PEMROGRAMAN JARINGAN
Tugas Praktikum Concurrency

Class E

05111840000127 - Salsabila Harlen

Lecturer :

Royyana M. Ijtihadie

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

Buatlah program yang mengimplementasikan :

1. multi process
2. multi thread
3. multi process asynchronous
4. multi thread asynchronous

Dengan menggunakan protokol transport UDP. kasus dapat didefinsikan sendiri. dan Buatlah arsitektur jaringan anda sendiri di simulator GNS3

Buatlah laporan dalam bentuk PDF yang berisikan screenshot dari

1. deskripsi kasus yang dibuat
2. gambar arsitektur jaringan (dalam simulator GNS3)
3. program yang dibuat (1-4)
4. hasil outputnya

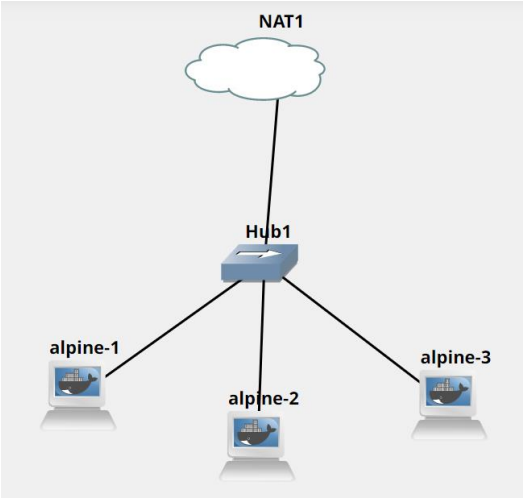
Jawab :

Berikut merupakan deskripsi untuk tugas praktikum implementasi kasus pada concurrency yaitu mengunduh sebuah file bertipe image menggunakan project GNS3 yang terdiri dari 3 alpine, dimana :

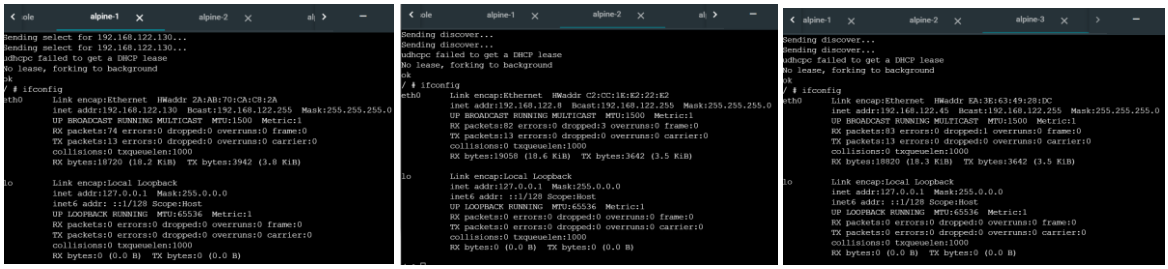
- Alpine 1 : 192.168.122.130 (server)
- Alpine 2 : 192.168.122.8 (client)
- Alpine 3 : 192.168.122.45 (server)

Selain itu, protocol transport udp akan mengirimkan file pdf kepada server. Pengiriman tersebut akan dengan menggunakan program multi process, multi thread, multi process asynchronous, multi thread asynchronous.

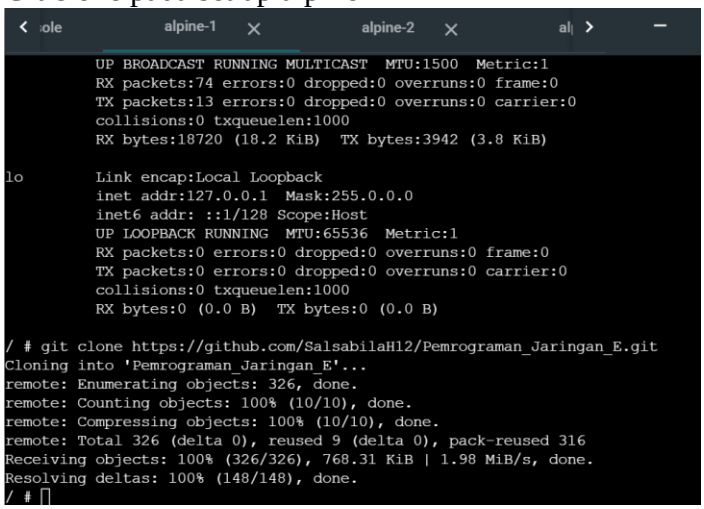
Berikut merupakan topologinya :



