UAS Distributed System (Data Sharing)

Dokumen ini menjelaskan proses lengkap untuk membangun dan mengkonfigurasi topologi jaringan yang aman di GNS3 untuk memenuhi skenario Ujian Akhir Semester (UAS) Distributed System.

Tujuan: Membuat simulasi jaringan multi-zona yang dikendalikan oleh satu router/firewall pusat (Alpine Linux). Jaringan internal akan memiliki akses internet (NAT) dan terlindung dari jaringan eksternal (Firewall). Pengecualian akan dibuat untuk layanan spesifik (Aplikasi Data Sharing) dan koneksi aman dari eksternal akan diizinkan melalui (VPN).

Modul yang Digunakan:

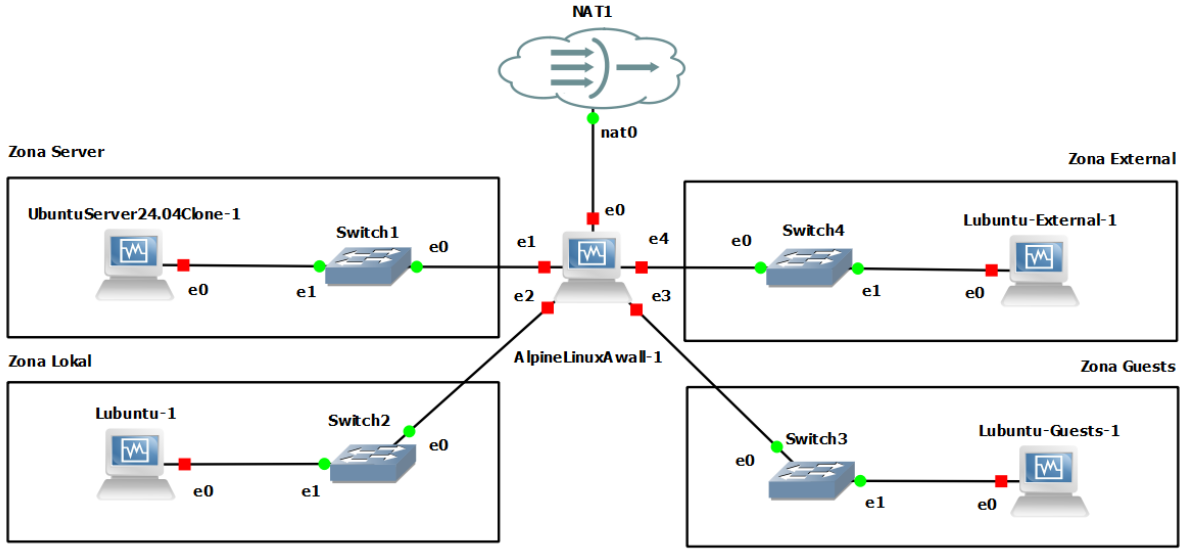
- **GNS3:** Simulator jaringan
- **VirtualBox:** Menjalankan VM
- **Alpine Linux:** VM yang berfungsi sebagai Router, Firewall (iptables), NAT, dan DHCP Server.
- **Ubuntu Server:** VM yang menghosting aplikasi server Python (Data Sharing).
- **Lubuntu (x3):** VM yang bertindak sebagai klien di zona Lokal, Tamu, dan Eksternal.
- **Python & Pastebin:** Bahasa untuk aplikasi dan metode transfer file.

Tahap 1: Prasyarat & Penyiapan Topologi GNS3

1.1 Desain Topologi & Rencana IP

Buat proyek baru di GNS3. Drag-and-drop perangkat berikut ke *canvas*:

- **Node Jaringan:**
    - 1x NAT (Ini akan menjadi Internet kita)
    - 4x Ethernet switch
- **VMs:**
    - 1x AlpineLinuxAwall (Ini adalah router utama kita)
    - 1x UbuntuServer
    - 1x Lubuntu (asli, untuk Zona Lokal)
    - 1x Lubuntu-Guests (klon)
    - 1x Lubuntu-External (klon)



