UNIVERSITAS INDONESIA

STUDI SIMULASI PERBANDINGAN KINERJA TUNNELING GRE DAN OPENVPN UNTUK TRAFIK-TRAFIK BERKELAS DAN TIDAK BERKELAS

SKRIPSI

FADRY SECONDARU
07 06 19 9306

FAKULTAS TEKNIK
PROGRAM STUDI TEKNIK ELEKTRO
DEPOK
DESEMBER, 2009

UNIVERSITAS INDONESIA

STUDI SIMULASI PERBANDINGAN KINERJA TUNNELING GRE DAN OPENVPN UNTUK TRAFIK-TRAFIK BERKELAS DAN TIDAK BERKELAS

TUGAS AKHIR

Diajukan sebagai salah satu syarat untuk memperoleh gelar Sarjana Teknik

FADRY SECONDARU
07 06 19 9306

FAKULTAS TEKNIK
PROGRAM STUDI TEKNIK ELEKTRO
DEPOK
DESEMBER, 2009

HALAMAN PERNYATAAN ORISINALITAS

Tugas Akhir ini adalah hasil karya saya sendiri,
Dan semua sumber baik yang dikutip maupun dirujuk
Telah saya nyatakan dengan benar.

Nama	**: Fadry Secondaru**
NPM	**: 0706199306**
Tanda Tangan	**:**
Tanggal	**: 15 Desember 2009**

HALAMAN PENGESAHAN

Tugas Akhir ini diajukan oleh:

Nama	: Fadry Secondaru
NPM	: 0706199306
Program Studi	: Strata 1 Ekstensi
Judul Tugas Akhir	: Studi Simulasi Perbandingan Kinerja Tunneling GRE Dan OpenVPN Untuk Trafik-Trafik Berkelas Dan Tidak Berkelas

Telah berhasil dipertahankan di hadapan Dewan Penguji dan diterima sebagai bagian pernyataan untuk memperoleh gelar Sarjana Teknik pada program studi strata 1 ekstensi, Fakultas Teknik, Universitas Indonesia.

DEWAN PENGUJI

Pembimbing	: Prof. Dr. Ir. Bagio Budiardjo M.Sc	(.....)
Penguji	: Prima Dewi Purnamasari ST, M.Sc	(.....)
Penguji	: Muhammad Salman ST., MIT	(.....)

Ditetapkan di : Universitas Indonesia, Depok

Tanggal : 29 Desember 2009

KATA PENGANTAR / UCAPAN TERIMA KASIH

Puji syukur saya panjatkan kepada Tuhan Yang Maha Esa, karena atas berkat dan rahmat-Nya, saya dapat menyelesaikan Tugas Akhir ini. Penulisan Tugas Akhir ini dilakukan dalam rangka memenuhi salah satu syarat untuk mencapai gelar Sarjana Teknik Jurusan Elektro pada Fakultas Teknik Universitas Indonesia.. Oleh karena itu, saya mengucapkan terima kasih kepada:

(1) Prof. Dr. Ir. Bagio Budiardjo M.Sc selaku dosen pembimbing yang telah menyediakan waktu, tenaga, ide dan pikiran untuk mengarahkan saya dalam penyusunan Tugas Akhir ini, Prima Dewi Purnamasari ST, M.Sc dan Muhammad Salman ST., MIT yang telah menyediakan waktu untuk menguji Tugas Akhir ini,

(2) Mas Irdan dan Mas Yono (P.T. Adiyasa Wicaksana) atas bimbingannya selama mengerjakan tugas akhir ini, serta Asrul teman dan rekan kerja yang juga merasakan pahit manisnya tugas akhir ini.

(3) Mas Irdan dan Mas Yono (P.T. Adiyasa Wicaksana) atas bimbingannya selama mengerjakan tugas akhir ini, Burhan dan Ardi (Ast Lab Jarkom), Mba Eka (Lab Cisco), Givano, Linda Widi, Hoya Tony, Mbak Heny, Fareza Nurmiati , Septice Jantika yang selalu mendo'akan dan memberikan motivasi

Akhir kata, saya berharap Tuhan Yang Maha Esa berkenan membalas segala kebaikan semua pihak yang telah membantu. Semoga Tugas Akhir ini membawa manfaat bagi pengembangan ilmu.

Depok, 30 Desember 2009

Penulis

HALAMAN PERNYATAAN PERSETUJUAN PUBLIKASI TUGAS AKHIR UNTUK KEPENTINGAN AKADEMIS

==========

Sebagai sivitas akademik Universitas Indonesia, saya yang bertanda tangan di bawah ini :

Nama	: Fadry Secondaru
NPM	: 0706199306
Program Studi	: S1 – Ekstensi
Departemen	: Teknik Elektro
Fakultas	: Teknik
Jenis karya	: Tugas Akhir

demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Universitas Indonesia **Hak Bebas Royalti Noneksklusif (*Non-exclusive Royalty - Free Right*)** atas karya ilmiah saya yang berjudul :

Studi Simulasi Perbandingan Kinerja Tunneling GRE Dan OpenVPN Untuk Trafik-Trafik Berkelas Dan Tidak Berkelas

beserta perangkat yang ada (jika diperlukan). Dengan Hak Bebas Royalti Noneksklusif ini Universitas Indonesia berhak menyimpan, mengalih media / formatkan, mengelola dalam bentuk pangkalan data (*database*), merawat, dan memublikasikan tugas akhir saya tanpa meminta izin dari saya selama tetap mencantumkan nama saya sebagai penulis / pencipta dan sebagai pemilik Hak Cipta.

Demikian pernyataan ini saya buat dengan sebenarnya.

Dibuat di : Depok
Pada tanggal : 30 Desember 2009
Yang menyatakan

(Fadry Secondaru)

ABSTRAK

Nama : Fadry Secondaru
Program Studi : S1 - Ekstensi
Judul : Desain dan Simulasi *Tunneling* Untuk Membandingkan Kinerja GRE, dan OPENVPN

Kebutuhan akan perluasan jaringan data/multimedia sekarang ini semakin tinggi. meningkatnya kebutuhan akan proses transfer data/multimedia dari jaringan satu ke jaringan lainnya melalui *internet* mendorong terpenuhinya kebutuhan pengguna akan efisiensi *bandwidth* dan tingkat keamanan proses transfer data yang akan dilakukan. Data yang melewati jaringan *internet* tidak terjamin keamanannya, oleh karenanya dibutuhkan sistem keamanan yang baik yang memungkinkan data yang dikirimkan tidak dapat di akses pengguna lain yang tidak berwenang. Tunneling memberikan solusi keamanan yang baik untuk permasalahan ini, dengan cara membentuk tunnel (terowongan) pada jaringan publik yang menghubungkan antara jaringan satu dengan jaringan yang lain. Tunneling ini tentunya juga akan menggunakan *byte* payload sebagai paket *header*, sehingga akan mengurangi kecepatan transfer data. Pada penelitian kali ini akan dianalisa kinerja tunneling protocol GRE, dan OPENVPN dari sisi proses enkapsulasi, besar *throughput*, dan perilaku protokol yang dilewatkan didalamnya dengan melakukan simulasi transfer data FTP (File Transfer Protocol), RDP (Remote Desktop Protocol) dan Video Streaming yang dilewatkan melalui IP Tunnel. Dari percobaan yang dilakukan didapatkan bahwa protokol openVPN pada network tanpa kelas, memiliki kinerja lebih baik dibandingkan dengan tunel GRE dalam hal efisiensi bandwidth hal ini dikarenakan openVPN melakukan kompresi pada paketnya. Penerapan *class of service* pada tunnel GRE akan mengoptimalkan penggunaan *bandwidth* menjadi 100%, dari yang sebelumnya tanpa *class of service* tunnel GRE hanya menggunakan 95% dari total *bandwidth* yang tersedia.

