

Implementasi VPN WireGuard untuk Secure Remote Access

Perancangan Keamanan Sistem dan Jaringan

Dosen Pengampu: Ferdi Chahyadi, S.Kom., M.Cs



Anggota Kelompok:

2201020023	Mohd Allifyan Baitul Nesam
2201020048	Janumelah
2201020085	Safitri Wulandari
2201020108	Anjas Revaldo
2201020110	Danish Arya Yudhistira
2201020141	M. Aditya Egi Dwi Nata

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNIK DAN TEKNOLOGI KEMARITIMAN
UNIVERSITAS MARITIM RAJA ALI HAJI
2025/2026

DAFTAR ISI

DAFTAR ISI.....	2
DAFTAR GAMBAR.....	4
BAB I.....	6
PENDAHULUAN.....	6
1.1 Latar Belakang.....	6
1.2 Tujuan Proyek.....	7
1.3 Ruang Lingkup Pekerjaan	7
BAB II	8
LANDASAN TEORI.....	8
2.1 Keamanan Sistem dan Jaringan.....	8
2.2 Virtual Private Network (VPN)	8
2.3 WireGuard.....	8
2.4 Secure Remote Access.....	9
2.5 Enkripsi dan Tunneling.....	9
2.6 Wireshark	9
BAB III.....	10
METODOLOGI DAN PERANCANGAN SISTEM.....	10
3.1 Metode Penelitian.....	10
3.2 Lingkungan dan Topologi Jaringan	10
3.3 Spesifikasi Perangkat Keras dan Perangkat Lunak.....	11
3.3.1 Spesifikasi VPN Server	11
3.3.2 Spesifikasi VPN Client.....	12
3.4 Alur Kerja Sistem	12

BAB IV	14
IMPLEMENTASI DAN PENGUJIAN	14
4.1 Implementasi Instalasi Linux Server dan Client	14
4.2 Implementasi Instalasi WireGuard	16
4.3 Implementasi Membuat Sertifikat Kunci & Konfigurasi	19
4.4 Implementasi Pengujian Koneksi Titik-Ke-Titik.....	21
4.5 Implementasi Capture Wireshark Sebelum & Sesudah VPN	30
BAB V.....	36
SARAN DAN KESIMPULAN	36
5.1 KESIMPULAN.....	36
5.2 SARAN	36
DAFTAR PUSTAKA	38

DAFTAR GAMBAR

Gambar 1. Topologi Jaringan	10
Gambar 2. Spesifikasi VPN Server	11
Gambar 3. Spesifikasi VPN Client	12
Gambar 4. Instalasi sistem operasi Linux pada kedua VM	14
Gambar 5. Proses instalasi Ubuntu Desktop pada VM client.....	15
Gambar 6. Melakukan ping	15
Gambar 7. Perintah ping	16
Gambar 8. VM Server	17
Gambar 9. VM Client	17
Gambar 10. Instalasi paket WireGuard	18
Gambar 11. Terminal menunjukkan bahwa paket wireguard-tools berhasil dipasang.....	18
Gambar 12. Private key dan public key untuk mesin server	19
Gambar 13. Private key dan public key baru.....	20
Gambar 14. Struktur konfigurasi wg0.conf	20
Gambar 15. Berkas /etc/wireguard/wg0.conf	21
Gambar 16. Penerusan paket (IP forwarding).	21
Gambar 17. Menjalankan interface WireGuard pada server dan client	22
Gambar 18. Perintah: sudo wg-quick up wg0	22
Gambar 19. Verifikasi status WireGuard antar server	23
Gambar 20. Verifikasi status WireGuard antar client	24
Gambar 21. Pengujian ping antar IP VPN.....	24
Gambar 22. Pengujian ping antar IP VPN.....	25
Gambar 23. Verifikasi IP routing pada client.....	25
Gambar 24. Verifikasi alamat IP interface WireGuard	26
Gambar 25. Verifikasi alamat IP interface WireGuard	27
Gambar 26. Pengujian handshake berulang	28
Gambar 27. Pengujian handshake berulang	28
Gambar 28. Pengujian akses jarak jauh (remote access) melalui SSH	29

Gambar 29. Melakukan ping antar mesin sebelum VPN	30
Gambar 30. Melakukan ping antar mesin sebelum VPN	31
Gambar 31. Membuka aplikasi Wireshark untuk melakukan packet capture	31
Gambar 32. Capture lalu lintas jaringan sebelum VPN	32
Gambar 33. Melakukan verifikasi handshake VPN WireGuard	32
Gambar 34. Komunikasi VPN sudah aktif.	33
Gambar 35. Melakukan verifikasi routing di Client.....	33
Gambar 36. Melakukan ping IP VPN dari Server	34
Gambar 37. Capture lalu lintas jaringan setelah VPN.....	34

BAB I

PENDAHULUAN

1.1 Latar Belakang

Perkembangan pesat teknologi informasi dan jaringan komputer telah meningkatkan permintaan akan akses jarak jauh. Banyak tugas administrasi sistem, transfer data, dan pengelolaan layanan jaringan kini dilakukan tanpa kehadiran fisik. Namun, akses jarak jauh melalui jaringan publik seperti internet menimbulkan berbagai risiko keamanan, termasuk penyadapan data, pencurian informasi, manipulasi paket, dan serangan man-in-the-middle.

Pada jaringan tanpa mekanisme keamanan, data yang dikirimkan antara perangkat tidak dienkripsi. Hal ini memungkinkan pihak yang tidak berwenang untuk menyadap dan membaca komunikasi menggunakan alat pemantauan jaringan seperti packet sniffers. Oleh karena itu, mekanisme perlindungan diperlukan untuk memastikan kerahasiaan, integritas, dan keaslian data selama transmisi.

Virtual Private Network (VPN) adalah solusi keamanan jaringan yang umum digunakan untuk masalah ini. VPN menciptakan terowongan komunikasi terenkripsi melalui jaringan publik, memastikan bahwa data yang dikirim hanya dapat dibaca oleh individu yang memiliki kunci otentikasi yang valid. Dengan VPN, pengguna dapat mengakses jaringan internal secara aman seolah-olah mereka terhubung langsung ke jaringan lokal.

WireGuard adalah protokol VPN modern yang dirancang untuk kesederhanaan, kinerja tinggi, dan keamanan yang tangguh. Berbeda dengan protokol VPN tradisional, WireGuard menggunakan algoritma kriptografi terbaru dan memiliki pengaturan yang lebih sederhana, sehingga implementasi dan pemeliharannya lebih mudah. Selain itu, kode sumber WireGuard yang ringkas memudahkan audit keamanan.

Sebagai bagian dari mata kuliah “Desain Keamanan Jaringan dan Sistem”, proyek ini bertujuan untuk mengimplementasikan VPN WireGuard sebagai solusi

akses jarak jauh yang aman dalam lingkungan Linux. Implementasi dilakukan menggunakan dua mesin virtual, satu bertindak sebagai server dan yang lain sebagai klien. Uji sistem dilakukan melalui uji koneksi, akses jarak jauh, dan analisis lalu lintas jaringan menggunakan Wireshark untuk menunjukkan bahwa komunikasi dienkripsi dan oleh karena itu lebih aman daripada tanpa VPN.

1.2 Tujuan Proyek

Tujuan dari proyek ini adalah:

1. Mengimplementasikan VPN WireGuard pada lingkungan Linux server dan client.
2. Menguji konektivitas jaringan melalui tunnel VPN yang aman.
3. Membuktikan adanya enkripsi lalu lintas jaringan menggunakan packet capture.
4. Menerapkan konsep keamanan jaringan yang dipelajari secara praktis.

1.3 Ruang Lingkup Pekerjaan

Ruang lingkup proyek ini meliputi:

- Implementasi VPN WireGuard pada dua mesin Linux.
- Pengujian koneksi titik-ke-titik (point-to-point).
- Pengujian secure remote access menggunakan SSH.
- Analisis lalu lintas jaringan sebelum dan sesudah VPN.

BAB II

LANDASAN TEORI

2.1 Keamanan Sistem dan Jaringan

Keamanan jaringan adalah upaya untuk melindungi data saat transit di jaringan dari ancaman seperti pencurian, penyadapan, dan manipulasi data. Konsep keamanan jaringan meliputi kerahasiaan (confidentiality), integritas (integrity), dan ketersediaan (availability) data. Analisis lalu lintas jaringan seperti packet sniffing menunjukkan bahwa jaringan tanpa pengamanan mudah terekspos oleh pihak tidak berwenang menggunakan tools network analyzer seperti Wireshark (Muhammad Agus Darmawan et al., 2023).

2.2 Virtual Private Network (VPN)

Virtual Private Network (VPN) adalah teknologi yang membentuk koneksi privat melalui jaringan publik dengan menerapkan teknik tunneling dan enkripsi, sehingga data yang ditransmisikan lebih aman dibandingkan koneksi langsung tanpa VPN. Dalam banyak skenario, VPN digunakan untuk secure remote access, yaitu akses jarak jauh yang aman dari luar jaringan internal. Penelitian menunjukkan bahwa implementasi VPN modern seperti WireGuard dapat meningkatkan keamanan data dibandingkan jaringan tanpa VPN karena penggunaan kriptografi yang lebih ringkas dan efisien (Phạm Anh Thu., 2025).

2.3 WireGuard

WireGuard adalah protokol VPN generasi baru yang dirancang dengan basis kriptografi modern dan arsitektur yang lebih sederhana dibandingkan protokol VPN tradisional seperti IPsec atau OpenVPN. Dalam *A WireGuard Exploration*, dijelaskan bahwa WireGuard menggunakan pendekatan pertukaran kunci berbasis public key yang efisien dan memudahkan implementasi server–client tanpa konfigurasi rumit, serta mampu menjamin keamanan komunikasi data (Master, A., & Garman, C., 2021).

Selain itu, penelitian lain menunjukkan mekanisme keamanan WireGuard dalam mengamankan trafik VPN dengan statistik QoS yang baik dalam uji VoIP, serta performa latency yang lebih rendah dibandingkan beberapa protokol lain, sehingga cocok diterapkan pada berbagai kebutuhan jaringan yang memerlukan keamanan dan efisiensi (Rahman, I. K., & Harnaningrum, L. N., 2024).

