

PEMROGRAMAN JARINGAN
Tugas Praktikum Multi-threaded Server



Class E

05111840000127 - Salsabila Harlen

Lecturer :

Royyana M. Ijtihadie

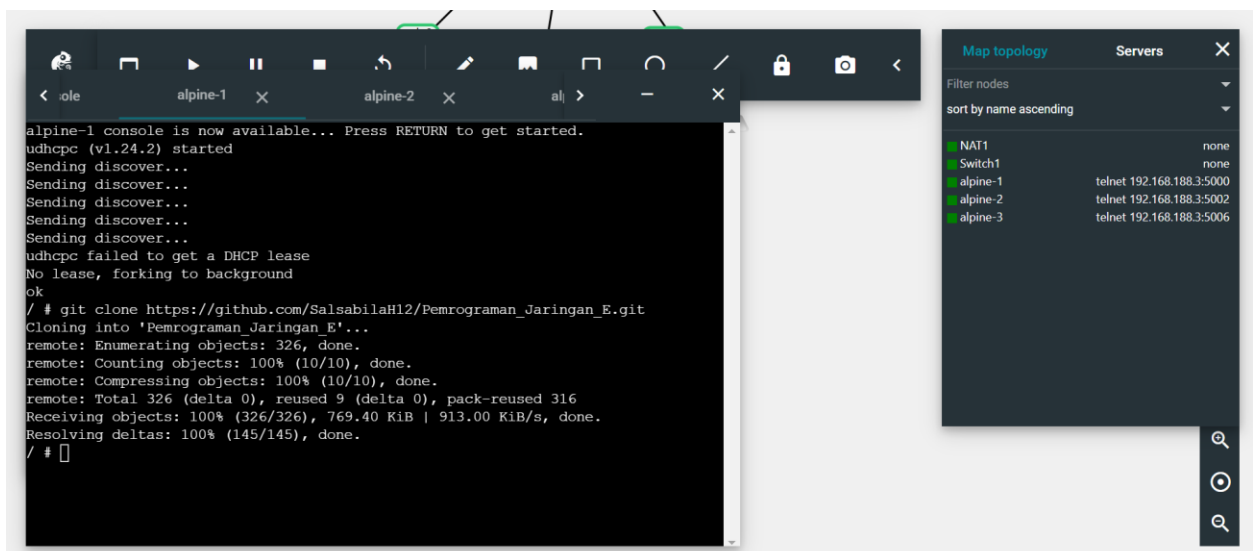
Informatics Department
Faculty of ELECTICS
Institut Teknologi Sepuluh Nopember (ITS) Surabaya
2021

Semua tugas berikut ini HARUS dijalankan di SIMULATOR

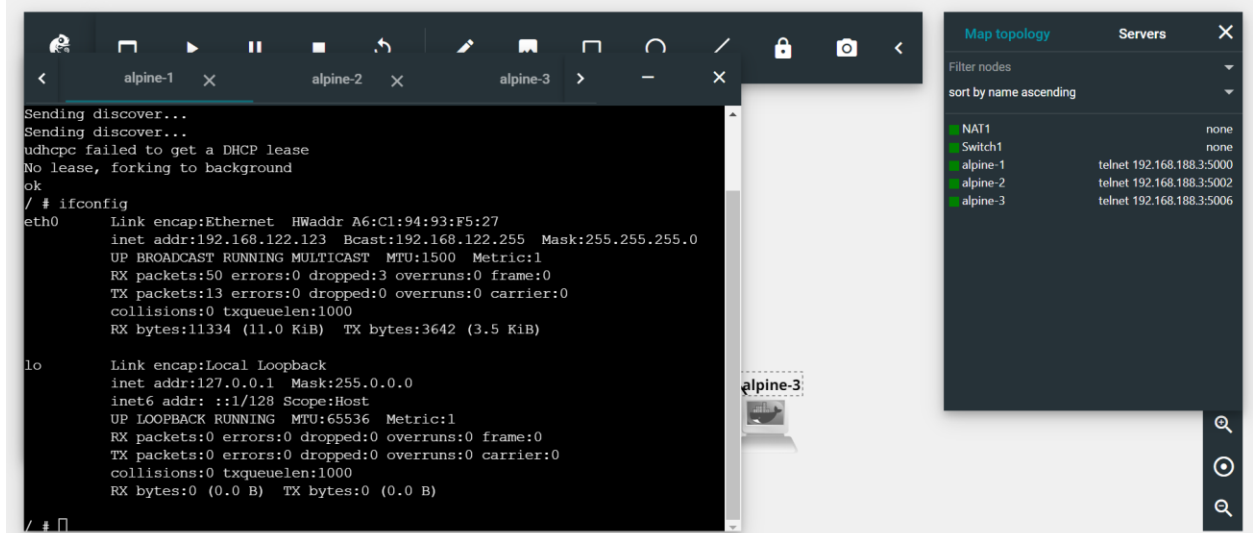
1. Jalankan program server seperti dalam pembahasan
2. Buatlah program client yang dapat melakukan 100 request get pada dalam satu saat untuk operasi get file "pokijan.jpg"
3. Capture dan submitlah poin 1 dan 2 dalam satu dokumen pdf. berikan DESKRIPSI dan PENJELASAN