WAN	**Perangkat Terhubung**	**Interface Alpine**	**Network Address**	**IP Gateway (di Alpine)**
WAN	NAT1	eth0	DHCP (dari NAT1)	DHCP
Server	UbuntuServer, Switch1	Eth1	192.168.10.0/24	192.168.10.1
Lokal	Lubuntu, Switch2	Eth2	192.168.20.0/24	192.168.20.1
Guests	Lubuntu-Guests, Switch3	Eth3	192.168.30.0/24	192.168.30.1
External	Lubuntu-External, Switch4	Eth4	172.16.1.0/24	172.16.1.1
VPN	(Virtual)	tun0	10.8.0.0/24	10.8.0.1

1.4 Menghubungkan Kabel Virtual

Hubungkan semua perangkat persis seperti gambar dan tabel ini:

- NAT1 -> eth0 di AlpineLinux
- **Zona Server:**
  - UbuntuServer -> Switch1
  - Switch1 -> eth1 di AlpineLinux
- **Zona Lokal:**
  - Lubuntu -> Switch2
  - Switch2 -> eth2 di AlpineLinux
- **Zona Guests:**
  - Lubuntu-Guests -> Switch3

    - Switch3 -> eth3 di AlpineLinux
- **Zona External:**
    - Lubuntu-External -> Switch4
    - Switch4 -> eth4 di AlpineLinux

Tahap 2: Konfigurasi AlpineLinuxAwall (Router & Firewall)

2.1 Instalasi Awal & IP Address

1. Login sebagai root.
2. Instal editor nano:

```
apk update
apk add nano
```

3. Buat direktori *network*:

```
mkdir /etc/network
```

4. Edit file konfigurasi *interface*:

```
nano /etc/network/interfaces
```

5. Hapus semua isinya dan ganti dengan konfigurasi ini:

```
auto lo
iface lo inet loopback

# eth0 (WAN) terhubung ke GNS3 NAT node
auto eth0
iface eth0 inet dhcp

# eth1 (Zona Server)
auto eth1
iface eth1 inet static
    address 192.168.10.1
    netmask 255.255.255.0

# eth2 (Zona Lokal)
auto eth2
iface eth2 inet static
    address 192.168.20.1
    netmask 255.255.255.0

# eth3 (Zona Guests)
auto eth3
iface eth3 inet static
    address 192.168.30.1
    netmask 255.255.255.0

# eth4 (Zona External)
auto eth4
iface eth4 inet static
```

```
    address 172.16.1.1
    netmask 255.255.255.0
```

6. Simpan (Ctrl+O, Enter) dan Keluar (Ctrl+X).
7. Terapkan perubahan jaringan:

```
rc-service networking restart
```

2.2 Aktifkan Routing (IP Forwarding)

1. Izinkan kernel untuk me-rute-kan paket:

```
nano /etc/sysctl.conf
```

2. Cari baris #net.ipv4.ip_forward=1 dan hapus tanda pagar (#) di depannya.
3. Simpan dan Keluar.
4. Terapkan perubahan:

```
sysctl -p
```

2.3 Konfigurasi DHCP Server & NAT

1. Instal paket yang diperlukan (dnsmasq untuk DHCP, iptables untuk NAT/Firewall):

```
apk add dnsmasq iptables
```

2. Konfigurasi NAT: Aturan ini membuat VM internal bisa "meminjam" IP eth0 (Internet) milik Alpine.

```
iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
```

3. Konfigurasi DHCP Server: Kita akan memberikan IP otomatis ke eth1, eth2, dan eth3. eth4 (External) sengaja tidak diberi.

```
nano /etc/dnsmasq.conf
```

