Jurnal Impresi Indonesia (JII)
Vol. 1, No. 12, Desember 2022
p-ISSN: 2828-1284 e-ISSN: 2810-062x
website: https: //rivierapublishing.id/JII/index.php/jii/index

IMPLEMENTASI *VIRTUAL PRIVATE NETWORK* (VPN) *OPEN VPN* DENGAN KEAMANAN SERTIFIKAT SSL PADA *NETWORK ATTACHED STORAGE* (NAS) FREENAS

Muhammad Affandi
Fakultas Teknik, Universitas Tanjungpura, Pontianak, Indonesia
m_affandi@student.untan.ac.id

Received: 28-11-2022
Accepted: 10-12-2022
Published: 20-12-2022

***Keywords*:** *Virtual Private Networks; OpenVPN; SSL; Network Attached Storage; FreeNAS*

Abstract

***Introduction**: The development of information and communication technology is very fast and rapid, this has led to the emergence of advances in information technology. Because of the many conveniences that are offered, information technology cannot be separated from various aspects of human life, which makes it possible to communicate and exchange information or data. Often security issues come second, or even last, on the list of things considered important. **Purpose**: produce a private network and secure data on a Network Attached Storage (NAS) with an SSL-based security system that can be accessed privately from the public network using VPN technology. **Method**: This method uses a Quantitative method which can take the calculated values of the four QoS parameters. The calculated values of the five QoS parameters are recapitulated based on the time and type of data, the next recapitulation process will show a more detailed value of each QoS parameter. **Result**: The host virtual machine runs 3 guest virtual machines, the first guest virtual machine is for FreeNAS Server, the second guest virtual machine is for Certificate Authority Server, and the third guest virtual machine is for OpenVPN Server. System implementation is divided into four steps, namely, RB2011 proxy router configuration, FreeNAS Server installation and configuration, CA Server installation and configuration, OpenVPN Server installation and configuration. **Conclusion**: Based on OpenVPN QoS performance testing of six parameters namely, packet loss, round trip, file transfer, Jitter, Download and Upload which were tested on six different Providers, the bandwidth capacity of the Internet Provider used greatly influences the quality of service when connected to an OpenVPN Client. The higher the bandwidth capacity of the Provider used, the better the service quality, the faster the data access speed.*

Kata kunci: *Virtual Private Network; OpenVPN; SSL; Network Attached Storage; FreeNAS*

Abstrak

Pendahuluan: Perkembangan teknologi informasi dan komunikasi sangat cepat dan pesat, hal ini yang menyebabkan munculnya kemajuan teknologi informasi. Karena banyak kemudahan yang di tawarkan, teknologi informasi tidak dapat lepas dari berbagai aspek kehidupan manusia, yang memungkinkan dapat berkomunikasi dan saling bertukar informasi atau data. Seringkali masalah keamanan berada di urutan kedua, atau bahkan di urutan terakhir dalam daftar hal-hal yang dianggap penting. **Tujuan**: menghasilkan jaringan privat dan mengamankan data pada Network Attached Storage (NAS) dengan sistem keamanan berbasis SSL yang dapat diakses dari jaringan publik secara privat menggunakan teknologi VPN. **Metode**: Metode ini

menggunakan metode Kuantitatif yang dimana dapat mengambil nilai hasil perhitungan empat parameter QoS. Nilai perhitungan lima parameter QoS dilakukan rekapitulasi berdasarkan waktu dan jenis data, proses rekapitulasi selanjutnya akan memperlihatkan nilai yang lebih detail dari masing-masing parameter QoS. **Hasil**: Host virtual machine menjalankan 3 guest virtual machine, guest virtual machine pertama untuk FreeNAS Server, guest virtual machine kedua untuk Certificate Authority Server, dan guest virtual machine ketiga untuk OpenVPN Server. Implementasi sistem dibagi menjadi empat langkah yaitu, konfigurasi router mikrotik RB2011, instalasi dan konfigurasi Server FreeNAS, instalasi dan konfigurasi CA Server, instalasi dan konfigurasi OpenVPN Server. **Kesimpulan**: Berdasarkan pengujian performa QoS OpenVPN terhadap enam parameter yakni, packet loss, round trip, transfer file, Jitter, Download dan Upload yang diujikan pada enam macam Provider, kapasitas bandwidth Provider internet yang dipakai sangat berpengaruh terhadap kualitas layanan ketika terkoneksi menjadi OpenVPN Client. Semakin tinggi kapasitas bandwidth dari Provider yang digunakan semakin baik kualitas layanannya, semakin cepat kecepatan akses datanya.

Corresponding Author: Muhammad Affandi
E-mail: m_affandi@student.untan.ac.id

PENDAHULUAN

Perkembangan teknologi informasi dan komunikasi sangat cepat dan pesat, hal ini yang menyebabkan munculnya kemajuan teknologi informasi. Secara langsung atau tidak, teknologi informasi telah menjadi bagian penting dari berbagai bidang kehidupan. Karena banyak kemudahan yang di tawarkan, teknologi informasi tidak dapat lepas dari berbagai aspek kehidupan manusia, yang memungkinkan dapat berkomunikasi dan saling bertukar informasi atau data. Kebutuhan mengakses data, sharing data baik file maupun folder dapat dilakukan dari satu komputer ke komputer lain menggunakan jaringan komputer lokal maupun jaringan komputer yang lebih besar yaitu internet. *Network Attached Storage* (NAS) adalah sebuah media penyimpanan jaringan yang dapat berupa sebuah *dedicated hardware* atau dapat pula berupa media penyimpanan yang dibangun dari sebuah komputer (Barb Gonzalez, 2015). Perangkat NAS *(Network-Attached Storage)* menjadi suatu solusi penyimpanan yang efektif, terukur, dan rendah biaya. NAS dapat melayani kebutuhan data seperti penyimpanan dan berbagi data secara cepat serta dapat diakses melalui jaringan lokal. Kendala dalam penyimpanan secara lokal adalah tidak dapat diakses dimana saja dan kapan saja, apabila pengguna membutuhkan data dan ingin mengakses data dari penyimpanan NAS, pengguna harus terhubung ke jaringan lokal tersebut yang artinya pengguna harus berada di kantor atau rumah tempat penyimpanan itu terhubung. Salah satu cara agar penyimpanan tersebut dapat diakses dimana saja dan kapan saja adalah penyimpanan tersebut di sharing ke jaringan internet, namun jaringan internet atau jaringan publik ini bersifat tidak aman, karena tidak ada protokol keamanan yang melindunginya. Protokol keamanan membahas tujuan keamanan, termasuk otentikasi atau kerahasiaan. Secara khusus, otentikasi, atau logging in, memungkinkan verifikasi identitas. Otentikasi dilakukan melalui verifikasi kata sandi yang dikirimkan (Frank, 2017). Setiap perangkat yang terhubung ke jaringan dapat memiliki resiko keamanan, maka diperlukan pembenahan terkait dengan teknologi kemanan jaringan. Resiko keamanannya dapat berupa akses tidak sah, serangan *Denial of Service* (DoS) dan penggunaan ulang perangkat dengan mengunduh malware, seperti halnya perangkat lain yang terhubung ke jaringan (Paul Garms dan Sean Murphy, 2008). Data-data yang tersimpan di media penyimpanan akan rentan terhadap pencurian data. Hal tersebut tentu saja menimbulkan resiko bila informasi yang

sensitif dan berharga tersebut di akses oleh orang yang tidak bertangung jawab. Yang mana jika hal tersebut sampai terjadi, kemungkinan besar akan cmerugikan bahkan membahayakan orang yang akan mengirim pesan, maupun organisasinya. Informasi yang terkandung di dalamnya pun bisa saja berubah sehingga menyebabkan salah penafsiran oleh penerima pesan. Selain itu data yang di bajak tersebut kemungkinan rusak atau hilang yang menimbulkan kerugian material yang besar.

Masalah keamanan merupakan salah satu aspek penting dari sebuah sistem informasi. Sayang sekali masalah keamanan ini sering kali kurang mendapat perhatian dari para pemilik dan pengelola sistem informasi. Seringkali masalah keamanan berada di urutan kedua, atau bahkan di urutan terakhir dalam daftar hal-hal yang dianggap penting (Patrick Dowd dan John T McHenry, 1998). Pada akhir Januari 2022 grup *ransomware* DeadBolt menyerang pengguna QNAP *Network Attached Storage* (NAS). Seluruh perangkat QNAP yang dapat diakses melalui internet rentan terhadap serangan tersebut. DeadBolt yang merupakan jenis *cryptovirus*. *Ransomware* DeadBolt mengenkripsi berbagai file teks, *spreadsheet*, presentasi, dan data dokumen lainnya serta file gambar, video, audio yang ada di dalam NAS dengan ekstensi "*.deadbolt*." (BSSN.go.id, 2022).

Dengan masalah ini dibutuhkan suatu sistem yang dapat menjaga keamanan dari serangan-serangan keamanan jaringan agar data yang berupa file-file penting dapat tersebar dengan aman. Dengan mempertimbangkan kerahasiaan data, salah satu solusi yang dapat digunakan adalah dengan penggunaan VPN *(Virtual Private Network)*, VPN akan membuat jaringan yang digunakan di institusi bersifat privasi tetapi bisa digunakan pada jaringan publik atau internet. VPN adalah sebuah teknologi komunikasi yang memungkinkan untuk dapat terkoneksi ke jaringan publik dan menggunakannya untuk dapat bergabung dengan jaringan lokal (Irawan Afrianto dan Eko Budi Setiawan, 2014). Dengan mengaplikasikan VPN pada jaringan yang ada institusi tidak perlu khawatir pengiriman data atau file melalui publik atau internet karena akan aman meski di tempat umum sekalipun. VPN adalah sebuah cara aman untuk mengakses local area network yang berada pada jangkauan, dengan menggunakan internet atau jaringan umum lainnya untuk melakukan transmisi data paket secara pribadi, dengan enkripsi perlu penerapan teknologi tertentu sehingga walaupun menggunakan medium yang umum, tetapi traffic (lalu lintas) antar *remote-site* atau antar VPN *Server* dan *Client* tidak dapat disadap dengan mudah, juga tidak memungkinkan pihak lain untuk menyusupkan yang tidak semestinya ke dalam *remote-site*.

Berdasarkan penjelasan yang sudah dipaparkan, penelitian ini memanfaatkan kinerja teknologi VPN untuk memudahkan penyimpanan jaringan NAS yang sebelumnya hanya dapat diakses secara lokal dapat juga diakses menggunakan jaringan publik internet sekaligus mengamankan komunikasi data antar *Server* dan *Client*. *Network Attached Storage* (NAS) menggunakan sistem operasi FreeNAS. *Virtual Private Network* (VPN) yang dibangun menggunakan perangkat lunak opensource OpenVPN. OpenVPN menggunakan protokol enkripsi SSL dengan menerapkan sertifikat digital dalam koneksi *Client* dan *Server*. *Secure Socket Layer* (SSL) adalah protokol keamanan yang diterapkan pada OpenVPN dengan menerapkan 3 jenis keamanan, yaitu CA *(Certificate Authority)*, *Private Key* dan algoritma *Elliptic Curve*. Dalam penelitian ini, akan dilakukan pengujian keamanan guna menghindari ancaman serangan berupa port scanning dan sniffing agar data yang tersimpan dalam NAS tidak dicuri dan disalahgunakan oleh orang tidak bertanggung jawab. Dan juga akan dilakukan perbandingan performa jaringan OpenVPN sebelum dan sesudah menggunakan sertifikat SSL. Performansi unjuk kerja yaitu packet loss, roundtrip, transfer file, jitter, download dan upload.

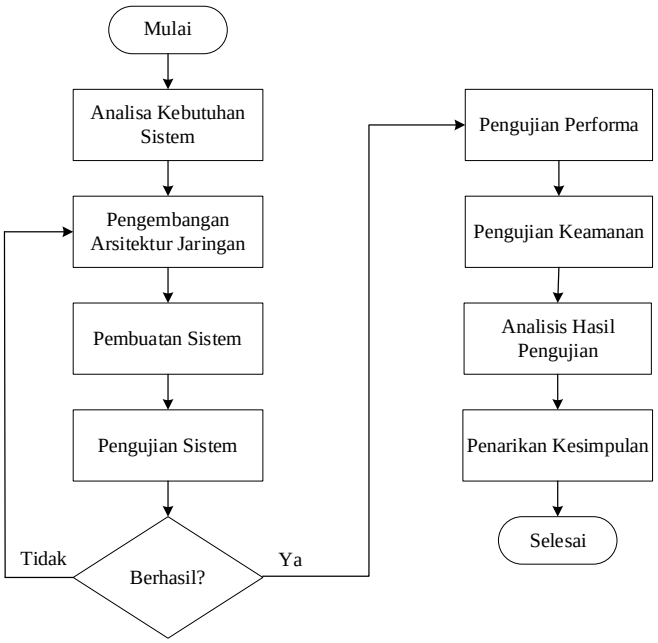
Adapun tujuan dari penelitian yang dilakukan adalah menghasilkan jaringan privat dan mengamankan data pada *Network Attached Storage* (NAS) dengan sistem keamanan berbasis SSL yang dapat diakses dari jaringan publik secara privat menggunakan teknologi VPN.

Beberapa penelitian terkait dengan penelitian ini diantaranya sebagai berikut : Devi Ruwaida dan Dian Kurnia (2018) melakukan penelitian dengan judul "Rancang Bangun File Transfer Protocol (FTP) dengan Pengamanan Open SSL pada Jaringan VPN Mikrotik di SMKS Dwiwarna". Pada penelitian ini dibangun File Transfer Protocol Server di SMKS

Dwiwarna. Pengamanan jaringan atau data menggunakan Open SSL, yang pada awalnya FTP Server berjalan pada protokol yang tidak terlindungi di dalam port 21. Sistem operasi dalam membangun FTP Server menggunakan sistem operasi Linux Debian 9.1. Ditambahkan juga pengamanan jaringan VPN *Point to Point Tunneling Protocol* (PPTP) di *router* mikrotik agar *Client* memungkinkan dalam pengiriman data secara aman melalui *remote Client* ke Server sekolah.

METODE PENELITIAN

Metodologi penelitian yang dilakukan dapat dijelaskan pada diagram alir penelitian berikut:



Gambar 1. Diagram Alir Penelitian

Analisis hasil pengujian dilakukan dengan cara membagi analisis dalam dua bagian. Bagian pertama yaitu analisis yang dilakukan untuk performa jaringan yang telah diimplementasi dan diuji, sedangkan untuk bagian kedua merupakan analisis yang dilakukan terhadap keamanan jaringan.

Analisis untuk mengukur performa dilakukan dengan cara mengambil nilai hasil perhitungan empat parameter QoS. Nilai perhitungan lima parameter QoS dilakukan rekapitulasi berdasarkan waktu dan jenis data, proses rekapitulasi selanjutnya akan memperlihatkan nilai yang lebih detail dari masing-masing parameter QoS.

Analisis terhadap keamanan dilakukan dengan cara menganalisa hasil pengujian keamanan jaringan sesuai dengan skema pengujian yang telah dirancang. Analisa dilakukan dengan melihat jenis enkapsulasi data, ketahanan terhadap serangan, dan protokol yang digunakan.