2.4 Secure Remote Access

Secure remote access adalah teknik untuk menghubungkan perangkat atau pengguna dari lokasi berbeda ke jaringan internal secara aman. Tanpa pengamanan, akses ini rentan terhadap serangan seperti *man-in-the-middle* atau penyadapan. VPN menjadi solusi umum untuk masalah ini karena menyediakan jalur komunikasi terenkripsi yang memberikan lapisan proteksi terhadap data yang dikirimkan melalui internet.

2.5 Enkripsi dan Tunneling

Enkripsi mengubah data asli (plaintext) menjadi bentuk yang tidak dapat dibaca tanpa kunci tertentu, sehingga hanya pihak yang berwenang yang dapat melihat isi data. WireGuard menggunakan kriptografi modern yang efisien untuk mengamankan data ini. Tunneling dalam konteks VPN adalah teknik pembungkusan paket data asli ke dalam format yang aman untuk ditransmisikan melalui jaringan publik.

2.6 Wireshark

Wireshark adalah tools yang digunakan untuk melakukan packet capture serta analisis terhadap lalu lintas jaringan. Tools ini memungkinkan pengamatan paket sebelum dan sesudah enkripsi VPN. Penelitian menggunakan Wireshark untuk *packet sniffing* menunjukkan bahwa tanpa proteksi jaringan yang memadai, paket data dapat dilihat secara jelas oleh analis jaringan, sedangkan penggunaan VPN akan mengubah tampilan paket menjadi bentuk terenkripsi yang tidak mudah dibaca (Muhammad Agus Darmawan et al., 2023).

BAB III

METODOLOGI DAN PERANCANGAN SISTEM

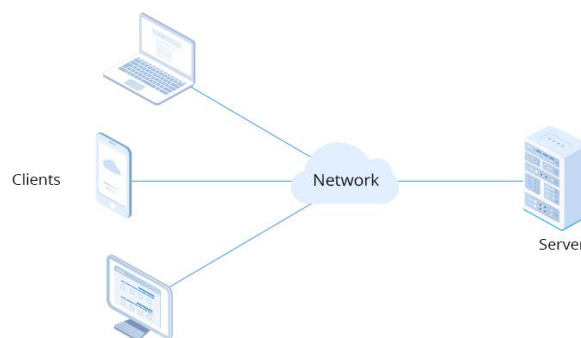
3.1 Metode Penelitian

Metode penelitian yang digunakan dalam laporan ini adalah metode eksperimen. Metode eksperimen dilakukan dengan cara membangun sebuah sistem, mengimplementasikan konfigurasi keamanan jaringan, serta melakukan pengujian secara langsung terhadap sistem yang telah dibuat.

Pada proyek ini, eksperimen dilakukan dengan mengimplementasikan VPN WireGuard pada dua mesin Linux yang berperan sebagai server dan client. Setiap tahap eksperimen diamati dan diuji untuk memastikan bahwa sistem berjalan sesuai dengan tujuan, yaitu menyediakan akses jarak jauh yang aman (secure remote access).

3.2 Lingkungan dan Topologi Jaringan

Lingkungan pengujian dibuat menggunakan VMware Workstation. Pada tahap ini, disiapkan dua mesin virtual dengan peran yang berbeda: satu sebagai server VPN dan yang lainnya sebagai klien VPN. Kedua mesin virtual dikonfigurasi dalam mode jaringan NAT untuk terhubung ke jaringan melalui host. Topologi jaringan yang digunakan adalah topologi client-server, di mana klien terhubung ke server melalui terowongan VPN WireGuard. Adapun gambaran umum topologi jaringan yang digunakan dalam proyek ini ditampilkan pada gambar berikut.

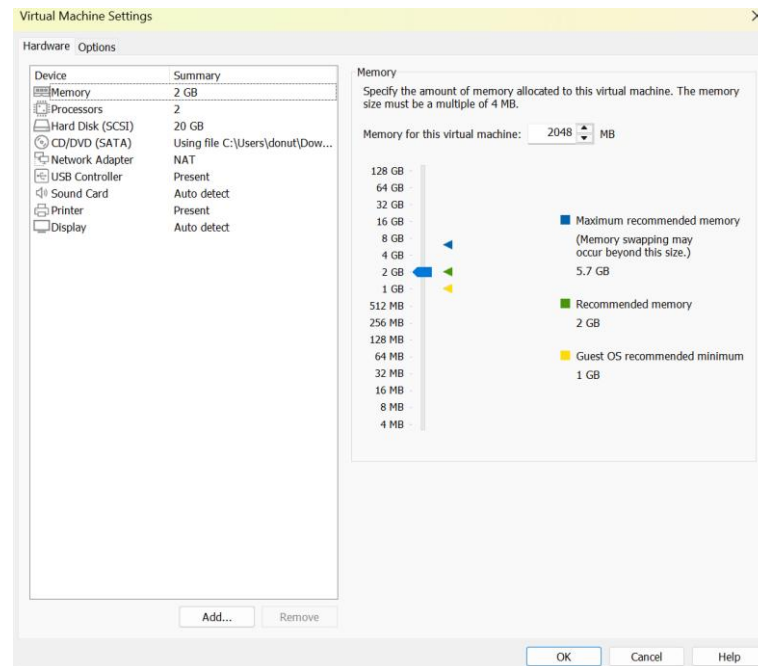


Gambar 1. Topologi Jaringan

3.3 Spesifikasi Perangkat Keras dan Perangkat Lunak

Spesifikasi perangkat keras dan perangkat lunak ditentukan agar sistem dapat berjalan stabil selama proses implementasi dan pengujian.

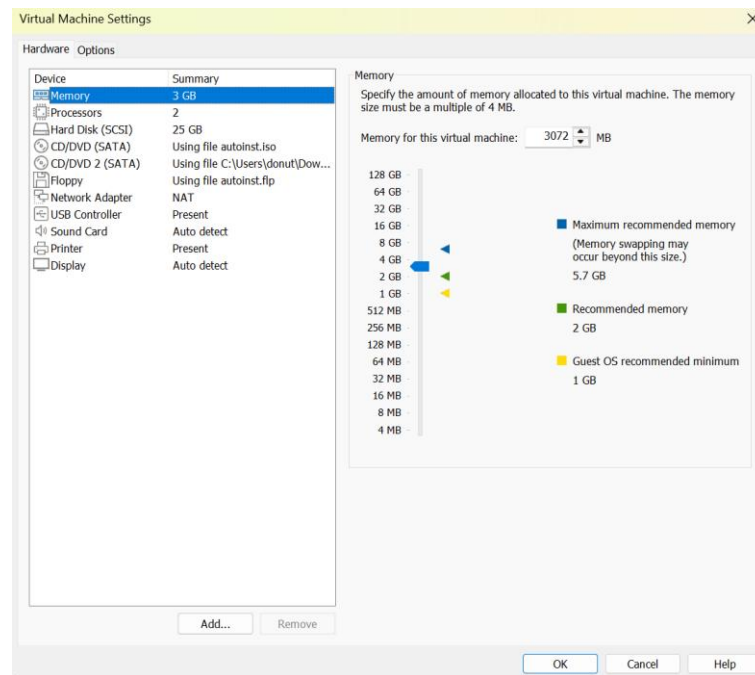
3.3.1 Spesifikasi VPN Server



Gambar 2. Spesifikasi VPN Server

- Sistem Operasi: Ubuntu Server 24.04 LTS
- Memory (RAM): 2 GB
Jumlah memori yang cukup untuk menjalankan layanan dasar Ubuntu Server.
- Processors: 2 core
Memberikan performa yang lebih stabil untuk proses server.
- Hard Disk: 20 GB
Kapasitas penyimpanan minimal untuk sistem operasi dan instalasi layanan tambahan.
- Network Adapter: NAT
Menggunakan mode NAT agar server dapat terkoneksi dengan jaringan melalui IP host.
- Perangkat Lunak Utama: WireGuard

3.3.2 Spesifikasi VPN Client



Gambar 3. Spesifikasi VPN Client

Pada gambar di atas, VM Client mendapatkan spesifikasi yang sedikit lebih tinggi karena menggunakan Ubuntu Desktop dengan antarmuka grafis (GUI):

- Sistem Operasi: Ubuntu Desktop 24.04 LTS
- Memory (RAM): 3 GB
Menyediakan memori tambahan untuk mendukung lingkungan desktop.
- Processors: 2 core
Memberikan kelancaran saat menjalankan aplikasi GUI.
- Hard Disk: 25 GB
Kapasitas lebih besar untuk menampung aplikasi dan kebutuhan pengguna.
- Network Adapter: NAT
Memungkinkan client memperoleh akses jaringan melalui host.
- Perangkat Lunak Pendukung: WireGuard dan Wireshark.

3.4 Alur Kerja Sistem

Alur kerja sistem VPN WireGuard disusun berdasarkan tahapan implementasi yang dilakukan setiap minggu. Alur ini dimulai dari persiapan

lingkungan hingga tahap analisis keamanan jaringan. Secara umum, alur kerja sistem adalah sebagai berikut:

1. Pembuatan dan konfigurasi mesin virtual (Minggu ke-1).
2. Instalasi sistem operasi Linux (Minggu ke-1).
3. Instalasi paket WireGuard (Minggu ke-2).
4. Pembuatan pasangan kunci dan konfigurasi WireGuard (Minggu ke-3).
5. Pengujian koneksi VPN dan secure remote access (Minggu ke-4).
6. Analisis lalu lintas jaringan menggunakan Wireshark (Minggu ke-5).

Alur kerja ini memastikan bahwa setiap tahapan saling berkesinambungan dan dapat dievaluasi secara bertahap.