4. Hapus semua isi file dan ganti dengan ini (konfigurasi ini sudah mencakup perbaikan *bug* bind-interfaces):

```
# Memaksa dnsmasq untuk HANYA
# mendengarkan di interface yang kita sebutkan
bind-interfaces

# Tentukan interface yang kita dengarkan
interface=eth1
interface=eth2
interface=eth3

# Range untuk Zona Server (eth1)
dhcp-
range=interface:eth1,192.168.10.100,192.168.10.150,255.255.
255.0,12h
```

```
# Range untuk Zona Lokal (eth2)
dhcp-
range=interface:eth2,192.168.20.100,192.168.20.150,255.255.
255.0,12h

# Range untuk Zona Guests (eth3)
dhcp-
range=interface:eth3,192.168.30.100,192.168.30.150,255.255.
255.0,12h

# Beri tahu klien apa Gateway mereka
dhcp-option=interface:eth1,option:router,192.168.10.1
dhcp-option=interface:eth2,option:router,192.168.20.1
dhcp-option=interface:eth3,option:router,192.168.30.1

# Beri tahu klien apa DNS Server mereka
dhcp-option=option:dns-server,8.8.8.8
dhcp-option=option:dns-server,1.1.1.1
```

5. Simpan dan Keluar.

2.4 Konfigurasi Firewall (iptables)

1. Bersihkan semua aturan FORWARD yang mungkin ada:

```
iptables -F FORWARD
```

2. Atur *policy* default untuk DROP (Blokir Semua):

```
iptables -P FORWARD DROP
```

3. Tambahkan Aturan (Jalankan satu per satu):

```
# (FIX WAJIB) Izinkan DHCP request (INPUT) agar VM bisa
dapat IP
iptables -I INPUT -p udp --dport 67 -j ACCEPT

# Izinkan koneksi balasan (PENTING untuk SEMUA koneksi)
iptables -A FORWARD -m conntrack --ctstate
ESTABLISHED,RELATED -j ACCEPT

# Izinkan Internal (Server, Lokal, Guests) mengakses
Internet (eth0)
iptables -A FORWARD -i eth1 -o eth0 -j ACCEPT
iptables -A FORWARD -i eth2 -o eth0 -j ACCEPT
iptables -A FORWARD -i eth3 -o eth0 -j ACCEPT

# Izinkan Lokal (eth2) dan Guests (eth3) berbicara ke
Server (eth1)
iptables -A FORWARD -i eth2 -o eth1 -j ACCEPT
iptables -A FORWARD -i eth3 -o eth1 -j ACCEPT

# === ATURAN PENGECUALIAN UAS (IZINKAN EKSTERNAL) ===
```

```
# Izinkan External (eth4) -> Server (eth1) untuk PING
iptables -A FORWARD -i eth4 -o eth1 -d 192.168.10.0/24 -p
icmp --icmp-type echo-request -j ACCEPT

# Izinkan External (eth4) -> Server (eth1) untuk SSH (Port
22)
iptables -A FORWARD -i eth4 -o eth1 -d 192.168.10.0/24 -p
tcp --dport 22 -j ACCEPT

# Izinkan External (eth4) -> Server (eth1) untuk Aplikasi
Data Sharing (Port 50001)
iptables -A FORWARD -i eth4 -o eth1 -d 192.168.10.0/24 -p
tcp --dport 50001 -j ACCEPT
```

2.5 Simpan & Jalankan Layanan

1. **Simpan semua aturan Firewall** secara permanen:

```
rc-service iptables save
```

2. Jalankan dan aktifkan dnsmasq saat *booting*:

```
rc-service dnsmasq start
rc-update add dnsmasq default
```

3. Aktifkan iptables saat *booting*:

```
rc-update add iptables default
```

AlpineLinux sekarang sudah selesai dikonfigurasi.

Tahap 3: Konfigurasi VM Klien

3.1 Klien Internal (UbuntuServer, Lubuntu, Lubuntu-Guests)

1. Nyalakan ketiga VM ini.
2. Mereka **seharusnya mendapatkan IP secara otomatis** dari dnsmasq.
3. Buka terminal di masing-masing VM dan verifikasi:
   - UbuntuServer akan mendapatkan 192.168.10.100 (atau serupa).
   - Lubuntu (Lokal) akan mendapatkan 192.168.20.100 (atau serupa).
   - Lubuntu-Guests akan mendapatkan 192.168.30.100 (atau serupa).
4. Verifikasi akses internet di ketiganya dengan ping 8.8.8.8. Ini harus berhasil.

3.2 Klien Eksternal (Lubuntu-External)

Zona ini sengaja tidak diberi DHCP. Kita harus mengaturnya secara manual.