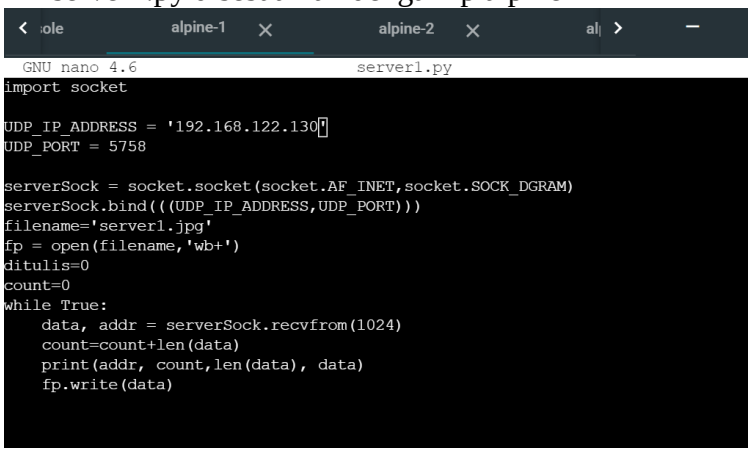
1. Ifconfig untuk mengecek ip setiap alpine



2. Git clone pada setiap alpine



3. Tambahkan file pada progjar3, server1.py dan server_2.py . Lalu, sesuaikan ip dengan ip masing-masing alpine.
   - server1.py disesuaikan dengan ip alpine 1



- server2.py disesuaikan dengan ip alpine 3

alpine-1 alpine-2 alpine-3

```
  GNU nano 4.6                          server2.py
import socket

UDP_IP_ADDRESS = '192.168.122.45'
UDP_PORT = 5758

serverSock = socket.socket(socket.AF_INET,socket.SOCK_DGRAM)
serverSock.bind(((UDP_IP_ADDRESS,UDP_PORT)))
filename='server2.jpg'
fp = open(filename,'wb+')
ditulis=0
count=0
while True:
    data, addr = serverSock.recvfrom(1024)
    count=count+len(data)
    print(addr, count,len(data), data)
    fp.write(data)
```

4. Modifikasi file – file yang terdapat pada folder progjar3 :
   - Library.py

```
import logging
import requests
import socket
import os
import time
import datetime

def get_url_list():
    urls = dict()
    urls['olivia']='https://m.media-amazon.com/images/M/MV5BYTYxODRmNzItMGFkNS00Mzk4LTk0NTMtYjdiYjgwYWNhODQ2XkEyXkFqcGdeQX
    urls['taylor']='http://jadiberita.com/wp-content/uploads/2014/06/taylor-swift-presenting-jpg.jpg'
    return urls

def download_gambar(url=None,tuliskefile='image'):
    waktu_awal = datetime.datetime.now()
    if (url is None):
        return False
    ff = requests.get(url)
    tipe = dict()
    tipe['image/png']='png'
    tipe['image/jpg']='jpg'
    tipe['image/gif']='gif'
    tipe['image/jpeg']='jpg'
    tipe['application/zip']='jpg'
    tipe['video/quicktime']='mov'
    # time.sleep(2) #untuk simulasi, diberi tambahan delay 2 detik

    content_type = ff.headers['Content-Type']
    logging.warning(content_type)
    if (content_type in list(tipe.keys())):
        namafile = os.path.basename(url)
        ekstensi = tipe[content_type]
        if (tuliskefile):
            fp = open(f"{tuliskefile}.{ekstensi}","wb")
            fp.write(ff.content)
            fp.close()
        waktu_process = datetime.datetime.now() - waktu_awal
        waktu_akhir =datetime.datetime.now()
        logging.warning(f"writing {tuliskefile}.{ekstensi} dalam waktu {waktu_process} {waktu_awal} s/d {waktu_akhir}")
        return waktu_process
```

```
    else:
        return False

def kirim_gambar(IP_ADDRESS, PORT, filename):
    print(IP_ADDRESS, PORT, filename)
    ukuran=os.stat(filename).st_size
    clientSock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)

    fp=open(filename,'rb')
    k=fp.read()
    terkirim=0
    for x in k:
        k_bytes=bytes([x])
        clientSock.sendto(k_bytes,(IP_ADDRESS,PORT))
        terkirim=terkirim+1

if __name__=='__main__':

    k = download_gambar('https://m.media-amazon.com/images/M/MV5BYTYxODRmNzItMGFkNS00Mzk4LTk0NTMtYjdiYjgwYWNhODQ2XkEyXkFqc
    print(k)
```