Jawab :

1. Clone Repository Github



2. Ifconfig Alpine 1



3. Buka dan edit file_server.py dang anti ip address sesuai ip address alpine 1

The image displays two screenshots of a network simulation environment. The top screenshot shows the initial state of the `file_server.py` script, which is configured to listen on IP address `192.168.122.123`. The bottom screenshot shows the script after modification, where the IP address has been changed to `192.168.122.123` (Note: the original image shows the same IP, but the instruction implies a change to match alpine-1's IP). The right sidebar in both screenshots shows the network topology, listing nodes and their associated IP addresses.

Top Screenshot: Initial `file_server.py`

```
GNU nano 4.6 file_server.py Modified
if data:
    d = data.decode()
    hasil = fp.proces_string(d)
    hasil=hasil+"\r\n\r\n"
    self.connection.sendall(hasil.encode())
else:
    break
self.connection.close()

class Server(threading.Thread):
    def __init__(self,ipaddress='192.168.122.123',port=8889):[]
    self.ipinfo=(ipaddress,port)
    self.the_clients = []
    self.my_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    self.my_socket.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
    threading.Thread.__init__(self)

    def run(self):
        logging.warning(f"server berjalan di ip address {self.ipinfo}")

        self.my_socket.bind(self.ipinfo)
        self.my_socket.listen(1)
        while True:
            self.connection, self.client_address = self.my_socket.accept()
            logging.warning(f"connection from {self.client_address}")

            clt = ProcessTheClient(self.connection, self.client_address)
            clt.start()
            self.the_clients.append(clt)

def main():
    svr = Server(ipaddress='192.168.122.123',port=8889)
    svr.start()

if __name__ == "__main__":
    main()
```

Bottom Screenshot: Modified `file_server.py`

```
GNU nano 4.6 file_server.py Modified
self.my_socket.bind(self.ipinfo)
self.my_socket.listen(1)
while True:
    self.connection, self.client_address = self.my_socket.accept()
    logging.warning(f"connection from {self.client_address}")

    clt = ProcessTheClient(self.connection, self.client_address)
    clt.start()
    self.the_clients.append(clt)

def main():
    svr = Server(ipaddress='192.168.122.123',port=8889)
    svr.start()

if __name__ == "__main__":
    main()
```

Network Topology (Servers):

Node	IP Address
NAT1	none
Switch1	none
alpine-1	telnet 192.168.188.3:5000
alpine-2	telnet 192.168.188.3:5002
alpine-3	telnet 192.168.188.3:5006

4. Jalankan file_server.py

The terminal window shows the following output:

```
lo      Link encap:Local Loopback
        inet addr:127.0.0.1  Mask:255.0.0.0
        inet6 addr: ::1/128 Scope:Host
        UP LOOPBACK RUNNING  MTU:65536  Metric:1
        RX packets:0 errors:0 dropped:0 overruns:0 frame:0
        TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:1000
        RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

/ # git clone https://github.com/rm77/progjar.git
Cloning into 'progjar'...
remote: Enumerating objects: 332, done.
remote: Counting objects: 100% (16/16), done.
remote: Compressing objects: 100% (14/14), done.
remote: Total 332 (delta 3), reused 14 (delta 2), pack-reused 316
Receiving objects: 100% (332/332), 770.38 KiB | 461.00 KiB/s, done.
Resolving deltas: 100% (148/148), done.
/ # cd progjar
/progjar # cd progjar4a
/progjar/progjar4a # nano fileserver.py
/progjar/progjar4a # nano file_server.py
/progjar/progjar4a # python3 file_server.py
WARNING:root:server berjalan di ip address ('192.168.122.123', 8889)
```

The network topology map shows the following servers:

