Department of Digital Business
Journal of Artificial Intelligence and Digital Business (RIGGS)
Homepage: https://journal.ilmudata.co.id/index.php/RIGGS
Vol. 5 No. 2 (2026) pp: 13532-13543
P-ISSN: 2963-9298, e-ISSN: 2963-914X

Analisis Perbandingan Kinerja WireGuard dan OpenVPN pada VPN Site-to-Site Server Debian

Yoel Simbolon, Devi Sari Delima Tumanggor, Efrans Surbakti, Lotar Mateus Sinaga
Teknik Informatika, Fakultas Ilmu Komputer, Universitas Katolik Santo Thomas
yoelsimbolon25@gmail.com; devisaridelima23@gmail.com; surbaktiefrans@gmail.com; lotarmateus88@gmail.com

Abstrak

Virtual Private Network (VPN) site-to-site banyak digunakan untuk menghubungkan dua jaringan lokal yang terpisah secara geografis melalui jaringan publik secara aman. Perbedaan arsitektur antara OpenVPN, yang berjalan di ruang pengguna (user space), dan WireGuard, yang berjalan di dalam kernel, diperkirakan menghasilkan perbedaan kinerja yang signifikan namun belum dievaluasi secara kuantitatif pada server Debian. Penelitian ini bertujuan membandingkan kinerja WireGuard dan OpenVPN pada VPN site-to-site yang diterapkan di server berbasis Debian. Eksperimen dilakukan pada lingkungan tervirtualisasi menggunakan Oracle VirtualBox dengan dua gateway yang membentuk terowongan, dan lima parameter diukur menggunakan iperf3, ping, dan mpstat: throughput, latency, jitter, packet loss, dan utilisasi CPU. Hasil penelitian menunjukkan bahwa WireGuard secara signifikan mengungguli OpenVPN, dengan throughput TCP sebesar 94 Mbit/s dibanding 27,5 Mbit/s (sekitar 3,4 kali lebih tinggi), latency rata-rata yang lebih rendah (1,493 ms berbanding 1,659 ms), serta packet loss 0% berbanding 71% pada beban UDP 100 Mbit/s, dengan utilisasi CPU yang setara sekitar 100%. Temuan ini menunjukkan bahwa arsitektur WireGuard yang berbasis kernel memberikan efisiensi dan kapasitas tunnel yang lebih tinggi dibanding OpenVPN yang berbasis ruang pengguna, sehingga dapat menjadi acuan praktis dalam pemilihan protokol VPN site-to-site yang mengutamakan throughput tinggi dan efisiensi sumber daya.

Kata kunci: *WireGuard; OpenVPN; VPN Site-to-Site; Kinerja; Server; Debian*

1. Latar Belakang

Kebutuhan komunikasi data yang aman antar-lokasi terus meningkat seiring berkembangnya organisasi yang memiliki lebih dari satu kantor atau pusat data. Ancaman terhadap data dalam transit juga meningkat tajam dalam beberapa tahun terakhir — hanya pada kuartal III 2024, lebih dari 241 juta individu di Amerika Serikat menjadi korban kebocoran data (Identity Theft Resource Center, 2024), sementara laporan We Are Social (2025) mencatat sekitar 23,1% pengguna internet global di atas 16 tahun telah menggunakan VPN dalam aktivitas hariannya pada kuartal I 2025. Akamai (2025) bahkan menemukan bahwa perangkat VPN justru menjadi salah satu target favorit serangan siber, terutama melalui eksploitasi celah keamanan dan pencurian kredensial, sehingga pemilihan protokol VPN yang tepat menjadi semakin krusial. Untuk menghubungkan dua jaringan lokal yang terpisah secara geografis melalui jaringan publik seperti internet, diperlukan mekanisme yang menjamin kerahasiaan dan integritas data selama transmisi. Virtual Private Network (VPN) site-to-site menjadi solusi yang umum digunakan karena membentuk terowongan terenkripsi antara dua gateway sehingga lalu lintas antar-jaringan tidak dapat dibaca oleh pihak yang tidak berwenang (Mackey et al., 2020).

Salah satu protokol VPN yang telah lama mapan adalah OpenVPN, yang berjalan pada ruang pengguna (user space) dan memanfaatkan pustaka OpenSSL. Meskipun andal dan fleksibel, arsitektur user space OpenVPN kerap dianggap menimbulkan overhead pemrosesan yang relatif tinggi. Sebagai alternatif yang lebih baru, WireGuard hadir dengan basis kode ringkas dan berjalan di dalam kernel sistem operasi, serta menggunakan primitif kriptografi modern seperti Curve25519 untuk pertukaran kunci dan ChaCha20-Poly1305 untuk enkripsi terotentikasi.

Beberapa penelitian terdahulu telah membandingkan kinerja WireGuard dan OpenVPN, namun hasilnya menunjukkan variasi yang besar antar-lingkungan pengujian sehingga belum dapat dijadikan generalisasi langsung untuk konteks site-to-site di server Debian. Mackey et al. (2020) membangun kerangka pengujian otomatis pada delapan node, termasuk instance AWS dan VM lokal, dan menemukan bahwa keunggulan utama WireGuard

terletak pada kinerjanya di mesin multi-core serta basis kodenya yang ringan dibanding OpenVPN. Anyam et al. (2025) mengevaluasi kedua protokol pada lingkungan cloud dan virtualisasi (Azure dan VMware) dan menemukan bahwa kinerja sangat bergantung pada konteks: pada VMware, WireGuard mencapai throughput TCP 210,64 Mbps berbanding 110,34 Mbps pada OpenVPN dengan packet loss yang jauh lebih rendah (12,35% berbanding 47,01%), namun pada Azure dengan kondisi latensi tinggi OpenVPN justru tampil lebih baik (120 Mbps berbanding 60 Mbps). Hussein dan Rehman (2025) menggunakan aplikasi Speedtest pada platform Windows 11 dan melaporkan WireGuard mencapai kecepatan unduh 50,4 Mbps dengan latency 8,01 ms dan utilisasi CPU hanya 3%, sedangkan OpenVPN turun menjadi 22,1 Mbps dengan latency 160,38 ms. Ali et al. (2025), dalam konteks Indonesia, mengembangkan model evaluasi VPERFSEC yang menggabungkan skor keamanan dan performa, dan menemukan bahwa VPN mampu mencegah 87% serangan sniffing dan man-in-the-middle, dengan OpenVPN mencatat latency tertinggi (80 ms) dibanding WireGuard yang paling efisien (45 ms). Dekker dan Spaans (2020) membandingkan WireGuard, strongSwan, dan OpenVPN pada lingkungan 1 Gbit/s dan menemukan bahwa strongSwan dengan cipher AES-GCM justru mencatat goodput tertinggi dan latency terendah, mengindikasikan bahwa keunggulan WireGuard tidak selalu konsisten di semua skenario pengujian.