- Multi_procress.py

```
from library import download_gambar, get_url_list, kirim_gambar
import time
import datetime
from multiprocessing import Process

def kirim_server():
    texec = dict()
    urls = get_url_list()
    temp = 0
    catat_awal = datetime.datetime.now()
    for k in urls:
        print(f"mendownload {urls[k]}")
        waktu = time.time()
        UDP_IP_ADDRESS = "192.168.122.130"
        UDP_IP_ADDRESS2 = "192.168.122.45"
        PORT = 5758
        #bagian ini merupakan bagian yang mengistruksikan eksekusi fungsi download gambar secara multiprocess
        if temp == 0:
            texec[k] = Process(target=kirim_gambar, args=(UDP_IP_ADDRESS,PORT,f"{k}.jpg"))
            print('Masuk server 1')
            temp = temp+1
        elif temp == 1:
            print('Masuk server 2')
            texec[k] = Process(target=kirim_gambar, args=(UDP_IP_ADDRESS2,PORT,f"{k}.jpg"))
        texec[k].start()
    #setelah menyelesaikan tugasnya, dikembalikan ke main process dengan join
    for k in urls:
        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 download_gambar akan dijalankan secara multi process
if __name__=='__main__':
    kirim_server()
```

- Multi_process_async.py

```python
from library import download_gambar, get_url_list, kirim_gambar
import time
import datetime
from multiprocessing import Process, Pool

def kirim_server():
    texec = dict()
    urls = get_url_list()
    status_task = dict()
    temp = 0
    task_pool = Pool(processes=20) #2 task yang dapat dikerjakan secara simultan, dapat diset sesuai jumlah core
    catat_awal = datetime.datetime.now()
    for k in urls:
        download_gambar(urls[k],k)
        print(f"mendownload {urls[k]}")
        #bagian ini merupakan bagian yang mengistruksikan eksekusi fungsi download gambar secara multiprocess
        UDP_IP_ADDRESS = "192.168.122.130"
        UDP_IP_ADDRESS2 = "192.168.122.45"
        PORT = 5758
        if temp == 0:
            texec[k] = task_pool.apply_async(func=kirim_gambar, args=(UDP_IP_ADDRESS,PORT,f"{k}.jpg"))
            print('Masuk server 1')
            temp = temp+1
        elif temp == 1:
            print('Masuk server 2')
            texec[k] = task_pool.apply_async(func=kirim_gambar, args=(UDP_IP_ADDRESS2,PORT,f"{k}.jpg"))
    #setelah menyelesaikan tugasnya, dikembalikan ke main process dengan mengambil hasilnya dengan get
    for k in urls:
        status_task[k]=texec[k].get(timeout=10)

    catat_akhir = datetime.datetime.now()
    selesai = catat_akhir - catat_awal
    print(f"Waktu TOTAL yang dibutuhkan {selesai} detik {catat_awal} s/d {catat_akhir}")
    print("status TASK")
    print(status_task)
#fungsi download_gambar akan dijalankan secara multi process
if __name__=='__main__':
    kirim_server()
```