1. Nyalakan Lubuntu-External.
2. Setelah *booting*, klik ikon jaringan di *taskbar* -> **Edit Connections...**
3. Pilih koneksi enp0s3 (atau yang serupa) -> klik ikon **Edit** (roda gigi).
4. Pindah ke tab **"IPv4 Settings"**.
5. Ubah "Method" dari "Automatic (DHCP)" menjadi **"Manual"**.
6. Klik tombol **"Add"** di bawah "Addresses" dan masukkan:

- **Address:** 172.16.1.100
- **Netmask:** 255.255.255.0
- **Gateway:** 172.16.1.1

7. Di kotak "DNS servers", masukkan 8.8.8.8.
8. Klik **Save**. Koneksi akan terputus dan tersambung kembali.
9. **Verifikasi:** Buka terminal dan ping 172.16.1.1. Ini harus berhasil. (Mencoba ping 8.8.8.8 akan gagal, ini normal dan sesuai desain firewall kita).

Tahap 4: Konfigurasi VPN (OpenVPN)

Ini adalah modul VPN. Semua perintah dijalankan di AlpineLinux.

4.1 Instalasi & Persiapan

```
apk add openvpn easy-rsa bash python3
mkdir -p /etc/openvpn/easy-rsa
cp -r /usr/share/easy-rsa/* /etc/openvpn/easy-rsa/
cd /etc/openvpn/easy-rsa/
```

4.2 Pembuatan Kunci & Sertifikat

Jalankan ini satu per satu di dalam direktori /etc/openvpn/easy-rsa/:

```
./easyrsa init-pki
./easyrsa build-ca nopass
./easyrsa gen-req server nopass
./easyrsa sign-req server server  # Ketik: yes
./easyrsa gen-req client1 nopass
./easyrsa sign-req client client1 # Ketik: yes
./easyrsa gen-dh
openvpn --genkey --secret pki/ta.key
```

4.3 Konfigurasi Server OpenVPN

Salin kunci-kunci penting:

```
cp /etc/openvpn/easy-rsa/pki/ca.crt /etc/openvpn/
cp /etc/openvpn/easy-rsa/pki/issued/server.crt /etc/openvpn/
cp /etc/openvpn/easy-rsa/pki/private/server.key /etc/openvpn/
cp /etc/openvpn/easy-rsa/pki/dh.pem /etc/openvpn/
cp /etc/openvpn/easy-rsa/pki/ta.key /etc/openvpn/
```

Buat file /etc/openvpn/openvpn.conf (kita gunakan nama ini untuk kompatibilitas rc-service).

```
nano /etc/openvpn/openvpn.conf
```

Tempelkan konfigurasi ini:

```
port 1194
proto udp
dev tun
ca ca.crt
cert server.crt
key server.key
```

```
dh dh.pem
tls-auth ta.key 0
topology subnet
server 10.8.0.0 255.255.255.0
push "route 192.168.10.0 255.255.255.0"
push "route 192.168.20.0 255.255.255.0"
push "route 192.168.30.0 255.255.255.0"
push "dhcp-option DNS 8.8.8.8"
client-to-client
cipher AES-256-GCM
auth SHA256
keepalive 10 120
user nobody
group nobody
persist-key
persist-tun
status /var/log/openvpn-status.log
verb 3
```

Jalankan dan aktifkan layanan:

```
rc-service openvpn start
rc-update add openvpn default
```

4.4 Memperbarui Firewall untuk VPN

Kita harus membuka port VPN dan mengizinkan *traffic* dari jaringan VPN.

```
# 1. Izinkan koneksi OpenVPN (INPUT) di port 1194
iptables -A INPUT -p udp --dport 1194 -j ACCEPT

# 2. Izinkan traffic dari VPN (tun0) ke Jaringan Internal
iptables -A FORWARD -i tun0 -o eth1 -j ACCEPT
iptables -A FORWARD -i tun0 -o eth2 -j ACCEPT

# 3. Simpan permanen
rc-service iptables save
```

Tahap 5: Penyiapan Aplikasi Python (Data Sharing)