Dari kelima penelitian tersebut terlihat sebuah kesenjangan (*gap*) yang jelas: pengujian sebelumnya sebagian besar dilakukan pada skenario remote-access, lingkungan cloud publik (Azure, AWS), atau platform desktop (Windows 11), dan belum ada yang secara khusus mengevaluasi implementasi VPN site-to-site pada server Debian — sistem operasi yang banyak digunakan sebagai basis infrastruktur server dan layanan cloud karena stabilitas dan keamanannya. Variasi hasil antar-lingkungan pada penelitian terdahulu juga menunjukkan bahwa kinerja WireGuard dan OpenVPN tidak dapat diasumsikan sama pada setiap platform, sehingga pengukuran kuantitatif khusus pada konfigurasi site-to-site berbasis Debian menjadi penting untuk dilakukan secara independen, terutama mengingat meningkatnya ancaman siber terhadap perangkat VPN dan kebutuhan organisasi akan koneksi antar-lokasi yang efisien sekaligus aman.

Perbedaan arsitektur antara kedua protokol diperkirakan menghasilkan perbedaan kinerja yang signifikan, terutama pada throughput, latency, dan penggunaan sumber daya prosesor. Oleh karena itu, penelitian ini bertujuan membandingkan kinerja protokol WireGuard dan OpenVPN pada implementasi VPN site-to-site di server Debian melalui pengukuran lima parameter, yaitu throughput, latency, jitter, packet loss, dan utilisasi CPU. Kontribusi penelitian ini adalah menyediakan data perbandingan kinerja yang dapat menjadi acuan praktis dalam memilih protokol VPN site-to-site sesuai kebutuhan kinerja dan efisiensi sumber daya.

2. Metode Penelitian

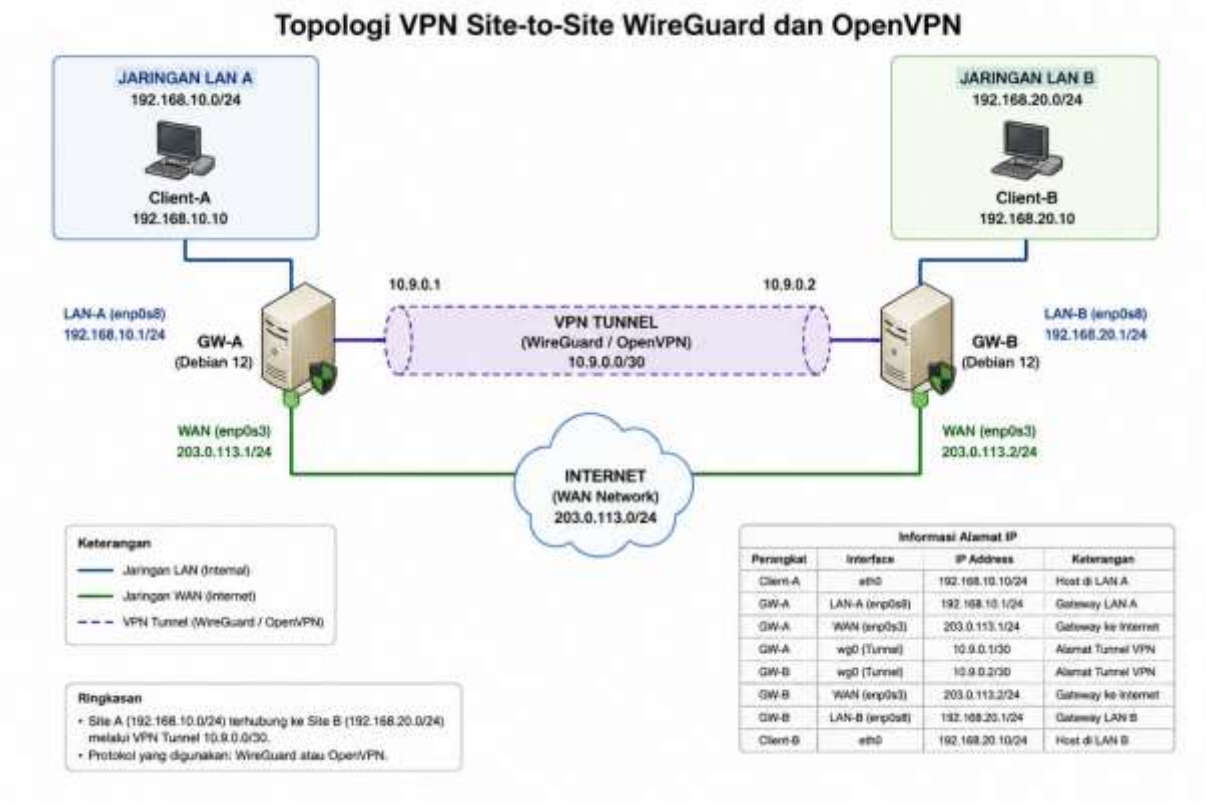
Penelitian ini menggunakan pendekatan eksperimental komparatif dengan mengukur kinerja WireGuard dan OpenVPN dalam kondisi pengujian yang identik, kemudian membandingkan hasilnya secara kuantitatif. Tahapan penelitian meliputi perancangan topologi jaringan, pembangunan lingkungan pengujian tervirtualisasi, konfigurasi masing-masing protokol secara bergantian, pengumpulan data melalui pengujian terukur, serta analisis dan perbandingan hasil.

2.1. Topologi dan Lingkungan Pengujian

Lingkungan pengujian dibangun pada Oracle VirtualBox menggunakan empat mesin virtual bersistem operasi Debian 12 yang terdiri dari dua gateway, yaitu GW-A dan GW-B, serta dua host pengujian yaitu Client-A dan Client-B. Jalur antar-gateway disimulasikan sebagai jaringan publik menggunakan Internal Network dengan alamat 203.0.113.1 dan 203.0.113.2. Masing-masing gateway juga terhubung ke jaringan lokal yang berbeda, yaitu LAN-A (192.168.10.0/24) dan LAN-B (192.168.20.0/24).

Setiap protokol VPN membentuk terowongan (tunnel) antar-gateway menggunakan alamat virtual 10.9.0.1 dan 10.9.0.2. Pengukuran kinerja dilakukan secara end-to-end melalui terowongan tersebut dengan lalu lintas data yang dikirim dari satu sisi jaringan ke sisi lainnya. Topologi pengujian ditunjukkan pada Gambar 1, sedangkan spesifikasi perangkat dan konfigurasi protokol ditunjukkan pada Tabel 1.

DOI: https://doi.org/10.31004/riggs.v5i2.11029
Lisensi: Creative Commons Attribution 4.0 International (CC BY 4.0)



Gambar 1. Topologi lingkungan pengujian

Berdasarkan topologi pada Gambar 1, pengujian dilakukan dengan menghubungkan dua jaringan lokal melalui tunnel VPN Site-to-Site yang dibangun antara GW-A dan GW-B. Lingkungan pengujian menggunakan empat mesin virtual berbasis Debian 12 yang terdiri dari dua gateway dan dua client. Spesifikasi perangkat dan konfigurasi yang digunakan selama penelitian ditunjukkan pada Tabel 1.

