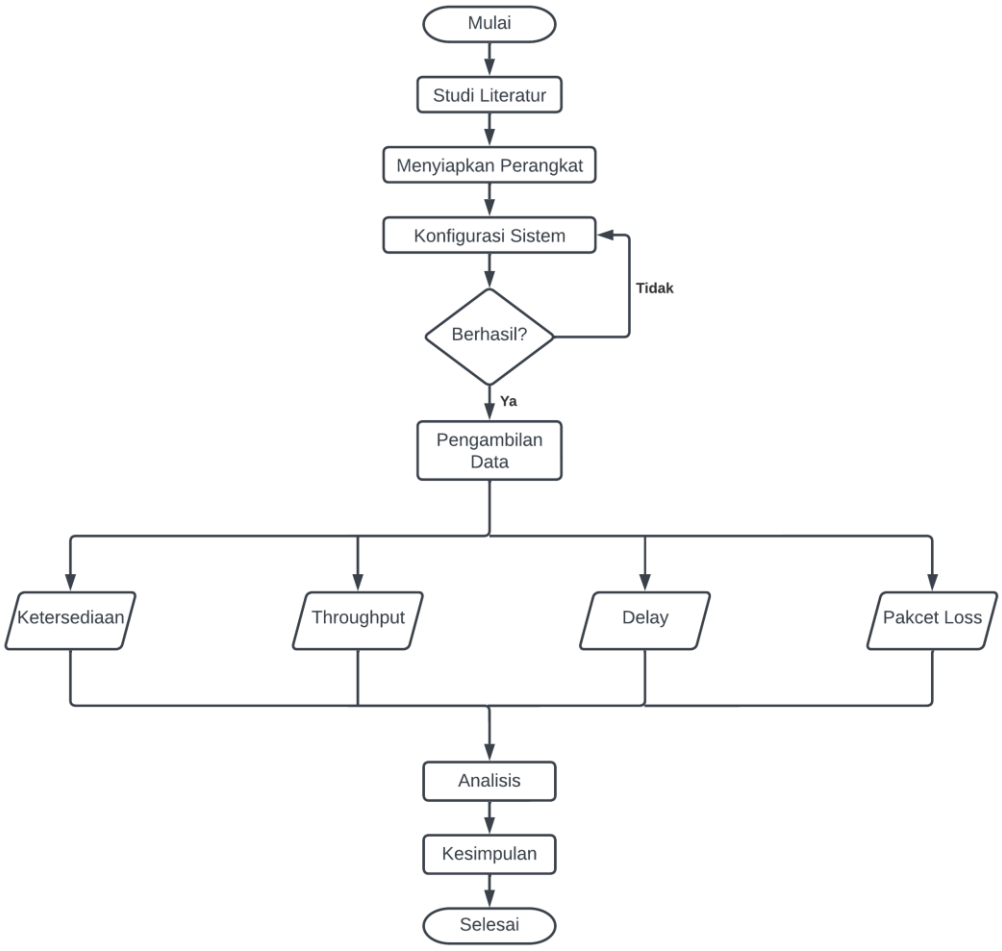
BAB 3
METODE PENELITIAN

3.1 ALUR PENELITIAN

Penelitian ini menggunakan beberapa tahapan yang sudah ditentukan, agar hasil penelitian ini tidak menyimpang. Tahapan-tahapan yang dilakukan penulis untuk menyusun penelitian ini direpresentasikan pada diagram alir di bawah ini.



Gambar 3.1 Diagram Alir Penelitian

Gambar 3.1 menjelaskan bagaimana alur penelitian ini dengan menggunakan 4 parameter yaitu ketersediaan, *throughput*, *delay*, dan *packet loss.*

3.2 PERSIAPAN PERANGKAT

Untuk mempersiapkan perangkat yang dibutuhkan, diperlukan sistem pendukung yang sesuai dengan kebutuhan dalam menerapkan skema yang akan dibuat. Sistem pendukung tersebut terdiri dari perangkat keras (*hardware*) dan perangkat lunak (*software*). Untuk menjalankan skema tersebut, diperlukan perangkat dengan spesifikasi sebagai berikut:

3.2.1 Spesifikasi *Hardware*

Penelitian ini menggunakan hardware berupa 3 buah PC, 2 buah Mikrotik, dan 4 kanel UTP untuk menghubungkan Miktotik dengan router utama dan Mikrotik dengan PC. Pada tabel 3.1, menjelaskan spesifikasi perangkat keras yang akan digunakan untuk melakukan penelitian ini:

Tabel 3.1 Spesifikasi *Hardware*

No	Perangkat	Spesifikasi	Fungsi	Jumlah
1	*Webserver*	• *Processor* Intel Core i7-7700 (2 CPUs) • RAM 3 GB, • *Hardisk* 80 GB	*Webserver* dengan OS Ubuntu 22.04 LTS	1
2	PC	• *Processor* Intel Core i7-7700 • RAM 8 GB, • *Hardisk* 1 TB	*Client* dengan OS Windows 10	1
3	PC	• *Processor* Intel Core i7-7700 (2 CPUs) • RAM 8 GB, • *Hardisk* 1 TB	*Attacker* dengan OS Windows 10	1
4	Mikrotik (RB952Ui-5ac2nD-TC)	• CPU QCA9531 650MHz • *Main Storage*/NAND 16MB • RAM 64MB	Sebagai *router* penghubung jaringan dan VPN	2