HASIL DAN PEMBAHASAN

A. Implementasi

a. Implementasi Pengembangan Arsitektur Jaringan

Pengembangan arsitektur jaringan yang dilakukan yaitu dengan meletakkan satu komputer Server yang berfungsi sebagai host virtual machine. Host virtual

machine menjalankan 3 *guest virtual machine*, guest virtual machine pertama untuk *FreeNAS Server*, guest virtual machine kedua untuk *Certificate Authority Server*, dan guest virtual machine ketiga untuk OpenVPN *Server*. Host virtual machine terhubung ke router melalui port 4 yang memiliki IP keluaran lokal 192.168.10.0/24 dan IP publik 202.162.213.180/29 sebagai IP masuk.

b. Implementasi Sistem

Implementasi sistem dibagi menjadi empat langkah yaitu, konfigurasi router mikrotik RB2011, instalasi dan konfigurasi *Server FreeNAS*, instalasi dan konfigurasi CA Server, instalasi dan konfigurasi *OpenVPN Server*.

a.) Konfigurasi router mikrotik RB2011

Router yang digunakan pada penelitian ini menggunakan *routerboard* mikrotik RB2011. Konfigurasi yang dilakukan antara lain konfigurasi IP Address, konfigurasi DNS Server, dynamic routing konfigurasi Network Address Translation (NAT), konfigurasi *Dynamic Host Client Protocol* (DHCP), pemilihan protocol, konfigurasi IP Pool. Berikut ini adalah sejumlah proses konfigurasi router yang dikerjakan untuk dapat menjalankan OpenVPN Server di langkah selanjutnya.

b.) Konfigurasi IP Address

Langkah pertama dalam mengkonfigurasi router adalah dengan memberikan IP address pada port IP masuk (IP publik) dan port IP keluar (IP lokal). Port IP masuk dalam implementasi ini menggunakan port ether1 sedangkan port IP keluar menggunakan port ether2 sampai ether10.

1. IP masuk

   IP masuk yang digunakan adalah IP publik yang didapat dari ISP ICON+. IP publik terhubung dengan jaringan Wide Area Network (WAN) sehingga router dapat mengakses internet. IP publik merupakan syarat utama untuk membangun sebuah VPN. Port yang digunakan untuk IP masuk IP publik adalah port ether1. Perintah yang digunakan untuk melakukan konfigurasi IP dapat dilihat pada Gambar 2

```
[admin@MikroTik-RB2011-KANTOR-PTK] > /ip address
[admin@MikroTik-RB2011-KANTOR-PTK] /ip address> add address=202.162.213.180/29
interface=ether1 network=202.162.213.176
[admin@MikroTik-RB2011-KANTOR-PTK] /ip address>
```

Gambar 2. Konfigrasi IP masuk

2. IP keluar

   IP keluar adalah IP yang akan dikonfigurasi untuk Client lokal pada routerboard mikrotik RB2011. IP keluar menggunakan IP class C dengan menggunakan port ether2 sampai ether10. Pertama port ether2 sampai ether10 digabungkan menjadi satu segment menggunakan teknik bridging. Untuk mengkonfigurasi bridge dan IP keluar lokal dapat dilihat pada Gambar 3.

Gambar 3. Konfigurasi bridge dan IP keluar

c.) Konfigurasi DNS Server

Konfigurasi DNS *Server* tujuannya agar router dapat melakukan ping ke hostname atau domain yang ada di internet. Jika tidak melakukan setting DNS maka router hanya

akan dapat melakukan ping melalui IP masing-masing website atau domain. Untuk dapat melakukan konfigurasi DNS dapat dilihat pada Gambar 4.3 berikut ini.

```
[admin@MikroTik-RB2011-KANTOR-PTK] >
[admin@MikroTik-RB2011-KANTOR-PTK] >
[admin@MikroTik-RB2011-KANTOR-PTK] > /ip dns
[admin@MikroTik-RB2011-KANTOR-PTK] /ip dns> set servers=202.162.220.220,8.8.8.8
[admin@MikroTik-RB2011-KANTOR-PTK] /ip dns>
```

Gambar 4. 1 Konfigurasi DNS

1.1.1.1.1 Konfigurasi *Dynamic Routing*

Dynamic routing dilakukan agar *router* dapat menentukan jalur akses data secara dinamis. *Routing* yang diperlukan adalah menghubungkan *router* ke internet sehingga *router* dapat mengakses ke luar dan *router* dapat diakses dari luar. Konfigurasi *dynamic routing* dapat dilihat pada Gambar 4.4 berikut.

```
[admin@MikroTik-RB2011-KANTOR-PTK] >
[admin@MikroTik-RB2011-KANTOR-PTK] > /ip route
[admin@MikroTik-RB2011-KANTOR-PTK] /ip route> add distance=1 gateway=202.162.213.177
[admin@MikroTik-RB2011-KANTOR-PTK] /ip route>
```

Gambar 4. 2 Konfigurasi *Dynamic Routing*

1.1.1.1.2 Konfigurasi *Network Address Translation (NAT)*

Konfigurasi NAT diperlukan untuk mentranslasi dari IP lokal ke IP publik agar IP lokal bisa terhubung ke internet. Berbeda dengan routing yang hanya menghubungkan *router* ke internet namun tidak dengan *Client* yang berada dibawah *router* sehingga perlu adanya konfigurasi NAT. Konfigurasi NAT lebih lanjut dapat dijelaskan pada Gambar 4.5 berikut.

```
[admin@MikroTik-RB2011-KANTOR-PTK] >
[admin@MikroTik-RB2011-KANTOR-PTK] > /ip firewall nat
[admin@MikroTik-RB2011-KANTOR-PTK] /ip firewall nat> add action=masquerade chain=srcnat
out-interface=ether1
[admin@MikroTik-RB2011-KANTOR-PTK] /ip firewall nat>
```

Gambar 4. 3 Konfigurasi NAT

1.1.1.1.3 Konfigurasi *DHCP Client* dan *IP Pool*

Konfigurasi IP secara DHCP dibutuhkan agar setiap *Client* yang terhubung pada *router* dapat langsung terkoneksi dan mendapat IP secara otomatis dengan *range* IP yang disetting pada IP *pool router*. Konfigurasi IP secara DHCP lebih lengkapnya dapat dilihat pada Gambar 4.6 berikut ini.

Gambar 4. 4 Konfigurasi DHCP dan IP Pool

1.1.1.1.4 Konfigurasi *Forwarding*

Agar OpenVPN *Server* bisa diakses dari internet, set fowarding di *router* dengan fitur firewall NAT. *Fowarding* ini akan membelokkan *traffic* yang menuju ke IP publik yang terpasang di router menuju ke IP lokal *Server*. Dengan begitu, seolah-olah *Client* dari internet berkomunikasi dengan OpenVPN *Server* meminjam IP *public router*. IP yang akan ditetapkan pada komputer OpenVPN *Server* menggunakan IP 192.168.10.3/24 dan menggunakan *protocol* UDP *port* 1194. Konfigurasi *Forwarding* dapat dilihat pada Gambar 4.7 berikut ini.

```
[admin@MikroTik-RB2011-KANTOR-PTK] > /ip firewall nat
[admin@MikroTik-RB2011-KANTOR-PTK] /ip firewall nat> add action=dst-nat chain=dstnat
dst-address=202.162.213.180 dst-port=1194 protocol=udp to-addresses=192.168.10.3
[admin@MikroTik-RB2011-KANTOR-PTK] /ip firewall nat>
```

Gambar 4. 5 Konfigurasi *Forwarding*

1.1.1.2 Instalasi dan Konfigurasi Server FreeNAS

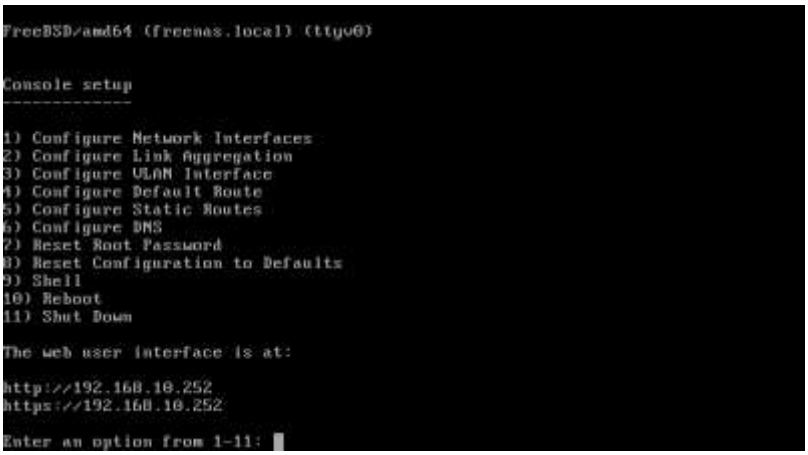
Setelah dilakukan penyiapan awal, instalasi dan konfigurasi perangkat jaringan yang telah dilakukan pada tahap sebelumnya. Tahap ini melakukan instalasi dan konfigurasi *Server* FreeNAS. *Server* FreeNAS dijalankan pada mesin *virtual* menggunakan *Oracle VM Virtual Box*. Setelah *Oracle VM Virtual Box* terinstal, kemudian disiapkan mesin *virtual* untuk menjalankan

FreeNAS. Gambar 4.8 menunjukkan mesin *virtual* FreeNAS yang siap untuk dijalankan pada *Virtual Box*. Mesin virtual untuk FreeNAS menggunakan sistem operasi FreeBSD, memorinya sebesar 8 Gb. Sementara untuk *disk*-nya digunakan dua buah *virtual disk* dengan kapasitas 40 Gb dan 1 Tb. FreeNAS diinstal pada virtual disk yang berkapasitas 40 Gb. Sedangkan virtual disk yang satunya digunakan untuk media penyimpan data. Untuk koneksi jaringan digunakan tipe *bridge* supaya bisa berbagi kartu jaringan antara komputer *host* dengan FreeNAS. Setelah pengaturan mesin *virtual* benar dan lengkap kemudian dijalankan. Jika *booting* dengan virtual disk berhasil maka akan tampil seperti Gambar 4.9. Kemudian tahapan instalasi FreeNAS dimulai.

Gambar 4. 6 FreeNAS siap di-*instal* di *Oracle VM Virtual Box*

Gambar 4. 7 Tampilan *Boot* FreeNAS

Setelah dilakukan proses instalasi sistem operasi FreeNAS, selanjutnya akan dilakukan konfigurasi FreeNAS berupa konfigurasi IP *address* dan jaringan, konfigurasi *Network Time Protocol* (NTP), konfigurasi media penyimpanan (storage), konfigurasi *user* dan *group*, konfigurasi ACL, dan konfigurasi *sharing*-nya. Tampilan FreeNAS OS yang telah selesai *booting* dapat dilihat pada Gambar 4. 10.



Gambar 4. 8 Tampilan FreeNAS OS

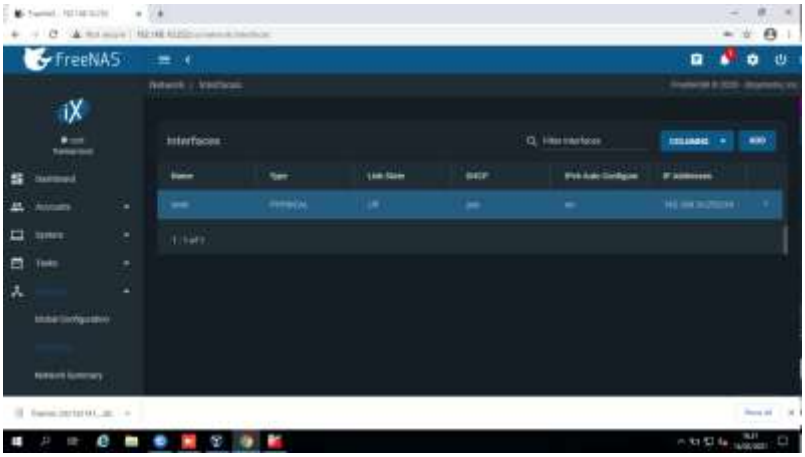
1.1.1.2.1 Konfigurasi IP *Address* FreeNAS

Untuk melakukan konfigurasi IP *Address* dapat dilakukan melalui *command line interface* (CLI) atau melaui *web interface* FreeNAS. IP 192.168.10.252 adalah IP DHCP dari *router*, *default* saat instalasi awal FreeNAS. Pada tahap ini dilakukan konfigurasi IP *Address Static* melalui *web interface*, dengan mengakses IP awal FreeNAS pada *browser*. Halaman *login web interface* FreeNAS dapat dilihat pada Gambar 4. 11.

Gambar 4. 9 Tampilan *Login Web Interface*

Masukkan *username* dan *password* yang telah dikonfigurasi saat awal instalasi sistem operasi FreeNAS.

Untuk melakukan konfigurasi IP *Address static*, pada *dashboard web*, pilih *tab network -> interfaces* (di tunjukkan pada Gambar 4. 12)

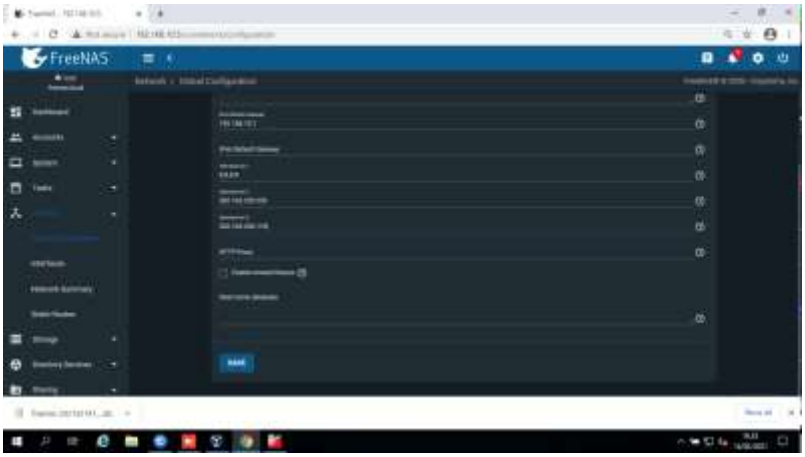


Gambar 4. 10 Tampilan *Network Interfaces*

Edit interfaces, untuk melakukan perubahan IP *address*. Konfigurasi IP *Address Static* 192.168.10.5/24 sesuai perancangan yang telah dibuat. *Apply* konfigurasi dan simpan seperti pada Gambar 4. 13.

Gambar 4. 11 Tampilan Konfigurasi IP *Static*

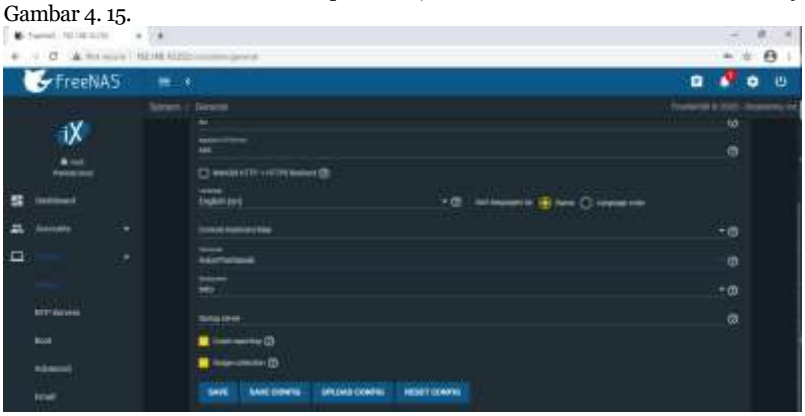
Selanjutnya setelah melakukan perubahan IP *Address* sesuai perancangan, lakukan konfigurasi jaringan lainnya yaitu konfigurasi IP *Gateway* dan DNS jaringan. Langkahnya yaitu pada *tab Network -> Global Configuration*. Masukkan IP *Default Gateway* dan DNS jaringan. IP *Gateway* nya yaitu 192.168.10.1, DNS menggunakan 3 DNS yaitu, 8.8.8.8, 202.162.220.220, 202.162.220.110. Seperti pada Gambar 4. 14.