Kata kunci:
Tunneling, *Bandwidth*, GRE, OPENVPN, FTP, RDP, Video Streaming

ABSRACT

Name : Fadry Secondaru
Study Program: S1 - Ekstensi
Title : Design and Simulation of Tunneling to Compare Performance of GRE, OPENVPN

The needs of data/multimedia *network* expansion nowadays are getting higher. The increase of data/multimedia transfer needs from one *network* to the other through the *Internet* encouraged the users' *bandwidth* efficiency fulfillment and data transfer process security level that is about to be conducted. Data, which passes through the *Internet network*s are not securely guaranteed, therefore, a thorough security system is needed to enables the sent data could not be accessed by another unauthorized parties. Tunneling provides a good security solution for this problem by forming a tunnel on public *network*, which connects one *network* with another. This tunneling would of course use *byte* payload as a *header* package, in order to decrease data transfer speed. On this research, tunneling protocol GRE, and OPENVPN performance would be analyzed from *encapsulation* process, throughtput and protocol characteristic which pass through inside tunnel by conducting FTP (File Transfer Protocol), RDP (Remote Desktop Protocol) data transfer simulation and Video Streaming passed through IP tunnel. From the simulation show that openVPN protocol in a network without class of service has better performance than GRE tunnel this because openVPN doing packet compression before sending the packet over tunnel. Implementation class of service in GRE tunnel would optimizing bandwidth consumption become 100% , whereas without class of service GRE tunnel using 95% from available bandwidth

Key Words:
Tunneling, *Bandwidth*, GRE, OPENVPN, FTP, RDP, *Video Streaming*

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I
PENDAHULUAN

1.1. Latar Belakang

*Tunnel*ing merupakan metode untuk transfer data dari satu jaringan ke jaringan lain dengan memanfaatkan jaringan *internet* secara terselubung. Disebut *tunnel* atau saluran karena aplikasi yang memanfaatkannya hanya melihat dua *end point* atau ujung, sehingga paket yang lewat pada *tunnel* hanya akan melakukan satu kali lompatan atau *hop*. Data yang akan ditransfer dapat berupa *frame* (atau paket) dari protokol yang lain.

Protokol *tunnel*ing tidak mengirimkan *frame* sebagaimana yang dihasilkan oleh *node* asalnya begitu saja melainkan membungkusnya (meng-enkapsulasi) dalam *header* tambahan. *Header* tambahan tersebut berisi informasi routing sehingga data (*frame*) yang dikirim dapat melewati jaringan *internet*. Jalur yang dilewati data dalam *internet* disebut *tunnel*. Saat data tiba pada jaringan tujuan, proses yang terjadi selanjutnya adalah dekapsulasi, kemudian data original akan dikirim ke penerima terakhir. *Tunnel*ing mencakup keseluruhan proses mulai dari enkapsulasi, transmisi dan dekapsulasi.

1.2 Tujuan Penulisan

Tujuan penulisan tugas akhir ini adalah untuk mengetahui dan memahami konsep *tunnel*ing, protokol-protokol seperti GRE, OpenVPN.Dari sisi enkapsulasi paket, *throughput*, dan perilaku protokol yang dilewatkan didalamnya.

1.3 Batasan Masalah

Masalah yang dibahas dalam tugas akhir ini adalah analisis kinerja GRE, OpenVPN dalam hal efisiensi *bandwidth* yang akan dilakukan dengan melakukan simulasi (*event simulation)* yang menggunakan FTP (*File Transfer Protocol*), RDP (Remote Desktop Protocol) dan *Video streaming* sebagai paket data yang dilewatkan dalam IP *Tunnel*

1.4 Sistematika Penelitian

Sistematika penelitian pada tugas akhir ini adalah :

Bab 1 Pendahuluan

Bagian pendahuluan terdiri atas latar belakang, tujuan penulisan, batasan masalah, dan sistematika penelitian.

Bab 2 Dasar Teori

Bagian ini akan membahas teori dasar yang digunakan pada penelitian yaitu mengenai teknologi *tunneling* dan protokol-protokol *tunneling*.

Bab 3 Perancangan

Bagian awal dari bab ini membahas mengenai perlengkapan dan parameter-parameter yang dibutuhkan dalam perancangan untuk membangun sebuah sistem *tunneling*.

Bab 4 Analisa Data

Bagian ini berisi tentang data-data hasil simulasi dan analisisnya. Hasil analisis merupakan dasar pembentukan kesimpulan pada penelitian ini.

Bab 5 Kesimpulan

Bab ini berisi kesimpulan yang diperoleh dari keseluruhan kegiatan penelitian yang telah dilakukan

BAB II
LANDASAN TEORI

2.1. GRE (*Generic Routing Encapsulation*)

Generic Routing Encapsulation adalah *tunneling protocol* yang di dikembangkan oleh *cisco*, protocol ini dapat melakukan encapsulasi berbagai macam jenis paket dalam lapisan *network protocol* dalam *tunnel*nya,dengan cara membuat *virtual* komunikasi *point to point* dari *router* asal ke *router* tujuan dengan menggunakan IP pada komunikasi *internetwork*. .[1]

Generic Routing Encapsulation (GRE) Protokol *tunnel*ing yang satu ini memiliki kemampuan membawa lebih dari satu jenis protokol pengalamatan komunikasi. Bukan hanya paket beralamat IP saja yang dapat dibawanya, melainkan banyak paket protokol lain seperti CNLP, IPX, dan banyak lagi. Namun, semua itu dibungkus atau dienkapsulasi menjadi sebuah paket yang bersistem pengalamatan IP. Kemudian paket tersebut didistribusikan melalui sistem *tunnel* yang juga bekerja di atas protokol komunikasi IP. Dengan menggunakan *tunnel*ing GRE, *router* yang ada pada ujung-ujung *tunnel* melakukan enkapsulasi paket-paket protokol lain di dalam *header* dari protokol IP. Hal ini akan membuat paket-paket tadi dapat dibawa ke manapun dengan cara dan metode yang terdapat pada teknologi IP. Dengan adanya kemampuan ini, maka protokol-protokol yang dibawa oleh paket IP tersebut dapat lebih bebas bergerak ke manapun lokasi yang dituju, asalkan terjangkau secara pengalamatan IP.

Aplikasi yang cukup banyak menggunakan bantuan protokol *tunnel*ing ini adalah menggabungkan jaringan-jaringan lokal yang terpisah secara jarak kembali dapat berkomunikasi. Atau dengan kata lain, GRE banyak digunakan untuk memperpanjang dan mengekspansi jaringan lokal yang dimiliki si penggunanya. Meski cukup banyak digunakan, GRE juga tidak menyediakan sistem enkripsi data yang lalu-lalang di *tunnel*-nya, sehingga semua aktivitas datanya dapat dimonitor menggunakan protokol *analyzer* biasa saja. .[2]

2.2. OpenVPN

OpenVPN adalah sebuah aplikasi VPN gratis dan *open source* yang digunakan untuk membuat koneksi *point to point* atau *point to mulipoint* antar komputer.OpenVPN dapat melakukan koneksi langsung antar *network* sekalipun *network* tersebut berada dibelakang sebuah NAT atau firewall.Aplikasi OpenVPN ini di tulis oleh James Yonan dan di *publish* dibawah naungan GNU General Public License (GPL). .[3]