5.1 Kode Python Server.py

Untuk UbuntuServer.

```
import socketserver
import os
import threading

# --- KONFIGURASI SERVER ---
HOST = '0.0.0.0'    # Mendengarkan di semua interface (Lokal &
VPN)
PORT = 50001        # Port yang dibuka di Firewall
SHARE_DIR = "share" # Folder penyimpanan file
```

```
BUFFER = 4096       # Ukuran buffer transfer data

class ClientHandler(socketserver.BaseRequestHandler):
    """Class Handler: Menangani satu koneksi klien dalam
thread terpisah"""

    def handle(self):
        addr = self.client_address
        print(f"[+] Koneksi masuk dari: {addr}")

        try:
            while True:
                # 1. Menerima data perintah mentah
                data =
self.request.recv(1024).strip().decode()
                if not data: break # Jika data kosong, klien
putus koneksi

                # 2. Memecah perintah (Format:
PERINTAH|ARGUMEN)
                parts = data.split('|', 1)
                cmd = parts[0].upper()

                # --- LOGIKA IF/ELSE UNTUK SETIAP PERINTAH ---

                # [A] FITUR LIST FILE
                if cmd == "LIST":
                    files = os.listdir(SHARE_DIR)
                    # Menggabungkan nama file dengan titik
koma (;) agar dikirim satu baris
                    msg = ";".join(files) if files else
"Folder kosong"
                    self.request.sendall(f"OK|{msg}\n".encode(
))

                # [B] FITUR UPLOAD (Terima file dari Klien)
                elif cmd == "UPLOAD":
                    filename = os.path.basename(parts[1]) #
Ambil nama file saja (keamanan)
                    filepath = os.path.join(SHARE_DIR,
filename)

                    # Beritahu klien server siap
                    self.request.sendall(b"READY\n")

                    # Terima ukuran file
                    filesize =
int(self.request.recv(1024).strip())

                    # Loop penerimaan data
                    with open(filepath, 'wb') as f:
```

```
                        received = 0
                        while received < filesize:
                            chunk = self.request.recv(BUFFER)
                            f.write(chunk)
                            received += len(chunk)
                    self.request.sendall(b"OK|Upload
Selesai\n")

                # [C] FITUR DOWNLOAD (Kirim file ke Klien)
                elif cmd == "DOWNLOAD":
                    filename = os.path.basename(parts[1])
                    filepath = os.path.join(SHARE_DIR,
filename)

                    if os.path.exists(filepath):
                        # Kirim ukuran file
                        size = os.path.getsize(filepath)
                        self.request.sendall(f"OK|{size}\n".en
code())

                        # Baca file dan kirim
                        with open(filepath, 'rb') as f:
                            while True:
                                chunk = f.read(BUFFER)
                                if not chunk: break
                                self.request.sendall(chunk)
                    else:
                        self.request.sendall(b"ERROR|File
tidak ditemukan\n")

                # [D] FITUR DELETE (Hapus File)
                elif cmd == "DELETE":
                    filename = os.path.basename(parts[1])
                    filepath = os.path.join(SHARE_DIR,
filename)

                    if os.path.exists(filepath):
                        try:
                            os.remove(filepath)
                            self.request.sendall(b"OK|File
berhasil dihapus\n")
                        except Exception as e:
                            self.request.sendall(f"ERROR|Gagal
menghapus: {e}\n".encode())
                    else:
                        self.request.sendall(b"ERROR|File
tidak ditemukan\n")

                else:
                    print(f"[-] Perintah tidak dikenal dari
{addr}")
```

```
        except Exception as e:
            print(f"[-] Error koneksi {addr}: {e}")

        print(f"[-] Koneksi ditutup: {addr}")

# --- CLASS SERVER UTAMA (MULTITHREADED) ---
class ThreadedServer(socketserver.ThreadingMixIn,
socketserver.TCPServer):
    """Mengaktifkan Multithreading agar bisa banyak klien
sekaligus"""
    pass

if __name__ == "__main__":
    # Buat folder jika belum ada
    if not os.path.exists(SHARE_DIR):
        os.makedirs(SHARE_DIR)
        print(f"[+] Folder '{SHARE_DIR}' siap.")

    # FIX PENTING: Mencegah error "Address already in use"
    socketserver.TCPServer.allow_reuse_address = True

    # Jalankan Server
    server = ThreadedServer((HOST, PORT), ClientHandler)
    print(f"[*] Server Data Sharing berjalan di
{HOST}:{PORT}")
    print("[*] Tekan Ctrl+C untuk berhenti.")

    try:
        server.serve_forever()
    except KeyboardInterrupt:
        print("\n[*] Server dimatikan.")
        server.shutdown()
```

Simpan dan Keluar (Ctrl+O, Ctrl+X).