Tabel 1. Spesifikasi perangkat dan konfigurasi pengujian

Hasil	Spesifikasi
Hypervisor	Oracle VirtualBox
Sistem operasi	Debian 12 (64-bit), kernel 6.1
Resource tiap VM	1 vCPU, 1 GB RAM
WireGuard	Curve25519, ChaCha20-Poly1305
OpenVPN	point-to-point, AES-256-CBC, auth SHA256
Alat ukur	iperf3, ping, mpstat

2.2. Konfigurasi Protokol

WireGuard dikonfigurasi menggunakan antarmuka virtual wg0 dengan alamat tunnel 10.9.0.1/24 pada GW-A dan 10.9.0.2/24 pada GW-B. Pertukaran kunci dilakukan menggunakan algoritma Curve25519 dan enkripsi ChaCha20-Poly1305. Selain itu, fitur IP Forwarding diaktifkan untuk memungkinkan proses routing antar jaringan lokal melalui tunnel VPN.

OpenVPN dikonfigurasi dalam mode Site-to-Site menggunakan enkripsi AES-256-CBC dan autentikasi SHA256. Sama seperti WireGuard, OpenVPN dikonfigurasi untuk menghubungkan jaringan 192.168.10.0/24 dan 192.168.20.0/24 melalui jaringan WAN virtual sehingga kedua jaringan dapat saling berkomunikasi melalui tunnel VPN yang aman.

DOI: https://doi.org/10.31004/riggs.v5i2.11029
Lisensi: Creative Commons Attribution 4.0 International (CC BY 4.0)

2.3. Parameter dan Skenario Pengujian

Lima parameter diukur untuk setiap protokol. Throughput TCP diukur menggunakan iperf3 selama 30 detik. Jitter dan packet loss diukur menggunakan iperf3 mode UDP pada bandwidth 30 Mbit/s selama 30 detik. Latency diukur dengan mengirim 100 paket ICMP menggunakan ping, dan nilai yang diambil adalah rata-rata round trip time (RTT). Utilisasi CPU direkam menggunakan mpstat selama proses transfer data berlangsung.

Setiap protokol diuji secara bergantian, dan protokol yang tidak sedang diuji selalu dinonaktifkan untuk menjaga keakuratan pengukuran throughput dan CPU. Seluruh hasil pengujian kemudian dicatat dan dianalisis secara kuantitatif untuk membandingkan kinerja WireGuard dan OpenVPN berdasarkan parameter throughput, latency, jitter, packet loss, dan utilisasi CPU.

3. Hasil dan Diskusi

Pengujian dilakukan pada lingkungan VPN site-to-site yang dibangun menggunakan server Debian di atas platform virtualisasi VirtualBox. Topologi terdiri dari empat mesin virtual: Gateway-A (GW-A), Gateway-B (GW-B), Client-A, dan Client-B, yang masing-masing dikonfigurasi untuk mensimulasikan dua sisi jaringan privat yang terhubung melalui terowongan VPN. Kedua protokol, yaitu WireGuard dan OpenVPN, diuji pada kondisi beban jaringan yang identik menggunakan alat ukur iperf3, ping, sysstat (sar), dan htop. Pengujian mencakup lima parameter utama: throughput, latency (latensi), packet loss, penggunaan CPU, dan penggunaan memori.

3.1. Topologi dan Lingkungan Pengujian

Pengujian dilakukan pada lingkungan VPN Site-to-Site yang dibangun menggunakan empat mesin virtual, yaitu GW-A, GW-B, Client-A, dan Client-B. Kedua gateway dikonfigurasi sebagai endpoint VPN yang menghubungkan jaringan LAN-A (192.168.10.0/24) dan LAN-B (192.168.20.0/24) melalui jaringan WAN virtual 203.0.113.0/24. Implementasi dilakukan menggunakan protokol WireGuard dan OpenVPN secara bergantian pada kondisi pengujian yang sama.



Gambar 2. Implementasi lingkungan pengujian pada Oracle VirtualBox

3.2. Implementasi dan Aktivasi WireGuard

Setelah konfigurasi alamat IP dan routing selesai dilakukan, protokol WireGuard dikonfigurasi pada kedua gateway menggunakan antarmuka virtual wg0. Aktivasi tunnel dilakukan menggunakan perintah wg-quick up wg0. Hasil menunjukkan bahwa tunnel WireGuard berhasil dibuat tanpa error dan kedua gateway memperoleh alamat VPN 10.9.0.1 dan 10.9.0.2.

DOI: https://doi.org/10.31004/riggs.v5i2.11029
Lisensi: Creative Commons Attribution 4.0 International (CC BY 4.0)

```
yoel@gw-a:~$ sudo wg-quick up wg0
sudo: unable to resolve host gw-a: Name or service not known
[#] ip link add wg0 type wireguard
[#] wg setconf wg0 /dev/fd/63
[#] ip -4 address add 10.9.0.1/24 dev wg0
[#] ip link set mtu 1420 up dev wg0
[#] ip -4 route add 192.168.20.0/24 dev wg0
yoel@gw-a:~$
```

Gambar 2. Aktivasi tunnel WireGuard pada GW-A

```
yoel@gw-b:~$ sudo wg-quick up wg0
[#] ip link add wg0 type wireguard
[#] wg setconf wg0 /dev/fd/63
[#] ip -4 address add 10.9.0.2/24 dev wg0
[#] ip link set mtu 1420 up dev wg0
[#] ip -4 route add 192.168.10.0/24 dev wg0
yoel@gw-b:~$
```

Gambar 3. Aktivasi tunnel WireGuard pada GW-B

3.3. Pengujian Konektivitas VPN

Untuk memastikan tunnel VPN berfungsi dengan baik, dilakukan pengujian menggunakan perintah ping dari GW-A ke GW-B melalui alamat VPN 10.9.0.2. Hasil pengujian menunjukkan seluruh paket berhasil diterima dengan packet loss sebesar 0%, sehingga tunnel VPN dinyatakan aktif dan siap digunakan untuk pengujian performa.