BAB IV

IMPLEMENTASI DAN PENGUJIAN

4.1 Implementasi Instalasi Linux Server dan Client

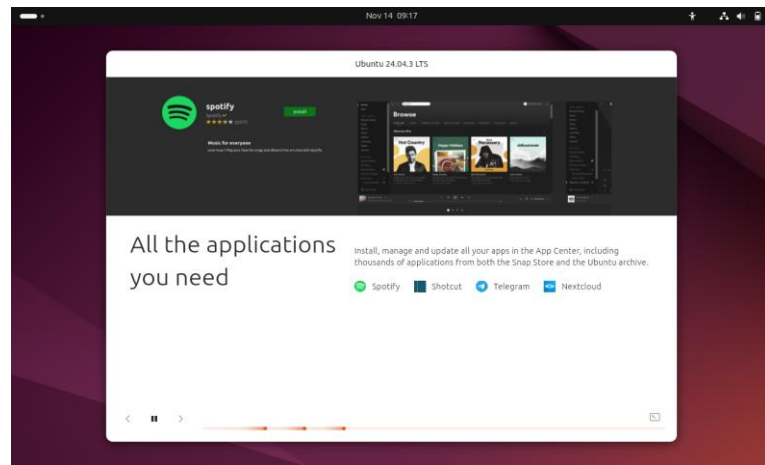
1. Melakukan instalasi sistem operasi Linux pada kedua VM

```
start: subiquity/install/install/curtin_install/run_curtin_step: executing curtin install extract step
start: subiquity/install/install/curtin_install/run_curtin_step/cmd-install/curl: curl command install
start: subiquity/install/install/curtin_install/run_curtin_step/cmd-install/stage-extract: writing install sources to disk
start: subiquity/install/install/curtin_install/run_curtin_step/cmd-install/stage-extract/built-in: running 'curtin extract'
start: subiquity/install/install/curtin_install/run_curtin_step/cmd-install/stage-extract/built-in/cde-extract: curtin command extract
start: subiquity/install/install/curtin_install/run_curtin_step/cmd-install/stage-extract/built-in/cde-extract/: acquiring and extracting image from cpi://
/ftp/freedom23/iso/
finish: subiquity/install/install/curtin_install/run_curtin_step/cmd-install/stage-extract/built-in/cde-extract/: acquiring and extracting image from cpi://
/ftp/freedom23/iso/
finish: subiquity/install/install/curtin_install/run_curtin_step/cmd-install/stage-extract/built-in: running 'curtin extract'
finish: subiquity/install/install/curtin_install/run_curtin_step/cmd-install/stage-extract: writing install sources to disk
finish: subiquity/install/install/curtin_install/run_curtin_step/cmd-install: curtin command install
start: subiquity/install/install/curtin_install/run_curtin_step: executing curtin install extract step
start: subiquity/install/install/curtin_install/setup_target/cmd-in-target: curtin command in-target
start: subiquity/install/install/curtin_install/setup_target/cmd-in-target: curtin command in-target
start: subiquity/install/install/curtin_install/run_curtin_step: executing curtin install curthooks step
start: subiquity/install/install/curtin_install/run_curtin_step/cmd-install/stage-curthooks: configuring installed system
start: subiquity/install/install/curtin_install/run_curtin_step/cmd-install/stage-curthooks/built-in: running 'curtin curthooks'
start: subiquity/install/install/curtin_install/run_curtin_step/cmd-install/stage-curthooks/built-in/cde-curthooks: curtin command curthooks
start: subiquity/install/install/curtin_install/run_curtin_step/cmd-install/stage-curthooks/built-in/cde-curthooks: writing apt-config: configuring apt co
nfiguring apt co
finish: subiquity/install/install/curtin_install/run_curtin_step/cmd-install/stage-curthooks/built-in/cde-curthooks: writing apt-config: configuring apt co
nfiguring apt co
start: subiquity/install/install/curtin_install/run_curtin_step/cmd-install/stage-curthooks/built-in/cde-curthooks: installing-missing-packages: installing
missing packages
start: subiquity/install/install/curtin_install/run_curtin_step/cmd-install/stage-curthooks/built-in/cde-curthooks/: Installing packages on target system
finish: subiquity/install/install/curtin_install/run_curtin_step/cmd-install/stage-curthooks/built-in/cde-curthooks/: Installing packages on target system
start: subiquity/install/install/curtin_install/run_curtin_step/cmd-install/stage-curthooks/built-in/cde-curthooks: installing-missing-packages: installing
missing packages
finish: subiquity/install/install/curtin_install/run_curtin_step/cmd-install/stage-curthooks/built-in/cde-curthooks: installing-missing-packages: installing
missing packages
start: subiquity/install/install/curtin_install/run_curtin_step/cmd-install/stage-curthooks/built-in/cde-curthooks: configuring-lscsi-service: configuring
lscsi service
finish: subiquity/install/install/curtin_install/run_curtin_step/cmd-install/stage-curthooks/built-in/cde-curthooks: configuring-lscsi-service: configuring
lscsi service
start: subiquity/install/install/curtin_install/run_curtin_step/cmd-install/stage-curthooks/built-in/cde-curthooks: configuring-udm-service: configuring
udm (udm) service
finish: subiquity/install/install/curtin_install/run_curtin_step/cmd-install/stage-curthooks/built-in/cde-curthooks: configuring-udm-service: configuring
udm (udm) service
start: subiquity/install/install/curtin_install/run_curtin_step/cmd-install/stage-curthooks/built-in/cde-curthooks: configuring-nm-over-top: configuring
nm over top
finish: subiquity/install/install/curtin_install/run_curtin_step/cmd-install/stage-curthooks/built-in/cde-curthooks: configuring-nm-over-top: configuring
nm over top
start: subiquity/install/install/curtin_install/run_curtin_step/cmd-install/stage-curthooks/built-in/cde-curthooks: installing-kernel: installing kernel
start: subiquity/network/send_update: DMDE2: emp03
finish: subiquity/network/send_update: DMDE2: emp03
```

Gambar 4. Instalasi sistem operasi Linux pada kedua VM

Pada gambar pertama, instalasi Ubuntu Server dilakukan melalui mode teks (terminal). Tahapan yang terlihat pada screenshot antara lain:

- Ekstraksi file instalasi ke disk VM.
- Instalasi paket inti (core system packages).
- Konfigurasi layanan sistem seperti network service.
- Instalasi kernel Linux.
- Persiapan sistem untuk reboot pertama setelah instalasi selesai.



Gambar 5. Proses instalasi Ubuntu Desktop pada VM client.

Gambar di atas menunjukkan proses instalasi Ubuntu Desktop pada VM client. Tampilan installer menunjukkan:

- a. Pengenalan fitur-fitur Ubuntu (App Center, aplikasi-aplikasi populer).
 - b. Proses instalasi berjalan di background.
 - c. Menunggu hingga instalasi sistem selesai sepenuhnya.
2. Melakukan ping pada kedua sistem

```

Welcome to Ubuntu 24.04.3 LTS (GNU/Linux 6.8.0-52-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

System information as of Fri Nov 14 09:07:58 AM UTC 2025

System load:          0.02
Usage of /:           10.5% of 24.44GB
Memory usage:         7%
Swap usage:           0%
Processes:            32
Users logged in:      0
IPV4 address for eth0: 10.0.2.15
IPv6 address for eth0: fd17:6252:1037::2:800:27ff:fe01:1c05
IPv6 address for eth0: fd17:6252:1037::2:800:27ff:fe01:1c05

Expanded Security Maintenance for Applications is not enabled.

No updates can be applied immediately.
To see these additional updates run: apt list --upgradable

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/*-copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo command".
See "man sudo_root" for details.

ubuntu@servervm:~$ ping 0.0.0.0
PING 0.0.0.0 (0.0.0.0) 56(84) bytes of data:
64 bytes from 0.0.0.0: icmp_seq=1 ttl=255 time=10.5 ms
64 bytes from 0.0.0.0: icmp_seq=2 ttl=255 time=7.32 ms
64 bytes from 0.0.0.0: icmp_seq=3 ttl=255 time=6.7 ms
64 bytes from 0.0.0.0: icmp_seq=4 ttl=255 time=0.44 ms
64 bytes from 0.0.0.0: icmp_seq=5 ttl=255 time=10.2 ms
64 bytes from 0.0.0.0: icmp_seq=6 ttl=255 time=7.22 ms

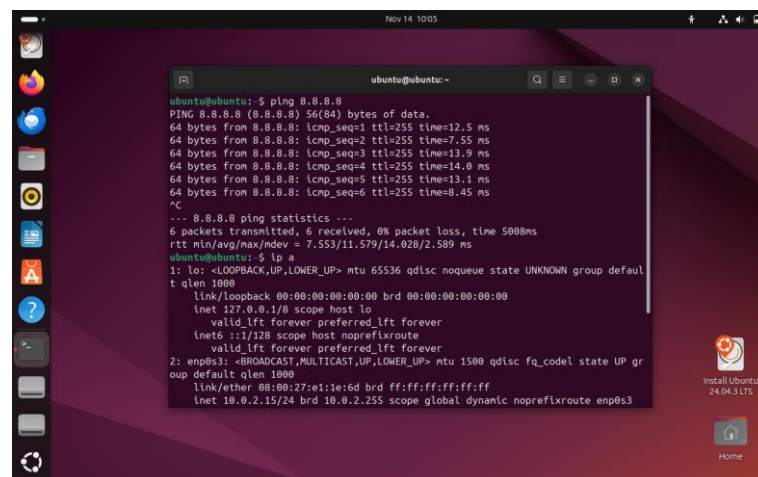
--- 0.0.0.0 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5006ms
rtt min/avg/max/mdev = 7.216/14.097/45.710/13.030 ms
ubuntu@servervm:~$

```

Gambar 6. Melakukan ping

Pada gambar di atas terlihat bahwa perintah ping 8.8.8.8 dijalankan pada sistem pertama. Hasil pengujian menunjukkan adanya reply dari host tujuan. Hal ini menandakan bahwa:

- a. Sistem pertama sudah memperoleh konfigurasi jaringan yang valid.
- b. Sistem dapat berkomunikasi keluar jaringan (akses internet tersedia).
- c. Tidak terdapat packet loss selama pengujian.



```
ubuntu@ubuntu:~$ ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data:
64 bytes from 8.8.8.8: icmp_seq=1 ttl=255 time=12.5 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=255 time=7.55 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=255 time=13.9 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=255 time=14.0 ms
64 bytes from 8.8.8.8: icmp_seq=5 ttl=255 time=13.1 ms
64 bytes from 8.8.8.8: icmp_seq=6 ttl=255 time=8.45 ms
^C
--- 8.8.8.8 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 500ms
rtt min/avg/max/ndev = 7.553/11.579/14.028/2.589 ms
ubuntu@ubuntu:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
        valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:e1:1e:6d brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.15/24 brd 10.0.2.255 scope global dynamic noprefixroute enp0s3
```

Gambar 7. Perintah ping

Pada gambar di atas, sistem kedua juga menjalankan perintah ping 8.8.8.8 dan memperoleh hasil reply yang sama. Setelah pengujian ping, ditampilkan pula hasil perintah ip a yang menunjukkan:

- a. Interface jaringan dalam kondisi aktif.
- b. Sistem telah menerima alamat IP secara dinamis.
- c. Rute jaringan dan konfigurasi interface berfungsi dengan baik.