- Multi_thread.py

```python
from library import download_gambar,get_url_list, kirim_gambar
import time
import datetime
import threading

def kirim_server():
    texec = dict()
    urls = get_url_list()
    temp = 0
    catat_awal = datetime.datetime.now()
    for k in urls:
        download_gambar(urls[k], k)
        print(f"mendownload {urls[k]}")
        waktu = time.time()
        UDP_IP_ADDRESS = "192.168.122.130"
        UDP_IP_ADDRESS2 = "192.168.122.45"
        PORT = 5758
        #bagian ini merupakan bagian yang mengistruksikan eksekusi fungsi download gambar secara multithread
        if temp == 0:
            texec[k] = threading.Thread(target=kirim_gambar, args=(UDP_IP_ADDRESS,PORT,f"{k}.jpg"))
            print('Masuk server 1')
            temp = temp+1
        elif temp == 1:
            print('Masuk server 2')
            texec[k] = threading.Thread(target=kirim_gambar, args=(UDP_IP_ADDRESS2,PORT,f"{k}.jpg"))
        texec[k].start()

    #setelah menyelesaikan tugasnya, dikembalikan ke main thread dengan join
    for k in urls:
        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 download_gambar akan dijalankan secara multithreading

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

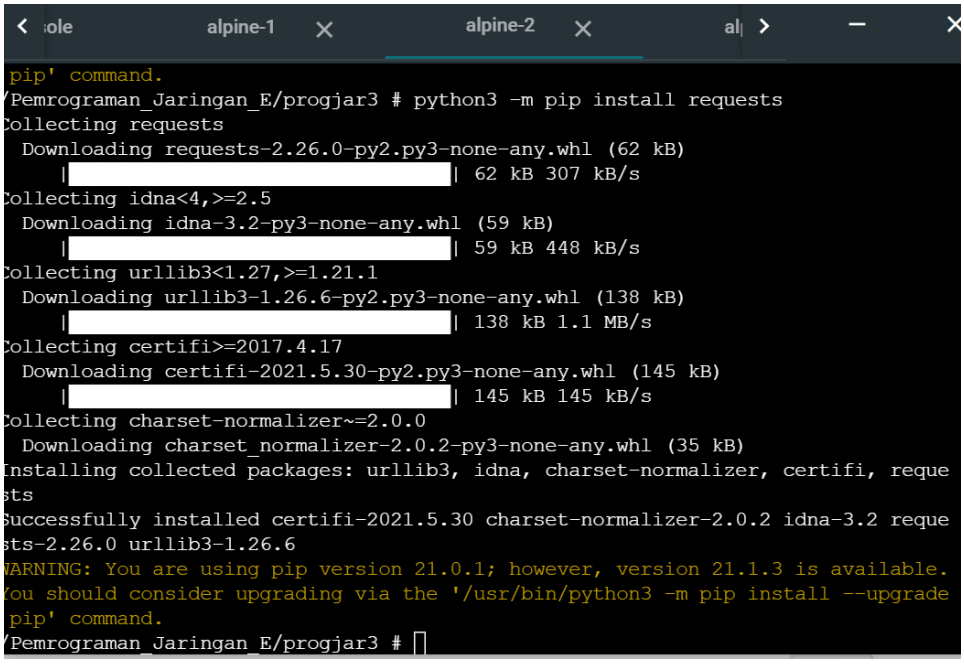
- Muti_thread_async.py

```python
from library import download_gambar,get_url_list, kirim_gambar
import time
import datetime
import concurrent.futures