Node	IP Address
NAT1	none
Switch1	none
alpine-1	telnet 192.168.188.3:5000
alpine-2	telnet 192.168.188.3:5002
alpine-3	telnet 192.168.188.3:5006

5. Membuat Program Client 100 Request

- Buka dan edit file_interface.py pada alpine 2. Ubah directori nya sesuai path saat meng clone

The terminal window shows the following code in file_interface.py:

```
GNU nano 4.6      file_interface.py      Modified
import os
import json
import base64
from glob import glob

class FileInterface:
    def __init__(self):
        os.chdir('/progjar/progjar4a')

    def list(self):
        try:
            filelist = glob('*.**')
            return dict(status='OK',data=filelist)
        except Exception as e:
            return dict(status='ERROR',data=str(e))

    def get(self,filename=''):
        if(filename==''):
            return None
```

The network topology map shows the following servers:

Node	IP Address
NAT1	none
Switch1	none
alpine-1	telnet 192.168.188.3:5000
alpine-2	telnet 192.168.188.3:5002
alpine-3	telnet 192.168.188.3:5006

- Ubah IP address sesuai ip server

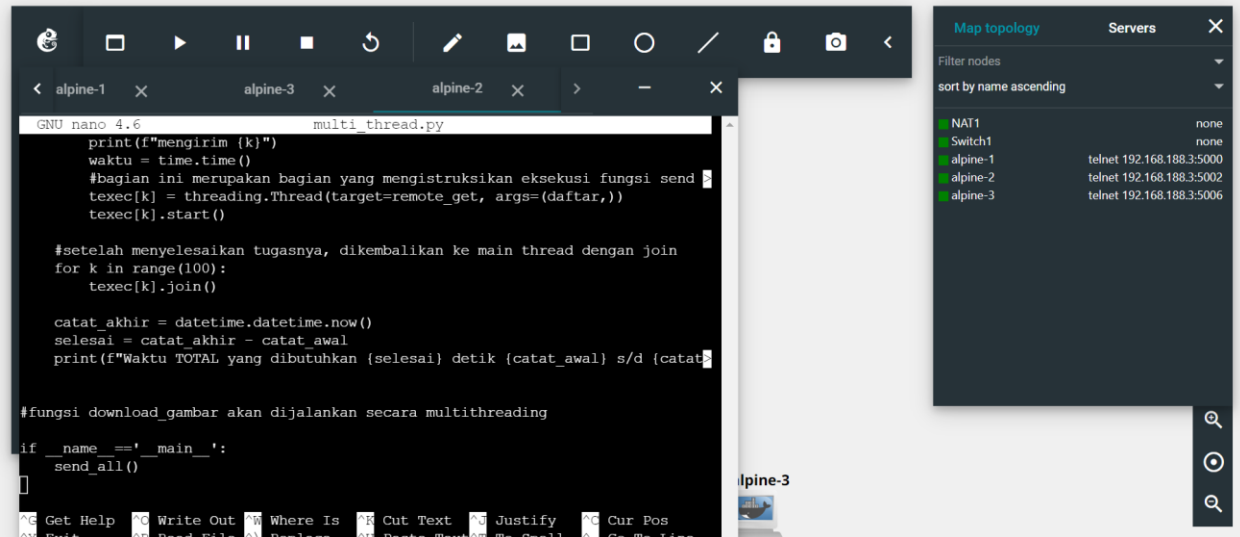
The screenshot shows a network simulator interface. On the left, a terminal window displays the contents of `file_client_cli.py` using GNU nano 4.6. The script defines a `send_command` function that sends a command to a server and a `remote_get` function that retrieves a file from a server. The main function sets the `server_address` to `('192.168.122.123', 8889)` and calls `remote_get('donalbebek.jpg')`. On the right, a 'Servers' panel lists the IP addresses for the servers in the topology:

Node	IP Address
NAT1	none
Switch1	none
alpine-1	telnet 192.168.188.3:5000
alpine-2	telnet 192.168.188.3:5002
alpine-3	telnet 192.168.188.3:5006

- Import (copy dan buat file baru `udp_multi_threaded.py` pada progjar4a) dan modifikasi program dari progjar 3a yaitu `multi_thread.py`.