4.2 Implementasi Instalasi WireGuard

Kemudian dilakukan proses instalasi paket VPN WireGuard pada kedua mesin Linux yang telah disiapkan pada minggu sebelumnya. Tujuan tahap ini adalah memastikan bahwa software WireGuard berhasil terpasang dan siap digunakan untuk proses konfigurasi pada minggu selanjutnya. Tahapan dan Proses:

1. Menginstal paket WireGuard pada kedua mesin Linux

Pada tahap ini dilakukan proses instalasi paket WireGuard pada kedua mesin virtual, yaitu Ubuntu Server dan Ubuntu Client. Instalasi dilakukan melalui terminal menggunakan perintah `sudo apt install wireguard -y`.

a. VM Server

```
ubuntu@24.04.3 LTS:~$ sudo apt install wireguard -y
[sudo] password for servervm:
ServerVM login: servervm
Welcome to Ubuntu 24.04.3 LTS (GNU/Linux 6.8.0-47-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://canonical.com/realtime
 * Support:        https://ubuntu.com/support

System information as of Wed Nov 29 05:41:58 PM UTC 2023

System load:  0.7%           Processes:           244
Usage of /:    95.7% of 8.76GB    Memory usage:      0%
Swap usage:    0%             IPv4 address for eth0: 192.168.63.109
Disk usage:    48%

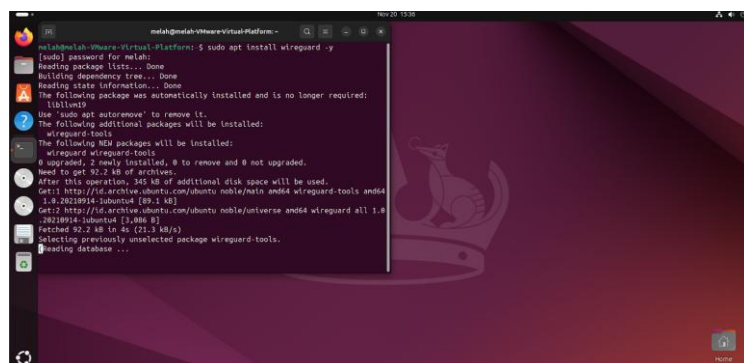
Extended Security Maintenance for Applications is not enabled.
# updates can be applied immediately.
Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo apt status

servervm@servervm:~$ sudo apt install wireguard -y
```

Gambar 8. VM Server

Pada VM Server, perintah tersebut dijalankan pada lingkungan terminal berbasis teks (CLI). Sistem kemudian mengunduh dan memasang paket WireGuard beserta dependensinya untuk menyiapkan fungsi VPN pada server.

b. VM Client



Gambar 9. VM Client

Pada VM Client, instalasi dilakukan melalui terminal di lingkungan Ubuntu Desktop. Prosesnya serupa, sistem melakukan

pembaruan daftar paket dan menginstal komponen wireguard dan wireguard-tools yang dibutuhkan untuk membuat koneksi ke server.

2. Memverifikasi keberhasilan instalasi WireGuard

```

The following NEW packages will be installed:
  wireguard wireguard-tools
0 upgraded, 2 newly installed, 0 to remove and 0 not upgraded.
Need to get 50.5 kB of archives.
After this operation, 345 kB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu noble/main amd64 wireguard-tools amd64 1.0.20210914-1ubuntu4 [69.1 kB]
Get:2 http://archive.ubuntu.com/ubuntu noble/universe amd64 wireguard all 1.0.20210914-1ubuntu4 [3.696 B]
Fetched 72.8 kB in 0s (43.5 kB/s)
Selecting previously unselected package wireguard-tools.
(Reading database ... 579477 files and directories currently installed.)
Preparing to unpack .../wireguard-tools-1.0.20210914-1ubuntu4.deb ...
Unpacking wireguard-tools (1.0.20210914-1ubuntu4) ...
Setting up previously unselected package wireguard.
Preparing to unpack .../wireguard-1.0.20210914-1ubuntu4_all.deb ...
Unpacking wireguard (1.0.20210914-1ubuntu4) ...
Setting up wireguard-tools (1.0.20210914-1ubuntu4) ...
wg-quick.target is disabled on a static unit, not starting it.
Setting up wireguard (1.0.20210914-1ubuntu4) ...
Processing triggers for man-db (2.12.0-4ubuntu1) ...
Scanning processes...
Scanning Linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
servervm@servervm:~$ wg --version
wireguard-tools v1.0.20210914 - https://git.z3dca.com/wireguard-tools/
servervm@servervm:~$ dkg --l | grep wireguard
dkg: error: unknown option -l

Type dkg --help for help about installing and deinstalling packages [4]:
Type wg --getconf for user-friendly package management.
Type dkg --help for a list of dkg debug flag values.
Type dkg --force for a list of forcing options.
Type dkg debug --help for help about manipulating .deb files.

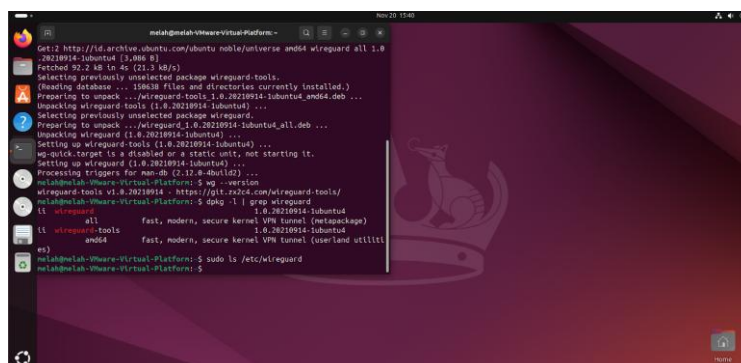
Options marked [a] produce a lot of output - pipe it through 'less' or 'more' !
servervm@servervm:~$ dkg --l | grep wireguard
11 wireguard      1.0.20210914-1ubuntu4      all          fast, modern, secure kernel VPN tunnel (metapackage)
servervm@servervm:~$ dkg --l | grep wireguard
11 wireguard      1.0.20210914-1ubuntu4      all          fast, modern, secure kernel VPN tunnel (metapackage)

```

Gambar 10. Instalasi paket WireGuard

Pada tahap ini, sistem melakukan proses instalasi paket WireGuard untuk VM Server dan dependensinya. Setelah proses instalasi selesai:

- Terminal menunjukkan bahwa paket wireguard-tools berhasil dipasang.
- Kernel modules yang dibutuhkan (seperti wireguard ataupun wg) telah dimuat.
- Sistem siap digunakan tanpa membutuhkan reboot.



Gambar 11. Terminal menunjukkan bahwa paket wireguard-tools berhasil dipasang.

Di sisi client, langkah verifikasi dilakukan dengan cara:

- a. Menjalankan perintah `sudo apt install wireguard` dan memastikan proses instalasi berjalan sukses.
- b. Mengecek apakah direktori konfigurasi `/etc/wireguard` sudah tersedia menggunakan `sudo ls /etc/wireguard`.
- c. Jika folder tersebut muncul (meski kosong), berarti instalasi WireGuard berhasil dan sistem siap untuk pembuatan file konfigurasi.

4.3 Implementasi Membuat Sertifikat Kunci & Konfigurasi

Pada minggu 3 dilakukan proses pembuatan pasangan kunci (key pair) serta penyusunan berkas konfigurasi awal WireGuard pada kedua mesin Linux. Tujuan dari tahap ini adalah menyiapkan identitas kriptografi server dan client sehingga keduanya dapat saling mengenali serta membentuk kanal komunikasi terenkripsi pada tahap pengujian minggu berikutnya.

Tahapan dan Proses:

1. Pembuatan key pair pada server dan client

```

Ubuntu 24.04.3 LTS servervm tty1
servervm login: servervmvm
Password:
Welcome to Ubuntu 24.04.3 LTS (GNU/Linux 6.8.0-87-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

System information as of Fri Nov 28 03:04:25 PM UTC 2025

System load: 0,29          Processes:      241
Usage of /:  45,7% of 9,75GB Users logged in:  0
Memory usage: 12%          IPv4 address for ens33: 192.168.63.128
Swap usage:  0%

Expanded Security Maintenance for Applications is not enabled.
0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

servervmvm@servervm:~$ wg genkey | tee server_private.key | wg pubkey > server_public.key
servervmvm@servervm:~$ cat server_private.key
oGR6sQdML4D6xQBYy+IRHB4XNMWVRgST5SHu40Jedck=
servervmvm@servervm:~$ cat server_public.key
Ddq9Ccx/MTEQ/sk0N3M7b793+st8LA1QY41jvuCfag8=
servervmvm@servervm:~$ _

```

Gambar 12. Private key dan public key untuk mesin server

Pada tahap ini dihasilkan private key dan public key untuk mesin server. Private key digunakan untuk autentikasi server, sedangkan public key nantinya dibagikan kepada client agar perangkat dapat melakukan

handshake

secara

aman.

```
melah@melah-VMware-Virtual-Platform:~$ wg genkey | tee client_private.key | wg p
ubkey > client_public.key
melah@melah-VMware-Virtual-Platform:~$ cat client_private.key
yMYofAzpVUY5elWy3UBCV1UMML090kJQlnpJlF/xjnA=
melah@melah-VMware-Virtual-Platform:~$ cat client_public.key
ktwbFhRtJz2+eg+PjjSh/5Lbj3W0cAfHHgCs9+cvqG4=
melah@melah-VMware-Virtual-Platform:~$
```

Gambar 13. Private key dan public key baru

Proses serupa dilakukan pada mesin client, yaitu menghasilkan private key dan public key baru. Key pair ini diperlukan agar server dapat mengenali identitas client saat proses koneksi VPN dilakukan.