def kirim_server():
    texec = dict()
    urls = get_url_list()
    status_task = dict()
    temp = 0
    task = concurrent.futures.ThreadPoolExecutor(max_workers=4)
    catat_awal = datetime.datetime.now()
    for k in urls:
        download_gambar(urls[k], k)
        print(f"mendownload {urls[k]}")
        waktu = time.time()
        UDP_IP_ADDRESS = "192.168.122.130"
        UDP_IP_ADDRESS2 = "192.168.122.45"
        PORT = 5758
        #bagian ini merupakan bagian yang mengistruksikan eksekusi fungsi download gambar secara multithread
        if temp == 0:
            texec[k] = task.submit(kirim_gambar, UDP_IP_ADDRESS,PORT,f"{k}.jpg")
            print('Masuk server 1')
            temp = temp+1
        elif temp == 1:
            print('Masuk server 2')
            texec[k] = task.submit(kirim_gambar, UDP_IP_ADDRESS2,PORT,f"{k}.jpg")
    #setelah menyelesaikan tugasnya, dikembalikan ke main thread dengan memanggil result
    for k in urls:
        status_task[k]=texec[k].result()
    catat_akhir = datetime.datetime.now()
    selesai = catat_akhir - catat_awal
    print(f"Waktu TOTAL yang dibutuhkan {selesai} detik {catat_awal} s/d {catat_akhir}")
    print("hasil task yang dijalankan")
    print(status_task)
#fungsi download_gambar akan dijalankan secara multithreading
if __name__=='__main__':
    kirim_server()
```

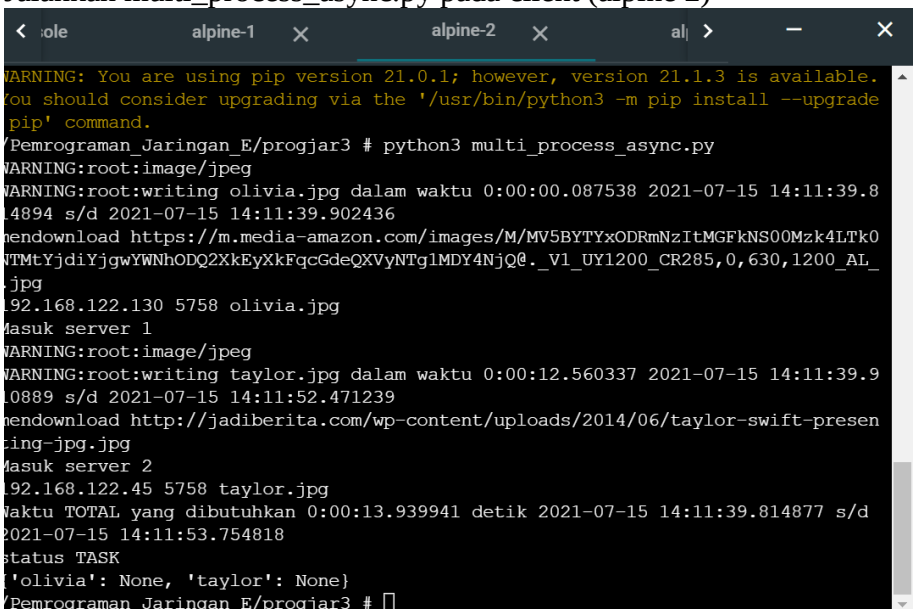
5. Masukkan file – file yang sudah dimodifikasi tadi kedalam alpine 2 (client)