Gambar 4. 12 Tampilan Konfigurasi *Network -> Global Configuration*

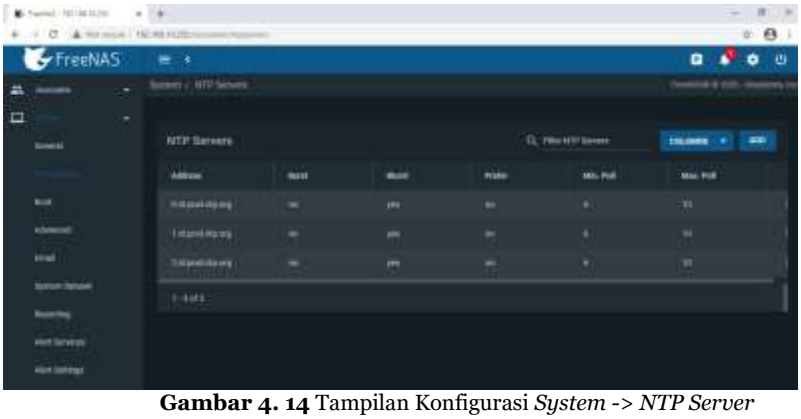
1.1.1.2.2 Konfigurasi *Network Time Protocol (NTP)*

NTP merupakan sebuah mekanisme atau protokol yang digunakan untuk melakukan sinkronisasi terhadap penunjuk waktu dalam sebuah sistem komputer dan jaringan. Proses sinkronisasi ini dilakukan di dalam jalur komunikasi data yang biasanya menggunakan protokol komunikasi TCP/IP. Pada FreeNAS tahapan konfigurasinya yaitu, *system -> general ->* tentukan *timezone* lokasi *Server* FreeNAS, pilih Asia/Pontianak sesuai lokasi *Server*. Ditunjukkan pada Gambar 4. 15.



Gambar 4. 13 Tampilan Konfigurasi *System -> General, Timezone*

Selanjutnya tahapan konfigurasinya, *System -> NTP Server* (Gambar 4. 16). Ganti *Address* NTP *Server* dengan NTP *Server* Indonesia, 0.id.pool.ntp.org, 1.id.pool.ntp.org, 2.id.pool.ntp.org.



Gambar 4. 14 Tampilan Konfigurasi *System -> NTP Server*

1.1.1.2.3 Konfigurasi *User* dan *Group*

Tahapan selanjutnya adalah pembuatan *User* dan Group yang akan menentukan hak akses dari setiap *group* dan *user*. Adapun tahapannya adalah sebagai berikut :

1. Membuat *group*

   Grup berfungsi untuk mengelompokan dan mengkategorikan user berdasarkan divisi kerja untuk menentukan hak akses yang dimilikinya. Berikut ini adalah daftar grup yang akan di buat:

Tabel 4. 1 Grup sesuai divisi kerja

No	Nama Grup	Keterangan
1	Aktivasi	Grup untuk pengguna divisi aktivasi
2	Pemeliharaan	Grup untuk pengguna divisi pemeliharaan
3	Aset	Grup untuk pengguna divisi aset
4	Sales	Grup untuk pengguna divisi sales
5	Umum	Grup untuk pengguna divisi umum

Untuk melakukan penambahan grup, pada *web interface* pilih menu *Accounts -> Groups -> Add*, seperti yang ditunjukan pada Gambar 4.17 dan Gambar 4.18

Gambar 4. 15 Tampilan Menu *Groups*

Klik tombol "*ADD*" untuk menambahkan grup baru. Maka selanjutnya, akan muncul tampilan seperti pada gambar 4.18

Gambar 4. 16 Tampilan Tambah *Group*

Selanjutnya isi nama grup sesuai nama divisi seperti pada tabel 4.1. Setelah selesai dilanjutkan dengan menekan tombol *save* untuk menyimpan grup baru yang dibuat. Tahap selanjutnya adalah membuat grup lainnya dengan cara yang sama.

2. Membuat *User*

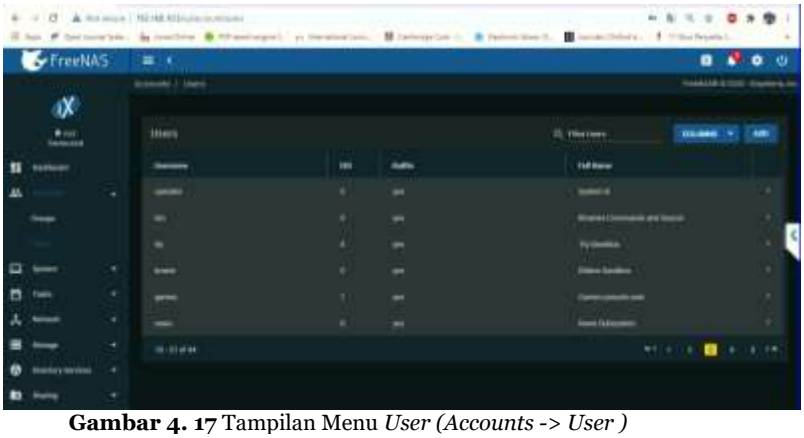
Berikutnya adalah pembuatan *account user*. Jumlah *user* tentunya tidak terbatas tergantung banyaknya pegawai atau karyawan yang menggunakan *Server* freenas. Berikut data user yang akan dibuat :

Tabel 4. 2 *User*

No	Nama	*Username*	*Group*
1	Paniel F Silitonga	paniel	pemeliharaan, aset, aktivasi, sales, umum
2	Imam Utomo Harahap	imam	pemeliharaan
3	Syarif Al Adnan Razi	razi	pemeliharaan
4	Henri Mettaloa	henri	aset
5	Cahaya Kamila	cahaya	aset
6	Muhammad Affandi	affandi	aktivasi
7	Chairul Fadly	fadly	aktivasi
8	M Rafiki	fiki	aktivasi
9	Ferry Aldiano	ferryald	aktivasi
10	Selvia Anggraini	selvia	aktivasi
11	Salmah	salmah	pemeliharaan, aset, aktivasi
12	Yusriansyah	byan	sales
13	Chrissye Amanda Solissa	mandha	sales
14	Sarifani Artayasa	sarifani	sales
15	Hirra Tursina	hirra	sales
16	Utin Devika Azzahroh	utin	umum
17	Vina Karina	vina	umum

Commented [MF1]: Gunakan tabel bergaris

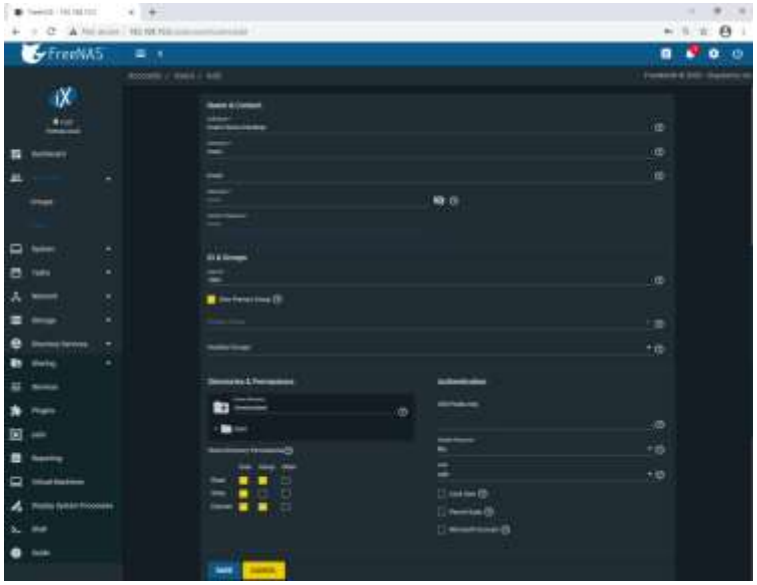
Untuk membuat user baru pada freenas, masih tetap di tampilan *web interface* freenas. Pada menu, pilih *Accounts -> User -> Add* seperti pada Gambar 4.19



Gambar 4. 17 Tampilan Menu *User (Accounts -> User)*

Maka akan muncul *form* pembuatan *user*, lalu isi *username* dengan yang akan digunakan oleh *user* untuk *login*, dapat dilihat seperti pada gambar 4.20

Gambar 4. 18 Tampilan Menu Tambah *User* (*Accounts -> User -> Add*)



Langkah selanjutnya yaitu mengelompokkan *user-user* yang telah ditambahkan ke dalam grup yang telah dibuat sebelumnya. Kembali ke menu *group* (*Accounts-> Groups*), pilih grup yang akan dimasukkan *user*, pilih *members* seperti pada Gambar 4.21.

Gambar 4. 19 Tampilan *Groups* Tambah *Members*

Maka akan tampil pengelompokkan *group* yang akan dimasukkan *username* dengan cara mengklik *username* dan tombol panah ke kolom group *members* seperti pada Gambar 4.22.

Gambar 4. 20 Gambar Pengelompokkan *username* ke *group members*

Setelah itu tekan tombol *save* untuk menyimpan *group members* yang telah dikelompokkan. Selanjutnya untuk menambahkan *User* dan *Groups* dilakukan dengan langkah yang sama.

1.1.1.2.4 *Konfigurasi Storage*

Pada tahapan ini dilakukan konfigurasi penyimpanan atau *storage*, konfigurasinya meliputi konfigurasi *pool*, *dataset*, dan ACL (*Access Control List*).

Pertama, pada *web interface* pilih menu *Storage -> Pools* seperti pada Gambar 4.23

Gambar 4. 21 Tampilan *Storage -> Pools*

Tekan button "*ADD*" untuk menambahkan *pool* baru, seperti pada Gambar 4. 24.

Gambar 4. 22 *Create New Pool*

Tekan *Create Pool*, setelah itu masuk ke *Pool Manager*. Masukkan nama *Pool* yang akan dibuat, pilih *disk storage* yang tersedia dan tekan tombol panah ke kanan untuk menambahkan *disk* ke dalam *pool* seperti pada Gambar 4.25.

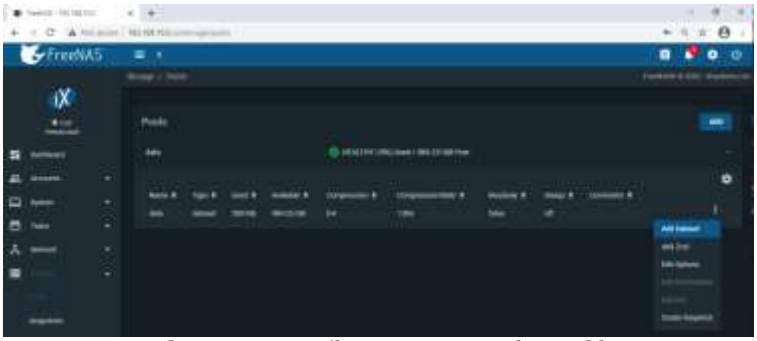
Gambar 4. 23 Tampilan *Pool Manager*, Tambah *Disk*

Selanjutnya jika sudah tekan tombol *create* untuk menyelesaikan pembuatan *pool* seperti pada gambar 4.26.

Gambar 4. 24 Gambar *Pool Manger, Create Pool*

Setelah *pool* dibuat, langkah selanjutnya menambahkan *dataset* pada *pool*. Pada dasarnya setelah mengkonfigurasi *Storage*, maka tidak ada data apapun didalamnya alias bersih. Dengan *Dataset* bisa membuat alokasi-alokasi khusus untuk pengguna nantinya, sederhananya *Dataset* adalah seperti folder yang dapat dikonfigurasi kapasitas, tipe *sharing*, izin akses, dan sebagainya.

Langkap menambahkan *dataset* sebagai berikut, pertama kembali ke menu *Storage* -> *Pools* lalu pada *pool* yang telah dibuat ada tombol titik 3 di sebelah kanan pilih "*Add Dataset*" seperti pada Gambar 4.27.



Gambar 4. 25 Tampilan *Storage ->Pools -> Add Dataset*

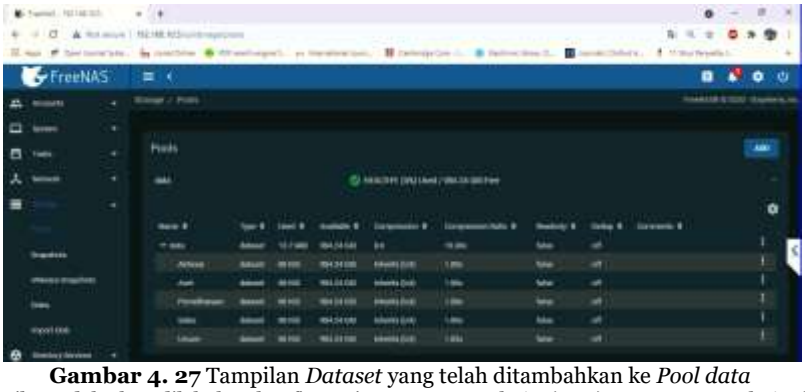
Selanjutnya isi nama dataset yang akan diberikan. Pada nama dataset diberi nama sesuai keperluan pengguna, pada penelitian ini dibuat berdasarkan grup pengguna yang dijelaskan pada tabel 4.3 berikut.

Tabel 4. 3 Nama Dataset

No	Nama Dataset
1	Pemeliharaan
2	Aset
3	Aktivasi
4	Sales
5	Umum

Gambar 4. 26 Tampilan Penambahan *Dataset* pada *Pool*

Langkah selanjutnya tambahkan *dataset* lain sesuai perancangan atau yang sudah dijelaskan pada tabel 4.3. Jika sudah ditambahkan semua dapat dilihat seperti pada Gambar 4.29.

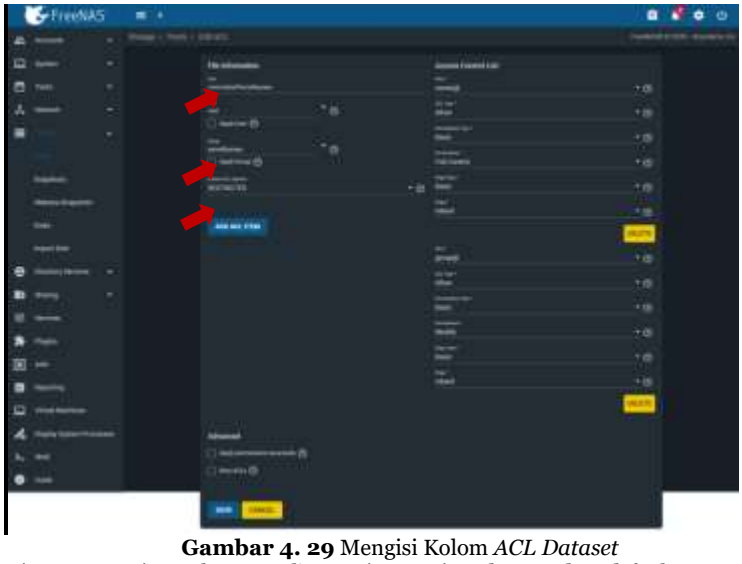


Gambar 4. 27 Tampilan *Dataset* yang telah ditambahkan ke *Pool data*

Jika sudah akan dilakukan konfigurasi *Access Control List* (ACL). *Access Control List* (ACL) pada FreeNAS merupakan daftar kondisi yang dirancang sedemikian rupa oleh *administrator* untuk mengontrol akses grup atau pengguna, mencegah grup atau pengguna yang tidak masuk ke daftar ACL untuk tidak dapat mengakses *folder* atau *dataset* yang telah ditentukan pada masing-masing grup atau pengguna. *Access List* dapat mengizinkan grup atau pengguna untuk mengakses *folder* atau *dataset* yang telah ditentukan.