The screenshot shows the same network simulator interface. The terminal window now displays the contents of `multi_thread.py`. The script imports `remote_get` from `file_client_cli` and uses it to send a list of commands to a server. The `send_all` function sends a list of commands to a server and waits for the results. The main function sets the `server_address` to `('192.168.122.123', 8889)` and calls `send_all` with a list of commands. On the right, the 'Servers' panel is the same as in the previous screenshot:

Node	IP Address
NAT1	none
Switch1	none
alpine-1	telnet 192.168.188.3:5000
alpine-2	telnet 192.168.188.3:5002
alpine-3	telnet 192.168.188.3:5006



- Isi dari file udp_multi_thread.py

```
from file_client_cli import remote_get
import time
import datetime
import threading
import socket

def send_all():
    texec = dict()
    daftar = 'pokijan.jpg'

    catat_awal = datetime.datetime.now()
    for k in range(100):
        print(f"mengirim {k}")
        waktu = time.time()
        #bagian ini merupakan bagian yang menginstruksikan eksekusi fungsi send >
        texec[k] = threading.Thread(target=remote_get, args=(daftar,))
        texec[k].start()

    #setelah menyelesaikan tugasnya, dikembalikan ke main thread dengan join
    for k in range(100):
        texec[k].join()

    catat_akhir = datetime.datetime.now()
    selesai = catat_akhir - catat_awal
    print(f"Waktu TOTAL yang dibutuhkan {selesai} detik {catat_awal} s/d {catat_akhir}")

#fungsi send all akan dijalankan secara multithreading
```

```
if __name__=='__main__':  
    send_all()
```

6. Jalankan file udp_multi_thread.py

- Hasil pada file server di alpine 1

WARNING:root:memproses request: GET
WARNING:root:connection from ('192.168.122.5', 32900)
WARNING:root:string diproses: GET pokijan.jpg
WARNING:root:memproses request: GET
WARNING:root:connection from ('192.168.122.5', 32902)
WARNING:root:string diproses: GET pokijan.jpg
WARNING:root:memproses request: GET
WARNING:root:connection from ('192.168.122.5', 32940)
WARNING:root:string diproses: GET pokijan.jpg
WARNING:root:memproses request: GET
WARNING:root:connection from ('192.168.122.5', 32942)
WARNING:root:string diproses: GET pokijan.jpg
WARNING:root:memproses request: GET
WARNING:root:connection from ('192.168.122.5', 32944)
WARNING:root:string diproses: GET pokijan.jpg
WARNING:root:memproses request: GET
WARNING:root:connection from ('192.168.122.5', 32946)
WARNING:root:string diproses: GET pokijan.jpg
WARNING:root:memproses request: GET
WARNING:root:string diproses: GET pokijan.jpg
WARNING:root:memproses request: GET
WARNING:root:connection from ('192.168.122.5', 32948)
WARNING:root:string diproses: GET pokijan.jpg
WARNING:root:memproses request: GET
WARNING:root:connection from ('192.168.122.5', 32950)

Map topology	Servers
Filter nodes	
sort by name ascending	
NAT1	none
Switch1	none
alpine-1	telnet 192.168.188.3:5000
alpine-2	telnet 192.168.188.3:5002
alpine-3	telnet 192.168.188.3:5006

- Hasil dari file udp_multi_thread.py

mengirim 75
mengirim 76
mengirim 77
mengirim 78
mengirim 79
mengirim 80
mengirim 81
mengirim 82
mengirim 83
mengirim 84
mengirim 85
mengirim 86
mengirim 87
mengirim 88
mengirim 89
mengirim 90
mengirim 91
mengirim 92
mengirim 93
mengirim 94
mengirim 95
mengirim 96
mengirim 97
mengirim 98
mengirim 99

Map topology	Servers
Filter nodes	
sort by name ascending	
NAT1	none
Switch1	none
alpine-1	telnet 192.168.188.3:5000
alpine-2	telnet 192.168.188.3:5002
alpine-3	telnet 192.168.188.3:5006

The screenshot displays a network simulation environment. On the left, a terminal window shows a series of 15 warning messages: "WARNING:root:data received from server:". Below these, it reports the total execution time as 0:00:03.295443 detik on 2021-07-10 at 13:58:09.002908 s/d, and the current time as 2021-07-10 13:58:09.002908 s/d 2021-07-10 13:58:12.298351. The prompt is "/progjar/progjar4a #".

On the right, a "Map topology" panel is visible, showing a list of nodes and their associated IP addresses. The nodes are NAT1, Switch1, alpine-1, alpine-2, and alpine-3. NAT1 and Switch1 have no IP addresses, while the alpine nodes have telnet IP addresses.

Node	IP Address
NAT1	none
Switch1	none
alpine-1	telnet 192.168.188.3:5000
alpine-2	telnet 192.168.188.3:5002
alpine-3	telnet 192.168.188.3:5006