3.2.2 Spesifikasi *Software*

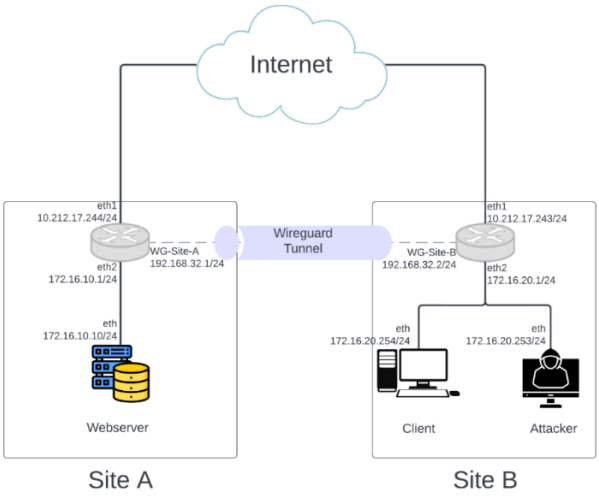
Tabel 3.2 menjelaskan spesifikasi perangkat lunak yang akan digunakan untuk melakukan penelitian ini:

Tabel 3.2 *Software* yang Digunakan

No	*Software*	Fungsi
1	*Virtual*box	Untuk membuat *virtual*isasi perangkat PC
2	Ubuntu 22.04 LTS	Sistem *webserver* dan *attacker*
3	Windows 10	Sistem operasi *client* dan *attacker*
4	Wireshark	*Tool* untuk melakukan menganalisis QOS pada jaringan dalam penelitian ini
5	LOIC	*Tool* untuk melakukan serangan DDOS/IP Floods
6	PHP 7.4	Bahasa pemrograman yang digunakan DVWA
7	Apache2	*Webserver*
8	Winbox	*Software* untuk konfigurasi MikroTik

3.3 PERANCANGAN TOPOLOGI JARINGAN

Topologi jaringan adalah suatu penentuan mengenai bagaimana struktur jaringan yang cocok untuk sistem yang sedang dikembangkan seperti yang digambarkan pada gambar 3.2 dan tabel 3.3 menjelaskan pengalamatan IP *address*. Hal ini bertujuan agar dapat memberikan gambaran yang jelas mengenai sistem yang akan dibangun.



Gambar 3.2 Topologi Jaringan

Tabel 3.3 Pengalamatan IP

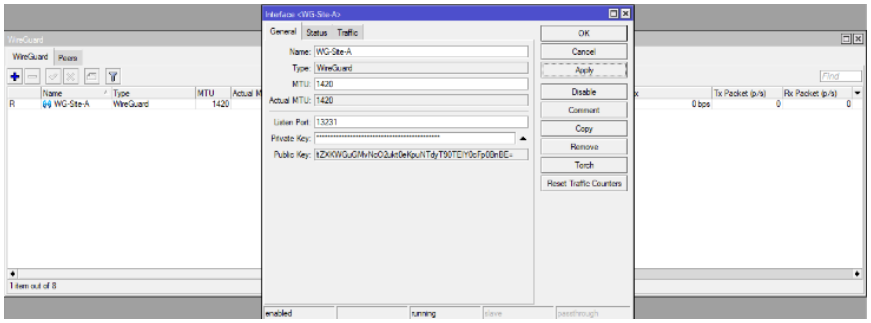
Perangkat	*Interfaces*	IP *Address*
Mikrotik *Site* A	*Ethernet* 1	10.17.212.17.244 /24
	Ethernet 2	172.16.10.1 /24
	WG-*Site*-A	192.168.32.1 /24
Mikrotik *Site* B	*Ethernet* 1	10.17.212.17.243 /24
	Ethernet 2	172.16.20.1 /24
	WG-*Site*-B	192.168.32.2 /24
Webserver	*Ethernet*	172.16.10.10
Client	*Ethernet*	172.16.10.254
Attacker	*Ethernet*	172.16.10.254

3.4 KONFIGURASI WIREGUARD

Untuk menghubungkan dua jaringan lokal pada *site* yang berbeda, digunakan *tunneling* VPN menggunakan protokol WireGuard. Tahapan dalam melakukan konfigurasi WireGuard adalah:

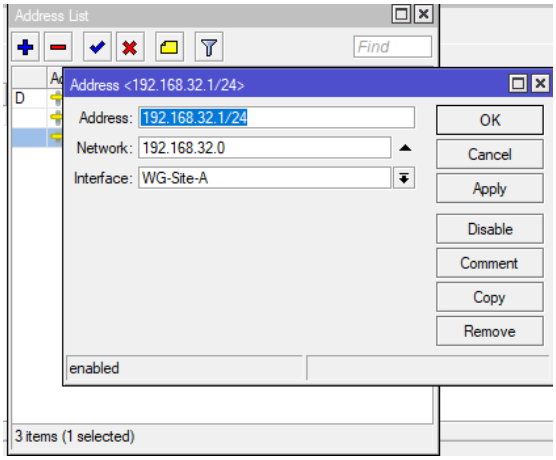
1) Konfigurasi pada *Site* A

   Dalam konfigurasi ini, dimulai dengan membuat *interface* baru untuk WireGuard dengan nama ”WG-*Site*-A” dan *port* yang digunakan secara *default* adalah 13231. Kemudian WireGuard secara otomatis akan men*generate private key* dan *public key* yang digunakan untuk verifikasi identitas dan digital *signature* untuk proses pengamanan atau enkripsi seperti pada gambar 3.3.