Untuk melakukannya pada masing-masing *dataset* yang telah dibuat, tekan tombol titik 3 pada sebelah kanan, pilih "*Edit ACL*" seperti pada Gambar 4. 30

Gambar 4. 28 Tampilan *Dataset -> Edit ACL*



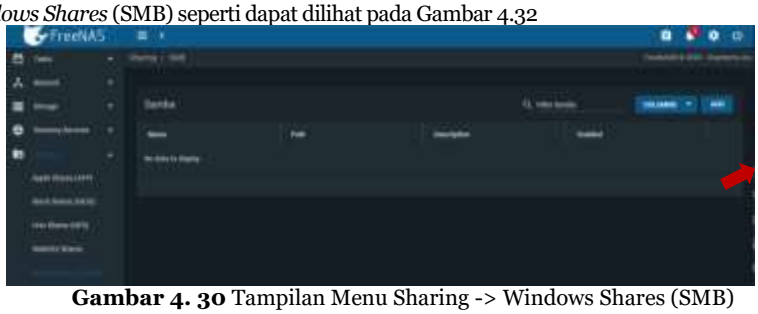
Gambar 4. 29 Mengisi Kolom *ACL Dataset*

Isi grup sesuai untuk *accest list* masing-masing *dataset* dan *default* ACL *options* pilih "*RESTRICTED*". Jika sudah tekan tombol *save*. Terlihat pada Gambar 4. 31.

1.1.1.2.5 Konfigurasi Layanan *Server*

Konfigurasi layanan *Server* dimaksudkan untuk mengkonfigurasi layanan *sharing data*. Pada penelitian ini tujuan penggunanya adalah Windows *user*, maka untuk layanan *Server* berbagi yang dikonfigurasi adalah *Windows Shares* (SMB). SMB (*Server Message Block*) adalah protokol *Client/Server* yang ditujukan sebagai layanan untuk berbagi berkas (*file sharing*) di dalam sebuah jaringan. Dalam suatu jaringan komputer, SMB (*Server Message Block*) beroperasi sebagai protokol jaringan layer aplikasi. SMB berguna untuk menyediakan akses berbagi file, *printer*, dan komunikasi lain-lain antar *node* dalam sebuah jaringan.

Langkah-langkah konfigurasi SMB di FreeNAS adalah dengan memilih menu *Sharing* -> *Windows Shares* (SMB) seperti dapat dilihat pada Gambar 4.32



Gambar 4. 30 Tampilan Menu Sharing -> Windows Shares (SMB)

Untuk menambahkan konfigurasi SMB baru, tekan tombol "*ADD*", maka akan muncul seperti pada Gambar 4.33.

Gambar 4. 31 Tampilan Konfigurasi Menambahkan SMB *Shares*

Pada *path* pilih lokasi direktori *dataset* yang akan di *shares*. Selanjutnya isi nama dan deskripsi SMB *Shares* yang akan dikonfigurasi, jika sudah tekan tombol *save* untuk menyimpannya. Lakukan langkah yang sama untuk mengaktifkan SMB *Shares* yang lain. Jika sudah semua dapat dilihat pada Gambar 4.34 berikut.

Gambar 4. 32 Tampilan SMB *Shares* yang telah dikonfigurasi

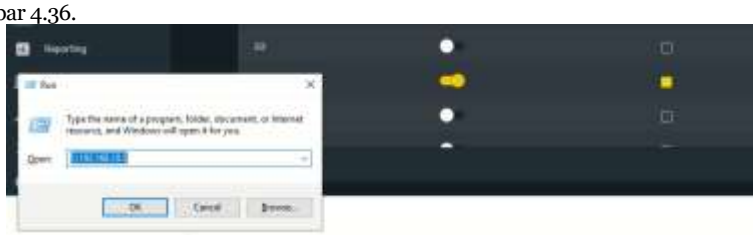
Jika semua *dataset* telah di-*shares* maka tahapan ini selesai.

Tahapan berikutnya adalah mengaktifkan *Service* SMB agar semua *dataset* yang telah dikonfigurasi dapat diakses di windows. Dimulai dengan memilih menu "*Services*" pada menu utama. Berikutnya adalah lakukan "*ON*" dan "*OFF*" untuk me-*restart service* pada SMB dan centang *Start Automatically* seperti pada Gambar 4.36.

Gambar 4. 33 Tampilan Menu Services

Sampai pada tahap ini SMB telah diaktifkan dan konfigurasi SMB telah selesai dilakukan. SMB FreeNAS siap digunakan.

Untuk menguji apakah SMB sudah bisa digunakan, dicoba akses dengan cara tekan tombol Windows + R ketik "\\192.168.10.5" (IP *Server* FreeNAS), lalu tekan OK dapat dilihat pada Gambar 4.36.



Gambar 4. 34 Mengakses SMB *Shares* FreeNAS

Selanjutnya akan tampil tampilan login untuk memasukkan *username* dan *password*. Masukkan salah satu *username* dan *password* yang telah dikonfigurasi sebelumnya pada *Accounts -> Users* (Gambar 4. 37).

Gambar 4. 35 Tampilan Muncul Login Akses SMB FreeNAS

Setelah dimasukkan salah satu *username* dan *password* yang telah dibuat sebelumnya, akan tampil *folder dataset* yang telah dibuat. Seperti terlihat pada Gambar 4. 38

Gambar 4. 36 Tampilan *Folder Dataset* SMB FreeNAS

Sampai tahap ini, pengguna bisa mengakses *folder* sesuai hak akses masing-masing. Pengguna dapat membuka *folder dataset* tersebut, pengguna dapat membuat *folder* di dalam *dataset*, pengguna dapat menambahkan *file* di dalam *folder*, menyalin atau meng-*copy paste* pada *dataset* sesuai akses pengguna yang telah ditentukan. Pengguna tidak dapat mengakses dataset

lain yang diluar hak aksesnya. Selanjutnya untuk pengujian penggunaan dilanjutkan pada tahap pengujian.

1.1.1.3 Konfigurasi CA *Server*

Certificate Authority (CA) adalah sebuah entitas bertanggung jawab untuk mengeluarkan sertifikat digital untuk memverifikasi identitas di internet. Meskipun CA publik merupakan pilihan populer untuk memverifikasi identitas situs *web* dan layanan lain yang disediakan untuk publik umum, CA privat dapat digunakan untuk grup tertutup dan layanan privat.

Membangun Otoritas Sertifikat pribadi akan memungkinkan untuk mengonfigurasi, menguji, dan menjalankan program yang memerlukan koneksi terenkripsi antara klien dan *Server*. Dengan CA privat, dapat menerbitkan sertifikat untuk pengguna, *Server*, atau *program* dan layanan individual dalam infrastruktur Anda. Contoh *program* di Linux yang menggunakan CA privatnya sendiri adalah OpenVPN.

Pada penelitian ini, CA *Server* merupakan sistem yang berdiri sendiri. Ini hanya akan digunakan untuk mengimpor, menandatangani, dan mencabut permintaan sertifikat. Tidak untuk menjalankan layanan lain, dan akan *offline* atau benar-benar mati ketika tidak aktif bekerja dengan CA.

1.1.1.3.1 Meng-*install Easy-RSA*

CA *Server* dibangun menggunakan aplikasi *Easy-RSA*. *Easy-RSA* adalah alat manajemen CA yang akan digunakan untuk membuat kunci pribadi, dan sertifikat akar publik, yang kemudian akan digunakan untuk menandatangani permintaan dari klien dan *Server* yang akan mengandalkan CA. Pada CA *Server* ini *Easy-RSA* berjalan pada sistem operasi Linux Ubuntu *Server* 20.04. Langkah pertama *install* paket *easy-rsa* pada komputer CA *Server*. Masuk ke CA *Server* sebagai pengguna *sudo non-root* dan jalankan perintah berikut:

$ sudo apt-update
$ sudo apt install easy-rsa

1.1.1.3.2 Menyiapkan Direktori Infrastruktur Kunci Publik

Setelah menginstal *easy-rsa*, setelah itu membuat kerangkat infrastruktur kunci publik (PKI) di CA *Server*. Masuk sebagai pengguna *non-root* dan buat *easy-rsa* direktori. Pastikan tidak menggunakan sudo untuk menjalankan salah satu dari perintah berikut, karena pengguna normal harus mengelola dan berinteraksi dengan CA tanpa hak istimewa yang lebih tinggi.

$ mkdir ~/easy-rsa

Ini akan membuat direktori baru easy-rsa di folder *home*. Direktori ini akan digunakan untuk membuat tautan simbolik yang menunjuk ke file paket *easy-rsa* yang telah diinstal di langkah sebelumnya. *File-file* ini terletak di *folder /usr/share/easy-rsa* di CA *Server*.

Buat *symlink* dengan *ln* perintah:

$ ln -s /usr/share/easy-rsa/ ~/easy-rsa/*

Batasi akses ke direktori PKI, hanya pemilik yang dapat mengaksesnya menggunakan *chmod* perintah:

$ chmod 700 /home/caserv/easy-rsa

Terakhir, inisialisasi PKI di dalam direktori *easy-rsa*:

$ cd ~/easy-rsa
$ ./easyrsa init-pki

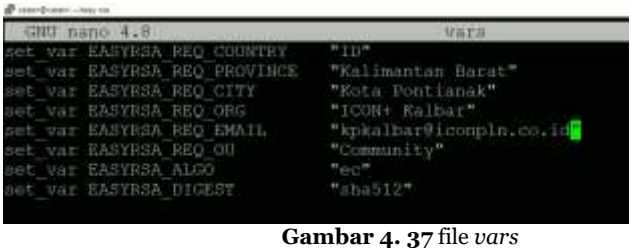
Setelah menyelesaikan bagian ini, CA *Server* memiliki direktori yang berisi semua berkas yang diperlukan untuk membuat CA (*Certificate Authority*). Di bagian selanjutnya membuat kunci privat dan sertifikat publik untuk CA.

1.1.1.3.3 Membuat *Certificate Authority*

Sebelum membuat kunci pribadi dan sertifikat CA, perlu membuat dan mengisi file yang disebut *vars* dengan beberapa nilai *default*. Pertama masuk ke direktori *easy-rsa*, lalu buat dan edit *vars* file dengan nano:

$ cd ~/easy-rsa
$ nano vars

Setelah file dibuka, isi file *vars* berikut seperti pada Gambar 4. 39 :



Gambar 4. 37 file *vars*

Setelah selesai, simpan dan tutup *file*. Untuk membuat pasangan kunci publik dan pribadi *root* untuk CA, jalankan *./easy-rsa*, dengan opsi *build-ca* :

$ ./easyrsa build-ca

Dalam *output*, akan muncul beberapa baris tentang versi OpenSSL dan diminta untuk memasukkan kata sandi untuk pasangan kunci. Pastikan untuk memilih frasa sandi yang kuat, dan catat di tempat yang aman. Perlu memasukkan frasa sandi setiap saat perlu berinteraksi dengan CA, misalnya untuk menandatangani atau mencabut sertifikat.

Selanjutnya juga diminta untuk mengkonfirmasi *Common Name* (CN) untuk CA. CN adalah nama yang digunakan untuk merujuk ke mesin CA. Tekan *ENTER* untuk menerima nama *default*.

CA *Server* sekarang memiliki dua *file* penting - *~/easy-rsa/pki/ca.crt dan ~/easy-rsa/pki/private/ca.key-* yang membentuk komponen publik dan pribadi dari CA.

- ca.crt adalah *file* sertifikat publik CA. Pengguna, *Server*, dan klien akan menggunakan sertifikat ini untuk memverifikasi bahwa mereka adalah bagian dari *web* kepercayaan yang sama. Setiap pengguna dan *Server* yang menggunakan CA perlu memiliki salinan *file* ini. Semua pihak akan mengandalkan sertifikat publik untuk memastikan bahwa seseorang tidak meniru sistem dan melakukan serangan *Man-in-the-middle* .
- ca.key adalah kunci pribadi yang digunakan CA untuk menandatangani sertifikat untuk *Server* dan klien. Jika penyerang mendapatkan akses ke CA dan pada gilirannya file ca.key, CA harus dihancurkan. Inilah sebabnya mengapa file ca.key seharusnya hanya ada di mesin CA dan idealnya, mesin CA Anda harus tetap *offline* ketika tidak menandatangani permintaan sertifikat sebagai tindakan keamanan tambahan.

Dengan itu, CA sudah siap digunakan untuk menandatangani permintaan sertifikat, dan untuk mencabut sertifikat.

1.1.1.4 Konfigurasi OpenVPN *Server*

OpenVPN adalah VPN *Transport Layer Security* (TLS) sumber terbuka lengkap yang mengakomodasi beragam konfigurasi. Dalam penelitian ini, OpenVPN di instal pada sistem operasi Ubuntu *Server* 20.04, lalu mengonfigurasinya supaya dapat diakses dari mesin klien. Komputer (VM) untuk OpenVPN *Server* ini berjalan sendiri tanpa digabung dengan CA *Server* guna meningkatkan keamanan PKI.

1.1.1.4.1 Menginstal OpenVPN dan *Easy-RSA*

Langkah pertama adalah menginstal OpenVPN dan *Easy-RSA*. *Easy-RSA* adalah alat manajemen infrastruktur kunci publik (PKI) yang akan digunakan pada OpenVPN *Server* untuk menghasilkan permintaan sertifikat yang selanjutnya akan diverifikasi dan ditanda tangani pada CA *Server*.

Untuk memulai, perbarui indeks paket OpenVPN *Server* serta instal OpenVPN dan *Easy-RSA*. Kedua paket ini tersedia di repositori Ubuntu, sehingga dapat menggunakan *apt* untuk instalasi:

$ sudo apt update

$ sudo apt install openvpn easy-rsa

Selanjutnya, buat direktori baru pada *Server* OpenVPN sebagai pengguna *non-root* yang bernama *~/easy-rsa*:

$ mkdir ~/easy-rsa

Sekarang buat *symlink* dari skrip *easyrsa* yang diinstal paket ke direktori *~/easy-rsa* yang baru saja dibuat:

$ ln -s /usr/share/easy-rsa/ ~/easy-rsa/*

Pastikan pemilik direktori adalah pengguna *sudo non-root* dan batasi akses ke pengguna itu menggunakan *chmod*:

$ sudo chown ovpn ~/easy-rsa

$ chmod 700 ~/easy-rsa

Setelah program-program ini terinstal dan dipindahkan ke lokasi yang tepat pada sistem, langkah selanjutnya adalah membuat Infrastruktur Kunci Publik (PKI) di *Server* OpenVPN supaya *Server* OpenVPN dapat meminta dan mengelola sertifikat TLS untuk klien dan *Server* lain yang akan terhubung ke VPN.

1.1.1.4.2 Menciptakan PKI untuk OpenVPN

Sebelum dapat membuat kunci dan sertifikat privat dari *Server* OpenVPN, perlu membuat direktori Infrastruktur Kunci Publik lokal pada *Server* OpenVPN. *Server* OpenVPN akan menggunakan direktori ini untuk mengelola permintaan sertifikat *Server* dan klien sebagai pengganti membuatnya secara langsung pada *Server* CA.

Untuk membangun direktori PKI pada *Server* OpenVPN, perlu mengisi berkas bernama *vars* dengan beberapa nilai. Pertama-tama, masukkan ke direktori *easy-rsa*, lalu buat dan *edit* berkas *vars* menggunakan *nano*.