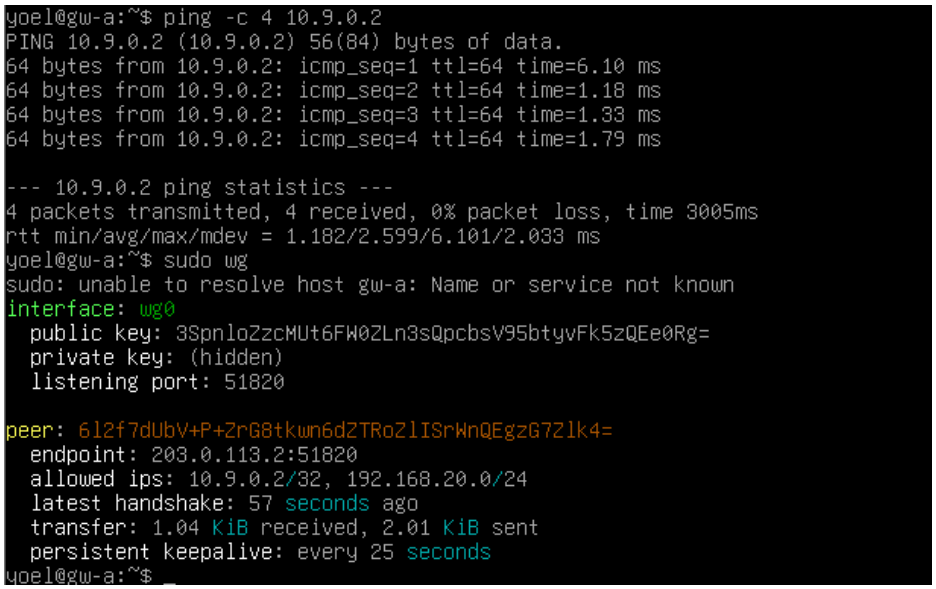
```
yoel@gw-a:~$ ping -c 100 10.9.0.2
PING 10.9.0.2 (10.9.0.2) 56(84) bytes of data.
64 bytes from 10.9.0.2: icmp_seq=1 ttl=64 time=6.30 ms
64 bytes from 10.9.0.2: icmp_seq=2 ttl=64 time=2.17 ms
64 bytes from 10.9.0.2: icmp_seq=3 ttl=64 time=1.34 ms
64 bytes from 10.9.0.2: icmp_seq=4 ttl=64 time=0.924 ms
64 bytes from 10.9.0.2: icmp_seq=5 ttl=64 time=1.53 ms
64 bytes from 10.9.0.2: icmp_seq=6 ttl=64 time=1.23 ms
64 bytes from 10.9.0.2: icmp_seq=7 ttl=64 time=1.01 ms
64 bytes from 10.9.0.2: icmp_seq=8 ttl=64 time=1.46 ms
64 bytes from 10.9.0.2: icmp_seq=9 ttl=64 time=0.938 ms
64 bytes from 10.9.0.2: icmp_seq=10 ttl=64 time=1.23 ms
64 bytes from 10.9.0.2: icmp_seq=11 ttl=64 time=2.11 ms
```

Gambar 4. Hasil pengujian konektivitas WireGuard

3.4. Status Tunnel WireGuard

Status koneksi VPN diperiksa menggunakan perintah wg. Hasil menunjukkan bahwa proses handshake berhasil dilakukan antara kedua gateway dan terdapat aktivitas transfer data pada tunnel.

DOI: https://doi.org/10.31004/riggs.v5i2.11029
Lisensi: Creative Commons Attribution 4.0 International (CC BY 4.0)

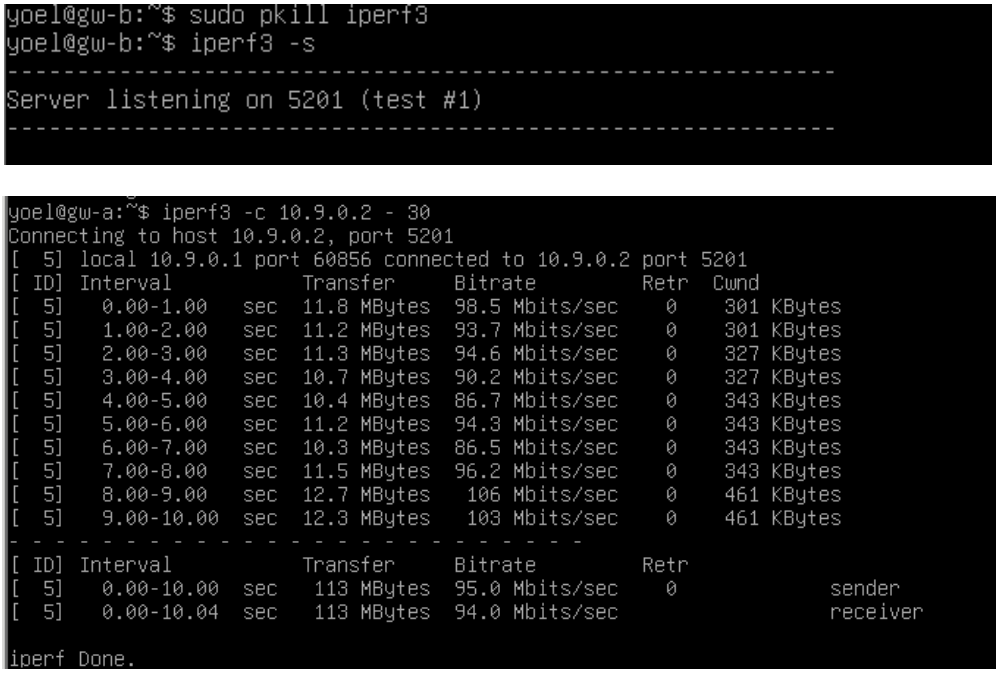


Gambar 5. Status koneksi WireGuard

3.4. Pengujian Throughput TCP

WireGuard mencapai throughput TCP rata-rata 94 Mbit/s, sedangkan OpenVPN hanya 27,5 Mbit/s, atau sekitar 3,4 kali lebih rendah. Perbedaan yang signifikan ini disebabkan oleh arsitektur WireGuard yang berjalan di dalam kernel sehingga meminimalkan overhead perpindahan data antara ruang kernel dan ruang pengguna. Sebaliknya, OpenVPN memproses paket di ruang pengguna dan memerlukan banyak system call serta penyalinan data melalui antarmuka tun, yang membatasi kapasitas penyaluran datanya.

Pengujian throughput dilakukan menggunakan aplikasi iperf3 selama 30 detik melalui tunnel VPN. Hasil pengujian menunjukkan bahwa WireGuard mampu menghasilkan throughput rata-rata sebesar 94 Mbps dengan total transfer data sebesar 113 MB.



Gambar 6. Hasil pengujian throughput TCP menggunakan iperf3

DOI: https://doi.org/10.31004/riggs.v5i2.11029
Lisensi: Creative Commons Attribution 4.0 International (CC BY 4.0)

Berdasarkan hasil pengujian menggunakan iperf3 selama 10 detik, WireGuard mampu menghasilkan throughput TCP rata-rata sebesar 94 Mbps dengan total transfer data mencapai 113 MB. Nilai throughput yang tinggi menunjukkan bahwa tunnel VPN yang dibangun dapat memanfaatkan kapasitas jaringan secara optimal dengan overhead yang rendah. Hasil ini mengindikasikan bahwa WireGuard mampu menjaga performa transmisi data meskipun proses enkripsi dan dekripsi dilakukan secara terus-menerus selama komunikasi berlangsung.