5.2 Kode Python Client.py

Untuk VM Lubuntu (Lokal, Guests, External)

```
import socket
import os
import sys
import time # Penting untuk sinkronisasi upload

BUFFER = 4096

def receive_line(sock):
    """Fungsi bantu: Menerima data sampai ketemu baris baru
(Enter)"""
    data = b""
    while True:
```

```
        chunk = sock.recv(1)
        if not chunk or chunk == b'\n':
            break
        data += chunk
    return data.decode()

def main():
    sock = None # Variabel untuk menyimpan status koneksi

    while True:
        # Prompt dinamis (menunjukkan status koneksi)
        prompt = "cli> " if sock else "cli (disconnected)> "

        try:
            cmd_input = input(prompt).strip()
        except EOFError:
            break # Handle Ctrl+D

        if not cmd_input: continue

        parts = cmd_input.split()
        cmd = parts[0].lower()

        try:
            # --- LOGIKA IF/ELSE PERINTAH KLIEN ---

            # [1] EXIT
            if cmd == "exit":
                break

            # [2] CONNECT
            elif cmd == "connect":
                if len(parts) != 3:
                    print("Usage: connect <ip> <port>")
                    continue

                # Reset socket lama jika ada
                if sock: sock.close()

                ip, port = parts[1], int(parts[2])
                sock = socket.socket(socket.AF_INET,
socket.SOCK_STREAM)
                sock.connect((ip, port))
                print(f"[+] Berhasil terhubung ke
{ip}:{port}")
                continue

            # Cek Koneksi sebelum menjalankan perintah lain
            elif not sock:
                print("[!] Belum terhubung. Gunakan: connect
<ip> <port>")
```

```
                continue

            # [3] LIST
            elif cmd == "list":
                sock.sendall(b"LIST\n")
                resp = receive_line(sock)

                # Parsing respons (OK|file1;file2)
                parts_resp = resp.split('|', 1)
                if len(parts_resp) > 1:
                    # Ganti titik koma (;) dengan Enter (\n)
untuk tampilan rapi
                    file_list = parts_resp[1].replace(';',
'\n')
                    print(f"--- File di Server ---
\n{file_list}\n----------------------")
                else:
                    print(resp)

            # [4] UPLOAD
            elif cmd == "upload":
                if len(parts) != 2:
                    print("Usage: upload <nama_file_lokal>")
                    continue

                local_file = parts[1]
                if not os.path.exists(local_file):
                    print("[!] File lokal tidak ditemukan.")
                    continue

                # Kirim Header
                sock.sendall(f"UPLOAD|{local_file}\n".encode()
)

                # FIX PENTING: Jeda sedikit agar server siap
menerima ukuran
                time.sleep(0.1)

                if receive_line(sock) == "READY":
                    size = os.path.getsize(local_file)
                    sock.sendall(f"{size}\n".encode()) # Kirim
ukuran

                    # Baca file dan kirim
                    with open(local_file, 'rb') as f:
                        while True:
                            chunk = f.read(BUFFER)
                            if not chunk: break
                            sock.sendall(chunk)
```

```
                    print(f"Server: {receive_line(sock)}") #
Terima OK

            # [5] DOWNLOAD
            elif cmd == "download":
                if len(parts) < 2:
                    print("Usage: download
<nama_file_server>")
                    continue

                remote_file = parts[1]
                sock.sendall(f"DOWNLOAD|{remote_file}\n".encod
e())

                resp = receive_line(sock)
                status, payload = resp.split('|', 1)

                if status == "OK":
                    size = int(payload)
                    received = 0
                    print(f"[*] Mendownload {remote_file}
({size} bytes)...")

                    with open(remote_file, 'wb') as f:
                        while received < size:
                            chunk = sock.recv(BUFFER)
                            f.write(chunk)
                            received += len(chunk)
                    print(f"[+] Download sukses.")
                else:
                    print(f"[!] Server Error: {payload}")

            # [6] DELETE
            elif cmd == "delete":
                if len(parts) != 2:
                    print("Usage: delete <nama_file_server>")
                    continue
                remote_file = parts[1]
                sock.sendall(f"DELETE|{remote_file}\n".encode(
))
                print(f"Server: {receive_line(sock)}")

            # [7] HELP
            elif cmd == "help":
                print("--- Bantuan ---")
                print("connect <ip> <port> : Hubungkan ke
server")
                print("list                : Lihat file di
server")
                print("upload <file>       : Upload file
lokal")
```

```
                print("download <file>     : Download file
server")
                print("delete <file>       : Hapus file di
server")
                print("exit                : Keluar")

            else:
                print(f"Perintah '{cmd}' tidak dikenal. Ketik
'help'.")

        except Exception as e:
            print(f"[!] Error: {e}")
            sock = None # Reset koneksi jika error fatal

    if sock: sock.close()
    print("[*] Program selesai.")

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

5.3 Transfer Kode Server.py (ke UbuntuServer)

Di Komputer Host Anda:

- Salin kode Server.py
- Buka Pastebin.com, tempel kode, klik "Create New Paste".
- Klik tombol "raw" di halaman berikutnya, lalu salin URL dari *address bar* Anda.

Di UbuntuServer:

- Buka terminal dan jalankan wget (ganti URL-nya):