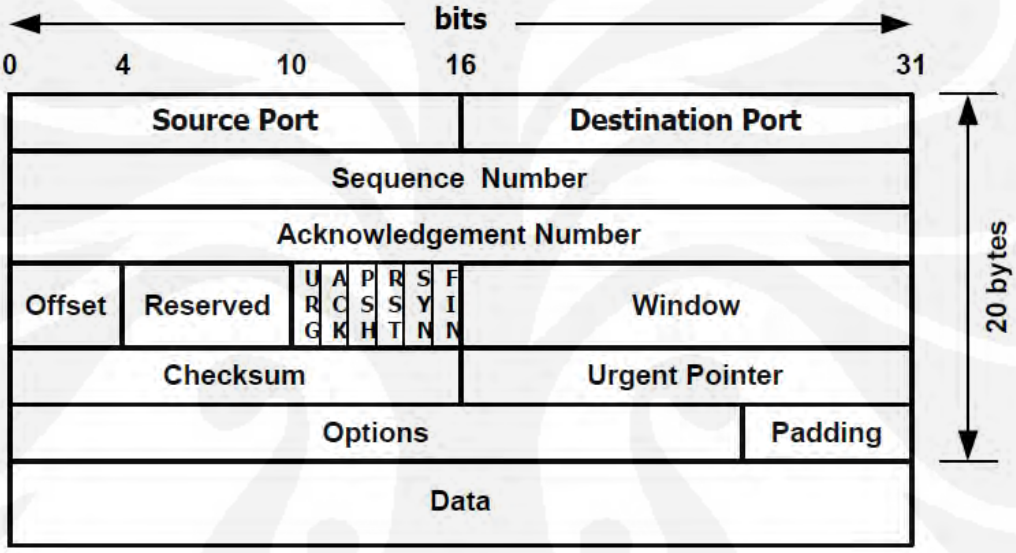
OpenVPN mengijinkan *remote peer* untuk melakukan autentikasi satu sama lain dengan menggunakan *pre-shared secret key, certificates*, atau username/password.Ketika OpenVPN digunakan dalam konfigurasi *multiclient-server OpenVPN* akan mengeluarkan sertifikat autentikasi untuk setiap *client* dengan menggunakan signature dan Certificate authority.OpenVPN menggunakan OpenSSL untuk mengenkripsi datanya sebaik protocol SSLv3/TLSv1.OpenVPN tersedia untuk Solaris, Linux, OpenBSD, FreeBSD, NetBSD, Mac OS X, dan windows 2000/XP/Vista.OpenVPN membawa banyak fitur kontrol dan sekuriti.OpenVPN bukan "*web-base*" VPN, dan juga tidak kompatibel dengan IPSec atau paket VPN yang lain. .[4]

2.3. TCP (*Transport Control Protocol*)

TCP (*Transport Control Protocol*) merupakan protokol yang berada pada *layer transport* dari *layer* TCP/IP. TCP adalah protokol yang bersifat *byte* stream, *connection-oriented* dan reliable dalam pengiriman data. TCP menggunakan komunikasi *byte-stream*, yang berarti bahwa data dinyatakan sebagai suatu urutan-urutan *byte*. *Connection-oriented* berarti sebelum terjadi proses pertukaran data antar komputer terlebih dahulu harus dibentuk suatu hubungan. Hal ini dapat dianalogikan dengan proses pendialan nomor telepon dan akhirnya terbentuk suatu hubungan.

Keandalan TCP dalam megirimkan data didukung oleh mekanisme yang disebut *Positive Acknowledgement with Re-transmission* (PAR).[5] Data yang dikirim dari *layer* aplikasi akan dipecah-pecah dalam bagian-bagian yang lebih kecil dan diberi nomor urut *(sequence number)* sebelum dikirimkan ke *layer*

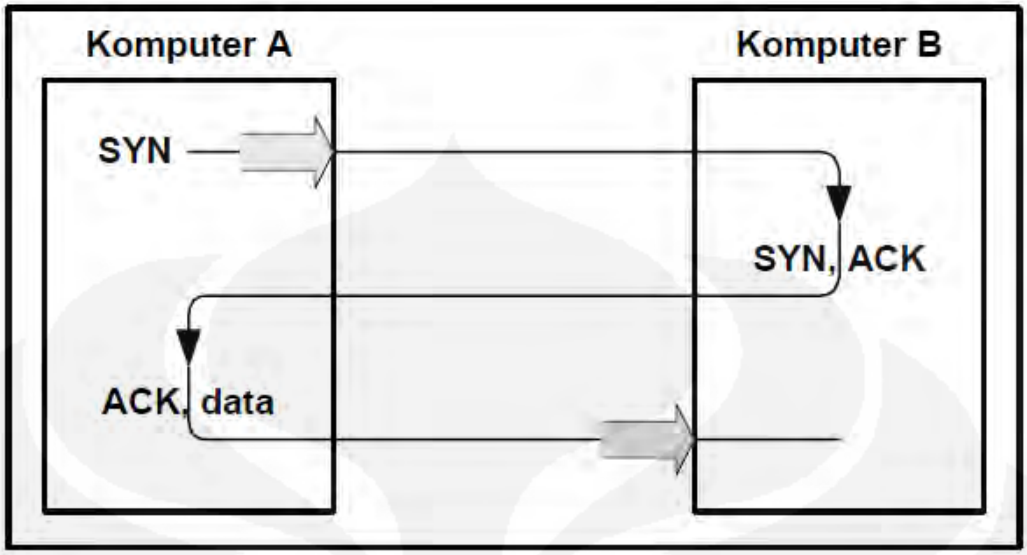
berikutnya. Unit data yang sudah dipecah-pecah tadi disebut *segmen* (*segmen*t). TCP selalu meminta konfirmasi setiap kali selesai mengirimkan data, apakah data tersebut sampai pada komputer tujuan dan tidak rusak. Jika data berhasil sampai mencapai tujuan, TCP akan mengirimkan data urutan berikutnya. Jika tidak berhasil, maka TCP akan melakukan pengiriman ulang urutan data yang hilang atau rusak tersebut. Dalam kenyataannya TCP menggunakan sebuah *acknowledgement* (ACK) sebagai suatu pemberitahuan antara komputer pengirim dan penerima. *Format segmen* TCP diperlihatkan pada gambar 2.1.



Gambar 2.1 : *Format* TCP

Data yang diterima pada sisi penerima akan disusun berdasarkan nomor urut yang diberikan oleh sisi pengirim. Untuk mengatasi kerusakan data yang diterima, TCP menggunakan sebuah *checksum* untuk memastikan bahwa data tersebut tidak rusak.

Model komunikasi dua arah antara komputer sisi kirim dan sisi terima sebelum terjadi proses pengiriman data disebut *handshake*. Tipe *handshake* yang digunakan TCP adalah threeway handshake, karena menggunakan tiga *segmen*. Tujuan *three-way handshake* ini adalah untuk pembentukan koneksi, sinkronisasi *segmen*, dan pemberitahuan besar data yang bisa diterima pada suatu saat antara sisi kirim dan sisi terima. Proses sederhana *three-way handshake* tersebut dapat ditunjukkan pada gambar 2.2.

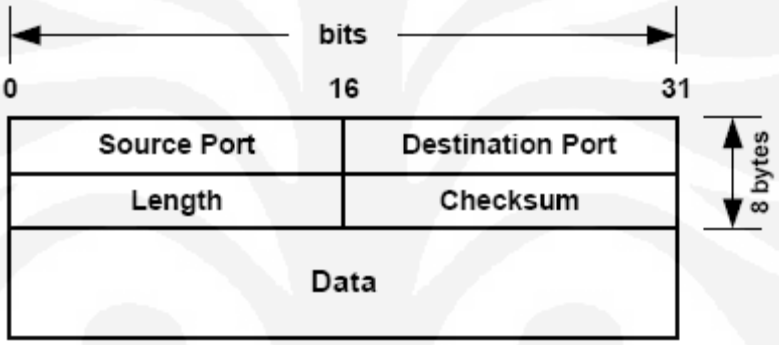


Gambar 2.2 : *three-way handshake*

Komputer A memulai hubungan dengan mengirimkan *segmen* sinkronisasi nomor urut (SYN) pada komputer B. *Segmen* tersebut merupakan pemberitahuan pada komputer B bahwa komputer A ingin melakukan sebuah hubungan dan menanyakan nomor urut berapa yang akan digunakan sebagai awal urutan *segmen* yang akan dikirim. (Nomor urut tersebut digunakan agar data tetap berada pada urutan yang benar). Komputer B memberikan respon pada komputer A dengan sebuah *segmen* yang memberikan ACK dan SYN. Dengan demikian komputer A akan tahu informasi nomor urut yang digunakan untuk komputer B. Akhirnya, komputer A pun mengirimkan sebuah *segmen* sebagai balasan dari *segmen* yang dikirim komputer B, sekaligus melakukan pengiriman data yang sebenarnya pertama kali. Setelah terjadi proses tersebut komputer A mendapati bahwa komputer B siap menerima data dan segera setelah hubungan dipastikan dapat terjadi data pun dikirim sepenuhnya ke komputer B. Pada saat seluruh data telah selesai dikirim, proses *three-way handshake* untuk mengakhiri hubungan pun terjadi untuk memastikan bahwa tidak ada lagi data yang dikirim.