$ cd ~/easy-rsa

$ nano vars

Setelah berkas dibuka, lekatkan dua baris berikut, pada Gambar 4. 40:

```
  GNU nano 4.8
set_var EASYRSA_ALGO "ec"
set_var EASYRSA_DIGEST "sha512"
```

Gambar 4. 38 Isi file *vars* di *Server* OpenVPN

Hanya dua baris inilah yang dibutuhkan di dalam berkas *vars* di *Server* OpenVPN karena tidak akan digunakan sebagai CA. Kedua baris ini akan memastikan bahwa permintaan kunci dan sertifikat privat dikonfigurasi untuk menggunakan *Elliptic Curve Cryptography* (ECC) yang menghasilkan kunci dan tanda tangan aman untuk klien dan *Server* OpenVPN.

Mengonfigurasi *Server* OpenVPN dan CA untuk menggunakan ECC berarti bahwa ketika klien dan *Server* berupaya membuat kunci simetris bersama, kedua *Server* itu dapat menggunakan algoritma *Elliptic Curve* untuk melakukan pertukaran di antara keduanya. Menggunakan ECC untuk pertukaran kunci secara signifikan lebih cepat daripada menggunakan *Diffie-Hellman* biasa dengan algoritma RSA klasik, karena angka-angkanya jauh lebih kecil dan komputasinya lebih cepat.

Setelah mengisi berkas *vars*, dilanjutkan dengan menciptakan direktori PKI. Untuk melakukan itu, jalankan skrip *easy-rsa* dengan opsi *init-pki*. Meskipun telah menjalankan perintah ini pada *Server* CA sebagai bagian dari prasyarat, ini perlu dijalankan di sini karena *Server* OpenVPN dan *Server* CA memiliki direktori PKI terpisah:

$ ./easyrsa init-pki

Pada *Server* OpenVPN, tidak perlu menciptakan CA. *Server* CA hanya bertanggung jawab atas pemvalidasian dan penandatanganan sertifikat. PKI pada *Server* VPN hanya digunakan sebagai tempat yang sesuai dan tersentralisasi untuk menyimpan permintaan sertifikat dan sertifikat publik.

Setelah menginisialisasi PKI pada *Server* OpenVPN, dilanjutkan ke langkah selanjutnya, yaitu menciptakan permintaan sertifikat *Server* OpenVPN dan kunci privat.

1.1.1.4.3 Menciptakan Permintaan Sertifikat dan Kunci Privat *Server* OpenVPN

Langkah selanjutnya adalah membuat kunci privat dan Permintaan Penandatanganan Serfifikat (CSR) di *Server* OpenVPN. Setelah itu, *transfer* permintaan ke CA untuk ditandatangani, yang membuat sertifikat yang dibutuhkan. Setelah menandatangani sertifikat, *transfer* kembali ke *Server* OpenVPN dan menginstalnya untuk digunakan *Server*.

Untuk memulai, bernavigasi ke direktori *~/easy-rsa* di *Server* OpenVPN sebagai pengguna *non-root*:

$ cd ~/easy-rsa

Sekarang panggil *easy-rsa* dengan opsi *gen-req* yang diikuti dengan *Common Name* (CN) untuk mesin itu. CN bisa apa pun akan membantu jika menggunakan nama yang lebih deskriptif. CN dari *Server* OpenVPN adalah *Server*. Pastikan untuk memasukkan opsi *nopass* juga. Jika itu gagal dilakukan, berkas permintaan akan dilindungi dengan kata sandi sehingga bisa menyebabkan masalah izin.

Ini akan menciptakan kunci privat untuk *Server* dan berkas permintaan sertifikat yang bernama *Server.req*. Salin kunci *Server* ke direktori */etc/openvpn/Server*:

$ sudo cp /home/sammy/easy-rsa/pki/private/Server.key /etc/openvpn/Server/

Setelah menyelesaikan langkah-langkah ini, kunci privat telah berhasil dibuat untuk *Server* OpenVPN. Selanjutnya buat Permintaan Penandatangan Sertifikat (CSR) untuk *Server* OpenVPN. CSR siap untuk penandatanganan oleh CA.

1.1.1.4.4 Menandatangani Permintaan Sertifikat *Server* OpenVPN

Dalam langkah sebelumnya, Permintaan Penandatanganan Sertifikat (CSR) dan kunci privat telah dibuat untuk *Server* OpenVPN. Sekarang, *Server* CA harus mengetahui sertifikat *Server* dan memvalidasinya. Setelah CA memvalidasi dan mengirimkan kembali sertifikat ke *Server* OpenVPN, klien yang memercayai CA akan dapat memercayai *Server* OpenVPN juga.

Pada *Server* OpenVPN, sebagai pengguna *non-root*, gunakan SCP untuk menyalin permintaan sertifikat *Server.req* ke *Server* CA untuk penandatanganan:

$ scp /home/ovpn/easy-rsa/pki/reqs/Server.req caserv@192.168.10.4:/tmp

Masuk ke *Server* CA sebagai pengguna *non-root* yang diciptakan untuk mengelola CA sebelumnya. masukkan ke direktori *~/easy-rsa* tempat membuat kunci, lalu mengimpor permintaan sertifikat menggunakan skrip *easy-rsa*:

$ cd ~/easy-rsa

$ ./easyrsa import-req /tmp/Server.req Server

Selanjutnya, tanda tangani permintaan dengan menjalankan skrip *easy-rsa* menggunakan opsi *sign-req*, diikuti oleh tipe permintaan dan *Common Name*. Tipe permintaan dapat berupa *Client* atau *Server*. Karena bekerja dengan permintaan sertifikat *Server* OpenVPN, pastikan menggunakan tipe permintaan *Server*:

$ ./easyrsa sign-req Server Server

Pada keluaran, akan diminta memverifikasi bahwa permintaan itu berasal dari sumber tepercaya. Ketik *yes*, lalu tekan *ENTER* untuk mengonfirmasi:

Setelah langkah-langkah ini selesai, permintaan sertifikat *Server* OpenVPN telah ditandatangani menggunakan kunci privat *Server* CA. Berkas *Server.crt* yang dihasilkan memuat kunci enkripsi publik dari *Server* OpenVPN dan juga tanda tangan dari *Server* CA. Maksud tanda tangan adalah memberi tahu siapa pun yang memercayai *Server* CA bahwa mereka juga dapat memercayai *Server* OpenVPN saat mereka terhubung dengannya.

Untuk menyelesaikan pengonfigurasian sertifikat, salin berkas *Server.crt* dan *ca.crt* dari *Server* CA ke *Server* OpenVPN:

$ scp pki/issued/Server.crt ovpn@192.168.10.3:/tmp

$ scp pki/ca.crt ovpn@192.168.10.3:/tmp

Sekarang kembali ke *Server* OpenVPN, salin berkas dari */tmp* ke */etc/openvpn/Server*:

$ sudo cp /tmp/{Server.crt,ca.crt} /etc/openvpn/Server

1.1.1.4.5 Mengonfigurasi Materi Kriptografis OpenVPN

Untuk lapisan keamanan tambahan, tambah kunci rahasia bersama yang akan digunakan *Server* dan semua klien dengan arahan *tls-crypt* dari OpenVPN. Opsi ini digunakan untuk mengaburkan sertifikat TLS yang digunakan saat *Server* dan klien saling terhubung pada awalnya. Ini juga digunakan oleh *Server* OpenVPN untuk melakukan pemeriksaan cepat pada paket masuk. Jika paket ditandatangani menggunakan kunci yang dibagikan sebelumnya, maka *Server* memprosesnya. Jika tidak ditandatangani, maka *Server* tahu kalau paket itu berasal dari sumber tidak tepercaya dan bisa mengabaikannya tanpa perlu melakukan pekerjaan dekripsi tambahan.

Opsi ini akan membantu memastikan bahwa *Server* OpenVPN dapat mengatasi lalu lintas tidak terautentikasi, *port*, pemindaian *port*, dan serangan *Denial of Service*, yang dapat membebani sumber daya *Server*. Ini juga akan membuat sulit identifikasi lalu lintas jaringan OpenVPN.

Untuk membuat kunci *tls-crypt* yang dibagikan sebelumnya, jalankan perintah berikut ini pada *Server* OpenVPN di direktori *~/easy-rsa*:

$ cd ~/easy-rsa

$ openvpn --genkey --secret ta.key

Hasilnya akan berupa berkas yang bernama *ta.key*. Salin ke direktori */etc/openvpn/Server/:*

$ sudo cp ta.key /etc/openvpn/Server

Dengan berkas-berkas ini berada di *Server* OpenVPN, *Server* OpenVPN siap membuat sertifikat klien dan berkas kunci untuk pengguna, yang akan digunakan untuk terhubung ke VPN.

1.1.1.4.6 Membuat Sertifikat Klien dan Pasangan Kunci

Buat kunci klien tunggal dan pasangan sertifikat. Untuk klien yang lain, dapat mengulangi proses ini untuk masing-masingnya. Setiap *Client* harus dibuat dengan nama yang unik ke skrip.

Mulai dengan membuat struktur direktori di dalam direktori *home* untuk menyimpan berkas sertifikat dan kunci klien:

$ mkdir -p ~/Client-configs/keys

Karena pasangan kunci/sertifikat akan disimpan disini dan berkas konfigurasi di dalam direktori ini, kunci izinnya sekarang sebagai tindakan keamanan:

$ chmod -R 700 ~/Client-configs

Selanjutnya, arahkan kembali ke direktori *EasyRSA* dan jalankan skrip *easy-rsa* dengan opsi *gen-req* dan *nopass*, bersama dengan *Common Name* untuk klien:

$ cd ~/easy-rsa

$ ./easyrsa gen-req Client1 nopass

Tekan *ENTER* untuk mengonfirmasi *Common Name*. Lalu, salin berkas *Client1.key* ke direktori *~/Client-configs/keys/* yang sudah dibuat sebelumnya:

$ cp pki/private/Client1.key ~/Client-configs/keys/

Selanjutnya, *transfer* berkas *Client1.req* ke *Server* CA menggunakan metode aman:

$ scp pki/reqs/*Client*1.req caserv@192.168.10.4:/tmp

Sekarang masuk ke *Server* CA. Lalu ke direktori *EasyRSA*, dan impor permintaan sertifikat:

$ cd ~/easy-rsa

$ ./easyrsa import-req /tmp/Client1.req Client1

Selanjutnya, tanda tangani permintaan sama seperti yang dilakukan untuk *Server* pada langkah sebelumnya. Namun kali ini, tipe permintaannya *Client*:

$ ./easyrsa sign-req Client Client1

Ketika diminta, ketikkan *yes* untuk mengonfirmasi bahwa berniat menandatangani permintaan sertifikat dan bahwa itu berasal dari sumber tepercaya:

Ini akan membuat berkas sertifikat klien yang bernama *Client1.crt*. *Transfer* berkas ini kembali ke *Server*:

$ scp pki/issued/Client1.crt ovpn@192.168.10.3:/tmp

Kembali ke *Server* OpenVPN, salin sertifikat klien ke direktori *~/Client-configs/keys/*:

$ cp /tmp/Client1.crt ~/Client-configs/keys/

Selanjutnya, salin berkas *ca.crt* dan *ta.key* ke direktori *~/Client-configs/keys/* juga, dan atur izin yang sesuai untuk pengguna *sudo*:

$ cp ~/easy-rsa/ta.key ~/Client-configs/keys/

$ sudo cp /etc/openvpn/Server/ca.crt ~/Client-configs/keys/

*$ sudo chown ovpn.ovpn ~/Client-configs/keys/**

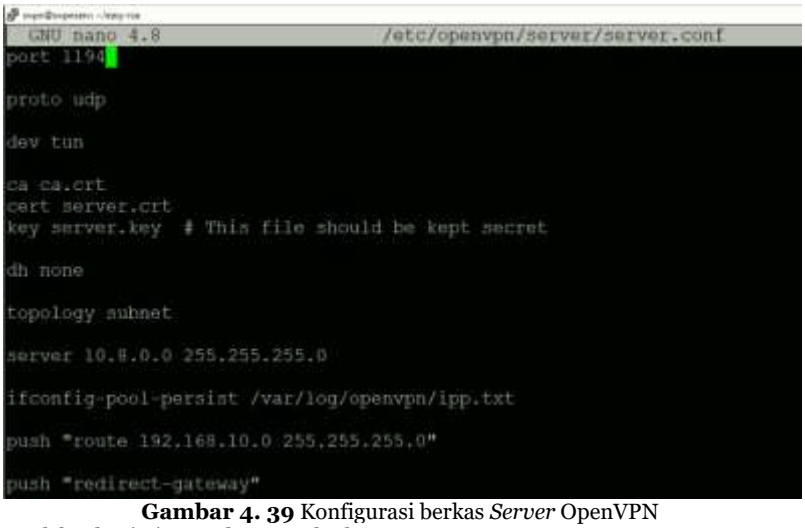
Dengan itu, sertifikat dan kunci *Server* dan *klien* telah dibuat dan disimpan di dalam direktori yang sesuai pada *Server* OpenVPN.

1.1.1.4.7 Mengonfigurasi OpenVPN

Buat berkas konfigurasi dengan nama *Server.conf*, buat dengan *nano*

$ sudo nano /etc/openvpn/Server/Server.conf

Isi berkas dengan konfigurasi OpenVPN seperti pada Gambar 4. 41 agar dapat berjalan sesuai perancangan.



Gambar 4. 39 Konfigurasi berkas *Server* OpenVPN

Setelah selesai, simpan dan tutup berkas.

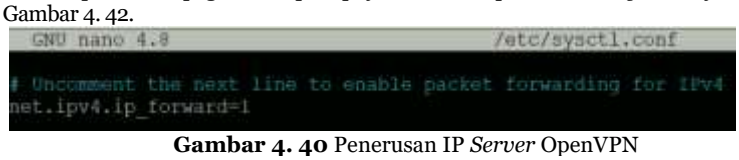
1.1.1.4.8 Menyesuaikan Konfigurasi Jaringan *Server* OpenVPN

Ada beberapa aspek dari konfigurasi jaringan *Server* yang perlu diubah supaya OpenVPN dapat mengalirkan lalu lintas melalui VPN dengan benar. Yang pertama adalah penerusan IP, suatu metode untuk menentukan ke mana lalu lintas IP harus dialirkan. Ini esensial untuk fungsionalitas VPN yang akan diberikan oleh *Server*.

Untuk menyesuaikan pengaturan penerusan IP dari *Server* OpenVPN, buka berkas */etc/sysctl.conf* menggunakan *nano*:

$ sudo nano /etc/sysctl.conf

Lalu hapus tanda pagar *#net.ipv4.ip_forward = 1* pada berkas *sysctl.conf* . Dapat dilihat pada Gambar 4. 42.



Gambar 4. 40 Penerusan IP *Server* OpenVPN

Sekarang *Server* OpenVPN bisa meneruskan lalu lintas masuk dari satu perangkat *ethernet* ke perangkat lain. Pengaturan ini memastikan bahwa *Server* dapat mengarahkan lalu lintas dari klien yang terhubung pada antarmuka VPN virtual melalui perangkat *ethernet* fisik lain miliknya. Konfigurasi ini akan mengalirkan semua lalu lintas *web* dari klien melalui alamat IP *Server*, dan alamat IP publik pada klien akan secara efektif disembunyikan.

1.1.1.4.9 Konfigurasi *Firewall*

Untuk mengizinkan OpenVPN melintasi *firewall*, perlu diaktifkan penyamaran, suatu konsep *iptables* yang menyediakan *Network Address Translation* (NAT) secara langsung untuk mengalirkan koneksi klien dengan benar.

Sebelum membuka berkas konfigurasi *firewall* untuk menambah aturan penyamaran, harus menemukan antarmuka jaringan publik dari mesin terlebih dahulu. Untuk melakukan ini, ketik:

$ ip route list default

Buka berkas */etc/ufw/before.rules* untuk menambahkan konfigurasi yang relevan:

$ sudo nano /etc/ufw/before.rules

Pada bagian atas berkas, tambahkan baris yang disorot di bawah ini. Ini akan mengatur *default policy* untuk *POSTROUTING chain* pada *tabel nat* dan menyamarkan lalu lintas apa pun yang berasal dari VPN. Dilihat pada Gambar 4. 43.

