	47546 → 45000 [PSH, ACK] Seq=1 Ack=1 Win=64256 Len=6 TSval=25...
943441410	172.16.16.101	172.16.16.102	TCP	68	45000 → 47546 [ACK] Seq=1 Ack=7 Win=65280 Len=0 TSval=1883478...
944111467	172.16.16.101	172.16.16.102	TCP	91	45000 → 47546 [PSH, ACK] Seq=1 Ack=7 Win=65280 Len=23 TSval=1...
944196720	172.16.16.102	172.16.16.101	TCP	68	47546 → 45000 [ACK] Seq=7 Ack=24 Win=64256 Len=0 TSval=252916...
731554506	172.16.16.102	172.16.16.101	TCP	74	47546 → 45000 [PSH, ACK] Seq=7 Ack=24 Win=64256 Len=6 TSval=2...
732429954	172.16.16.101	172.16.16.102	TCP	77	45000 → 47546 [PSH, ACK] Seq=24 Ack=13 Win=65280 Len=9 TSval=...
732526430	172.16.16.102	172.16.16.101	TCP	68	47546 → 45000 [ACK] Seq=13 Ack=33 Win=64256 Len=0 TSval=25291...
732593026	172.16.16.101	172.16.16.102	TCP	68	45000 → 47546 [FIN, ACK] Seq=33 Ack=13 Win=65280 Len=0 TSval=...
732650804	172.16.16.102	172.16.16.101	TCP	68	47546 → 45000 [FIN, ACK] Seq=13 Ack=34 Win=64256 Len=0 TSval=...
732660420	172.16.16.101	172.16.16.102	TCP	68	45000 → 47546 [ACK] Seq=34 Ack=14 Win=65280 Len=0 TSval=18835...

Frame 8759: 91 bytes on wire (728 bits), 91 bytes captured (728 bits) on interface any, id 0	
Interface id: 0 (any)	
Encapsulation type: Linux cooked-mode capture v1 (25)	
Arrival Time: May 23, 2025 12:28:29.175786013 UTC	
[Time shift for this packet: 0.000000000 seconds]	
Epoch Time: 1748003309.175786013 seconds	
[Time delta from previous captured frame: 0.000670057 seconds]	
[Time delta from previous displayed frame: 0.000670057 seconds]	
[Time since reference or first frame: 103.944111467 seconds]	
Frame Number: 8759	
Frame Length: 91 bytes (728 bits)	
Capture Length: 91 bytes (728 bits)	
[Frame is marked: False]	
[Frame is ignored: False]	

0000	00 04 00 01 00 06 02 42	ac 10 10 65 65 62 08 00B.....eeb..
0010	45 00 00 4b 02 a5 40 00	40 06 bf 1c ac 10 10 65	E..K...@...@.....e
0020	ac 10 10 66 af c8 b9 ba	14 53 09 b2 bc d8 bf 87	...f...@...S.....
0030	80 18 01 fe 79 29 00 00	01 01 08 0a 0b 39 f5 bb	...y).....9.....
0040	96 c0 08 fb 4a 41 4d 20	32 33 20 30 35 20 32 35JAM 23 05 25
0050	20 31 32 3a 32 38 3a 32	39 0d 0a	12:28:2 9...

Server ini hanya menerima permintaan (request) dari client jika pesan dimulai dengan kata "TIME" dan diakhiri dengan dua karakter khusus: `\r\n` yang berarti *carriage return* dan *line feed* (kode ASCII 13 dan 10). Ini adalah standar penanda akhir baris dalam banyak protokol jaringan. Jika format tidak sesuai (misalnya hanya `\n`), server akan menolak permintaan dengan mengirimkan pesan error. Selain itu, client dapat mengakhiri sesi dengan mengirimkan "QUIT\r\n", yang akan dibalas oleh server dengan "Goodbye" dan menutup koneksi.

Sebagai respon terhadap permintaan "TIME\r\n", server mengirimkan waktu saat ini dalam format UTF-8, dimulai dengan string "JAM " diikuti oleh waktu dalam format "hh:mm:ss", dan diakhiri dengan `\r\n`. Format waktu ini dihasilkan menggunakan modul `datetime`, khususnya dengan `datetime.now().strftime("%H:%M:%S")`. Contoh balasan dari server adalah: **JAM 12:28:29\r\n**