2.4. UDP (User Datagram Protocol)

UDP (*User Datagram Protocol*) merupakan protokol yang juga berada pada *layer transport* selain TCP. Protokol ini bersifat *connectionless* dan *unreliable* dalam pengiriman data.*Connectionless* berarti tidak diperlukannya suatu bentuk hubungan terlebih dahulu untuk mengirimkan data. *Unreliable* berarti pada protokol ini data tidak dijamin akan sampai pada tujuan yang benar dan dalam kondisi yang benar pula. Keandalan pengiriman data pada protocol ini menjadi tanggung jawab dari program aplikasi pada *layer* di atasnya. Gambar 2.3.. menunjukkan *format header* UDP.



Gambar 2.3 : *format header* UDP

Jika dibandingkan dengan TCP, UDP adalah protokol yang lebih sederhana dikarenakan proses yang ada di dalamnya lebih sedikit. Dengan demikian aplikasi yang memanfaatkan UDP sebagai protokol *transport* dapat mengirimkan data tanpa melalui proses pembentukan koneksi terlebih dulu. Hal ini pun terjadi pada saat mengakhiri suatu koneksi, sehingga dalam banyak hal proses yang terjadi sangatlah sederhana dibanding jika mengirimkan data melalui protokol TCP. Secara teknis protokol UDP memiliki *header* yang lebih kecil dibanding protokol TCP seperti terlihat pada *format header* masing-masing.[5]

Bila suatu program aplikasi memanfaatkan protokol UDP untuk mengirimkan informasi,protokol UDP melakukan fungsi *multiplexing /demultiplexing* seperti yang dilakukan protokol TCP dengan menentukan nomor *port* pengirim (*source*

port) dan nomor *port* penerima (*destination port*), kemudian menambahkan sedikit fungsi koreksi kesalahan lalu meneruskan *segmen* yang
terbentuk ke protokol *layer Internet*. Pada *layer internet segmen* tersebut ditambahi informasi dalam bentuk datagram IP dan kemudian ditentukan cara terbaik untuk mengantarkan *segmen* tersebut ke sisi penerima. Jika *segmen* tersebut tiba pada sisi penerima, protokol UDP menggunakan nomor *port* informasi IP pengirim dan penerima untuk mengantarkan data dalam *segmen* ke proses *program* aplikasi yang sesuai.

Beberapa hal yang harus diperhatikan jika suatu program aplikasi akan menggunakan protokol UDP sebagai protokol tans*port*:

- Tidak ada pembentukan koneksi. Protokol UDP hanya mengirim informasi begitu saja tanpa melakukan proses awal sebelumnya.
- Tidak ada pengkondisian koneksi. Protokol UDP tidak melakukan penentuan kondisi koneksi yang berupa parameter-parameter seperti *buffer* kirim dan terima, kontol kemacetan, nomor urutan *segmen* dan *acknowledgement.*
- Memiliki *header* yang kecil. Protokol UDP memiliki 8 *byte header* dibanding 20 *byte header* pada TCP.
- Tidak ada pengaturan laju pengiriman data. Protokol UDP hanya menekankan kecepatan kirim pada laju program aplikasi dalam menghasilkan data, kemampuan sumber kirim data (berdasarkan CPU, laju pewaktuan, dll) dan *bandwidth* akses menuju *Internet*. Jika terjadi kemacetan jaringan sisi penerima tidak perlu menerima seluruh data yang dikirim.Dengan demikian laju penerimaan data dibatasi oleh faktor kemacetan jaringan yang terjadi walaupun pada sisi kirim tidak memperhatikannya.

Protokol UDP lebih sering diimplementasikan untuk aplikasi-aplikasi yang mengarah proses *realtime* seperti aplikasi *multimedia*, dimana rugi-rugi paket data yang kecil lebih ditoleransi daripada nilai *delay* yang terjadi

2.5. RDP (*Remote Desktop Protocol*)

Remote Desktop Protocol (sering disingkat menjadi RDP) adalah sebuah protokol jaringan yang digunakan oleh *Microsoft Windows Terminal Services* dan *Remote Desktop*. RDP dibuat berdasarkan protokol T.120 yang spesifikasinya diumumkan oleh *International Telecommunication Union* (ITU), yang juga merupakan protokol yang digunakan di dalam perangkat lunak konferensi jarak jauh *Microsoft NetMeeting*.[6]

Klien-klien yang mendukungnya bervariasi, mulai dari sebagian besar sistem operasi *Windows* 32-bit (termasuk *Windows CE* dan *PocketPC*), hingga sistem operasi lainnya, seperti *Linux, FreeBSD, UNIX Solaris*, dan *Apple Mac OS X*. Secara *default*, *server* yang membuka protokol ini, akan membuka *port* **TCP 3389**.

Protocol ini memiliki layanan seperti dukungan terhadap kedalaman warna 32-bit, enkripsi 128-bit (dengan menggunakan algoritma enkripsi RC4), dukungan terhadap protokol *Transport Layer Security* (TLS), redireksi suara (suara yang sebenarnya keluar di *server* bisa didengarkan pada klien local), redireksi sistem berkas, (sistem berkas lokal yang menyimpan berkas-berkas pengguna dapat digunakan di dalam sebuah sesi terminal *server*), redireksi *port* (para pengguna dapat mengakses *port* serial dan paralel lokal secara langsung).[6]

BAB III
PERANCANGAN *NETWORK*

Pada bab ini dibahas mengenai perancangan sistem *network* melalui *tunnel*, baik itu menggunakan *tunnel* GRE dan, OpenVPN.

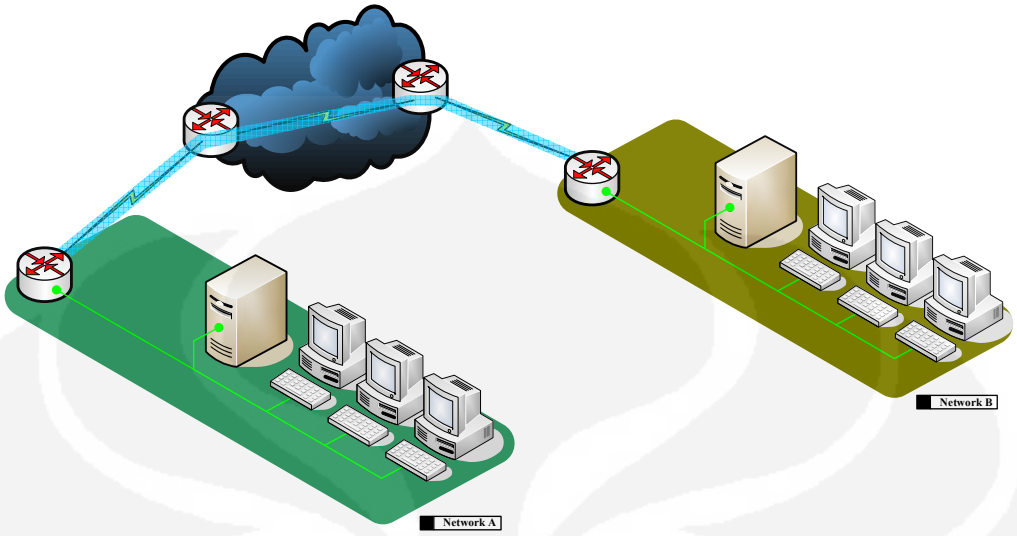
Dalam perancangan *network* ini tiap-tiap *network* terbagi menjadi tiga bagian, yaitu *network* A, *network* B dan *Network* ISP. *Network* A bertindak sebagai *network client* yang akan mengakses *network* B yang merupakan *server farm network* yang terdiri dari FTP *server* dan *streaming server* melalui *network* ISP yang bertindak sebagai penyedia layanan *internet*.