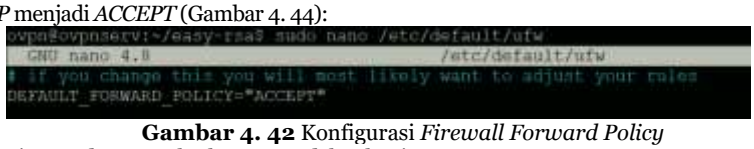
Gambar 4. 41 Konfigurasi *firewall before.rules*

Simpan dan tutup berkas saat sudah selesai.

Selanjutnya, izinkan paket yang diteruskan secara default juga. Untuk melakukan ini, buka berkas */etc/default/ufw*:

$ sudo nano /etc/default/ufw

Di dalam berkas itu, temukan arahan *DEFAULT_FORWARD_POLICY* dan ubah nilai dari *DROP* menjadi *ACCEPT* (Gambar 4. 44):



Gambar 4. 42 Konfigurasi *Firewall Forward Policy*

Simpan dan tutup berkas saat sudah selesai.

Selanjutnya, sesuaikan *firewall* itu untuk mengizinkan lalu lintas ke OpenVPN. Buka lalu lintas UDP ke port 1194.

$ sudo ufw allow 1194/udp

$ sudo ufw allow OpenSSH

Setelah menambahkan aturan itu, aktifkan kembali UFW untuk memulai ulang dan memuat perubahan dari semua berkas yang sudah dimodifikasi:

$ sudo ufw disable

$ sudo ufw enable

Server kini terkonfigurasi untuk menangani lalu lintas OpenVPN dengan benar. Dengan sudah adanya aturan *firewall*, layanan OpenVPN pada *Server* dapat dimulai.

1.1.1.4.10 Memulai *OpenVPN*

OpenVPN berjalan sebagai layanan *systemd*, sehingga dapat menggunakan *systemctl* untuk mengelolanya. Konfigurasi OpenVPN untuk memulai saat melakukan *boot* sehingga dapat terhubung ke VPN kapan saja selama *Server* aktif. Untuk melakukan ini, aktifkan layanan OpenVPN dengan menambahkan itu ke *systemctl*:

$ sudo systemctl -f enable openvpn-Server@Server.service

Lalu, mulai layanan OpenVPN:

$ sudo systemctl start openvpn-Server@Server.service

Periksa ulang bahwa layanan OpenVPN sudah aktif dengan perintah berikut. Anda akan melihat *active* (*running*) pada keluaran:

$ sudo systemctl status openvpn-Server@Server.service

Konfigurasi bagian *Server* untuk OpenVPN telah selesai. Selanjutnya konfigurasi untuk *Client* agar terhubung ke *Server* OpenVPN.

1.1.1.4.11 Menciptakan Infrastruktur Konfigurasi Klien

Buat berkas konfigurasi "*base*", lalu membuat skrip yang akan mengizinkan untuk membuat berkas konfigurasi klien, sertifikat, dan kunci yang unik sesuai kebutuhan.

Mulai dengan membuat direktori baru tempat akan menyimpan berkas konfigurasi klien di dalam direktori *Client-configs* yang dibuat sebelumnya:

$ mkdir -p ~/Client-configs/files

Selanjutnya, salin berkas konfigurasi klien contoh ke direktori *Client-configs* yang digunakan sebagai konfigurasi dasar:

$ cp /usr/share/doc/openvpn/examples/sample-config-files/Client.conf ~/Client-configs/base.conf

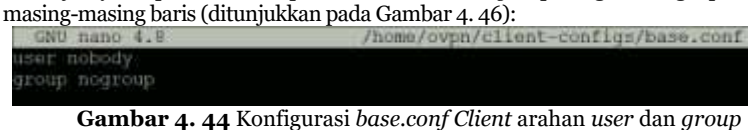
Buka berkas baru ini menggunakan *nano*:

$ nano ~/Client-configs/base.conf

Di dalam berkas, temukan arahan remote. Ketik alamat IP Publik *Server* OpenVPN dan *port* yang digunakan *Server* OpenVPN. Dapat dilihat pada Gambar 4. 45

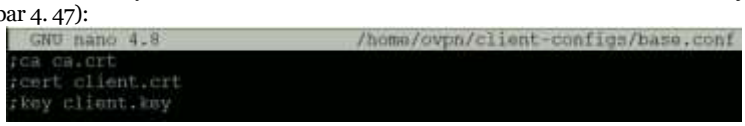
Gambar 4. 43 Konfigurasi *base.conf Client Protocol*, IP *Public* dan *Port* OpenVPN

Selanjutnya, hapus komentar pada arahan *user* dan *group* dengan menghapus tanda ; pada awal masing-masing baris (ditunjukkan pada Gambar 4. 46):



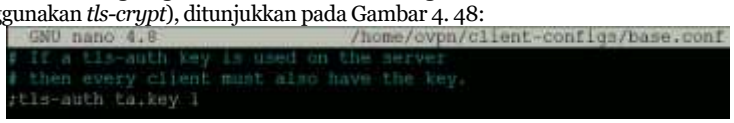
Gambar 4. 44 Konfigurasi *base.conf Client* arahan *user* dan *group*

Temukan arahan yang mengatur *ca*, *cert*, dan *key*. Beri tanda komentar pada arahan ini karena cert dan key akan ditambahkan di dalam berkas itu sendiri nanti (ditunjukkan pada Gambar 4. 47):



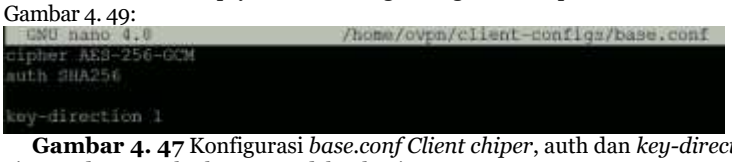
Gambar 4. 45 Konfigurasi *base.conf Client ca, cert* dan *key*

Dengan cara yang serupa, beri tanda komentar pada arahan *tls-auth*, karena ta.key akan ditambahkan langsung ke dalam berkas konfigurasi klien (dan *Server* disiapkan untuk menggunakan *tls-crypt*), ditunjukkan pada Gambar 4. 48:



Gambar 4. 46 Konfigurasi *base.conf Client* arahan *tls-auth*

Salin pengaturan *cipher* dan *auth* yang sudah diatur pada berkas */etc/openvpn/Server/Server/Server.conf* dan tambahkan arahan *key-direction* di dalam berkas. Atur ke "1" supaya VPN berfungsi dengan benar pada mesin klien, seperti pada Gambar 4. 49:



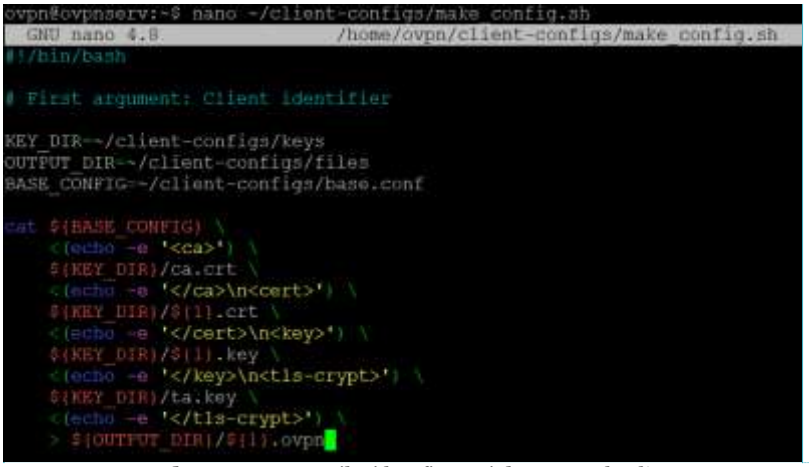
Gambar 4. 47 Konfigurasi *base.conf Client chiper*, auth dan *key-direction*

Simpan dan tutup berkas saat sudah selesai.

Buat skrip yang akan mengompilasi konfigurasi dasar dengan berkas sertifikat, kunci, dan enkripsi yang relevan, lalu menaruh konfigurasi yang dibuat di dalam direktori *~/Client-configs/files*. Buka berkas baru yang bernama *make_config.sh* di dalam direktori *~/Client-configs*:

$ nano ~/Client-configs/make_config.sh

Di dalam berkas itu, tambahkan konfigurasi berikut (ditunjukkan pada Gambar 4. 50):



Gambar 4. 48 Kompilasi konfigurasi dasar untuk *Client*

Simpan dan tutup berkas saat sudah selesai.

Sebelum melanjutkan, pastikan untuk menandai berkas ini agar dapat dieksekusi dengan mengetik:

$ chmod 700 ~/Client-configs/make_config.sh

Skrip ini akan membuat salinan berkas *base.conf* yang sudah dibuat, mengumpulkan semua berkas sertifikat dan kunci yang dibuat untuk klien, mengekstrak kontennya, menambahkannya ke salinan berkas konfigurasi dasar, dan mengekspor semua konten ini ke dalam berkas konfigurasi klien yang baru. Setiap kali menambahkan klien baru, perlu membuat kunci dan sertifikat baru untuk klien itu sebelum dapat menjalankan skrip ini dan membuat berkas konfigurasinya.

1.1.1.4.12 Membuat Konfigurasi Klien

Sebelumnya sudah dibuat sertifikat dan kunci klien yang bernama *Client1.crt* dan *Client1*, secara berurutan. Selanjutnya buat berkas konfigurasi untuk kredensial ini dengan pindah ke direktori *~/Client-configs* dan menjalankan skrip yang sudah dibuat pada langkah sebelumnya:

$ cd ~/Client-configs

$ ./make_config.sh Client1

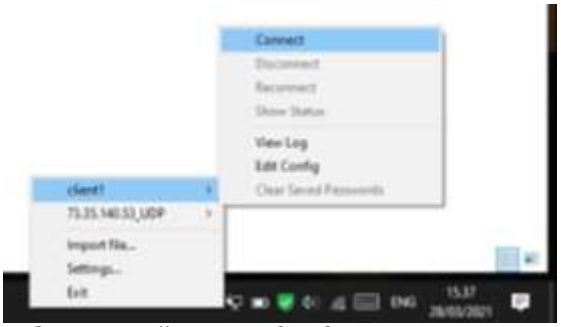
Ini akan membuat berkas bernama *Client1.ovpn* pada direktori *~/Client-configs/files*:

Transfer berkas ini ke perangkat yang digunakan sebagai klien.

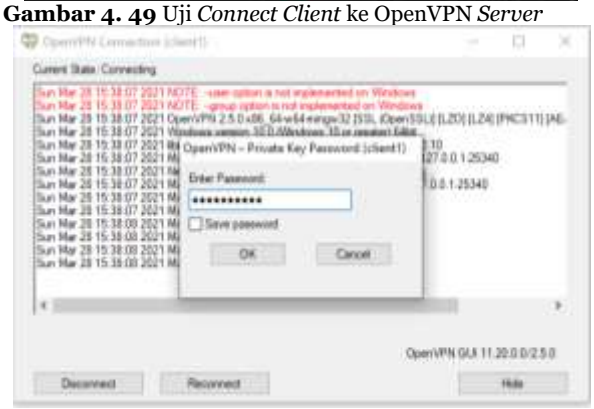
1.1.1.4.13 Koneksi Klien ke OpenVPN *Server*

Konfigurasi *Client* telah dibuat, selanjutnya di uji apakah *Client* sudah dapat terhubung ke OpenVPN *Server*. Perangkat *Client* yang akan di uji menggunakan sistem operasi Windows 10. Instal aplikasi OpenVPN *Client* pada Windows, dan salin konfigurasi *Client* ke direktori *config* OpenVPN. Uji koneksi seperti pada Gambar 4. 51, Gambar 4. 52 dan Gambar 4. 53.

Commented [MF2]: Mohon untuk gambar disamaratakan ya ukurannya.



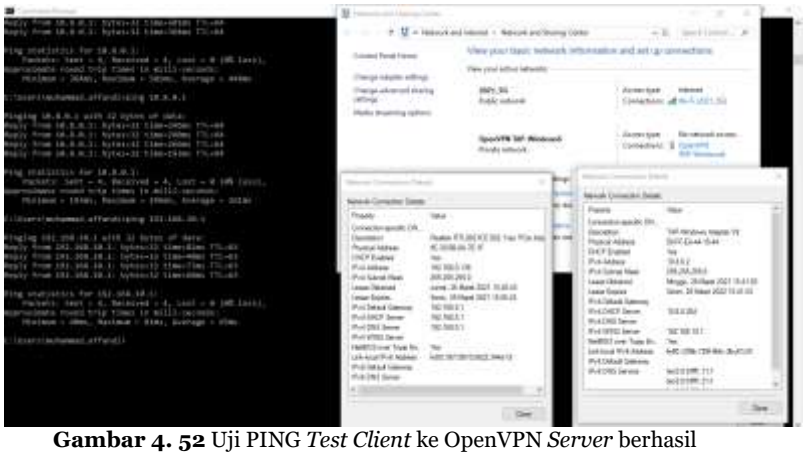
Gambar 4. 49 Uji *Connect Client* ke OpenVPN *Server*



Gambar 4. 50 Masukkan *Password Client* untuk terhubung ke OpenVPN *Server*

Gambar 4. 51 *Client* Berhasil terhubung ke OpenVPN *Server*

Selanjutnya dilakukan uji PING *Test* dari *client* ke OpenVPN *server*, seperti ditunjukkan pada Gambar 4.54.



Gambar 4. 52 Uji PING *Test Client* ke OpenVPN *Server* berhasil

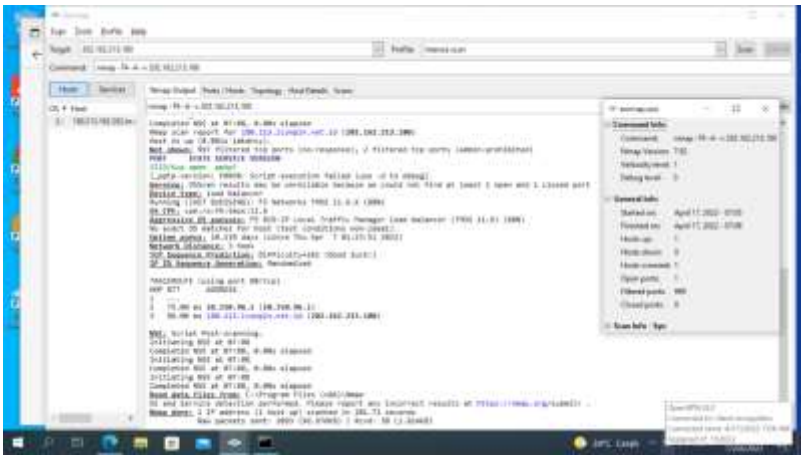
1.1.2 Pengamatan Keamanan Setelah Terkoneksi OpenVPN

Pengujian keamanan setelah terkoneksi VPN berfungsi untuk menganalisa keamanan jika *Client* telah terkoneksi pada OpenVPN. Berikut adalah hasil pengamatan keamanan setelah menggunakan OpenVPN.