2. Penyusunan konfigurasi WireGuard pada server dan client

Struktur konfigurasi wg0.conf terdiri dari [Interface] untuk pengaturan identitas, alamat IP, dan port dari perangkat dan [Peer] untuk informasi tentang perangkat tujuan dan IP mana saja yang boleh diarahkan melalui tunnel.

```
GNU nano 7.2 /etc/wireguard/wg0.conf
[Interface]
Address = 10.8.0.1/24
PrivateKey = oGR6SgdmL4D6xQBYy+1RHB4XNMvXRgST59Hu0je2Gk=
ListenPort = 51820

[Peer]
PublicKey = ktwbFhRtJz2+eg+PjjSh/5Lbj3W0cAfHHgCs9+cvqG4=
AllowedIPs = 10.8.0.2/32
```

Gambar 14. Struktur konfigurasi wg0.conf

Pada gambar di atas sebuah berkas konfigurasi /etc/wireguard/wg0.conf dibuat berisi:

- Alamat IP virtual server (10.8.0.1/24).
- Private key server.
- Port yang digunakan (51820).
- Serta peer client yang ditambahkan setelah client selesai dikonfigurasi.



Gambar 15. Berkas /etc/wireguard/wg0.conf

Pada gambar di atas, client juga dibuatkan berkas /etc/wireguard/wg0.conf yang memuat:

- a. Alamat IP virtual client (10.8.0.2/24).
- b. Private key client.
- c. Informasi public key server.
- d. Serta endpoint IP server untuk tujuan koneksi.

3. Pengaktifan IP forwarding pada server

```
servervm@servervm:~$ echo "net.ipv4.ip_forward=1" | sudo tee -a /etc/sysctl.conf
sudo sysctl -p
net.ipv4.ip_forward=1
net.ipv4.ip_forward = 1
servervm@servervm:~$
```

Gambar 16. Penerusan paket (IP forwarding).

Server diatur untuk mengizinkan penerusan paket (IP forwarding). Pengaturan ini diperlukan agar server mampu meneruskan lalu lintas jaringan dari client ke jaringan lain melalui interface VPN.

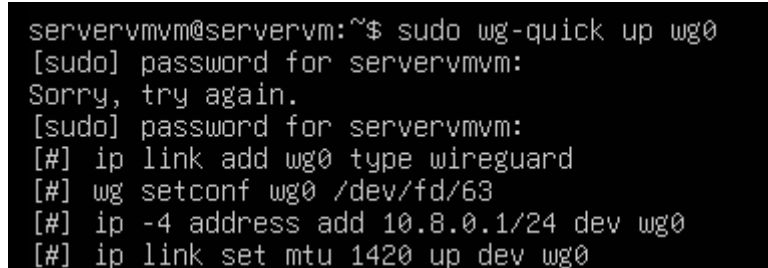
4.4 Implementasi Pengujian Koneksi Titik-Ke-Titik

Pada minggu 4 dilakukan proses implementasi konfigurasi VPN yang telah disusun pada minggu sebelumnya. Tujuan dari tahap ini adalah memastikan bahwa koneksi terowongan (tunnel) WireGuard dapat berjalan dengan baik, ditandai dengan keberhasilan handshake antara server dan client, serta kemampuan kedua

mesin untuk saling berkomunikasi melalui jaringan VPN.

Tahapan dan Proses:

1. Menjalankan interface WireGuard pada server dan client



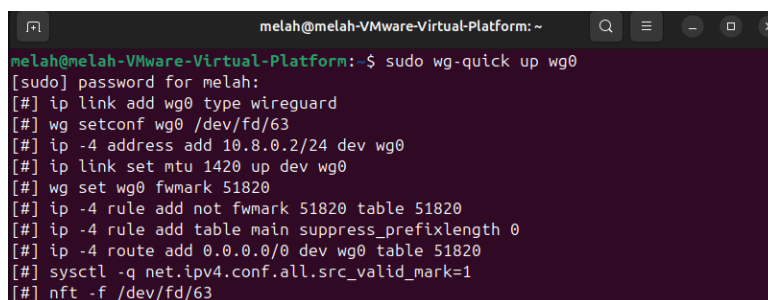
```
servervmvm@servervm:~$ sudo wg-quick up wg0
[sudo] password for servervmvm:
Sorry, try again.
[sudo] password for servervmvm:
[#] ip link add wg0 type wireguard
[#] wg setconf wg0 /dev/fd/63
[#] ip -4 address add 10.8.0.1/24 dev wg0
[#] ip link set mtu 1420 up dev wg0
```

Gambar 17. Menjalankan interface WireGuard pada server dan client

Perintah: `sudo wg-quick up wg0`

Analisis pengujian server:

- a. Sistem berhasil membuat interface baru bernama wg0 dengan tipe WireGuard.
- b. Konfigurasi dimuat dari file konfigurasi sistem.
- c. IP Address 10.8.0.1/24 berhasil dipasang (assigned) pada interface wg0. Ini bertindak sebagai alamat gateway untuk jaringan VPN.
- d. MTU (Maximum Transmission Unit) diatur ke 1420 byte untuk mengakomodasi header enkripsi WireGuard tanpa menyebabkan fragmentasi paket yang berlebihan.



```
melah@melah-VMware-Virtual-Platform: ~$ sudo wg-quick up wg0
[sudo] password for melah:
[#] ip link add wg0 type wireguard
[#] wg setconf wg0 /dev/fd/63
[#] ip -4 address add 10.8.0.2/24 dev wg0
[#] ip link set mtu 1420 up dev wg0
[#] wg set wg0 fwmark 51820
[#] ip -4 rule add not fwmark 51820 table 51820
[#] ip -4 rule add table main suppress_prefixlength 0
[#] ip -4 route add 0.0.0.0/0 dev wg0 table 51820
[#] sysctl -q net.ipv4.conf.all.src_valid_mark=1
[#] nft -f /dev/fd/63
```

Gambar 18. Perintah: `sudo wg-quick up wg0`

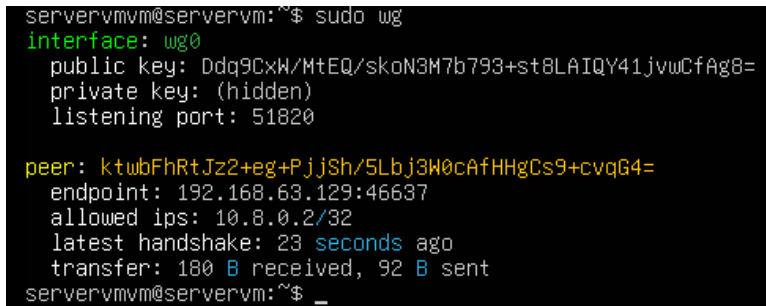
Perintah: `sudo wg-quick up wg0`

Analisis pengujian client:

- a. Sama seperti server, client berhasil membuat interface wg0.
- b. IP Address 10.8.0.2/24 berhasil dipasang. Ini menunjukkan bahwa client telah memiliki identitas dalam jaringan privat VPN.

- c. Terlihat adanya penambahan aturan routing dan firewall (via nft dan ip rule), yang menandakan bahwa rute trafik jaringan telah dimodifikasi agar melewati tunnel VPN (termasuk fwmark 51820).

2. Verifikasi status WireGuard antar server dan client

A terminal window with a black background and green text. The prompt is 'servervmvm@servervm:~\$'. The command 'sudo wg' has been executed. The output shows the status of the 'wg0' interface, including its public key, private key (hidden), and listening port (51820). It also shows a peer with a specific public key, endpoint IP (192.168.63.129), allowed IPs (10.8.0.2/32), latest handshake time (23 seconds ago), and transfer statistics (180 B received, 92 B sent).

```
servervmvm@servervm:~$ sudo wg
interface: wg0
  public key: Ddq9CxW/MtEQ/skoN3M7b793+st8LAIQY41jvwCfAg8=
  private key: (hidden)
  listening port: 51820

peer: ktwbFhRtJz2+eg+PjjSh/5Lbj3W0cAfHHgCs9+cvqG4=
  endpoint: 192.168.63.129:46637
  allowed ips: 10.8.0.2/32
  latest handshake: 23 seconds ago
  transfer: 180 B received, 92 B sent
servervmvm@servervm:~$ _
```

Gambar 19. Verifikasi status WireGuard antar server

Perintah: sudo wg

Analisis pengujian server:

- a. Interface wg0: Server mendengarkan (listening) pada port UDP 51820.
- b. Peer Terdeteksi: Server berhasil mengenali peer (Client) dengan public key yang berawalan “ktwbFh...”.
- c. Endpoint: Terlihat alamat IP asli Client yaitu 192.168.63.129 yang terhubung ke port server.
- d. Latest Handshake: Status menunjukkan “23 seconds ago”. Ini adalah indikator utama keberhasilan. Jika handshake berhasil, berarti kunci kriptografi cocok dan koneksi aman telah aktif.
- e. Transfer: Terjadi pertukaran data (180 B diterima, 92 B dikirim), yang membuktikan jalur data terbuka.

```

melah@melah-VMware-Virtual-Platform:~$ sudo wg
interface: wg0
  public key: ktwbFhRtJz2+eg+PjjSh/5Lbj3W0cAfHHGcs9+cvqG4=
  private key: (hidden)
  listening port: 46637
  fwmark: 0xca6c

peer: Ddq9CxW/MtEQ/skoN3M7b793+st8LAIQY41jvwCfAg8=
  endpoint: 192.168.63.128:51820
  allowed ips: 0.0.0.0/0
  latest handshake: 7 seconds ago
  transfer: 92 B received, 180 B sent
  persistent keepalive: every 25 seconds
melah@melah-VMware-Virtual-Platform:~$

```

Gambar 20. Verifikasi status WireGuard antar client

Perintah: sudo wg

Analisis pengujian client:

- a. Interface wg0: Client menggunakan port acak (46637) untuk koneksi keluar.
- b. Peer Terdeteksi: Client berhasil mengenali Server dengan public key berawalan “Ddq9Cx...”.
- c. Endpoint: Client terhubung ke IP asli Server 192.168.63.128 pada port 51820.
- d. Allowed IPs: Terkonfigurasi 0.0.0.0/0, yang artinya Client diset untuk melewatkan semua lalu lintas internetnya melalui VPN ini.
- e. Latest Handshake: Status “7 seconds ago” mengkonfirmasi koneksi aktif.
- f. Keepalive: Fitur persistent keepalive aktif setiap 25 detik untuk menjaga koneksi tetap hidup meski tidak ada lalu lintas data (penting untuk menembus NAT).