3.1 Perancangan *Network* Secara Global

3.1.1. GRE *Tunnel*

Secara umum *tunneling* GRE terdiri dari dua buah atau lebih *router* yang kesemua *router* tersebut terhubung pada *cloud provider* baik itu *cloud internet,* MPLS, OSPF, *Frame Relay*, dan lain lain.Ada beberapa pertimbangan sebuah *corporate* memilih GRE sebagai *tunnel.*

1. Kebutuhan akses dua *network* yang terpisah dalam hal ini LAN(*Local Area Connection)* yang dibatasi oleh NAT (*Network Address Translator)*
2. Data yang akan dilalui pada *tunnel* tidak memerlukan enkripsi data seperti IPSec atau SSL
3. Data yang akan dilewatkan merupakan data yang berbasiskan IP



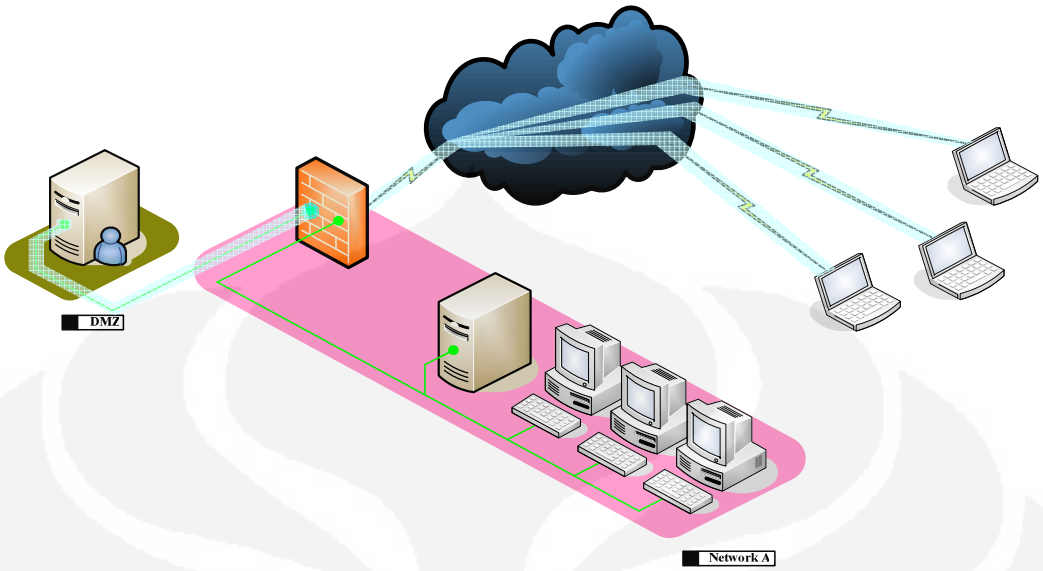
Gambar 3.1 GRE *Network Topology*

3.1.2. OpenVPN

OpenVPN adalah aplikasi open source untuk Virtual Private *Network* (VPN), di mana aplikasi tersebut dapat membuat koneksi point-to-point tunnel yang telah terenkripsi.

OpenVPN biasanya digunakan oleh *mobile user* sebuah *corporate* yang ingin mengakses data perusahaan dengan aman melalui *cloud internet*

Secara umum *Network* OpenVPN terdiri dari 1 buah OpenVPN *server* yang terhubung pada *cloud provider* yang biasanya merupakan *cloud internet* dan satu atau lebih *mobile user* baik itu user menggunakan laptop ataupun *smartphone* yang didalam masing masing *device* user tersebut telah teristal aplikasi OpenVPN *client*.



Gambar 3.2 OpenVPN *Network Topology*

3.2. Simulasi Perancangan *Network*

Secara umum model jaringan yang akan disimulasikan terbagi atas tiga buah model, yang masing masing terdiri dari topologi yang berbeda.Masing-masing topologi terdiri atas 3 buah *network* yakni *network* A, *network* B, dan *network* ISP.*Network* A merupakan *network client* yang akan mengakses *network* B yang berisikian *server farm* melalui *network* ISP yang merupakan penyedia jasa layanan *internet*.

Peralatan yang digunakan dalam perancangan ini terdiri dari perangkat keras dan perangkat lunak.Adapun daftar peralatan yang digunakan adalah :

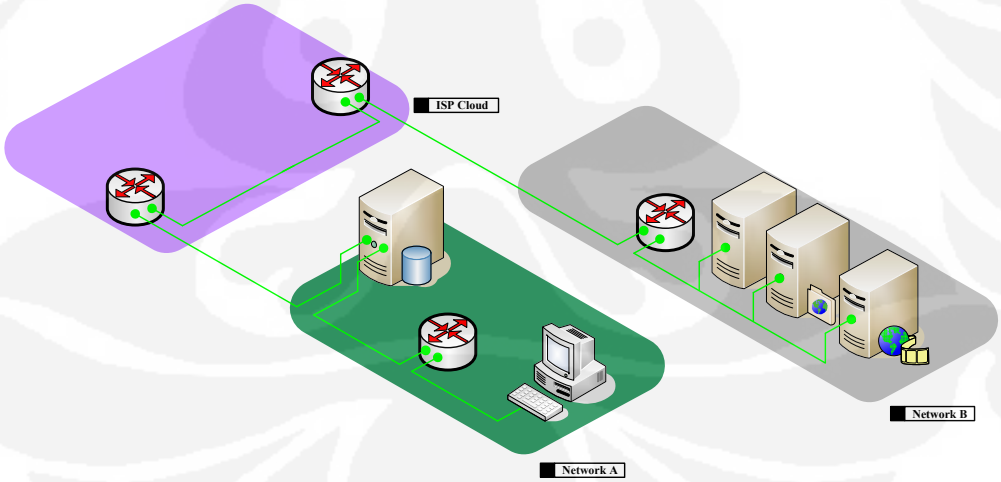
- **Perangkat Keras**
  1. Cisco *router* 2801 ; 4 buah yang digunakan untuk :
     - 1 buah sebagai *router* ISP A
     - 1 buah sebagai *router* ISP B
     - 1 buah sebagai *router network* A
     - 1 buah sebagai *router network* B
  2. Personal Computer ; 4 buah
     - 1 buah sebagai OpenVPN *server*
     - 1 buah sebagai FTP *server*

- 1 buah sebagai RDP *server*
- 1 buah sebagai PC *client*

- **Perangkat Lunak**
  1. Cisco IOS c2801-advipservicesk9-mz.124-11.T
  2. FileZilla *Server*-0.9.31
  3. FileZilla *Client* 3.2.4.1
  4. VLC *Streaming server*
  5. VLC *Streaming client*
  6. wireshark *network protocol analyzer* v.1.0.8
  7. *Windows XP Professional Edition Service Pack* 2
  8. OpenVPN *server*
  9. OpenVPN *client*