Untuk mempermudah analisis, hasil pengukuran throughput TCP dirangkum pada Tabel 3.

Tabel 3. Hasil Throughput TCP

Parameter	Nilai
Transfer Data	113 MB
Throughput TCP	94 Mbps

3.5. Pengujian Throughput UDP

Pengujian UDP dilakukan dengan bandwidth target 30 Mbps menggunakan iperf3. Hasil menunjukkan throughput rata-rata sebesar 29,8 Mbps dengan jitter 0,091 ms dan packet loss 0%.

```
yoel@gw-a:~$ iperf3 -c 10.9.0.2 -u -b 100M -t 30
Connecting to host 10.9.0.2, port 5201
[  5] local 10.9.0.1 port 45728 connected to 10.9.0.2 port 5201
[ ID] Interval           Transfer     Bitrate         Total Datagrams
[  5]   0.00-1.00   sec  3.64 MBytes  30.5 Mbits/sec  2787
[  5]   1.00-2.00   sec  3.54 MBytes  29.7 Mbits/sec  2715
[  5]   2.00-3.00   sec  3.58 MBytes  30.0 Mbits/sec  2742
[  5]   3.00-4.00   sec  3.56 MBytes  29.9 Mbits/sec  2728
[  5]   4.00-5.00   sec  3.55 MBytes  29.8 Mbits/sec  2722
[  5]   5.00-6.00   sec  3.55 MBytes  29.8 Mbits/sec  2720
[  5]   6.00-7.00   sec  3.55 MBytes  29.8 Mbits/sec  2719
[  5]   7.00-8.00   sec  3.56 MBytes  29.9 Mbits/sec  2729
[  5]   8.00-9.00   sec  3.52 MBytes  29.5 Mbits/sec  2696
[  5]   9.00-10.00  sec  3.54 MBytes  29.7 Mbits/sec  2712
[  5]  10.00-11.00  sec  3.57 MBytes  30.0 Mbits/sec  2737
[  5]  11.00-12.00  sec  3.53 MBytes  29.6 Mbits/sec  2709
[  5]  12.00-13.00  sec  3.54 MBytes  29.7 Mbits/sec  2714
[  5]  13.00-14.00  sec  3.55 MBytes  29.8 Mbits/sec  2719
[  5]  14.00-15.00  sec  3.54 MBytes  29.7 Mbits/sec  2717
[  5]  15.00-16.00  sec  3.56 MBytes  29.9 Mbits/sec  2728
[  5]  16.00-17.00  sec  3.57 MBytes  29.9 Mbits/sec  2733
[  5]  17.00-18.00  sec  3.58 MBytes  30.0 Mbits/sec  2746
[  5]  18.00-19.00  sec  3.56 MBytes  29.9 Mbits/sec  2729
[  5]  19.00-20.00  sec  3.57 MBytes  30.0 Mbits/sec  2739
[  5]  20.00-21.00  sec  3.60 MBytes  30.2 Mbits/sec  2758
[  5]  21.00-22.00  sec  3.56 MBytes  29.9 Mbits/sec  2728
[  5]  22.00-23.00  sec  3.55 MBytes  29.8 Mbits/sec  2719
[  5]  23.00-24.00  sec  3.56 MBytes  29.8 Mbits/sec  2725
[  5]  24.00-25.00  sec  3.54 MBytes  29.7 Mbits/sec  2714
[  5]  25.00-26.00  sec  3.57 MBytes  29.9 Mbits/sec  2735
[  5]  26.00-27.00  sec  3.54 MBytes  29.7 Mbits/sec  2711
[  5]  27.00-28.00  sec  3.58 MBytes  30.0 Mbits/sec  2741
[  5]  28.00-29.00  sec  3.58 MBytes  30.0 Mbits/sec  2745
[  5]  29.00-30.00  sec  3.54 MBytes  29.7 Mbits/sec  2710
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bitrate         Jitter    Lost/Total Datagrams
[  5]   0.00-30.00  sec   107 MBytes  29.9 Mbits/sec  0.000 ms  0/81827 (0%)  sender
[  5]   0.00-30.05  sec   107 MBytes  29.8 Mbits/sec  0.911 ms  0/81825 (0%)  receiver

iperf Done.
```

Gambar 7. Hasil pengujian throughput UDP

Pengujian throughput UDP dilakukan menggunakan iperf3 dengan target bandwidth sebesar 30 Mbps selama 30 detik. Berdasarkan hasil pengujian, tunnel WireGuard mampu mempertahankan throughput rata-rata sebesar 29,8 Mbps atau mendekati bandwidth yang ditentukan. Selain itu, nilai jitter yang diperoleh hanya sebesar 0,091 ms

DOI: https://doi.org/10.31004/riggs.v5i2.11029
Lisensi: Creative Commons Attribution 4.0 International (CC BY 4.0)

dengan packet loss sebesar 0%, yang menunjukkan bahwa kualitas transmisi data berlangsung sangat stabil. Rendahnya nilai jitter menunjukkan variasi waktu kedatangan paket yang sangat kecil, sedangkan packet loss 0% mengindikasikan bahwa seluruh paket berhasil diterima tanpa kehilangan selama proses pengiriman data. Hasil ini menunjukkan bahwa WireGuard mampu memberikan performa yang baik untuk layanan yang sensitif terhadap delay seperti VoIP, video conference, dan aplikasi real-time lainnya.

Ringkasan hasil pengujian throughput UDP ditunjukkan pada Tabel 4.

Tabel 4. Hasil Throughput UDP

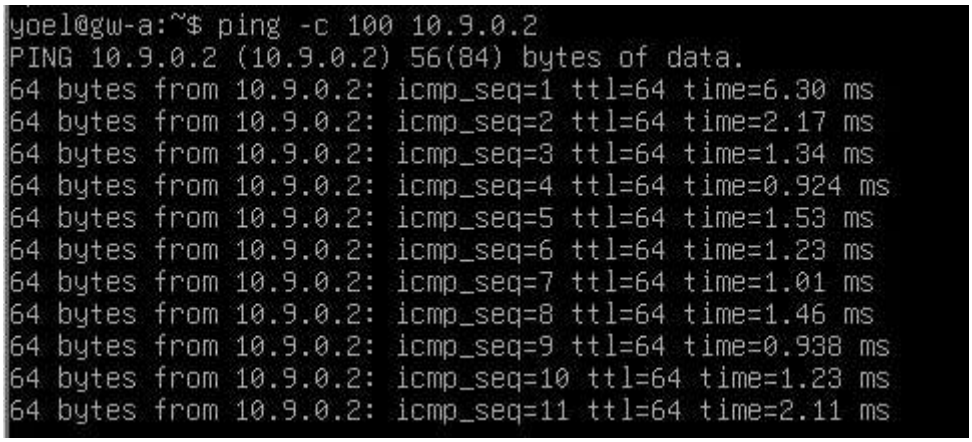
Parameter	Nilai
Transfer UDP	29,8 Mbps
Jitter	0,091 ms
Packet Loss	0%

3.6. Pengujian *Latency*

Latency rata-rata WireGuard sebesar 1,493 ms sedikit lebih rendah dibanding OpenVPN sebesar 1,659 ms. Keduanya tergolong rendah karena pengujian dilakukan pada jaringan lokal tervirtualisasi, namun overhead pemrosesan di ruang pengguna pada OpenVPN tetap menambah waktu tempuh paket sehingga latency-nya lebih tinggi.