3. Pengujian ping antar IP VPN

```

servervmvm@servervm:~$ ping 10.8.0.2
PING 10.8.0.2 (10.8.0.2) 56(84) bytes of data.
64 bytes from 10.8.0.2: icmp_seq=1 ttl=64 time=3.81 ms
64 bytes from 10.8.0.2: icmp_seq=2 ttl=64 time=1.46 ms
64 bytes from 10.8.0.2: icmp_seq=3 ttl=64 time=1.19 ms
^C
--- 10.8.0.2 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2005ms
rtt min/avg/max/mdev = 1.187/2.152/3.807/1.175 ms
servervmvm@servervm:~$ _

```

Gambar 21. Pengujian ping antar IP VPN

Perintah: ping 10.8.0.2 (IP VPN client)

Analisis pengujian server:

- a. Server berhasil mengirim 3 paket (packets transmitted: 3) ke IP VPN Client.
- b. Server berhasil menerima 3 balasan (3 received).
- c. Packet Loss: 0%.
- d. Waktu Respon (RTT): Rata-rata waktu respon sangat cepat, sekitar 1.187 ms hingga 3.807 ms, menunjukkan latensi yang rendah.

```
melah@melah-VMware-Virtual-Platform:~$ ping 10.8.0.1
PING 10.8.0.1 (10.8.0.1) 56(84) bytes of data.
64 bytes from 10.8.0.1: icmp_seq=1 ttl=64 time=1.56 ms
64 bytes from 10.8.0.1: icmp_seq=2 ttl=64 time=1.46 ms
64 bytes from 10.8.0.1: icmp_seq=3 ttl=64 time=1.43 ms
^C
--- 10.8.0.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2005ms
rtt min/avg/max/mdev = 1.431/1.484/1.561/0.055 ms
melah@melah-VMware-Virtual-Platform:~$
```

Gambar 22. Pengujian ping antar IP VPN

Perintah: ping 10.8.0.1 (IP VPN server)

Analisis pengujian client:

- a. Client berhasil mengirim 3 paket ke IP VPN Server.
- b. Client berhasil menerima 3 balasan.
- c. Packet Loss: 0%.
- d. Waktu Respon (RTT): Waktu respon juga sangat rendah, rata-rata sekitar 1.431 ms hingga 1.561 ms.

4. Verifikasi IP routing pada client

```
melah@melah-VMware-Virtual-Platform:~$ ip route
default via 192.168.63.2 dev ens33 proto dhcp src 192.168.63.129 metric 100
10.8.0.0/24 dev wg0 proto kernel scope link src 10.8.0.2
192.168.63.0/24 dev ens33 proto kernel scope link src 192.168.63.129 metric 100
```

Gambar 23. Verifikasi IP routing pada client

Perintah: ip route

Analisis pengujian:

- a. Rute default jaringan fisik:

1. Sistem masih mempertahankan rute default yang ada sebelum VPN diaktifkan, yaitu default via 192.168.63.2 dev ens33.
 2. Ini mengarahkan semua lalu lintas yang tidak secara eksplisit dirutekan ke gateway fisik (192.168.63.2) melalui interface fisik (ens33). Rute ini digunakan untuk akses ke jaringan lokal (LAN) dan internet non-VPN (jika tidak ada full tunneling).
- b. Rute jaringan lokal fisik:
1. Terdapat rute untuk jaringan lokal fisik (192.168.63.0/24 dev ens33). Rute ini memastikan komunikasi dalam jaringan lokal tetap menggunakan ens33.
- c. Rute kunci VPN (WireGuard):
1. Ini adalah bukti konfigurasi VPN yang berhasil: 10.8.0.0/24 dev wg0.
 2. Baris ini secara eksplisit menginstruksikan sistem bahwa setiap paket yang ditujukan ke subnet VPN (10.8.0.0/24) harus dikirim melalui interface virtual wg0.
 3. IP sumber (src) untuk paket-paket ini adalah alamat VPN Client itu sendiri (10.8.0.2).

5. Verifikasi alamat IP interface WireGuard

```
servervm@servervm:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
        valid_lft forever preferred_lft forever
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:e7:aa:58 brd ff:ff:ff:ff:ff:ff
    altname enp2s1
    inet 192.168.63.128/24 metric 100 brd 192.168.63.255 scope global dynamic ens33
        valid_lft 1736sec preferred_lft 1736sec
    inet6 fe80::20c:23ff:fe7:aa58/64 scope link
        valid_lft forever preferred_lft forever
3: wg0: <POINTOPOINT,NOARP,UP,LOWER_UP> mtu 1420 qdisc noqueue state UNKNOWN group default qlen 1000
    link/none
    inet 10.8.0.1/24 scope global wg0
        valid_lft forever preferred_lft forever
servervm@servervm:~$
```

Gambar 24. Verifikasi alamat IP interface WireGuard

Perintah: ip a

Analisis pengujian server:

- a. Ditemukan interface 3: wg0 dengan status <POINTOPOINT, NOARP, UP, LOWER_UP>. Status UP menunjukkan interface aktif.
- b. MTU (Maximum Transmission Unit) diatur ke 1420, yang merupakan nilai standar untuk WireGuard.
- c. Alamat IP VPN: Ditemukan baris inet 10.8.0.1/24 scope global wg0. Ini membuktikan bahwa Server berhasil memasang alamat 10.8.0.1 pada interface wg0.

```
melah@melah-VMware-Virtual-Platform:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
        valid_lft forever preferred_lft forever
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:8b:cf:34 brd ff:ff:ff:ff:ff:ff
    altname enp2s1
    inet 192.168.63.129/24 brd 192.168.63.255 scope global dynamic noprefixroute ens33
        valid_lft 1161sec preferred_lft 1161sec
    inet6 fe80::20c:29ff:fe8b:cf34/64 scope link
        valid_lft forever preferred_lft forever
3: wg0: <POINTOPOINT,NOARP,UP,LOWER_UP> mtu 1420 qdisc noqueue state UNKNOWN group default qlen 1000
    link/none
    inet 10.8.0.2/24 scope global wg0
        valid_lft forever preferred_lft forever
```

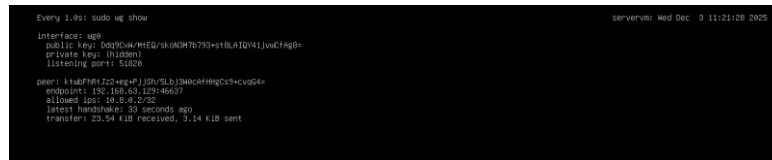
Gambar 25. Verifikasi alamat IP interface WireGuard

Perintah: ip a

Analisis pengujian client:

- a. Ditemukan interface 3: wg0 dengan status <POINTOPOINT, NOARP, UP, LOWER_UP>, mengkonfirmasi interface aktif.
- b. Alamat IP VPN: Ditemukan baris inet 10.8.0.2/24 scope global wg0. Ini membuktikan bahwa Client berhasil memasang alamat 10.8.0.2 pada interface wg0.
- c. Alamat IP fisik (interface ens33) Client juga terlihat (inet 192.168.63.129/24).

6. Pengujian handshake berulang



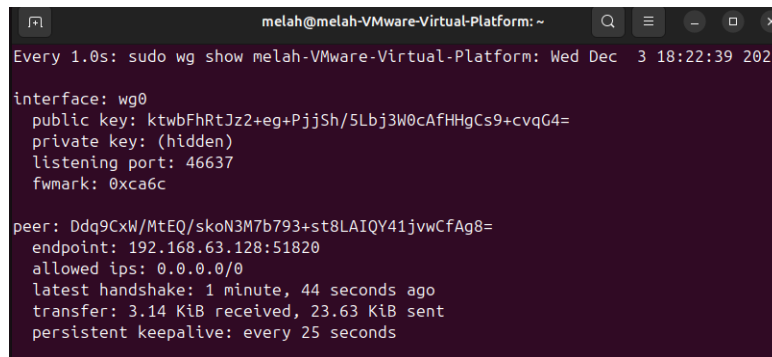
```
Every 1.0s: sudo wg show                               serverid: Wed Dec 3 11:21:28 2025
interface: wg0
  public key: 9dz9CwX/MtEQ/skoN3M7b793+st8LAIQY41jvwCfAg8=
  private key: (hidden)
  listening port: 51820
peer: k4wFhRtJz2+eg+PjjSh/5Lbj3W0cAfHHgCs9+cvqG4=
  endpoint: 192.168.63.129:46637
  allowed ips: 10.8.0.2/32
  latest handshake: 33 seconds ago
  transfer: 23.54 KiB received, 3.14 KiB sent
```

Gambar 26. Pengujian handshake berulang

Perintah: `watch -n 1 sudo wg show`

Analisis pengujian server:

- Latest Handshake: Menunjukkan nilai “33 seconds ago”. Nilai yang terus diperbarui ini membuktikan bahwa tunnel antara Server dan Client masih aktif dan key exchange kriptografi berjalan lancar.
- Transfer Data: Terlihat statistik data 23.54 KiB received, 3.14 KiB sent. Angka ini mengkonfirmasi adanya pertukaran data yang berkelanjutan, biasanya dikarenakan persistent keepalive atau aktivitas latar belakang sistem.



```
melah@melah-VMware-Virtual-Platform: ~
Every 1.0s: sudo wg show melah-VMware-Virtual-Platform: Wed Dec 3 18:22:39 2025
interface: wg0
  public key: ktwbFhRtJz2+eg+PjjSh/5Lbj3W0cAfHHgCs9+cvqG4=
  private key: (hidden)
  listening port: 46637
  fwmark: 0xca6c
peer: Ddq9CxW/MtEQ/skoN3M7b793+st8LAIQY41jvwCfAg8=
  endpoint: 192.168.63.128:51820
  allowed ips: 0.0.0.0/0
  latest handshake: 1 minute, 44 seconds ago
  transfer: 3.14 KiB received, 23.63 KiB sent
  persistent keepalive: every 25 seconds
```

Gambar 27. Pengujian handshake berulang

Perintah: `watch -n 1 sudo wg show`

Analisis pengujian client:

- Latest Handshake: Menunjukkan “1 minute, 44 seconds ago”. Meskipun berbeda dengan server karena pengecekan dilakukan pada waktu yang berbeda, nilai ini berada dalam rentang waktu keepalive 25 detik yang dikonfigurasi, memastikan bahwa koneksi tidak terputus (timed out).