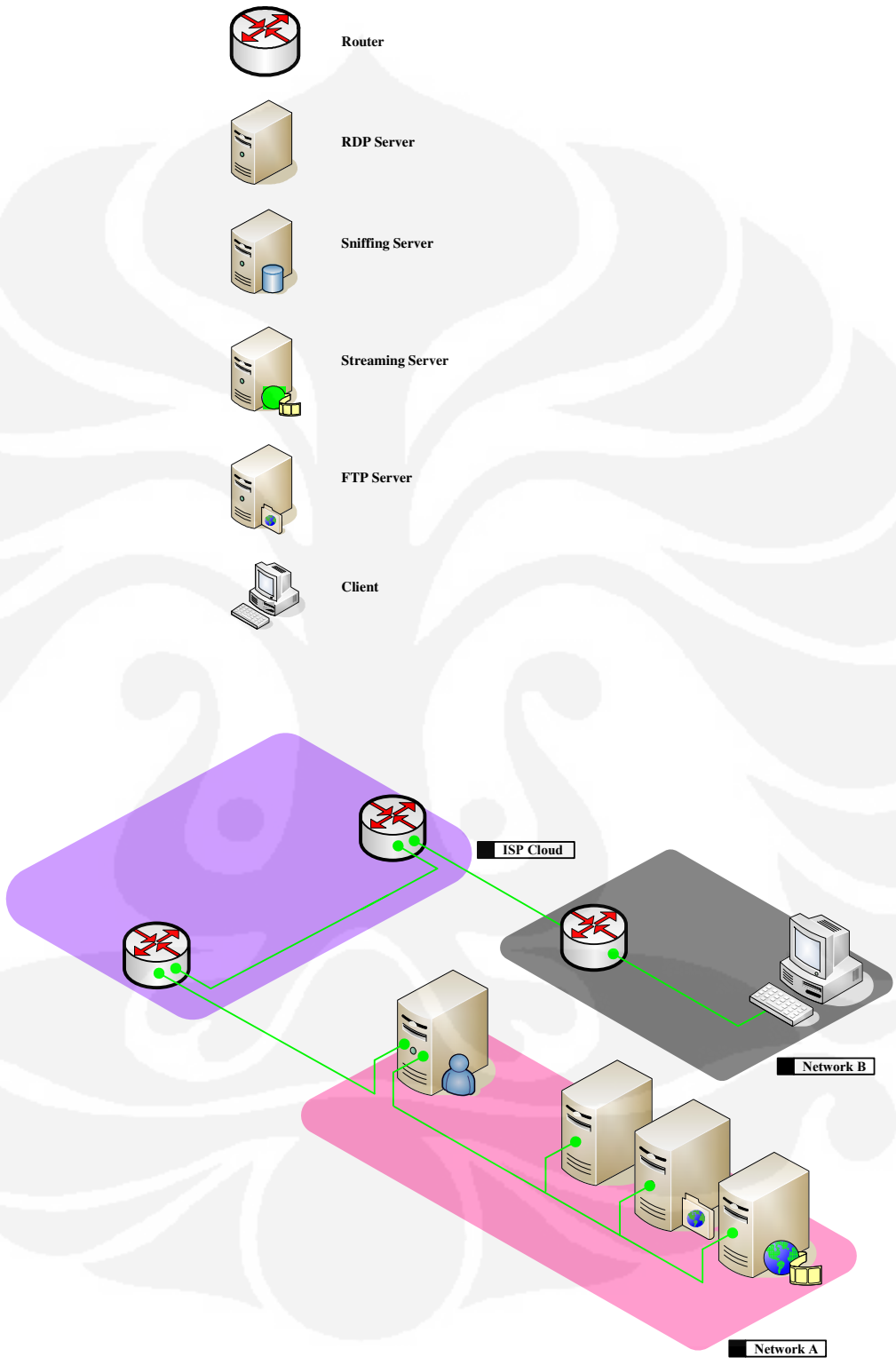
3.3. *Setup* Topologi

Untuk penelitian ini pertama-tama yaitu membuat perancangan topologi *network* GRE dan OpenVPN.Adapun topologi yang dibuat adalah sebagai berikut :



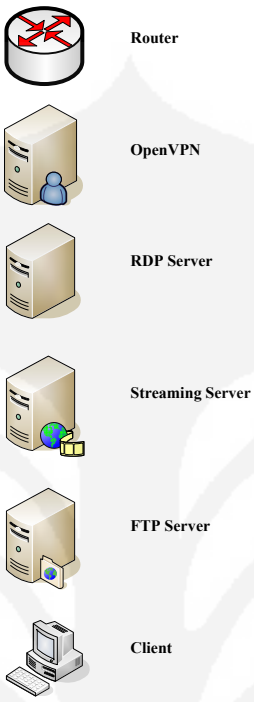
Gambar 3.3. perancangan topologi GRE

Keterangan :



Gambar 3.4. topologi OpenVPN

Keterangan :



- Topologi terdiri dari 3 *Network*: *Network* A (*Server*). *Network* B (*Client*) dan *Network* ISP
- *Network* B (*Client*) akan mengakses *Network* A (*Server*) melalui *Network* ISP dengan menggunakan GRE tunnel.
- *Bandwidth* pada *network* ISP di set sebesar 1 Mbps (ISP A dan ISP B).
- *IP address* yang digunakan adalah *IP address* class C yaitu 192.168.1.0/24 yang disubnet menjadi 32 subnet.

Tabel 3.1. Tabel *IP address*

Device	*IP address*	*Interface*	*Connected To*
FTP server	*172.16.1.123*	*FastEthernet 0*	*Network A*
RDP server	*172.16.1.25*	*FastEthernet 0*	*Network A*
Stream server	*172.16.1.103*	*FastEthernet 0*	*Network A*
Router Network A	*172.16.1.60*	*FastEthernet 0/1*	*Network A*
Router Network A	*192.168.1.9*	*FastEthernet 0/0*	*Network A to ISP A*
Router Network A	*192.168.100.1*	*Tunnel 0*	*Network A to Nwteork B*

OpenVPN server	*172.16.1.60*	*FastEthernet 0*	*Network A*
OpenVPN server	*192.168.1.9*	*FastEthernet 1*	*Network A to ISP A*
Router ISP A	*192.168.1.10*	*FastEthernet 0/1*	*ISP A to Network A*
Router ISP A	*192.168.1.17*	*FastEthernet 0/0*	*ISP A to ISP B*
Router ISP B	*192.168.1.18*	*FastEthernet 0/1*	*ISP B to ISP A*
Router ISP B	*192.168.1.25*	*FastEthernet 0/0*	*ISP B to Sniffing server*
Sniffing server	*192.168.1.26*	*FastEthernet 0*	*Sniffing server to ISP B*
Sniffing server	*192.168.1.33*	*FastEthernet 1*	*Sniffing server to Network B*
Router Network B	*192.168.1.34*	*FastEthernet 0/1*	*Network B to sniffing server*
Router Network B	*192.168.1.41*	*FastEthernet 0/0*	*Network B*
Router Network B	*192.168.100.2*	*Tunnel 0*	*Network B to Network A*
PC Client	*192.168.1.42*	*FastEthernet 0*	*Network B*

3.4.Konfigurasi *Network*

Setelah menentukan topologi *network* tahapan selanjutnya adalah melakukan konfigurasi network. Konfigurasi *network* yang dilakukan adalah konfigurasi *network* tanpa *class of service*. Adapun konfigurasi yang dilakukan pada *network* secara rinci dapat dilihat pada lampiran 1 halaman 33

3.5. Pengujian *Network*

Setelah merancang sesuai dengan topologi,, pengujian *network* dilakukan dengan langkah-langkah sebagai berikut:

1. Pada simulasi ini data yang akan dilewatkan adalah aplikasi-aplikasi yang bekerja dengan protocol TCP (*Transmission Control Protocol*), UDP (*User Datagram Protocol*) dan RDP (*Remote Desktop Protocol*) .
2. Simulasi transfer data TCP dengan aplikasi FTP (*File Transfer Protocol*) dari *network* A ke *network* B menggunakan *software FileZilla Server*/*Client*.
3. Simulasi transfer data UDP dengan aplikasi *video streaming* dari *network* A ke *network* B menggunakan *software* VLC Streaming *Server*/*Client*.

4. Simulasi transfer data UDP (*Video Streaming*) dan TCP (FTP) bersamaan dengan aplikasi *Remote Desktop* (RDP) dari *network* A ke *network* B.
5. Simulasi transfer data UDP (*Video Streaming*) dan TCP (FTP) bersamaan dengan aplikasi *Remote Desktop* (RDP) dari *network* A ke *network* B dengan *QoS*

3.6. Pengambilan Data

Pengambilan data dilakukan dengan metode *sniffing* menggunakan *software* wireshark. *Sniffing* data dilakukan di Ethernet *Router Network* B untuk GRE tunnel dan *Interface* TAP pada OpenVPN interface pada saat :

1. PC *client* melakukan transfer FTP dari *Network* A
2. PC *client* melakukan streaming video dari *Network* A
3. PC *client* melakukan transfer FTP dan *streaming video* dari *Network* A bersamaan dengan aplikasi *Remote Desktop* dari *client* ke *server*.
4. PC *client* melakukan transfer FTP dan *streaming video* dari *Network* A bersamaan dengan aplikasi *Remote Desktop* dari *client* ke *server* dengan diberlakukannya *Class of Service* pada masing-masing trafik (dengan *QoS*)