Meskipun selisih latency pada lingkungan lokal ini tampak kecil, perbedaan tersebut diperkirakan akan membesar secara proporsional pada jaringan dengan beban tinggi atau kondisi jaringan yang lebih nyata. Hussein dan Rehman (2025) mencatat selisih yang jauh lebih besar pada pengujian berbasis internet, yaitu 8,01 ms pada WireGuard berbanding 160,38 ms pada OpenVPN, yang mengonfirmasi bahwa overhead ruang pengguna OpenVPN semakin terasa pada kondisi lalu lintas yang lebih berat.

Pengukuran latency dilakukan menggunakan 100 paket ICMP. Berdasarkan hasil pengujian diperoleh nilai rata-rata round-trip time sebesar 1,493 ms dengan packet loss 0%.



Gambar 7. Hasil pengujian latency menggunakan ping

Pengukuran latency dilakukan dengan mengirimkan 100 paket ICMP dari GW-A menuju GW-B melalui tunnel WireGuard menggunakan perintah ping. Berdasarkan hasil pengujian, diperoleh nilai RTT (Round Trip Time) minimum sebesar 0,860 ms, rata-rata 1,493 ms, dan maksimum 6,297 ms. Selain itu, tidak ditemukan kehilangan paket (packet loss 0%) selama pengujian berlangsung. Nilai latency yang rendah menunjukkan bahwa proses transmisi data melalui tunnel VPN dapat berlangsung dengan cepat dan stabil. Kondisi ini mengindikasikan bahwa WireGuard mampu memberikan respons jaringan yang baik sehingga cocok digunakan untuk komunikasi data yang membutuhkan waktu tanggap rendah.

Ringkasan hasil pengujian latency ditampilkan pada Tabel 5.

DOI: https://doi.org/10.31004/riggs.v5i2.11029
Lisensi: Creative Commons Attribution 4.0 International (CC BY 4.0)

Tabel 5. Hasil Latency

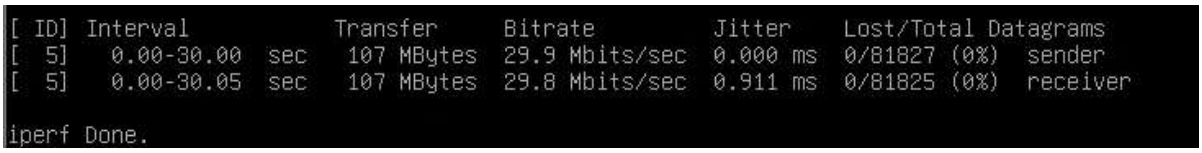
Parameter	Nilai
RTT Minimum	0,860 ms
RTT Rata-rata	1,493 ms
RTT Maksimum	6,297 ms
Packet Loss	0%

3.7. Pengujian *Jitter dan Packet Loss*

Jitter kedua protokol relatif kecil, dengan OpenVPN sedikit lebih rendah (0,627 ms) dibanding WireGuard (0,911 ms). Perbedaan yang paling mencolok terlihat pada packet loss ketika kedua protokol dibebani lalu lintas UDP sebesar 100 Mbit/s. WireGuard mencatat packet loss 0%, sedangkan OpenVPN mencapai 71%. Hal ini terjadi karena OpenVPN tidak mampu menyalurkan beban penuh sehingga banyak paket dibuang pada buffer antarmuka tun, sementara jalur kernel WireGuard secara otomatis menahan laju kirim sehingga seluruh paket yang dikirim berhasil diterima. Temuan ini menegaskan kapasitas tunnel WireGuard yang jauh lebih tinggi dibanding OpenVPN.

Nilai jitter OpenVPN yang sedikit lebih rendah kemungkinan disebabkan oleh laju pengiriman efektifnya yang lebih kecil, sehingga variasi antar-paket yang berhasil diterima menjadi lebih stabil. Dengan demikian, nilai jitter OpenVPN yang lebih baik tidak mencerminkan kualitas transmisi yang unggul, melainkan merupakan akibat langsung dari kapasitas tunnel yang lebih terbatas. Hasil ini sejalan dengan Anyam et al. (2025) yang juga melaporkan packet loss WireGuard jauh lebih rendah dibanding OpenVPN pada lingkungan VMware.

Berdasarkan hasil pengujian UDP menggunakan iperf3, diperoleh nilai jitter sebesar 0,091 ms dan packet loss sebesar 0%. Nilai jitter yang rendah menunjukkan bahwa variasi waktu kedatangan paket relatif stabil selama transmisi data berlangsung. Sementara itu, packet loss sebesar 0% menunjukkan bahwa seluruh paket yang dikirim berhasil diterima tanpa adanya kehilangan data. Hasil ini mengindikasikan bahwa tunnel WireGuard mampu menjaga kualitas layanan jaringan dengan baik, terutama untuk aplikasi yang sensitif terhadap keterlambatan dan kehilangan paket seperti VoIP, video conference, dan layanan multimedia real-time.



Gambar 8. Hasil Jitter dan Packet Loss

3.8. Pengujian *Utilisasi CPU*

Beban CPU pada kedua protokol didominasi oleh waktu kernel. Pada OpenVPN, tingginya komponen waktu sistem mencerminkan banyaknya system call serta penyalinan data akibat pemrosesan di ruang pengguna. Kondisi ini menunjukkan bahwa pada lingkungan dengan sumber daya terbatas seperti VM dengan 1 vCPU, OpenVPN mencapai batas kapasitasnya jauh lebih cepat dibanding WireGuard. Oleh karena itu, WireGuard lebih cocok diterapkan pada infrastruktur server dengan spesifikasi rendah hingga menengah yang mengutamakan efisiensi sumber daya komputasi.

Penggunaan CPU diamati menggunakan utilitas mpstat selama proses transfer data berlangsung. Hasil menunjukkan bahwa sistem masih mampu menangani lalu lintas VPN dengan baik meskipun beban CPU meningkat selama pengujian throughput.