Gambar 3.3 Penambahan *Interface* WireGuard

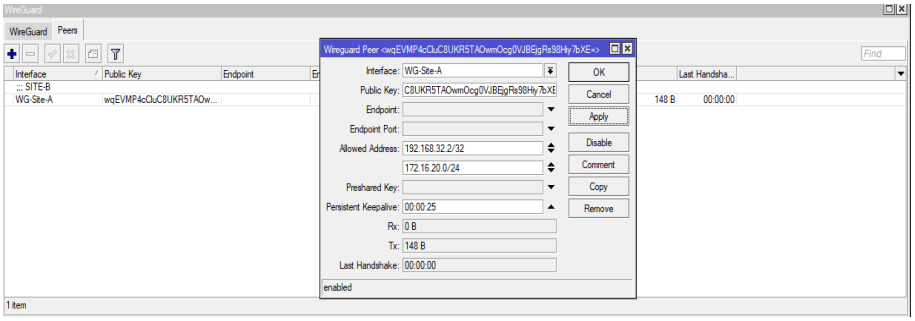
Gambar 3.4 merupakan penambahan IP *address* untuk *interface* ”WG-*Site*-A” untuk bisa berkomunikasi dengan klien manapun yang tergabung dalam jaringan WireGuard. Nantinya, IP dalam *interface* WireGuard di klien *site* manapun akan dijadikan satu alamat *network*.



Gambar 3.4 Penambahan IP *Address*

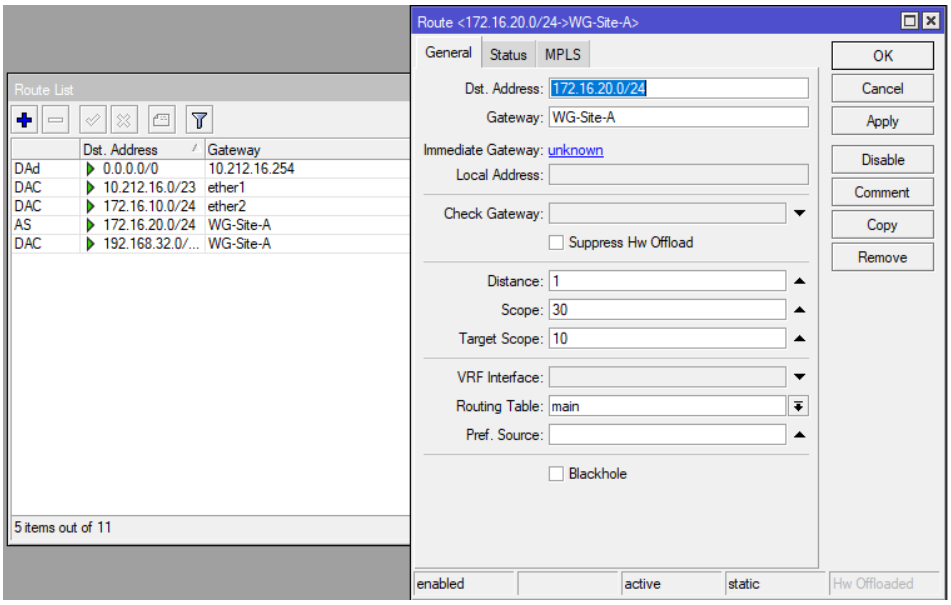
Untuk menghubungkan jaringan antar *site* yang menggunakan protokol WireGuard, diperlukan penambahan *Peer*s atau *router* pada setiap *site*. Pada penelitian ini, *Peer*s yang ditambahkan terletak pada *Site* B, dengan cara memasukkan *public key* yang dimiliki oleh *router* pada *Site* B ke dalam konfigurasi *router* pada *Site* A. Hal ini memungkinkan kedua *router* untuk terhubung dan membentuk *tunnel* VPN yang aman untuk mengirimkan data antar *site*.

Namun, untuk memastikan jaringan dalam *tunnel* tetap hidup bahkan ketika tidak ada *traffic* yang sedang mengalir, diatur presistent *keepalive* pada 25 detik. Fungsi dari persistent keep*alive* adalah untuk mengecek koneksi antara kedua *Peer*s secara berkala dan mengirimkan paket data kecil yang tidak membebani jaringan, sehingga dapat mempertahankan koneksi VPN dalam kondisi yang stabil. Dengan demikian, *Peer*s atau *router* pada kedua *site* dapat terus berkomunikasi dan mempertukarkan data dengan aman melalui jaringan VPN yang telah dibentuk. Gambar 3.5 menjelaskan cara konfigurasinya.



Gambar 3.5 Penambahan *Peer*s

Untuk membuat jaringan lokal antar *site* bisa berkomunikasi, maka dibuat jalur *routing* menggunaka *routing* statis seperti pada gambar 3.6. Dengan menggunakan jalur *routing* statis, pengguna dapat menentukan secara manual rute yang harus diambil oleh paket data saat melewati jaringan tersebut. Hal ini memungkinkan pengguna untuk mengatur dan mengoptimalkan lalu lintas jaringan dengan lebih baik, serta memperkuat keamanan dan stabilitas jaringan. Dalam pengaturan jalur *routing* statis, pengguna harus menentukan subnet jaringan yang akan dihubungkan dan menambahkan informasi rute yang sesuai ke dalam tabel *routing* perangkat jaringan.