- b. Transfer Data: Statistik transfer menunjukkan 3.14 KiB received, 23.63 KiB sent. Angka konsisten dengan data yang dikirim dan diterima oleh Server, menunjukkan bahwa transfer data melalui tunnel berjalan dua arah dengan baik.

7. Pengujian akses jarak jauh (*remote access*) melalui SSH

```
melah@melah-VMware-Virtual-Platform:~$ ssh servervmvm@10.8.0.1
The authenticity of host '10.8.0.1 (10.8.0.1)' can't be established.
ED25519 key fingerprint is SHA256:ipF3S0838AsQ6QBcp8t9uj6FuWimZBe6RfGUSbUpZag.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.8.0.1' (ED25519) to the list of known hosts.
servervmvm@10.8.0.1's password:
Welcome to Ubuntu 24.04.3 LTS (GNU/Linux 6.8.0-87-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

System information as of Wed Dec  3 12:39:17 PM UTC 2025

System load:  0.0               Processes:    229
Usage of /:   50.4% of 9.75GB    Users logged in: 1
Memory usage: 15%              IPv4 address for ens33: 192.168.63.128
Swap usage:   0%

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

Last login: Wed Dec  3 12:34:13 2025 from 192.168.63.1
servervmvm@servervm:~$
```

Gambar 28. Pengujian akses jarak jauh (*remote access*) melalui SSH

Perintah: `ssh servervmvm@10.8.0.1`

Analisis pengujian:

- a. Koneksi Dibuat: Perintah ini berhasil menemukan dan terhubung ke alamat IP VPN Server (10.8.0.1).
- b. Verifikasi Kunci: Sistem memberikan peringatan authenticity of host (standard pada koneksi SSH pertama), yang kemudian diterima oleh pengguna (yes).
- c. Akses Berhasil: Setelah memasukkan password, Client berhasil mendapatkan akses shell ke Server. Hal ini dibuktikan dengan perubahan prompt terminal menjadi `servervmvm@servervm:~$` (sama seperti terminal Server).

- d. Verifikasi IP Sumber: Informasi last login menunjukkan bahwa koneksi masuk berasal dari 192.168.63.1, namun yang paling penting, koneksi SSH itu sendiri berjalan di atas alamat VPN (10.8.0.1).

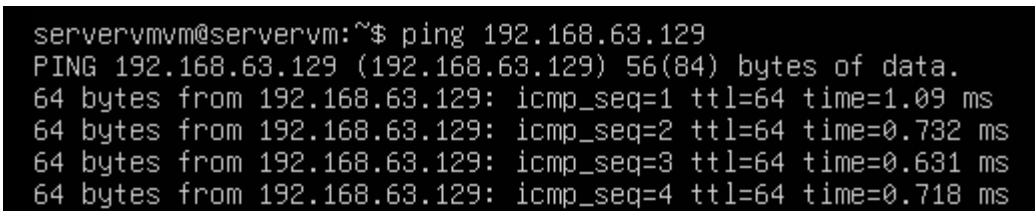
4.5 Implementasi Capture Wireshark Sebelum & Sesudah VPN

Pada minggu 5 dilakukan proses pengujian keamanan komunikasi jaringan dengan melakukan packet capture menggunakan Wireshark. Tujuan tahap ini adalah membandingkan kondisi lalu lintas jaringan sebelum menggunakan VPN (non-encrypted traffic) dan sesudah VPN aktif (encrypted traffic), sehingga dapat dibuktikan bahwa WireGuard berhasil memberikan enkripsi pada komunikasi antar mesin.

Tahapan dan Proses:

1. Melakukan ping antar mesin sebelum VPN

Pada tahap ini dilakukan pengujian konektivitas dasar antara kedua mesin (server dan client) dengan menggunakan perintah ping melalui jaringan lokal sebelum VPN diaktifkan. Tujuannya adalah memastikan bahwa kedua VM dapat saling terhubung secara langsung tanpa enkripsi, serta menjadi dasar pembandingan sebelum dilakukan pengujian melalui WireGuard VPN.



```
servervm@servervm:~$ ping 192.168.63.129
PING 192.168.63.129 (192.168.63.129) 56(84) bytes of data.
64 bytes from 192.168.63.129: icmp_seq=1 ttl=64 time=1.09 ms
64 bytes from 192.168.63.129: icmp_seq=2 ttl=64 time=0.732 ms
64 bytes from 192.168.63.129: icmp_seq=3 ttl=64 time=0.631 ms
64 bytes from 192.168.63.129: icmp_seq=4 ttl=64 time=0.718 ms
```

Gambar 29. Melakukan ping antar mesin sebelum VPN

Pada sisi server, dilakukan pengujian konektivitas dengan menjalankan perintah ping menuju alamat IP client (192.168.63.129). Hasil output menunjukkan respons ICMP yang stabil dengan waktu latensi rendah, menandakan bahwa server dapat terhubung ke client melalui jaringan lokal tanpa hambatan. Pengujian ini memastikan bahwa jalur

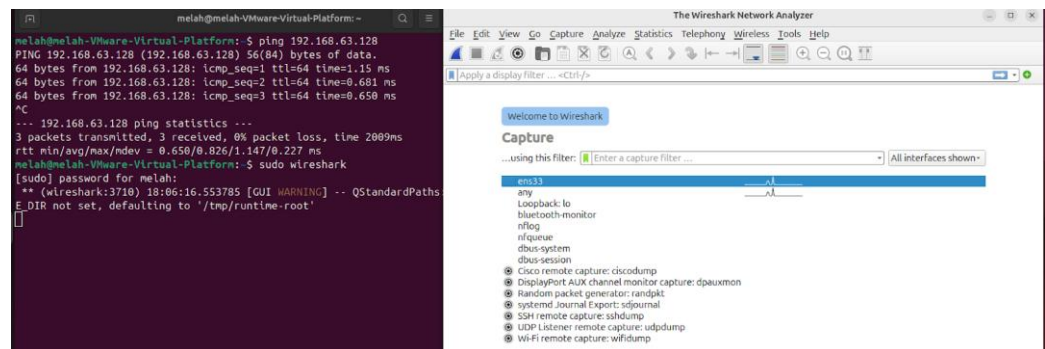
komunikasi dasar antara kedua mesin sudah berjalan sebelum VPN diaktifkan.

```
melah@melah-VMware-Virtual-Platform:~$ ping 192.168.63.128
PING 192.168.63.128 (192.168.63.128) 56(84) bytes of data:
64 bytes from 192.168.63.128: icmp_seq=1 ttl=64 time=1.15 ms
64 bytes from 192.168.63.128: icmp_seq=2 ttl=64 time=0.681 ms
64 bytes from 192.168.63.128: icmp_seq=3 ttl=64 time=0.650 ms
```

Gambar 30. Melakukan ping antar mesin sebelum VPN

Pada sisi client, dilakukan pengujian serupa dengan mengirimkan paket ICMP ke alamat IP server (192.168.63.128). Hasil ping menunjukkan balasan yang konsisten dan tidak ada packet loss, sehingga membuktikan bahwa client dapat menjangkau server secara langsung.

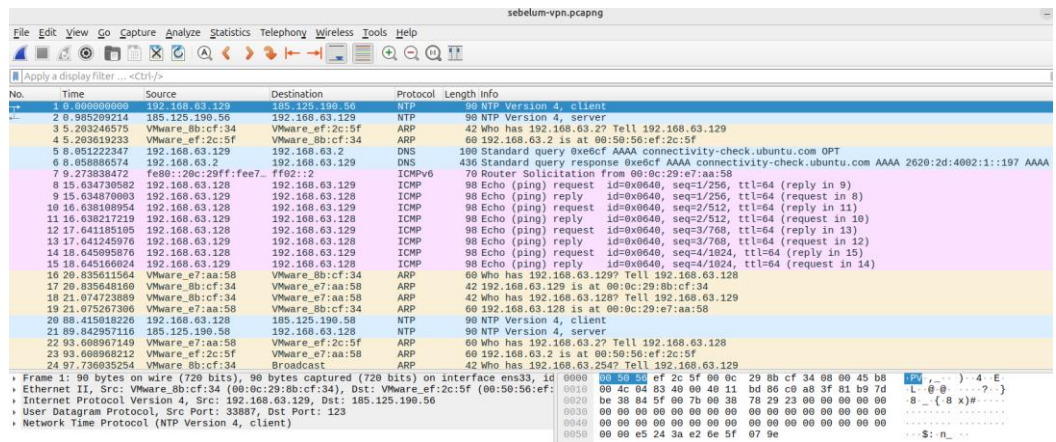
2. Membuka aplikasi Wireshark untuk melakukan *packet capture*



Gambar 31. Membuka aplikasi Wireshark untuk melakukan packet capture

Pada tahap ini, client membuka aplikasi Wireshark menggunakan perintah `sudo wireshark` untuk memulai proses packet capture. Setelah aplikasi berjalan, Wireshark menampilkan daftar interface jaringan yang tersedia seperti `ens33` yang akan digunakan untuk merekam lalu lintas jaringan sebelum dan sesudah VPN diaktifkan.

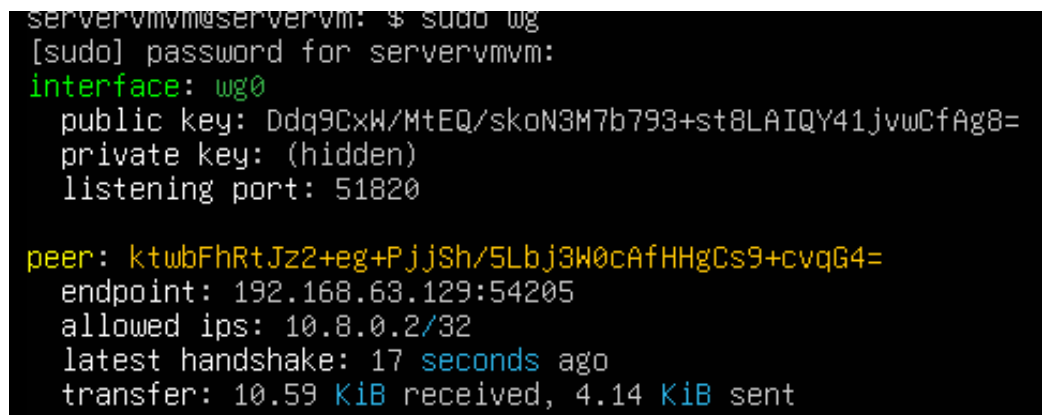
3. *Capture lalu lintas jaringan sebelum VPN*



Gambar 32. *Capture lalu lintas jaringan sebelum VPN*

Pada tahap ini dilakukan proses packet capture menggunakan Wireshark sebelum VPN WireGuard dijalankan. Hasil capture memperlihatkan bahwa seluruh paket jaringan seperti ICMP (ping), DNS query, ARP, dan NTP ditampilkan dalam bentuk plaintext, sehingga seluruh informasi sumber, tujuan, dan payload dapat terbaca dengan jelas. Kondisi ini menunjukkan bahwa komunikasi antar mesin masih berjalan tanpa enkripsi, sehingga rentan untuk disadap jika melalui jaringan publik.