DOI: https://doi.org/10.31004/riggs.v5i2.11029
Lisensi: Creative Commons Attribution 4.0 International (CC BY 4.0)

```
yoel@gw-a:~$ iperf3 -c 10.9.0.2 -t 60 >/dev/null 2>&1 &
[1] 929
yoel@gw-a:~$ mpstat 1 30
Linux 6.1.0-49-amd64 (gw-a)     06/20/2026      _x86_64_        (1 CPU)

12:03:05 PM  CPU    %usr   %nice    %sys %iowait    %irq   %soft  %steal  %guest  %gnice   %idle
12:03:06 PM  all    0.00    0.00   65.35    0.00    0.00   34.65    0.00    0.00    0.00    0.00
12:03:07 PM  all    0.00    0.00   69.31    0.00    0.00   30.69    0.00    0.00    0.00    0.00
12:03:08 PM  all    0.00    0.00   72.00    0.00    0.00   28.00    0.00    0.00    0.00    0.00
12:03:09 PM  all    0.00    0.00   75.00    0.00    0.00   25.00    0.00    0.00    0.00    0.00
12:03:10 PM  all    0.00    0.00   67.00    0.00    0.00   33.00    0.00    0.00    0.00    0.00
12:03:11 PM  all    0.00    0.00   67.00    0.00    0.00   33.00    0.00    0.00    0.00    0.00
12:03:12 PM  all    0.00    0.00   70.00    0.00    0.00   30.00    0.00    0.00    0.00    0.00
12:03:13 PM  all    0.00    0.00   68.00    0.00    0.00   32.00    0.00    0.00    0.00    0.00
12:03:14 PM  all    0.00    0.00   65.00    0.00    0.00   35.00    0.00    0.00    0.00    0.00
12:03:15 PM  all    0.00    0.00   65.00    0.00    0.00   35.00    0.00    0.00    0.00    0.00
12:03:16 PM  all    0.00    0.00   65.35    0.00    0.00   34.65    0.00    0.00    0.00    0.00
12:03:17 PM  all    0.00    0.00   66.00    0.00    0.00   34.00    0.00    0.00    0.00    0.00
12:03:18 PM  all    0.00    0.00   71.00    0.00    0.00   29.00    0.00    0.00    0.00    0.00
12:03:19 PM  all    0.00    0.00   69.00    0.00    0.00   31.00    0.00    0.00    0.00    0.00
12:03:20 PM  all    0.00    0.00   70.30    0.00    0.00   29.70    0.00    0.00    0.00    0.00
12:03:21 PM  all    0.00    0.00   70.00    0.00    0.00   30.00    0.00    0.00    0.00    0.00
12:03:22 PM  all    0.00    0.00   68.00    0.00    0.00   32.00    0.00    0.00    0.00    0.00
12:03:23 PM  all    0.00    0.00   69.00    0.00    0.00   31.00    0.00    0.00    0.00    0.00
12:03:24 PM  all    0.00    0.00   68.00    0.00    0.00   32.00    0.00    0.00    0.00    0.00
12:03:25 PM  all    0.00    0.00   73.00    0.00    0.00   27.00    0.00    0.00    0.00    0.00
12:03:26 PM  all    0.00    0.00   67.00    0.00    0.00   33.00    0.00    0.00    0.00    0.00
12:03:27 PM  all    0.00    0.00   72.00    0.00    0.00   28.00    0.00    0.00    0.00    0.00
12:03:28 PM  all    0.00    0.00   71.29    0.00    0.00   28.71    0.00    0.00    0.00    0.00
12:03:29 PM  all    0.00    0.00   68.00    0.00    0.00   32.00    0.00    0.00    0.00    0.00
12:03:30 PM  all    0.00    0.00   68.00    0.00    0.00   32.00    0.00    0.00    0.00    0.00
12:03:31 PM  all    0.00    0.00   68.00    0.00    0.00   32.00    0.00    0.00    0.00    0.00
12:03:32 PM  all    0.00    0.00   66.34    0.00    0.00   33.66    0.00    0.00    0.00    0.00
12:03:33 PM  all    0.00    0.00   73.00    0.00    0.00   27.00    0.00    0.00    0.00    0.00
12:03:34 PM  all    0.00    0.00   67.68    0.00    0.00   32.32    0.00    0.00    0.00    0.00
12:03:35 PM  all    0.00    0.00   71.29    0.00    0.00   28.71    0.00    0.00    0.00    0.00
Average:     all    0.00    0.00   68.86    0.00    0.00   31.14    0.00    0.00    0.00    0.00
yoel@gw-a:~$
```

Gambar 9. Monitoring penggunaan CPU menggunakan mpstat

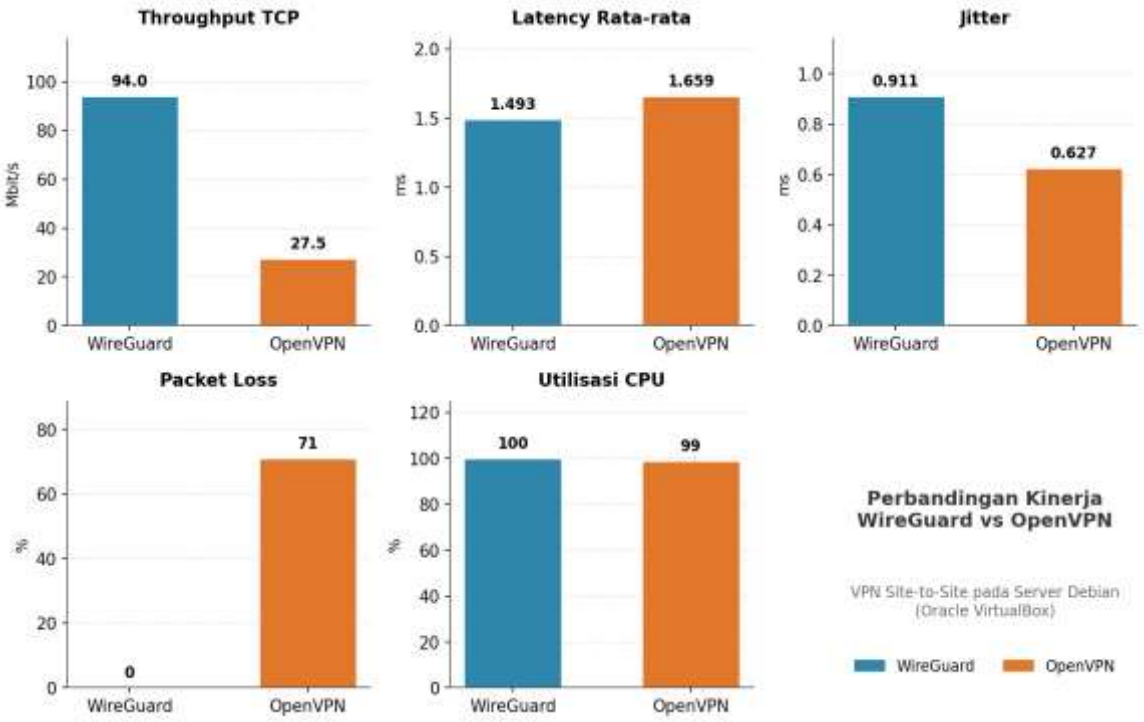
3.9. Diskusi

Hasil pengukuran kelima parameter kinerja untuk kedua protokol dirangkum pada Tabel 6 dan divisualisasikan pada Gambar 10. Secara umum, WireGuard menunjukkan kinerja yang lebih unggul pada hampir seluruh parameter yang diukur.