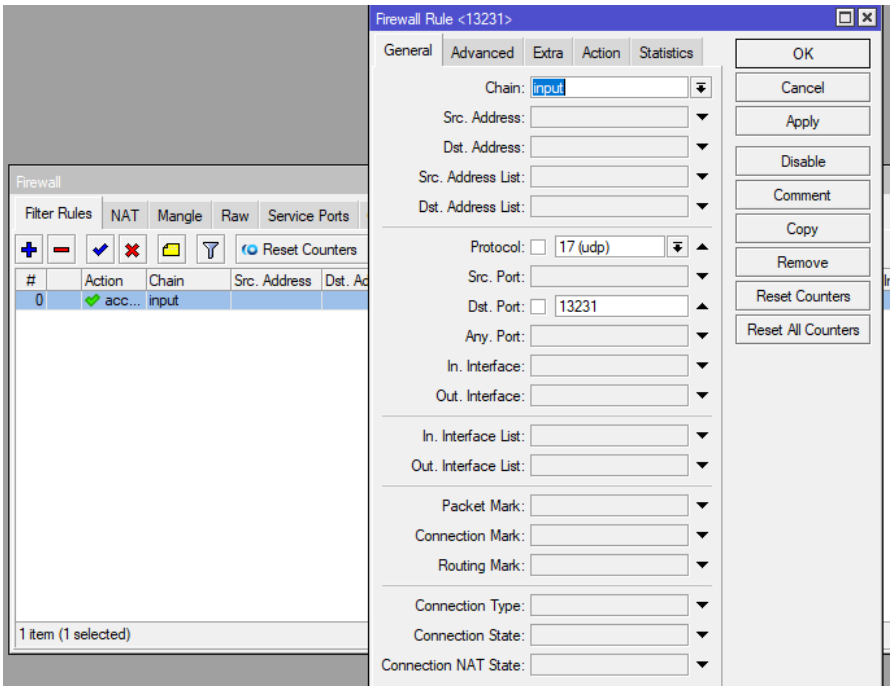


Gambar 3.6 *Static Routing*

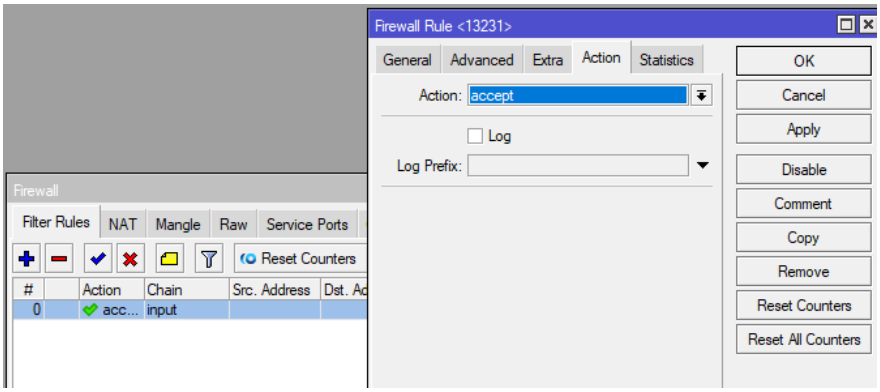
Ketika menggunakan protokol WireGuard pada jaringan VPN yang dibangun dengan Mikrotik, terkadang terdapat kendala di mana *firewall* Mikrotik secara *default* akan memblokir *port* 13231 pada protokol UDP. Padahal *port* 13231 adalah *port* yang digunakan oleh WireGuard sebagai *listening port* untuk menghubungkan *Peer*s atau *router* antar *site*.

Untuk mengatasi masalah ini, kita perlu menambahkan *rules* pada *firewall* Mikrotik yang mengizinkan *port* 13231 untuk berjalan pada protokol UDP seperti gambar 3.7 dan gambar 3.8. Dalam hal ini, kita dapat membuat

sebuah *rule* baru pada Mikrotik yang mengizinkan lalu lintas pada *port* 13231 dengan protokol UDP, baik untuk incoming maupun outgoing.



Gambar 3.7 Penambahan Filter *Rules*



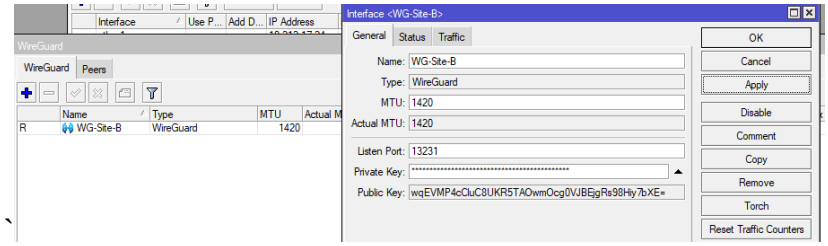
Gambar 3.8 Penambahan Filter *Rules*

2) Konfigurasi pada *Site* B

Konfigurasi pada *Site* B kurang lebih sama dengan konfigurasi pada *Site* A, yang membedakan adalah *rules* yang dipakai dalam *router* ini.

a. Penambahan *interface* ”WG-*Site*-B”