```
wget -O Server.py https://pastebin.com/raw/URL_SERVER_ANDA
```

5.4 Transfer Kode Client.py (ke Lubuntu-1 & Lubuntu-Guests)

- Buka terminal di kedua VM ini.
- Jalankan wget dengan URL klien baru Anda:

```
wget -O Client.py https://pastebin.com/raw/URL_KLIEN_ANDA
```

5.5 Transfer Kode Client.py (ke Lubuntu-External - Metode Khusus)

Lubuntu-External sengaja diblokir dari internet oleh *firewall* kita. Kita harus "membuka lubang" sementara.

1. **Di AlpineLinuxAwall-1 (Buka Lubang):**
    - Jalankan perintah ini untuk mengizinkan External (eth4) mengakses Internet (eth0):

    ```
    iptables -A FORWARD -i eth4 -o eth0 -j ACCEPT
    ```

2. **Di Lubuntu-External-1 (Unduh File):**
    - Sekarang jalankan perintah wget yang sama seperti di langkah 4.2:

    ```
    wget -O Client.py https://pastebin.com/raw/URL_KLIEN
    ```

3. **Di AlpineLinuxAwall-1 (Tutup Lubang):**
    - **SANGAT PENTING:** Hapus aturan yang baru saja Anda buat untuk mengembalikan keamanan:

```
iptables -D FORWARD -i eth4 -o eth0 -j ACCEPT
```

4. **Di AlpineLinuxAwall-1 (Simpan Perubahan):**
    - Simpan kembali konfigurasi *firewall* Anda yang sudah aman:

```
rc-service iptables save
```

Tahap 6: Penyiapan Klien VPN (OpenVPN)

Ini adalah konfigurasi di sisi Lubuntu-External-1.

6.1 Instalasi openvpn di Klien

Di AlpineLinux (Buka Lubang): `iptables -A FORWARD -i eth4 -o eth0 -j ACCEPT`

Di Lubuntu-External-1 (Instalasi):

```
sudo apt update
sudo apt install openvpn
```

Di AlpineLinux (Tutup Lubang): `iptables -D FORWARD -i eth4 -o eth0 -j ACCEPT rc-service iptables save`

6.2 Transfer File Konfigurasi .ovpn

Di AlpineLinux (Mulai Server Web):

```
mkdir /tmp/vpn-keys
cp /etc/openvpn/easy-
rsa/pki/{ca.crt,issued/client1.crt,private/client1.key,ta.key}
/tmp/vpn-keys/
iptables -I INPUT -p tcp --dport 8000 -j ACCEPT
cd /tmp/vpn-keys
python3 -m http.server 8000
```

Di Lubuntu-External-1 (Unduh Kunci):