Tabel 6. Hasil perbandingan kinerja WireGuard dan OpenVPN

Parameter	WireGuard	OpenVPN	Keterangan
Throughput TCP (Mbit/s)	94	27,5	WG lebih tinggi
Latency rata-rata (ms)	1,493	1,659	WG lebih rendah
Jitter (ms)	0,911	0,627	OVPN lebih rendah
Packet loss (%)	0	71	WG lebih rendah
Utilisasi CPU (%)	100	99	Setara

DOI: https://doi.org/10.31004/riggs.v5i2.11029
Lisensi: Creative Commons Attribution 4.0 International (CC BY 4.0)



Gambar 10. Perbandingan lima parameter kinerja WireGuard dan OpenVPN

4. Kesimpulan

Berdasarkan hasil pengujian, WireGuard menunjukkan kinerja yang lebih unggul dibanding OpenVPN pada implementasi VPN site-to-site di server Debian untuk empat dari lima parameter, yaitu throughput TCP sebesar 94 Mbit/s berbanding 27,5 Mbit/s, latency rata-rata 1,493 ms berbanding 1,659 ms, packet loss 0% berbanding 71% pada beban UDP 100 Mbit/s, serta efisiensi pemanfaatan CPU yang jauh lebih tinggi pada tingkat utilisasi prosesor yang setara. OpenVPN hanya sedikit lebih baik pada parameter jitter, yaitu 0,627 ms berbanding 0,911 ms. Keunggulan WireGuard berakar pada arsitekturnya yang berjalan di dalam kernel sehingga mengurangi overhead pemrosesan dibanding OpenVPN yang beroperasi di ruang pengguna. Dengan demikian, WireGuard direkomendasikan untuk kebutuhan VPN site-to-site yang mengutamakan throughput tinggi dan efisiensi sumber daya. Penelitian selanjutnya disarankan untuk menguji pada perangkat keras nyata, menggunakan beban jaringan yang lebih variatif, serta menambah jumlah iterasi pengujian agar diperoleh nilai rata-rata dan simpangan baku yang lebih representatif.

Referensi

1. Akamai Technologies. (2025). State of the internet security report 2025.
2. Ali, S., Pratama, R., & Nugroho, H. (2025). VPERFSEC: VPN performance and security evaluation model in Indonesia. Journal of Information Security Research, 11(3), 101–115.
3. Anyam, P., Smith, J., & Kumar, A. (2025). Performance comparison of WireGuard and OpenVPN in Azure and VMware environments. Journal of Cloud Computing, 14(2), 45–58.
4. Bernstein, D. J. (2006). Curve25519: New Diffie-Hellman speed records. In Public Key Cryptography (PKC 2006) (pp. 207–228). Springer. https://doi.org/10.1007/11745853_14
5. Chen, L., Wu, Y., & Zhao, H. (2023). Evaluating VPN tunneling protocols in virtualized infrastructures. Electronics, 12(7), 1635. https://doi.org/10.3390/electronics12071635
6. Dekker, M., & Spaans, M. (2020). Performance comparison between WireGuard, strongSwan and OpenVPN on high-speed networks. Network Engineering Journal, 8(2), 88–97.
7. Donenfeld, J. A. (2017). WireGuard: Next generation kernel network tunnel. In Proceedings of the Network and Distributed System Security Symposium (NDSS). https://doi.org/10.14722/ndss.2017.23160
8. Garcia, P., Lopez, J., & Torres, M. (2022). Security and performance trade-offs in VPN implementations: A comparative study. Computer Networks, 216, 109246. https://doi.org/10.1016/j.comnet.2022.109246
9. Hussein, M., & Rehman, A. (2025). A comparative analysis of VPN protocol performance on Windows 11 platforms. International Journal of Computer Networks, 18(1), 22–31.
10. Identity Theft Resource Center. (2024). Data breach report 2024.

DOI: https://doi.org/10.31004/riggs.v5i2.11029
Lisensi: Creative Commons Attribution 4.0 International (CC BY 4.0)

11. Khan, M. A., Rehman, S., & Ahmed, I. (2021). Comparative analysis of modern VPN protocols for secure communications. IEEE Access, 9, 145231–145245. https://doi.org/10.1109/ACCESS.2021.3121456
12. Langley, A., Hamburg, M., & Turner, S. (2016). Elliptic curves for security (RFC 7748). Internet Engineering Task Force. https://doi.org/10.17487/RFC7748
13. Mackey, T. K., Khera, B., & Cuomo, R. (2020). Evaluating the performance of WireGuard and OpenVPN in virtualized environments. International Journal of Network Security, 22(4), 567–575.
14. Nguyen, T., Pham, D., & Le, H. (2024). Resource utilization and throughput comparison of contemporary VPN protocols. Sensors, 24(4), 1372. https://doi.org/10.3390/s24041372
15. Nir, Y., & Langley, A. (2018). ChaCha20 and Poly1305 for IETF protocols (RFC 8439). Internet Engineering Task Force. https://doi.org/10.17487/RFC8439
16. OpenVPN Inc. (2024). OpenVPN reference manual. https://openvpn.net/community-resources/
17. Patel, R., Singh, A., & Kumar, V. (2022). Benchmarking WireGuard and OpenVPN under enterprise network workloads. Journal of Network and Computer Applications, 203, 103389. https://doi.org/10.1016/j.jnca.2022.103389
18. Rahman, F., Ismail, N., & Abdullah, M. (2023). Performance analysis of WireGuard for secure site-to-site communication. International Journal of Advanced Computer Science and Applications, 14(5), 412–420. https://doi.org/10.14569/IJACSA.2023.0140545
19. Silva, J., Costa, R., & Pereira, A. (2024). WireGuard versus OpenVPN: A performance assessment in virtual private networks. Applied Sciences, 14(3), 1187. https://doi.org/10.3390/app14031187
20. We Are Social. (2025). Digital 2025 global overview report.
21. Zhang, Y., Li, H., & Wang, X. (2021). Performance evaluation of VPN protocols in cloud computing environments. Future Internet, 13(8), 201. https://doi.org/10.3390/fi13080201

DOI: https://doi.org/10.31004/riggs.v5i2.11029
Lisensi: Creative Commons Attribution 4.0 International (CC BY 4.0)