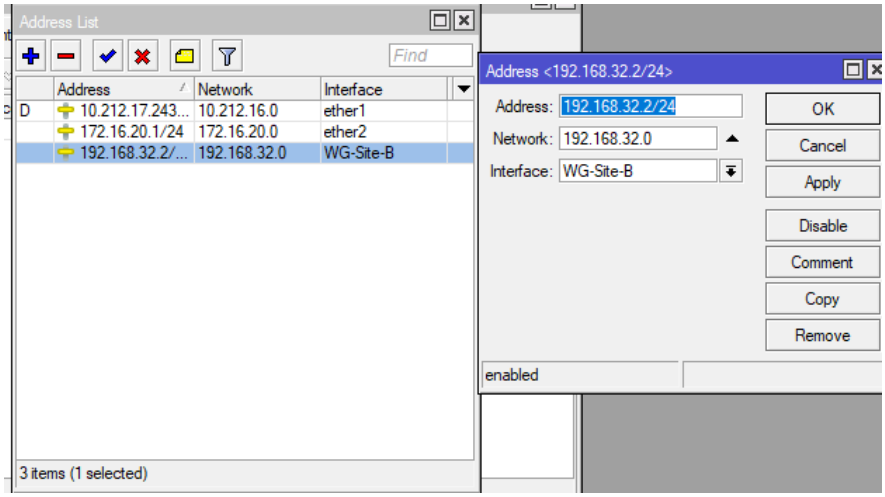
Gambar 3.9 menjelaskan konfigurasi penambahan *interfaces* pada *router site* B.



Gambar 3.9 Penambahan *Interface* WireGuard

b. Penambahan IP *Address*

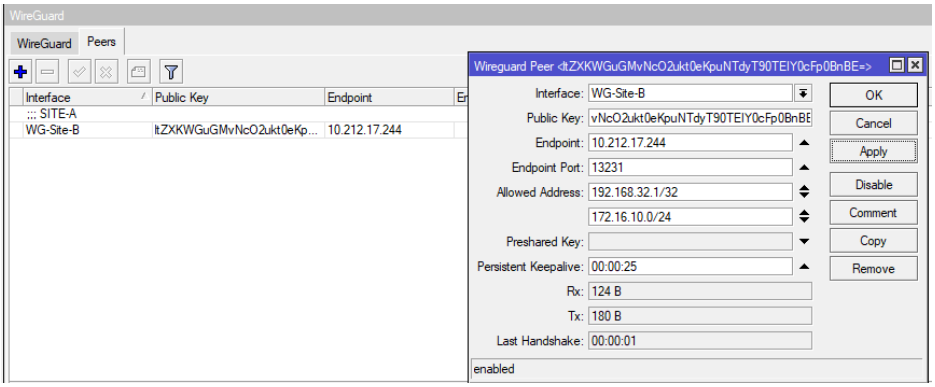
Gambar 3.10 menjelaskan konfigurasi penambahan IP *address* pada *router site B.*



Gambar 3.10 Penambahan IP *Address*

c. Penambahan *Peer*s

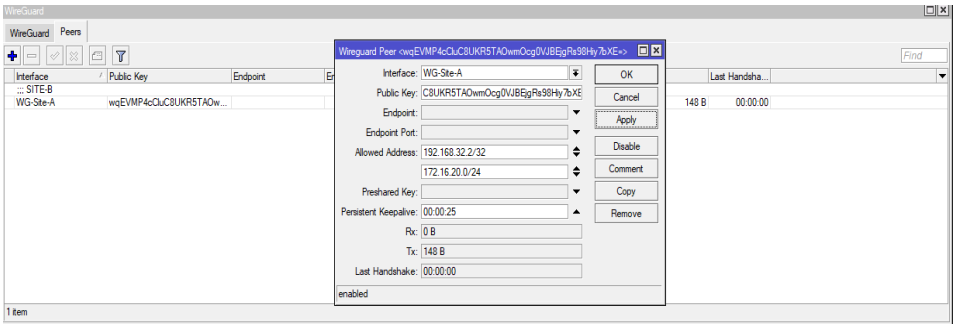
Gambar 3.11 menjelaskan konfigurasi penambahan *interfaces* pada *router site B.*



Gambar 3.11 Penambahan *Peer*s

d. Penambahan *Routing* Statis

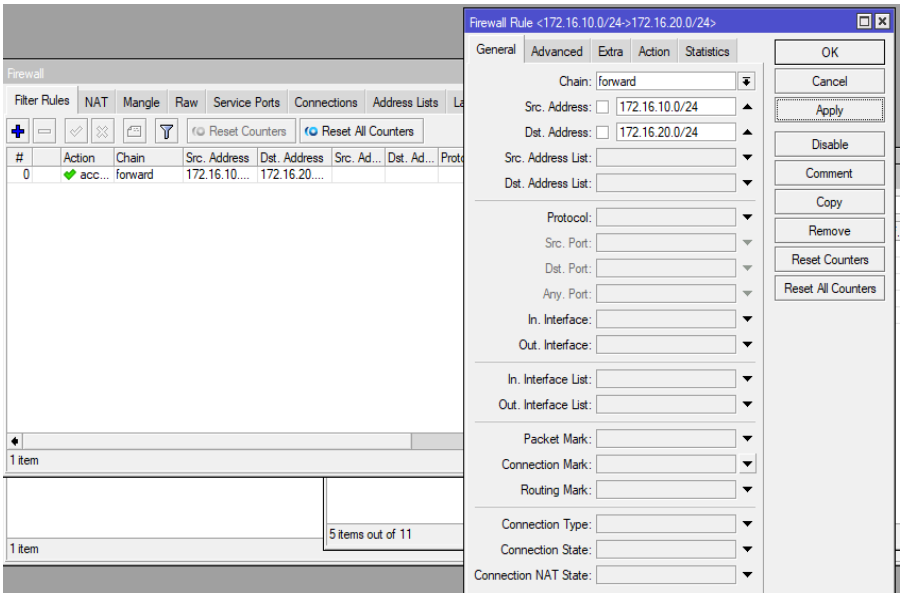
Gambar 3.12 menjelaskan konfigurasi penambahan *Routing* statis pada *router site B.*