4. Melakukan verifikasi *handshake* VPN WireGuard



Gambar 33. *Melakukan verifikasi handshake VPN WireGuard*

Pada sisi server, perintah `sudo wg` digunakan untuk menampilkan status koneksi WireGuard. Output menunjukkan bahwa server telah mengenali peer (client) melalui publik key yang sesuai, serta menampilkan informasi endpoint client dan allowed IP. Bagian “latest handshake”

menunjukkan waktu terakhir kedua perangkat berhasil bertukar kunci dan membentuk sesi terenkripsi. Data transfer (received/sent) juga mulai bertambah, menandakan bahwa komunikasi VPN sudah aktif.

```
melah@melah-VMware-Virtual-Platform:~$ sudo wg
interface: wg0
  public key: ktwbFhRtJz2+eg+PjjSh/5Lbj3W0cAfHHgCs9+cvqG4=
  private key: (hidden)
  listening port: 54205
  fwmark: 0xca6c

peer: Ddq9CxW/MtEQ/skoN3M7b793+st8LAIQY41jvwCfAg8=
  endpoint: 192.168.63.128:51820
  allowed ips: 0.0.0.0/0
  latest handshake: 6 seconds ago
  transfer: 92 B received, 180 B sent
  persistent keepalive: every 25 seconds
```

Gambar 34. Komunikasi VPN sudah aktif.

Pada sisi client, perintah yang sama (sudo wg) memperlihatkan bahwa client berhasil terhubung dengan server. Terlihat informasi publik key server, endpoint server, allowed IP, serta nilai “latest handshake” yang mengonfirmasi bahwa pertukaran kunci terjadi dengan sukses. Nilai transfer data juga mulai muncul, menandakan bahwa client telah berhasil mengirim dan menerima paket melalui tunnel VPN.

5. Melakukan verifikasi *routing* di Client

```
melah@melah-VMware-Virtual-Platform:~$ ip route
default via 192.168.63.2 dev ens33 proto dhcp src 192.168.63.129 metric 100
10.8.0.0/24 dev wg0 proto kernel scope link src 10.8.0.2
192.168.63.0/24 dev ens33 proto kernel scope link src 192.168.63.129 metric 100
```

Gambar 35. Melakukan verifikasi routing di Client

Pada tahap ini client menjalankan perintah ip route untuk memastikan bahwa rute jaringan untuk interface VPN (wg0) sudah terbentuk. Hasilnya menunjukkan adanya rute menuju jaringan 10.8.0.0/24 melalui interface wg0, dengan alamat sumber 10.8.0.2. Ini menandakan bahwa client sudah terhubung ke tunnel VPN dan dapat mengirim trafik ke server VPN melalui jalur tersebut.

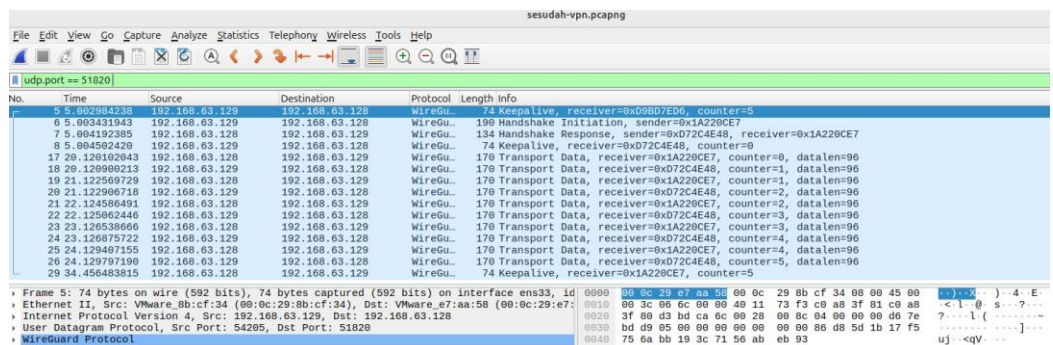
6. Melakukan ping IP VPN dari Server

```
servervmvm@servervm:~$ ping 10.8.0.2
PING 10.8.0.2 (10.8.0.2) 56(84) bytes of data.
64 bytes from 10.8.0.2: icmp_seq=1 ttl=64 time=2.35 ms
64 bytes from 10.8.0.2: icmp_seq=2 ttl=64 time=1.43 ms
64 bytes from 10.8.0.2: icmp_seq=3 ttl=64 time=1.38 ms
64 bytes from 10.8.0.2: icmp_seq=4 ttl=64 time=1.22 ms
64 bytes from 10.8.0.2: icmp_seq=5 ttl=64 time=1.23 ms
```

Gambar 36. Melakukan ping IP VPN dari Server

Server melakukan pengujian konektivitas ke client melalui perintah ping 10.8.0.2. Hasil ping menunjukkan balasan (reply) dengan waktu respon sangat rendah, menandakan bahwa tunnel VPN telah bekerja dengan baik dan komunikasi antara server dan client melalui jaringan privat berjalan tanpa masalah.

7. Capture lalu lintas jaringan setelah VPN



Gambar 37. Capture lalu lintas jaringan setelah VPN

Gambar di atas menunjukkan tampilan Wireshark saat melakukan packet capture setelah koneksi VPN WireGuard diaktifkan. Pada daftar paket terlihat bahwa seluruh komunikasi jaringan ditandai dengan protokol WireGuard (Wg), yang terdiri dari beberapa tipe paket seperti Handshake Initiation, Handshake Response, Keepalive, serta Transport Data.

Paket-paket Transport Data menampilkan informasi seperti receiver, counter, dan datalen, namun tidak menampilkan konten asli data karena seluruh payload telah dienkripsi. Hal ini menjadi bukti bahwa setelah VPN aktif, lalu lintas jaringan tidak lagi terbaca sebagai data mentah (non-

encrypted traffic), melainkan hanya terlihat sebagai paket terenkripsi oleh WireGuard.

BAB V

SARAN DAN KESIMPULAN

5.1 KESIMPULAN

Berdasarkan hasil perancangan, implementasi, dan pengujian yang telah dilakukan pada mata kuliah Perancangan Keamanan Sistem dan Jaringan, dapat disimpulkan bahwa penerapan Virtual Private Network (VPN) menggunakan WireGuard berhasil diimplementasikan dengan baik. Sistem VPN WireGuard mampu menyediakan akses jarak jauh (secure remote access) antara client dan server secara aman melalui tunnel terenkripsi.

Implementasi dilakukan secara bertahap berdasarkan progres mingguan, dimulai dari persiapan lingkungan virtual, instalasi sistem operasi, instalasi WireGuard, konfigurasi kunci dan interface, hingga tahap pengujian koneksi dan analisis keamanan jaringan. Hasil pengujian menunjukkan bahwa koneksi VPN dapat berjalan dengan stabil, ditandai dengan keberhasilan handshake, pengujian ping antar IP VPN, serta akses jarak jauh menggunakan protokol SSH.

Selain itu, analisis lalu lintas jaringan menggunakan Wireshark membuktikan bahwa komunikasi setelah VPN diaktifkan telah terenkripsi. Hal ini terlihat dari packet capture yang hanya menampilkan paket WireGuard tanpa memperlihatkan isi data secara langsung, berbeda dengan kondisi sebelum VPN aktif yang masih dapat dibaca. Dengan demikian, tujuan pembelajaran mengenai penerapan keamanan jaringan dan perlindungan data pada akses jarak jauh telah tercapai.

5.2 SARAN

Adapun beberapa saran yang dapat diberikan untuk pengembangan selanjutnya adalah sebagai berikut:

1. Implementasi VPN WireGuard dapat dikembangkan dengan menambahkan lebih banyak client untuk menguji performa dan skalabilitas sistem.

2. Sistem keamanan dapat diperkuat dengan menambahkan firewall rules atau integrasi dengan sistem autentikasi tambahan.
3. Pengujian dapat diperluas dengan melakukan simulasi serangan jaringan untuk melihat efektivitas VPN dalam menghadapi ancaman keamanan.
4. Analisis jaringan dapat dilakukan lebih mendalam dengan memanfaatkan fitur lanjutan pada Wireshark untuk mendapatkan hasil evaluasi yang lebih komprehensif.

Diharapkan hasil dari proyek ini dapat menjadi dasar pemahaman dalam penerapan keamanan sistem dan jaringan, khususnya dalam penggunaan VPN sebagai solusi secure remote access.

DAFTAR PUSTAKA

- Master, A., & Garman, C. (2021). *A WireGuard Exploration*. Purdue University.
DOI: 10.5703/1288284317610 docs.lib.purdue.edu
- Phạm Anh Thu. (2025). *Performance Evaluation of WireGuard Protocol in Next Generation VPNs*. JSTIC Journal. jstic.ptit.edu.vn
- Rahman, I. K., & Harnaningrum, L. N. (2024). *Analisa QoS Pada Jaringan L2TP IPSec dan WireGuard VPN untuk mengamankan VoIP*. Jurnal RESISTOR. ejournal.instiki.ac.id
- Bongga Arifwidodo. (2023). *Mekanisme Keamanan Jaringan Menggunakan Protokol WireGuard pada Jaringan Privat*. JICT. ejournal.akademitelkom.ac.id
- Muhammad Agus Darmawan et al. (2023). *Analisis Keamanan Jaringan terhadap Sniffing menggunakan Wireshark*. Jurnal Just-IT. [Jurnal UMJ](#)