BAB IV
ANALISA DATA

4.1. Struktur paket GRE dan OpenVPN

Pada teorinya ukuran *frame* pada *Ethernet* adalah 1518 *byte*, yang terdiri dari :

- 6 *byte* dest addr
- 6 *byte* src addr
- [4 *byte optional* 802.1q VLAN Tag]
- 2 *byte* length/type
- 46-1500 *byte* data (payload)
- 4 *byte* CRC

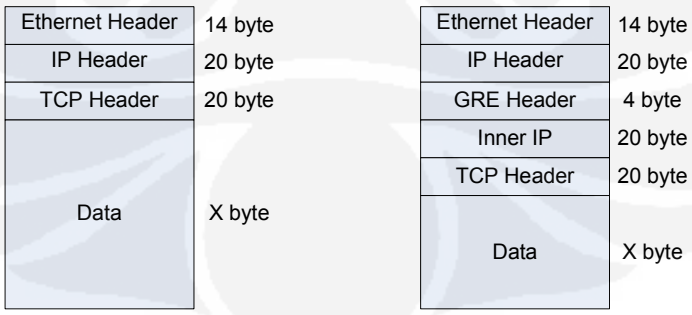
Sehingga jika dilakukan perhitungan manual pada sebuah paket yang berukuran 10 MB atau 10000000 *byte*, maka paket tersebut dapat dibagi menjadi 6667 paket yang dihitung denga menggunakan rumus :

Paket = ukuran file / payload maksimum

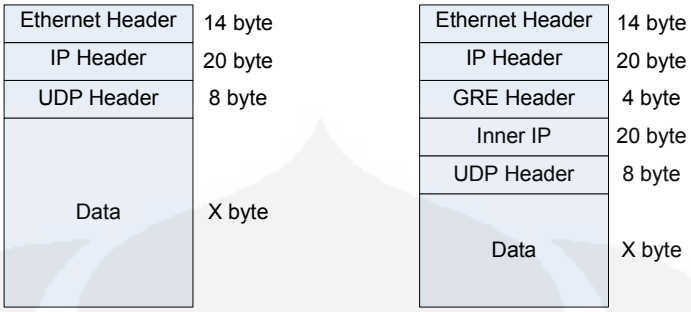
Adapun penjabaran dari perhitungan tersebut adalah

Payload data = 1500 - 20

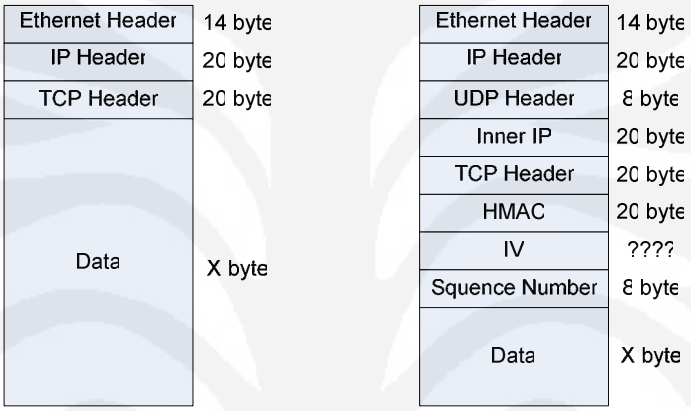
Dimana 20 *byte* terebut merupakan 20 *byte* IP *header*.Dengan menambahkan tunnel GRE ataupun OpenVPN berarti kita mengurangi *payload* paket pada data yang berefek mengurangi jumlah data yang dikirim.Dengan berkurangnya jumlah paket yang dapat dikirim dalam satuan waktu maka secara langsung akan memperlambat waktu pengiriman data.



Gambar 4.1. perbandingan paket TCP pada *Ethernet* tanpa GRE dan dengan GRE



Gambar 4.2. perbandingan paket UDP pada Ethernet tanpa GRE dan dengan GRE

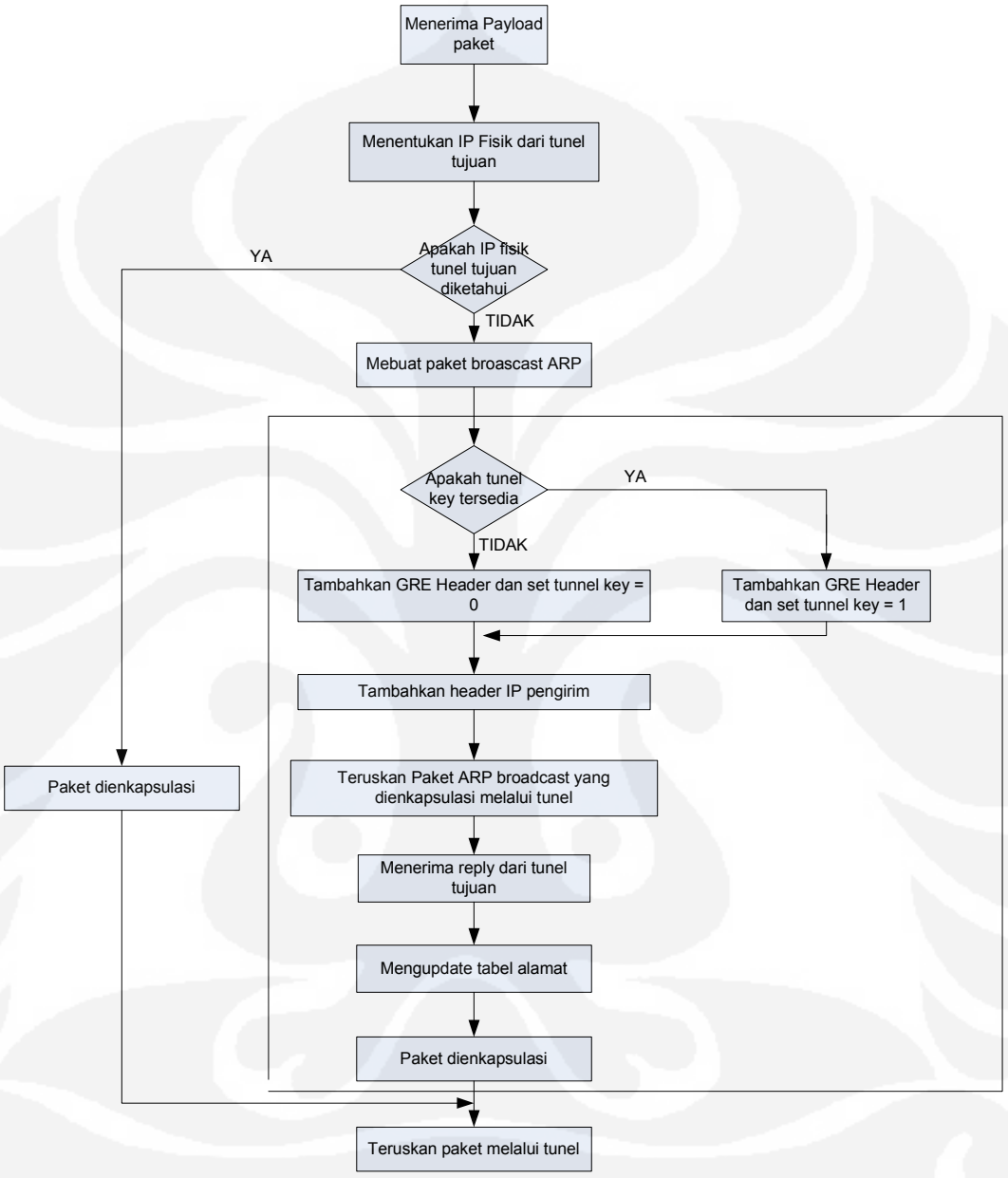


Gambar 4.3. perbandingan paket TCP pada Ethernet dengan Tanpa OpenVPN dan dengan OpenVPN

4.2. Enkapsulasi pada Tunneling Protocol

Pengambilan data dilakukkan pada saat *client* melakukkan transfer data dari *network* A (*server*) ke *network* B (*client*).Yang akan dianalisa dari pengambilan data ini adalah bagaimana cara GRE dan OpenVPN menenkapsulasi data didalam tunnel.