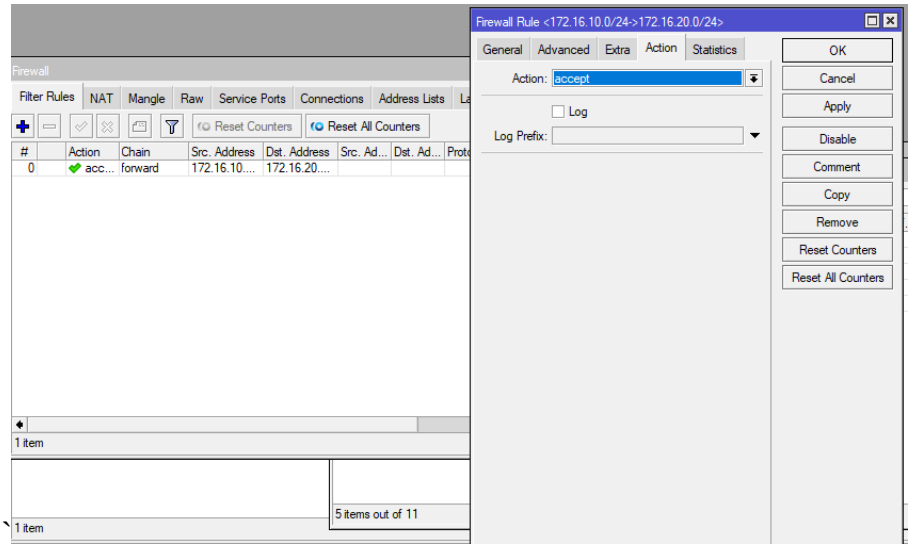
Gambar 3.12 Penambahan *Routing* Statis

e. Penambahan *rules* pada *firewall*

Rules yang ditambahkan berbeda dengan *rules* yang ada pada *router site* A. Dimana *rules* yang ditambahkan adalah memberi izin kepada lalu lintas apapun antar subnet yang akan diteruskan melalui *tunnel. Rules* pertama pada gambar 3.13 dan gambar 3.14 menunjukan *chain forward*, pada kolom Src. *Address* dimasukan alamat *network* dari jaringan lokal *site* A dan kolom Dst. *Address* ditambahkan alamat *Network* dari jaringan lokal *site* B dengan *action accept*.