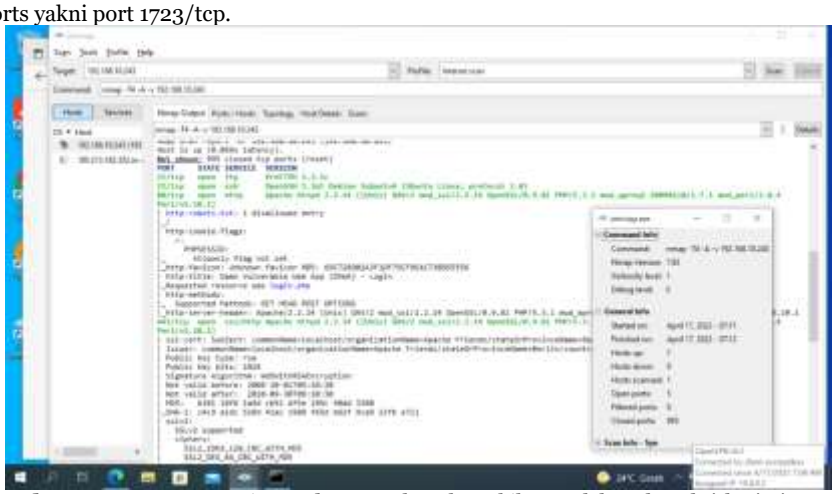
1.1.2.1 Pengamatan Keamanan Setelah Terkoneksi OpenVPN Tanpa *pre shared keys* dan sertifikat SSL

1. Pengujian *port scanning*
   Pengujian dilakukan menggunakan *tool* nmap atau zenmap pada windows. *Port scanning* dilakukan pada IP *Address* IP *Public Server* OpenVPN, IP *Address* FreeNAS, dan IP *Address Web Vulnerability*. Berikut *capture*—nya, dapat dilihat pada Gambar 4. 71.

   Gambar 4. 53 *Port scanning* pada IP Public *Server* OpenVPN setelah terkoneksi ke jaringan OpenVPN Tanpa *pre shared keys* dan sertifikat SSL



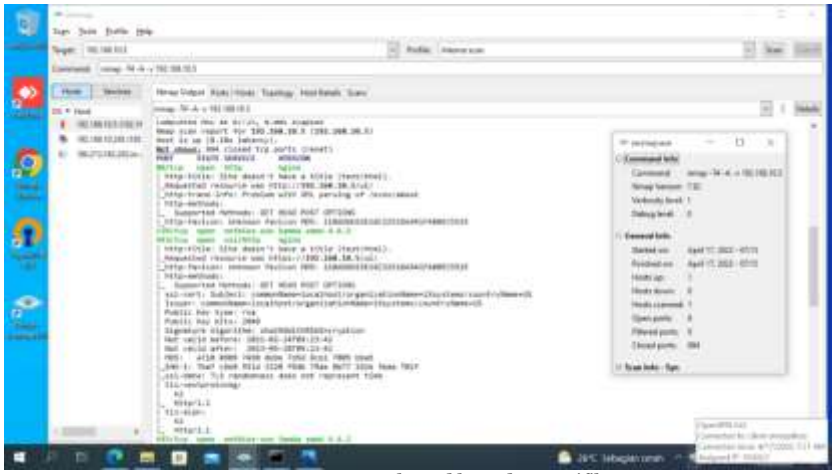
Pada Gambar 4. 71 dapat dilihat dari 1000 port yang di-scan terdapat 999 filtered ports, 1 open ports yakni port 1723/tcp.



Gambar 4. 54 *Port scanning* pada IP *Web Vulnerabilty* setelah terkoneksi ke jaringan OpenVPN Tanpa *pre shared keys* dan sertifikat SSL

Pada Gambar 4. 72 dapat dilihat dari 1000 *port* yang di-*scan* terdapat 0 *filtered ports,* 995 *closed ports*, 5 *open ports* yakni port 21/tcp, 22/tcp, 80/tcp, 443/tcp, 3306/tcp.

Gambar 4. 55 *Port scanning* pada IP FreeNAS setelah terkoneksi ke jaringan OpenVPN

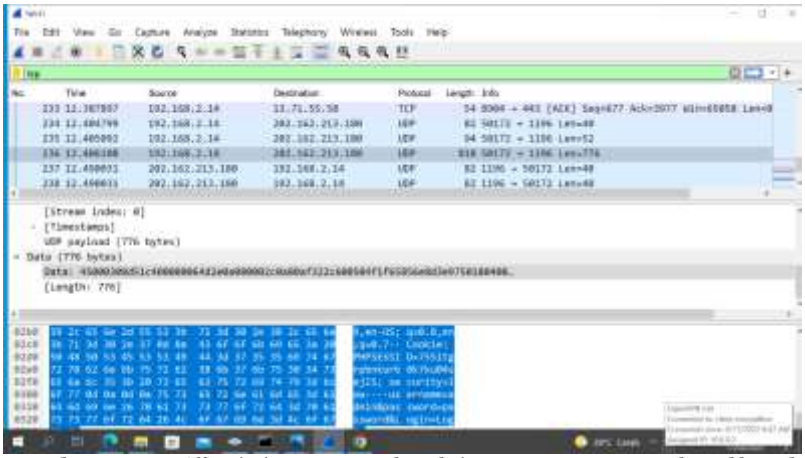


Tanpa *pre shared keys* dan sertifikat SSL

Pada Gambar 4. 73 dapat dilihat dari 1000 *port* yang di-*scan* terdapat 0 *filtered ports*, 994 *closed ports*, 6 *open ports* yakni port 80/tcp, 139/tcp, 443/tcp, 445/tcp, 5357/tcp, 6000/tcp.

2. Pengujian *sniffing*

Berikut merupakan hasil *capture* jaringan yang terkoneksi OpenVPN Tanpa *pre shared keys* dan sertifikat SSL (Gambar 4. 74).

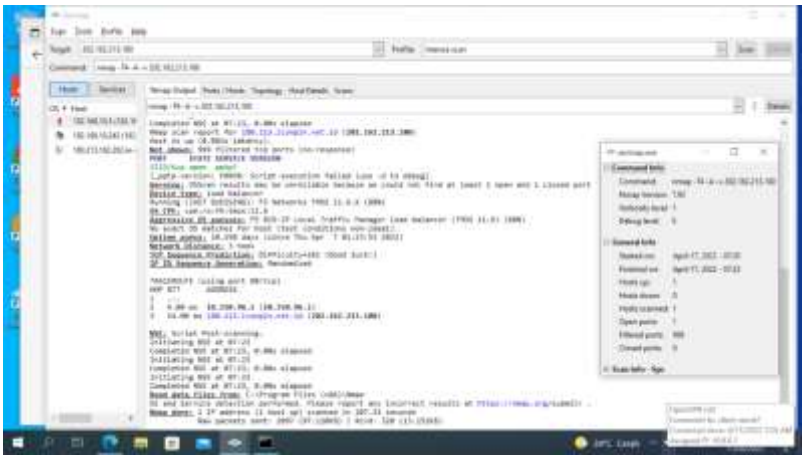


Gambar 4. 56 *Sniffing* jaringan yang terkoneksi OpenVPN Tanpa *pre shared keys* dan sertifikat SSL

Pada Gambar 4. 74 terlihat bahwa protokol yang sebelumnya pada jaringan lokal menggunakan protokol ICMP, ARP, TCP dan HTTP berubah menjadi protokol UDP dan masih bisa membaca paket yang berisi *username* dan *password*. Ini menunjukkan bahwa jaringan OpenVPN Tanpa *pre shared keys* dan sertifikat SSL tidak ter-enkripsi sehingga dapat disadap oleh *attacker*.

1.1.2.2 Pengamatan Keamanan Setelah Terkoneksi OpenVPN Menggunakan *pre shared keys*

1. Pengujian *port scanning*
   Pengujian dilakukan menggunakan *tool* nmap atau zenmap pada windows. *Port scanning* dilakukan pada IP *Address* IP *Public Server* OpenVPN, IP Address FreeNAS, dan IP Address Web Vulnerability. Berikut *capture*-nya seperti pada Gambar 4. 75.

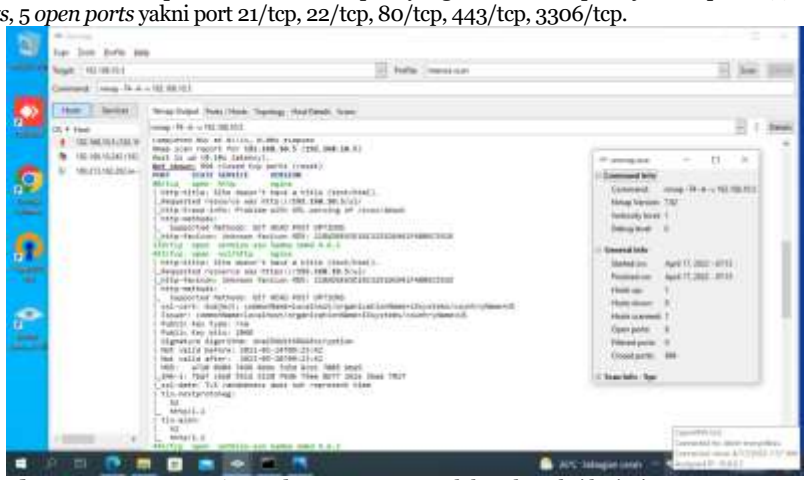


Gambar 4. 57
Port scanning pada IP Public *Server* OpenVPN setelah terkoneksi ke jaringan OpenVPN menggunakan *pre shared keys*.

Pada Gambar 4. 75 dapat dilihat dari 1000 *port* yang di-*scan* terdapat 999 *filtered ports*, 1 *open ports* yakni port 1723/tcp.

Gambar 4. 58
Port scanning pada IP *Web Vulnerabilty* setelah terkoneksi ke jaringan OpenVPN menggunakan *pre shared keys*

Pada Gambar 4. 76 dapat dilihat dari 1000 *port* yang di-*scan* terdapat 0 *filtered ports,* 995 *closed ports*, 5 *open ports* yakni port 21/tcp, 22/tcp, 80/tcp, 443/tcp, 3306/tcp.

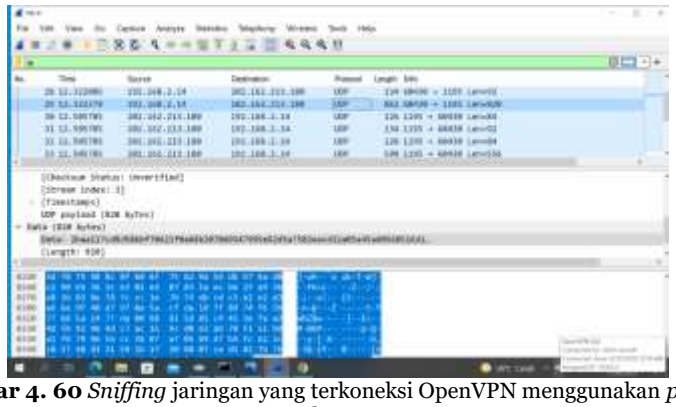


Gambar 4. 59 *Port scanning* pada IP FreeNAS setelah terkoneksi ke jaringan OpenVPN menggunakan *pre shared keys*

Pada Gambar 4. 77 dapat dilihat dari 1000 *port* yang di-*scan* terdapat 0 *filtered ports*, 994 *closed ports*, 6 *open ports* yakni port 80/tcp, 139/tcp, 443/tcp, 445/tcp, 5357/tcp, 6000/tcp.

2. Pengujian *sniffing*

Berikut merupakan hasil *capture* jaringan yang terkoneksi OpenVPN menggunakan *pre shared keys*, ditunjukkan pada Gambar 4. 78).



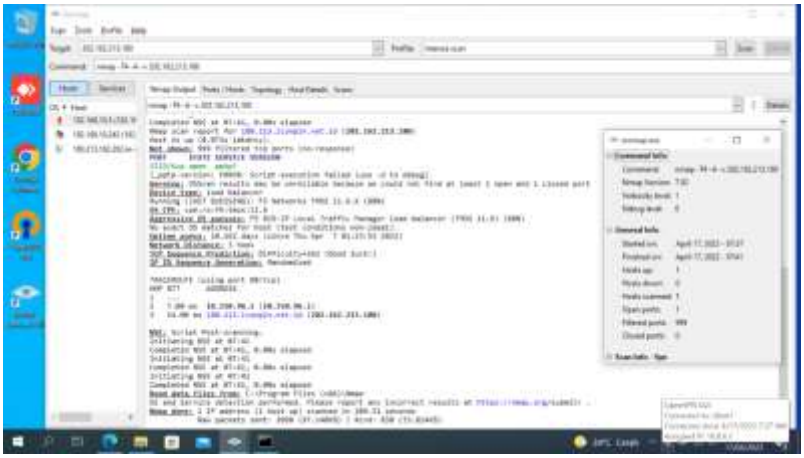
Gambar 4. 60 *Sniffing* jaringan yang terkoneksi OpenVPN menggunakan *pre shared keys*

Pada Gambar 4. 78 terlihat bahwa protokol yang digunakan menggunakan protokol UDP dan paket berisi *username* dan *password* yang sebelumnya tidak ter-enkripsi sekarang sudah ter-enkripsi. Ini menunjukkan bahwa jaringan OpenVPN menggunakan *pre shared keys* sudah ter-enkripsi dan menjadi lebih aman dibandingkan OpenVPN Tanpa *pre shared keys* dan sertifikat SSL sehingga walaupun disadap oleh *attacker* paket sudah terlindungi isinya.

1.1.2.3 Pengamatan Keamanan Setelah Terkoneksi OpenVPN Menggunakan Sertifikat SSL dan Kunci TLS

1. Pengujian *port scanning*

Pengujian dilakukan menggunakan *tool* nmap atau zenmap pada windows. *Port scanning* dilakukan pada IP *Address* IP *Public Server* OpenVPN, IP Address FreeNAS, dan IP *Address Web Vulnerability*. Berikut *capture*-nya seperti pada Gambar 4. 79.



Gambar 4. 61 *Port scanning* pada IP *Public Server* OpenVPN setelah terkoneksi ke jaringan OpenVPN menggunakan Sertifikat SSL dan Kunci TLS.

Pada Gambar 4. 79 dapat dilihat dari 1000 *port* yang di-*scan* terdapat 999 *filtered ports*, 1 *open ports* yakni port 1723/tcp.

Gambar 4. 62 *Port scanning* pada IP Web Vulnerabilty setelah terkoneksi ke jaringan

OpenVPN menggunakan Sertfikat SSL dan Kunci TLS

Pada Gambar 4. 80 dapat dilihat dari 1000 *port* yang di-*scan* terdapat 0 *filtered ports,* 995 *closed ports*, 5 *open ports* yakni port 21/tcp, 22/tcp, 80/tcp, 443/tcp, 3306/tcp.

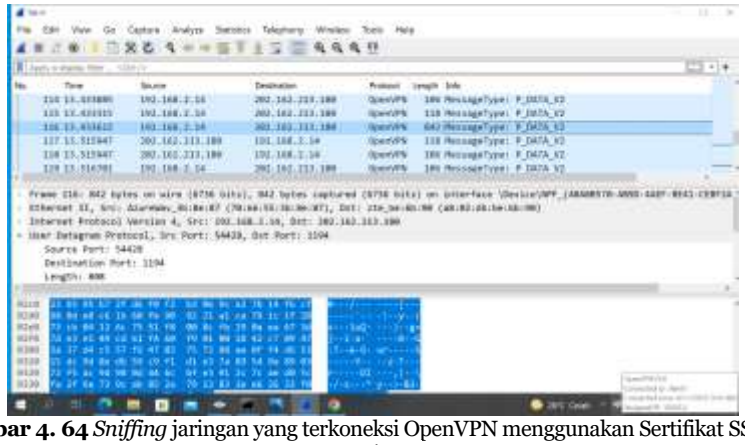
Gambar 4. 63 *Port scanning* pada IP FreeNAS setelah terkoneksi ke jaringan OpenVPN

menggunakan Sertifikat SSL dan Kunci TLS

Pada Gambar 4. 81 dapat dilihat dari 1000 *port* yang di-*scan* terdapat 0 *filtered ports*, 994 *closed ports*, 6 *open ports* yakni *port* 80/tcp, 139/tcp, 443/tcp, 445/tcp, 5357/tcp, 6000/tcp.