```
ole          alpine-1 ×          alpine-2 ×          al
          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/SalsabilaH12/Pemrograman_Jaringan_E.git
Cloning into 'Pemrograman_Jaringan_E'...
remote: Enumerating objects: 326, done.
remote: Counting objects: 100% (10/10), done.
remote: Compressing objects: 100% (10/10), done.
remote: Total 326 (delta 0), reused 9 (delta 0), pack-reused 316
Receiving objects: 100% (326/326), 767.23 KiB | 1.58 MiB/s, done.
Resolving deltas: 100% (153/153), done.
/ # cd Pemrograman_Jaringan_E
/Pemrograman_Jaringan_E # cd progjar3
/Pemrograman_Jaringan_E/progjar3 # nano library.py
/Pemrograman_Jaringan_E/progjar3 # nano multi_process.py
/Pemrograman_Jaringan_E/progjar3 # nano multi_process_async.py
/Pemrograman_Jaringan_E/progjar3 # nano multi_thread.py
/Pemrograman_Jaringan_E/progjar3 # nano multi_thread_async.py
/Pemrograman_Jaringan_E/progjar3 #
```

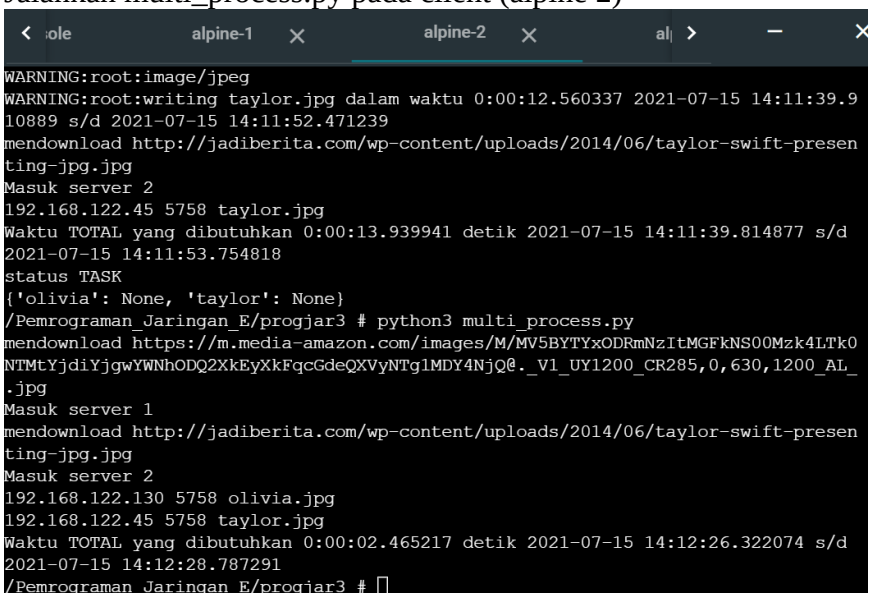
6. Jalankan server1.py pada alpine 1 dan server2.py pada alpine 3, serta install module request pada client (alpine 2)



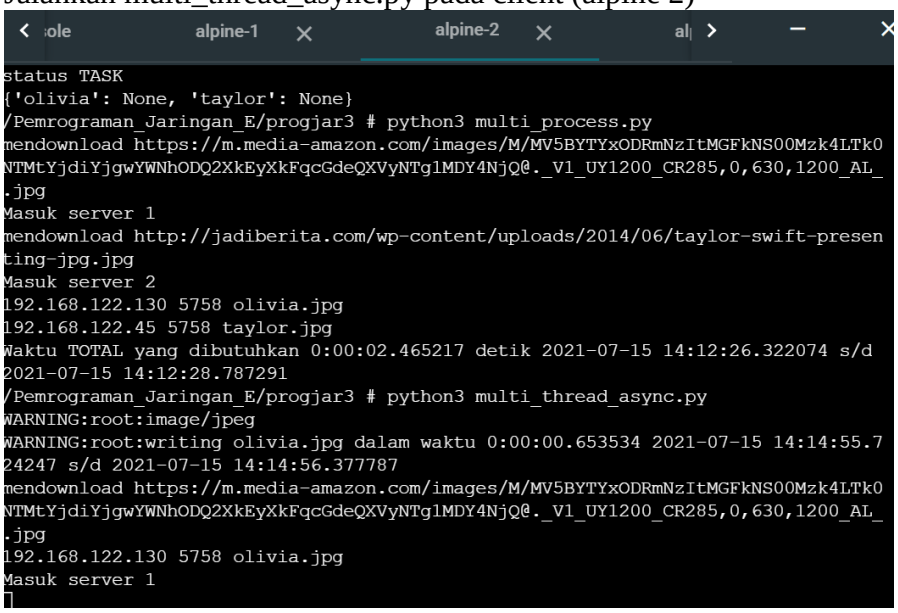
7. Jalankan multi_process_async.py pada client (alpine 2)



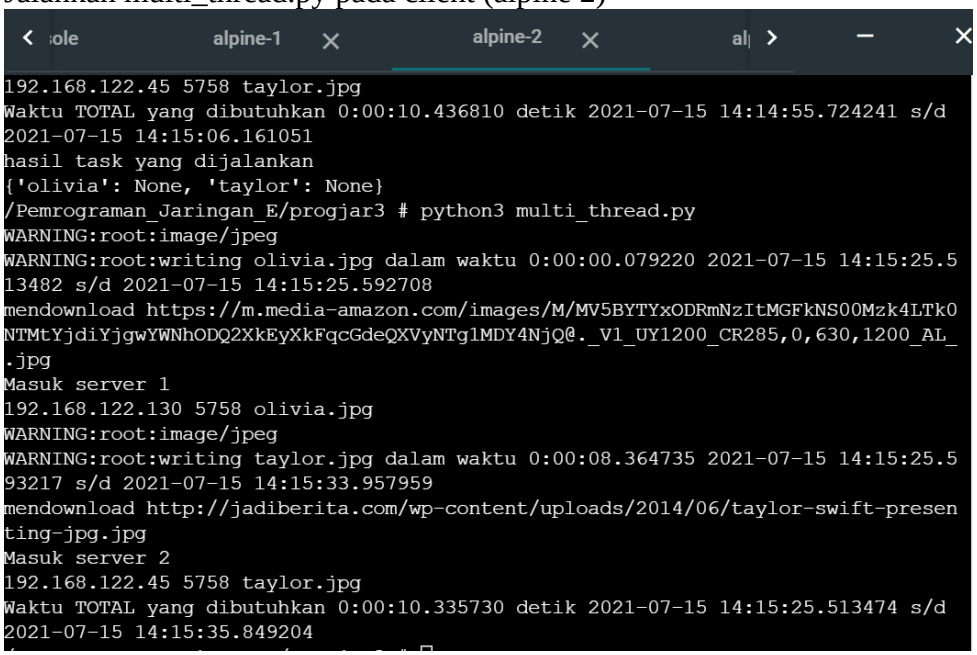
8. Jalankan multi_process.py pada client (alpine 2)



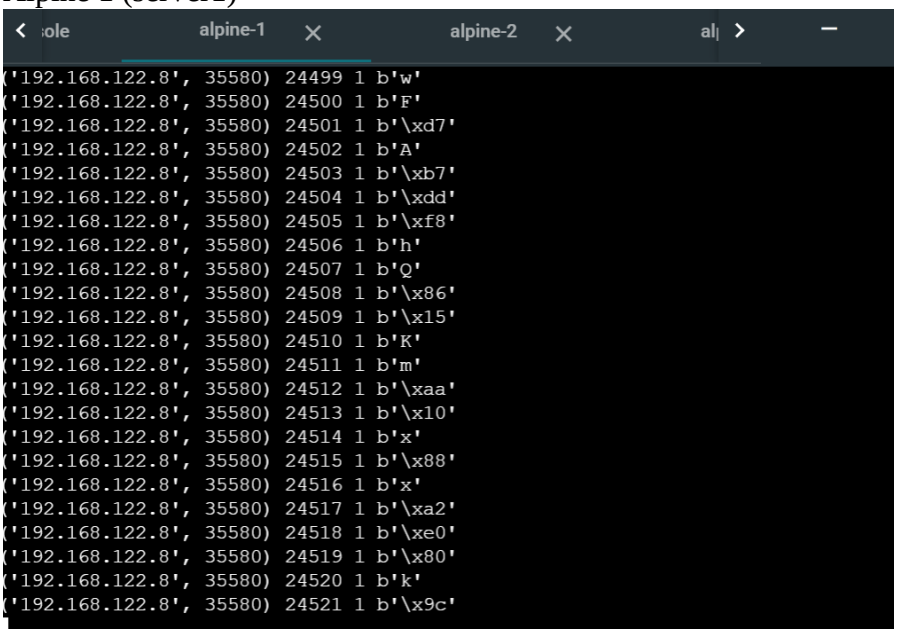
9. Jalankan multi_thread_async.py pada client (alpine 2)



10. Jalankan multi_thread.py pada client (alpine 2)



11. Maka, pada alpine 1 dan 3 sebagai server1 dan server2 akan mengeluarkan hasil berikut.
    - Alpine 1 (server1)



- Alpine 3 (server2)