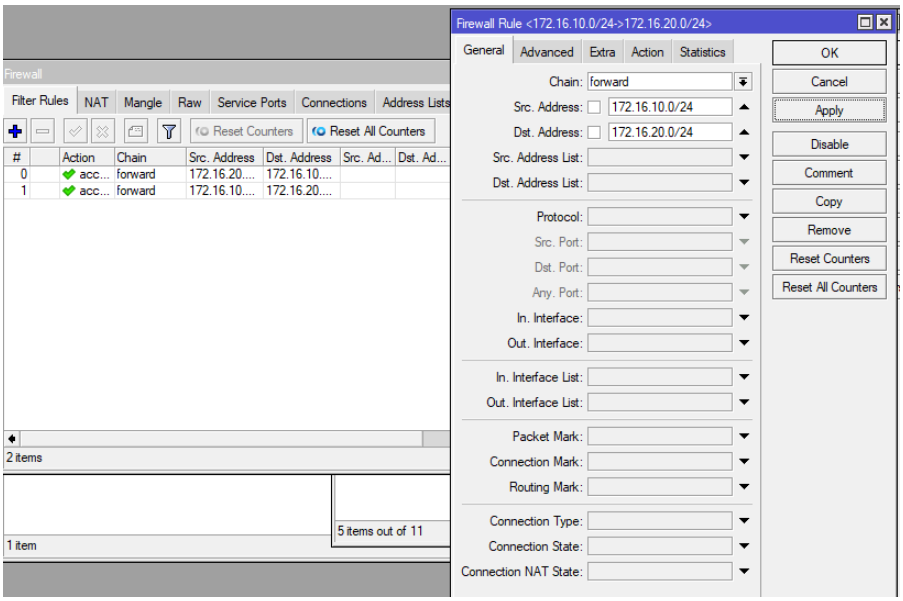


Gambar 3.13 Penambahan *Rules* pada *Firewall*

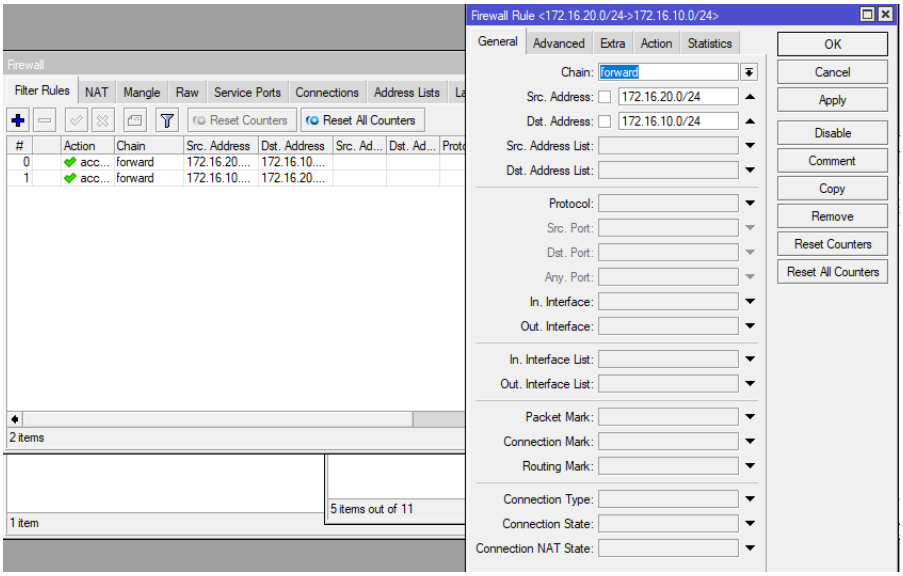


Gambar 3.14 Penambahan *Rules* pada *Firewall*

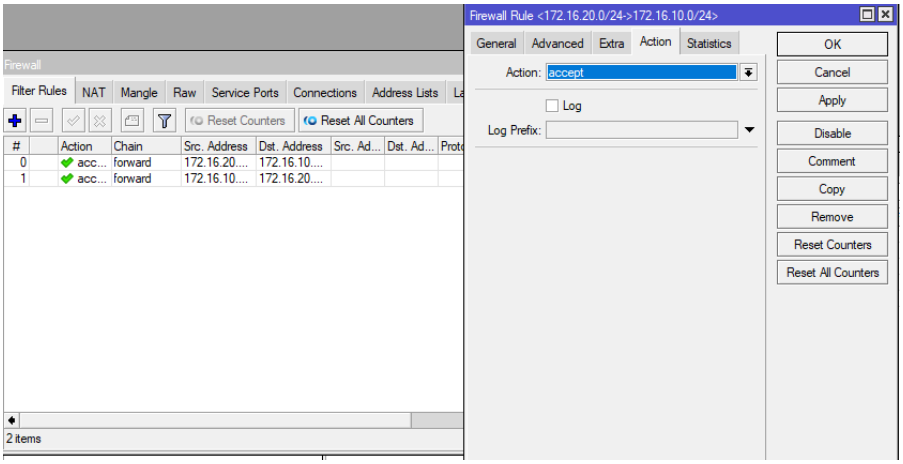
Rules kedua pada gambar 3.15, gambar 3.16 dan gambar 3.17 berkebalikan dari rules pertama dimana pada kolom Src. *Address* dimasukan alamat *network* dari jaringan lokal *site* B dan kolom Dst. *Address* ditambahkan alamat *Network* dari jaringan lokal *site* A dengan *action accept*.



Gambar 3.15 Penambahan R*ules* pada *Firewall*



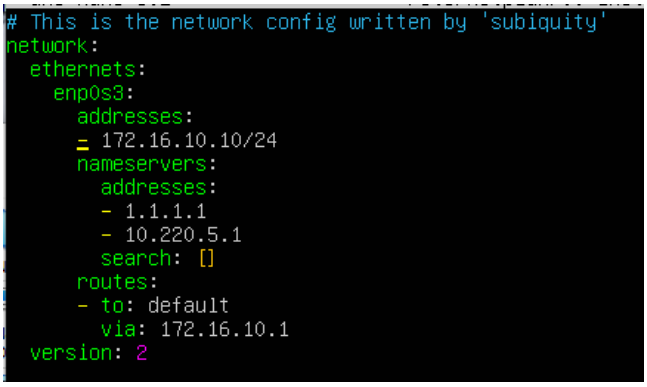
Gambar 3.16 Penambahan R*ules* pada *Firewall*



Gambar 3.17 Penambahan *Rules* pada *Firewall*

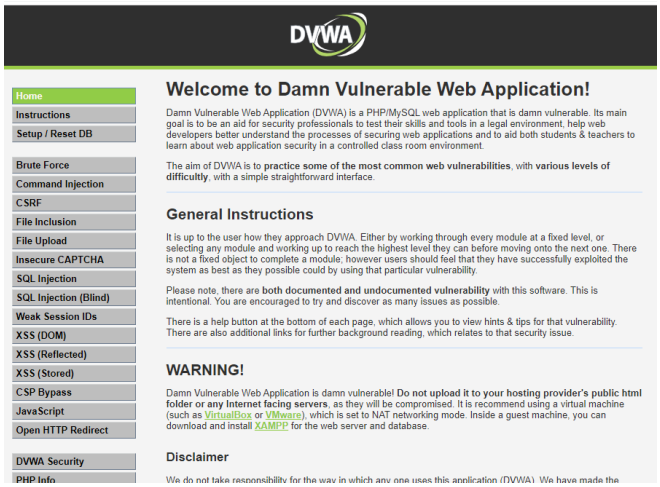
3.5 KONFIGURASI SERVER

Server disini merupakan sebuah *webserver* yang diletakan pada jaringan lokal pada *site* A.Server pada penelitian kali ini menggunakan sistem operasi Ubuntu 22.04.02 LTS. Nantinya server akan diakses oleh klien menggunakan IP *address* ini. IP *Address* pada server ditambahkan secara *static* dengan rincian ditampilkan pada gambar 3.18. IP *address* dari server adalah 172.16.10.10 dengan *subnetmask* 255.255.255.0