2. Pengujian *sniffing*

Berikut merupakan hasil *capture* jaringan yang terkoneksi OpenVPN menggunakan Sertifikat SSL dan Kunci TLS, ditunjukkan pada Gambar 4. 82.



Gambar 4. 64 *Sniffing* jaringan yang terkoneksi OpenVPN menggunakan Sertifikat SSL dan Kunci TLS

Pada Gambar 4. 82 terlihat bahwa protokol yang digunakan menggunakan protokol OpenVPN, tidak terbaca protokol UDP lagi. Semua paket yang melintas telah ter-enkripsi. Ini menunjukkan bahwa jaringan OpenVPN menggunakan Sertifikat SSL dan Kunci TLS sudah ter-enkripsi dengan baik dan juga melalui protokol sendiri sehingga jaringan yang telah terkoneksi OpenVPN menggunakan Sertifikat SSL dan Kunci TLS menjadi aman dari *attacker*.

1.2 Analisis Hasil Pengujian

Berdasarkan hasil pengujian sistem pada jaringan lokal maupun pada jaringan OpenVPN yang diuji pada 6 *Provider* penyedia layanan internet, jaringan OpenVPN yang dibangun berhasil terkoneksi dengan *Server* OpenVPN sehingga membuat jaringan OpenVPN ini teruji fleksibel dari segi koneksi tanpa terbatas ruang dan waktu. Baik jaringan lokal dan jaringan OpenVPN berhasil melakukan akses ke *Server* FreeNAS yang telah dibangun, *Client* dapat melakukan *copy data* dari komputer *Client* ke *Server* SMB FreeNAS maupun sebaliknya dari *Server* FreeNAS ke komputer *Client*. *Client* juga dapat mengakses *file* langsung dari *Server* FreeNAS, contohnya menjalankan *file* .mp4.

Berdasarkan hasil pengujian performa yang dilakukan pada jaringan OpenVPN yang telah di implementasi, OpenVPN tanpa *pre shared keys* dan sertifikat SSL, OpenVPN menggunakan *pre shared keys* dan OpenVPN menggunakan sertifikat SSL dan kunci tls dengan parameter QoS *packet loss*, *round trip*, *transfer file*, *Jitter*, *Upload* dan *Download*. Ketika *Client* melakukan koneksi ke OpenVPN Tanpa *pre shared keys* dan sertifikat SSL. 6 *Provider* memiliki nilai *packet loss* 0%. Pada pengujian *round trip* waktu rata-rata tercepat selama 28 ms (Biznet) dan terlama 74 ms (ICONNET). Pengujian *transfer* file 100 Mb dari *Client* ke *Server* waktu rata-rata terlama yakni 719 s (Indihome) dan tercepat 23 s (Biznet). Pengujian *transfer* file 100 Mb dari *Server* ke *Client* waktu rata-rata terlama yakni 359 s (Indosat Ooreedoo) dan tercepat 25 s (Biznet). Pengujian *Speedtest* memiliki parameter *Jitter*, *Download* dan *Upload*, *Jitter* terbesar 9,53 ms (Indosat Ooreedoo), *Jitter* terkecil 0,26 ms (ICONNET). *Download* terbesar 31,9 Mbps (Biznet), *Download* terkecil 10,1 Mbps (Indihome). *Upload* terbesar 49,3 Mbps (Biznet), *Upload* terkecil 2,64 Mbps (Indihome). Ketika *Client* melakukan koneksi ke OpenVPN Menggunakan *pre shared keys*. 6 *Provider* memiliki nilai *packet loss* 0%. Pada pengujian *round trip* waktu rata-rata tercepat selama 28 ms (Biznet) dan terlama 75 ms (ICONNET). Pengujian *transfer* file 100 Mb dari *Client* ke *Server* waktu rata-rata terlama yakni 387 s (Indihome) dan tercepat 23 s (Biznet). Pengujian *transfer* file 100 Mb dari *Server* ke *Client* waktu rata-rata terlama yakni 358 s (Indosat Ooreedoo) dan tercepat 27 s (Biznet). Pengujian *Speedtest* memiliki parameter *Jitter*, *Download* dan *Upload*, *Jitter* terbesar 10,9 ms (Tri), *Jitter* terkecil 0,38 ms (ICONNET). *Download* terbesar 31,7 Mbps (Biznet), *Download* terkecil 10,1 Mbps (Indihome). *Upload* terbesar 47,9 Mbps (Biznet), *Upload* terkecil 2,56 Mbps (Indihome). Ketika *Client* melakukan koneksi ke OpenVPN Menggunakan Sertifikat SSL dan Kunci TLS. 6 *Provider* memiliki nilai *packet loss* 0%. Pada pengujian *round trip*

waktu rata-rata tercepat selama 28 ms (Biznet) dan terlama 75 ms (ICONNET). Pengujian *transfer* file 100 Mb dari *Client* ke *Server* waktu rata-rata terlama yakni 415 s (Indihome) dan tercepat 23 s (Biznet). Pengujian *transfer* file 100 Mb dari *Server* ke *Client* waktu rata-rata terlama yakni 476 s (Indosat Ooreedoo) dan tercepat 27 s (Biznet). Pengujian *Speedtest* memiliki parameter *Jitter*, *Download* dan *Upload*, *Jitter* terbesar 18,7 ms (Simpati), *Jitter* terkecil 0,39 ms (ICONNET). *Download* terbesar 32,9 Mbps (Biznet), *Download* terkecil 10,1 Mbps (Indihome). *Upload* terbesar 47,8 Mbps (Biznet), *Upload* terkecil 2,58 Mbps (Indihome). Perbandingan performa OpenVPN dengan tiga skenario/konfigurasi, OpenVPN tanpa *pre shared keys* dan sertifikat SSL, OpenVPN menggunakan *pre shared keys* dan OpenVPN menggunakan sertifikat SSL dan kunci tls, hasilnya cenderung mirip atau tidak jauh berbeda hasil pengujian performa nya, dari pengujian *packet loss*, *round trip*, *transfer file*, *Jitter*, *Download* dan *Upload*. Yang mempengaruhi pengujian performanya yaitu pada *Provider* internet yang digunakan, kapasitas *bandwidth* masing-masing *Provider* yang digunakan saat pengujian juga mempengaruhi performa pada implementasi OpenVPN ini.

Berdasarkan pengujian keamanan yang dilakukan pada jaringan lokal, dan jaringan OpenVPN dengan tiga skenario/konfigurasi (OpenVPN tanpa *pre shared keys* dan sertifikat SSL, OpenVPN menggunakan *pre shared keys* dan OpenVPN menggunakan sertifikat SSL dan kunci tls) menunjukan hasil bahwa pengujian *port scanning* menunjukkan bahwa OpenVPN tidak mempengaruhi *port* yang dapat dibaca pada *tool* nmap, *port* yang dibaca baik tanpa terkoneksi jaringan OpenVPN dan terkoneksi jaringan OpenVPN dengan tiga skenario/konfigurasi memiliki hasil yang sama. Pengujian *sniffing* yang dilakukan terhadap jaringan lokal, jaringan tidak aman, jaringan tidak ter-enkripsi, jaringan dapat disadap sehingga *attacker* dapat menyadap informasi yang melintas pada jaringan, seperti protokol yang digunakan dan bahkan dapat membaca paket yang berisi *username* dan *password*. Pengujian pada jaringan OpenVPN tanpa *pre shared keys* dan sertifikat SSL menunjukkan protokol yang sebelumnya pada jaringan lokal menggunakan protokol ICMP, ARP, TCP dan HTTP berubah menjadi protokol UDP dan masih bisa membaca paket yang berisi *username* dan *password*. Ini menunjukkan bahwa jaringan OpenVPN Tanpa *pre shared keys* dan sertifikat SSL tidak ter-enkripsi sehingga dapat disadap oleh *attacker*. Pada pengujian OpenVPN menggunakan *pre shared keys* protokol yang digunakan menggunakan protokol UDP dan paket berisi *username* dan *password* yang sebelumnya tidak ter-enkripsi sekarang sudah ter-enkripsi. Ini menunjukkan bahwa jaringan OpenVPN menggunakan *pre shared keys* sudah ter-enkripsi dan menjadi lebih aman dibandingkan OpenVPN Tanpa *pre shared keys* dan sertifikat SSL sehingga walaupun disadap oleh *attacker* paket sudah terlindungi isinya. Pada pengujian OpenVPN menggunakan sertifikat SSL dan kunci tls menunjukkan protokol yang digunakan menggunakan protokol OpenVPN, tidak terbaca protokol UDP. Semua paket yang melintas telah ter-enkripsi. Ini menunjukkan bahwa jaringan OpenVPN menggunakan Sertifikat SSL dan Kunci TLS sudah ter-enkripsi dengan baik dan juga melalui protokol sendiri sehingga jaringan yang telah terkoneksi OpenVPN menggunakan Sertifikat SSL dan Kunci TLS menjadi aman dari *attacker*. Dari beberapa penjelasan tersebut menunjukan bahwa untuk keamanan yang diterapkan, OpenVPN menggunakan Sertifikat SSL dan Kunci TLS lebih baik tingkat keamananya dibanding dengan *tunnel* OpenVPN menggunakan *pre shared keys* dan OpenVPN tanpa *pre shared keys* dan sertifikat SSL.

KESIMPULAN

Setelah dilakukan implementasi dan pengujian OpenVPN pada Network Attached Storage (NAS) FreeNAS dengan mengukur fleksibilitas, performa QoS dan keamanan. Dapat disimpulkan bahwa: Berdasarkan pengujian performa QoS OpenVPN terhadap enam parameter yakni, packet loss, round trip, transfer file, Jitter, Download dan Upload yang diujikan pada enam macam Provider, kapasitas bandwidth Provider internet yang dipakai sangat berpengaruh terhadap kualitas layanan ketika terkoneksi menjadi OpenVPN Client. Semakin tinggi kapasitas bandwidth dari Provider yang digunakan semakin baik kualitas layanannya, semakin cepat kecepatan akses datanya. Sedangkan pengujian performa QoS terhadap tiga skenario/konfigurasi OpenVPN yang di implementasi (OpenVPN tanpa pre shared keys dan sertifikat SSL, OpenVPN menggunakan pre shared keys dan OpenVPN

menggunakan sertifikat SSL dan kunci tls) tidak terlalu mempengaruhi terhadap performa layanan ketika terkoneksi menjadi OpenVPN Client.

BIBLIOGRAFI

Abrianingsih, A. 2017. Rancang Bangun Aplikasi untuk Mempermudah Konfigurasi Port Knocking dalam Mikrotik. Jurnal semanTIK. Vol. 3 (1). Hlm. 107-114.

Afrianto, I., dan E. B. Setiawan. 2014. Kajian Virtual Private Network (VPN) sebagai Sistem Pengamanan (Studi Kasus Jaringan Komputer Unikom). Majalah Ilmiah UNIKOM. Vol. 12 (1). Hlm. 43-52.

Biro Hukum dan Komunikasi Publik BSSN. Peringatan Keamanan Ransomware Deadbolt pada Perangkat QNAP Network Attached Storage (NAS). Februari 15, 2022. BSSN. https://bssn.go.id/peringatan-keamanan-ransomware-deadbolt-pada-perangkat-qnap-network-attached-storage-nas/

Carter, G.; Ts, J., dan Eckstein, R. 2007. Using SAMBA. California: O Reilly Media, Inc.

Defni, dan C. Prabowo. 2013. Perancangan dan Implementasi Data Loss Prevention System dengan Menggunakan Network Attached Storage. Jurnal TEKNOIF. Vol. 1 (2). Hlm. 45-60.

Doraswamy, N., dan D. Harkins, 2003. IPSec: The New Security Standard for Internet, Intranet, and Virtual Private Network. 2nd Ed. New Jersey: Prentice Hall.

Dowd, P. W., dan J. T. McHenry. 1998. Network Security: It's Time to Take It Seriously. IEEE Computer. Vol. 31 (9). Hlm. 24-28.

ETSI (European Telecommunications Standards Institute). 1999. Telecommunications and Internet Protocol Harmonization Over Networks (TIPHON); General Aspects of Quality of Service (QoS). Valbonne: ETSI.

Feilner, M. 2006. OpenVPN: Building and Integrating Virtual Private Networks. Birmingham: Packt Publishing Ltd.

Feit, S.; Ranade, J., dan Advisor, S. 1993. TCP/IP Architecture, Protocols, and Implementation. New York: McGraw-Hill.

Frank, M. 2017. Introduction to networks v6: Companion guide. Indianapolis, IN: Cisco Press.

Garms, P., dan S. Murphy. 2018. Consider Data Security. October 15, 2020. Security Today. https://securitytoday.com/articles/2018/09/01/consider-data-security.aspx

Gonzales, B. 2015. What Is a NAS (Network Attached Storage Device)?. October 15, 2020. Lifewire. https://www.lifewire.com/what-is-a-nas-1847428

Gustriansyah, R. 2006. Remote Virtual the Data Storage. Jurnal Informatika. Vol. 7 (2). Hlm. 120-125.

Halim, R. M. N. 2019. Penerapan Network Attached Storage (NAS) Berbasis Raspberry Pi di LP3SDM AZRA Palembang. Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK). Vol. 6 (3). Hlm. 309-314.

Hansen, B. P. 2001. Operating System Principles. New Jersey: Prentice Hall.

ITU-T. 2003. Security in Telecommunication and Information Technology: An Overview of Issues and the Deployment of Existing ITU-T Recommendations for Secure Telecommunications. Switzerland: ITU Publications.

Nugroho, T. A., Suakanto, S., & A, S. F. (2016). Sistem Logging Data Menggunakan FTP Berbasis Jaringan 3G. Jurnal Telematika, 10(1), 13–20.

Ruslianto, I., dan U. Ristian. 2019. Perancangan dan Implementasi Virtual Private Network (VPN) Menggunakan Protokol SSTP (Secure Socket Tunneling Protocol) Mikrotik di Fakultas MIPA Universitas Tanjungpura. CESS (Journal of Computer Engineering System and Science). Vol. 4 (1). Hlm. 74-77.

Ruwaida, D., dan D. Kurnia. 2018. Rancang Bangun File Transfer Protocol (FTP) dengan Pengamanan Open SSL pada Jaringan VPN Mikrotik di SMKS Dwiwarna. CESS (Journal of Computer Engineering System and Science). Vol. 3 (1). Hlm. 45-49.

Sahala, A. 2014. Implementasi Jaringan dengan Linux Ubuntu. Yogyakarta: Andi Yogyakarta.

Sims, G. 2008. Learning FreeNAS (Configure and Manage Network Attached Storage Solution). Birmingham: Packt Publishing Ltd.

Stern, H.; Eisler, M., dan Labiaga, R. 2007. Managing NFS and NIS. California: O'Reilly Media, Inc.

Tharom, T. 2001. Teknologi VoIP. Jakarta: Elex Media Komputindo.

Yuniati, Y.; Fitriawan, H., dan Patih, D F. J. 2014. Analisa Perancangan Server VOIP (Voice Internet Protocol) dengan Open Source Asterisk dan VPN (Virtual Private Network) sebagai Pengaman Jaringan Antar Client. Jurnal Sains, Teknologi dan Industri. Vol. 12 (1). Hlm. 112-121.