```
wget http://172.16.1.1:8000/ca.crt
wget http://172.16.1.1:8000/client1.crt
wget http://172.16.1.1:8000/client1.key
wget http://172.16.1.1:8000/ta.key
```

Di AlpineLinux: Hentikan server web (Ctrl+C), tutup port (`iptables -D INPUT -p tcp --dport 8000 -j ACCEPT`), dan simpan (`rc-service iptables save`).

6.3 Rakit client1.ovpn

Di Lubuntu-External-1: Buat nano client1.ovpn dengan template ini (Ganti YOUR_ALPINE_WAN_IP dengan IP eth0 Alpine, misal 192.168.122.x):

```
client
dev tun
proto udp
remote YOUR_ALPINE_WAN_IP 1194
float
resolv-retry infinite
nobind
persist-key
persist-tun
cipher AES-256-GCM
auth SHA256
verb 3
key-direction 1
```

Simpan, lalu jalankan 8 perintah cat ini untuk merakitnya:

```
echo "<ca>" >> client1.ovpn
cat ca.crt >> client1.ovpn
echo "</ca>" >> client1.ovpn
```

```
echo "<cert>" >> client1.ovpn
cat client1.crt >> client1.ovpn
echo "</cert>" >> client1.ovpn
```

```
echo "<key>" >> client1.ovpn
cat client1.key >> client1.ovpn
echo "</key>" >> client1.ovpn
```

```
echo "<tls-auth>" >> client1.ovpn;
cat ta.key >> client1.ovpn
echo "</tls-auth>" >> client1.ovpn
```

7 Demo

7.1 Persiapan

- Di UbuntuServer: Jalankan mkdir share lalu `python3 Server.py` Biarkan berjalan.
- Di Lubuntu-1: Buat file `echo "file LOKAL" > file_lokal.txt`
- Di Lubuntu-External-1: Buat file `echo "file EKSTERNAL" > file_eksternal.txt`

7.2 Demo 1: Akses Internal & NAT (di Lubuntu-1)

1. Aksi: Jalankan `ping 8.8.8.8 -c 4`
   - Hasil: Berhasil.
2. Aksi: Jalankan `python3 Client.py`
3. Aksi: `connect 192.168.10.130 50001`
   - Hasil: Berhasil.
4. Aksi: `upload file_lokal.txt`
   - Hasil: Berhasil. (Akses Internal Terbukti).

7.3 Demo 2: Blokir & Pengecualian Firewall (di Lubuntu-External-1)

5. Aksi: Jalankan `ping 192.168.20.104` (IP Lubuntu-1).
   - Hasil: GAGAL (Timeout). (Blokir Firewall Terbukti).
6. Aksi: Jalankan `ping 192.168.10.130` (IP Server).
   - Hasil: BERHASIL. (Pengecualian Ping Terbukti).
7. Aksi: Jalankan `python3 Client.py`
8. Aksi: `connect 192.168.10.130 50001`
   - Hasil: BERHASIL. (Pengecualian Port Aplikasi Terbukti).
9. Aksi: `list` (akan menampilkan file_lokal.txt).
10. Aksi: `upload file_eksternal.txt`
    - Hasil: Berhasil.

7.4 Demo 3: Demo VPN (Demo Terakhir)

1. Aksi: Tunjukkan kembali bahwa `ping 192.168.20.104` (ke Lokal) GAGAL.
2. Di AlpineLinux (Router & VPN Server): Pastikan layanan VPN menyala, dengan `rc-service openvpn restart`
3. Aksi (di Lubuntu-External-1): Jalankan koneksi VPN.

```
sudo openvpn --config client1.ovpn --daemon
```

4. Aksi: Tunjukkan `ip a` (akan menampilkan tun0 dengan IP 10.8.0.x).
   - Hasil: "Sekarang saya terhubung ke VPN."
5. Aksi: Jalankan ulang ping yang gagal tadi:

```
ping 192.168.20.104
```

   - Hasil: BERHASIL. (Modul VPN).