Gambar 3.18 Penambahan IP Server

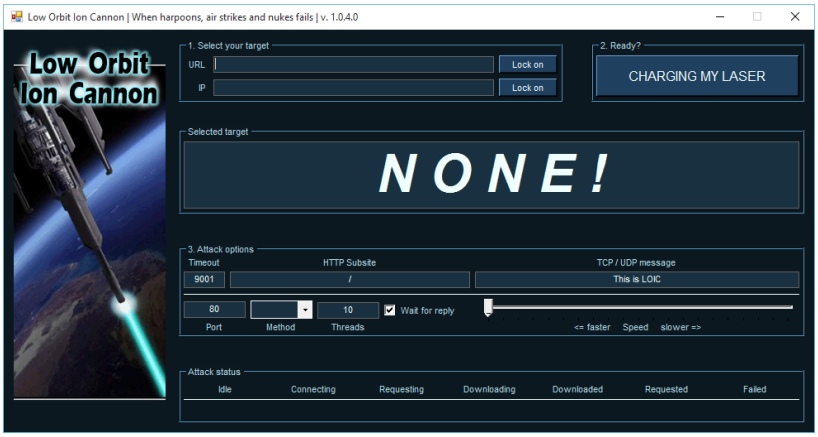
Berikut adalah tampilan *web* yang telah dikonfigurasi pada *webserver.*



Gambar 3.19 Tampilan *Webserver*

3.6 KONFIGURASI *ATTACKER*

Sistem operasi yang digunakan dalam penelitian ini adalah Windows 10 dengan menggunakan *tool* LOIC (*Low Orbit Ion Cannon*) untuk melakukan DDOS *Attack*. LOIC adalah *tool open-source* yang digunakan untuk melakukan serangan *Denial-of-Service* (DDOS) pada suatu *website* atau jaringan. Dalam penelitian keamanan digital, LOIC dapat digunakan sebagai *tool* untuk menguji kekuatan suatu *website* atau jaringan dalam menghadapi serangan DDOS. Penggunaan *tool* seperti LOIC dalam penelitian keamanan digital memungkinkan para profesional keamanan untuk memperkuat sistem keamanan jaringan dan aplikasi agar lebih tahan terhadap jenis serangan DDoS. Tampilan awal dari aplikasi LOIC ini bisa dilihat pada gambar 3.20.



Gambar 3.20 Tampilan LOIC

3.7 SKEMA PENGUJIAN

Dalam skenario ini, seorang penyerang berupaya untuk menyusup ke dalam jaringan lokal di *site* B dengan niatan melakukan serangan dari dalam jaringan tersebut. Pada saat yang sama, seorang klien mengunggah *file* berukuran 66 MB ke *webserver*. Selama proses ini, penyerang akan melancarkan serangan DDoS terhadap *webserver* sebagai targetnya.

Pada sekenario pertama, serangan dilakukan tanpa membatasi alamat IP dalam konfigurasi WireGuard. Kemudian, pada sekenario kedua, dilakukan setelah mengaktifkan pembatasan alamat IP dalam konfigurasi WireGuard. Dalam eksperimen ini, tujuannya adalah untuk mengevaluasi sejauh mana kemampuan protokol WireGuard dalam menghadapi serangan serta menjaga ketersediaan layanan pada server yang terhubung. Dalam tabel 3.4 dijabarkan bagaimana sekenario pengujian dijalankan.

Tabel 3.4 Sekenario Pengujian

Sekenario	*Upload File*	DDoS Attack	Pembatasan IP *Address*	Percobaan
1	66 MB	Ya	Tidak	5 kali
2	66 MB	Ya	Ya	5 kali

3.8 PENGAMBILAN HASIL DATA DAN ANALISIS

Pada penelitian ini, pengambilan hasil data sudah diatur berdasarkan beberapa parameter yang telah ditentukan. Setelah data terkumpul, langkah selanjutnya adalah menganalisis parameter-parameter tersebut untuk menghasilkan

sebuah evaluasi komprehensif mengenai performansi protokol WireGuard. Dalam proses analisis, data-data yang terkumpul akan dianalisis secara statistik dan dibandingkan antara sekenario 1 dan sekenario 2. Hasil analisis ini akan menghasilkan sebuah evaluasi komprehensif tentang keamanan dan kinerja protokol WireGuard dalam hal ketersediaan, *throughput*, *delay*, dan *packet loss*. Evaluasi ini dapat mengidentifikasi kelebihan dan kekurangan protokol, serta memberikan wawasan tentang area-area yang perlu diperbaiki atau dioptimalkan. Dengan adanya informasi mendalam yang dihasilkan melalui evaluasi ini, akan muncul wawasan berharga mengenai berbagai aspek yang dapat dioptimalkan atau diperbaiki pada protokol WireGuard. Melalui langkah-langkah perbaikan yang ditunjukkan oleh evaluasi, diharapkan bahwa protokol ini dapat mengalami peningkatan yang signifikan dalam hal kinerja dan keamanan.

3.9 PARAMETER

Untuk menguji keamanan dan performansi pada jaringan VPN menggunakan protokol WireGuard, dilakukan serangan DDoS menggunakan beberapa parameter yang diujikan diantaranya ketersediaan, *Throughput*, *Delay*, dan *Packet Loss* seperti yang dijelaskan pada tabel 3.5.

Tabel 3.5 Parameter Pengujian

Threads Request	Percobaan	Parameter
100	5 kali	Ketersediaan
		Throughput
		Delay
		Packet Loss
1000	5 kali	Ketersediaan
		Throughput
		Delay
		Packet Loss
2000	5 kali	Ketersediaan
		Throughput
		Delay
		Packet Loss
10000	5 kali	Ketersediaan
		Throughput
		Delay
		Packet Loss