4.2.1. Enkapsulasi data pada GRE



Gambar 4.4. flowchart proses pembentukan tunnel GRE

Proses pembentukan tunnel GRE pertama-tama adalah :

1. Interface tunnel pada *router* dalam status down

*Network*_A# show ip interface brief

Interface	IP-Address	OK?	Method	Status	Protocol
FastEthernet0/0	192.168.1.9	YES	NVRAM	up	up
FastEthernet0/1	172.16.1.60	YES	NVRAM	up	up
Serial0/0	unassigned	YES	unset	administratively down	down
Serial0/1	unassigned	YES	unset	administratively down	down
Tunnel0	**192.168.100.1**	**YES**	**manual**	**up**	**down**

Gambar 4.5. Status *interface router down*

2. Salah satu *router* akan mengirimkan paket GRE sebesar 4 *byte* ke *router destination*,
3. Jika *router destination* up maka *router destination* akan meng-upkan tunnelnya sehingga proses pertukaran data dapat dilakukan.

Router# show ip interface brief

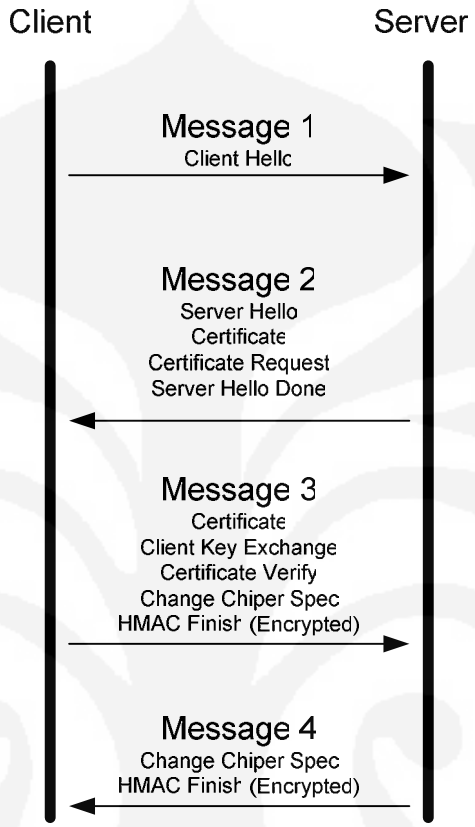
Interface	IP-Address	OK?	Method	Status	Protocol
FastEthernet0/0	192.168.1.26	YES	NVRAM	up	up
FastEthernet0/1	192.168.1.33	YES	NVRAM	up	up
Serial0/0	unassigned	YES	unset	administratively down	down
Serial0/1	unassigned	YES	unset	administratively down	down
Tunnel0	**192.168.100.1**	**YES**	**manual**	**up**	**up**

Gambar 4.6. Status *interface router up*

GRE (*Generic Routing Encapsulation*) melakukan enkapsulasi *frame* IP yang berisi paket PPP menjadi paket GRE, kemudian paket GRE tersebut dibungkus dalam sebuah paket IP untuk dilewatkan dalam tunnel.

4.2.2. Enkapsulasi data pada OpenVPN

Proses pembentukan tunnel pada OpenVPN adalah sebagai berikut :



Gambar 4.7.proses handshaking OpenVPN

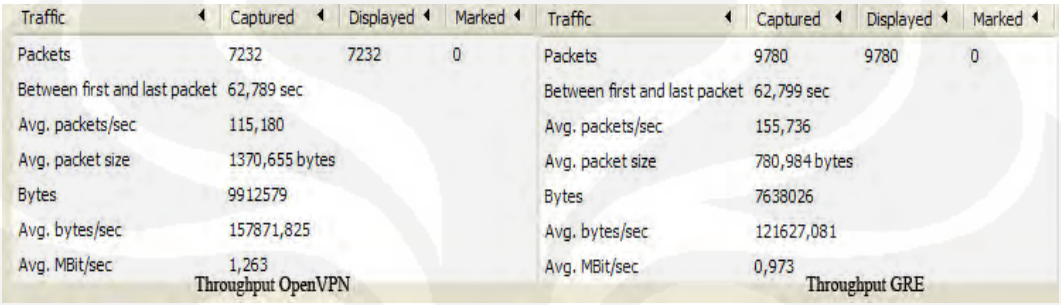
1. Pertama-tama *client* akan mengirimkan *hello* paket ke *server* memulai *handsake*.Termasuk didalamnya adalah daftar *chipper* pendukung yang merupakan salah satu parameter RSA atau *Diffie-Hellmann key*
2. *Server* membalas hello paket dari *client* lalu *server* mengirimkan *server public key* (ca.crt) dan merequest public key dari *client* (ca.crt)
3. *Client* merespon *request* dari *server* dengan mengirimkan *public key* dan *client* juga mengirimkan *client certificate* dan mengenkripsinya.
4. *Server* merespon kiriman sertifikat dari *client* dan membentuk koneksi

OpenVPN mengenkapsulasi dan mengenkripsi frame IP dengan SSL sebelum dilewatkan ke tunnel.

4.3. Throughput Tunnel GRE dan OpenVPN pada *network* tanpa kelas

Pengambilan data dilakukan pada saat GRE/OpenVPN melakukan *streaming video* (paket UDP) dari *Network* A, yang akan di analisa dari pengambilan data ini adalah besarnya *throughput* pada masing-masing jaringan.

Dari data hasil *sniffing* pada simulasi ini di ambil sampel data kedua model dengan transfer time selama 60 s



Traffic	Captured	Displayed	Marked
Packets	7232	7232	0
Between first and last packet	62,789 sec		
Avg. packets/sec	115,180		
Avg. packet size	1370,655 bytes		
Bytes	9912579		
Avg. bytes/sec	157871,825		
Avg. MBit/sec	1,263		

Throughput OpenVPN

Traffic	Captured	Displayed	Marked
Packets	9780	9780	0
Between first and last packet	62,799 sec		
Avg. packets/sec	155,736		
Avg. packet size	780,984 bytes		
Bytes	7638026		
Avg. bytes/sec	121627,081		
Avg. MBit/sec	0,973		

Throughput GRE

Gambar 4.8. *Throughtput* GRE dan OpenVPN

Dengan *bandwidth* sebesar 1024 Kbps, didapatkan *throughput* OpenVPN sebesar 1263 Kbps, maka dalam waktu 60 s jaringan OpenVPN dapat mengirim data sebesar:

Throughput = 1263 Kbps

Transfer Time = 60 s

$$\text{Data } sent = \frac{1263000 \times 60}{8} = 9472500 \ bytes$$

Sedangkan besar *throughput* GRE sebesar 973 Kbps, maka dalam waktu 60s jaringan GRE dapat mengirim data sebesar:

Throughput = 973 Kbps

Transfer Time = 60 s

$$\text{Data sent} = \frac{973000 \times 60}{8} = 7297500 \ bytes$$

Dari data diatas terlihat besar throughput pada OpenVPN pada *network* tanpa kelas lebih besar dari GRE, dikarenakan proses enkapsulasi pada OpenVPN melakukan kompresi *header* paketnya sehingga pengompresan *byte header* yang akan mengurangi jumlah *byte payload* pada OpenVPN, dengan demikian *byte*

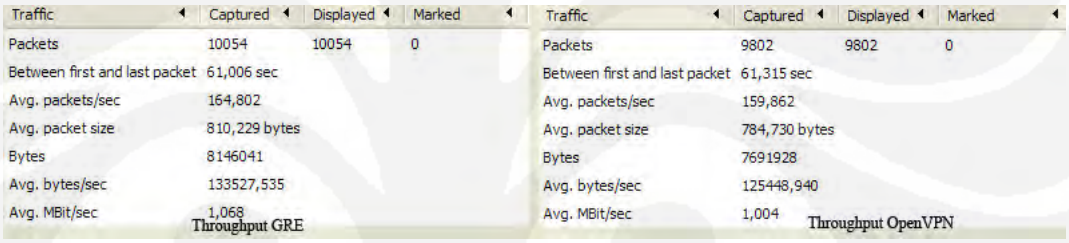
payload yang dapat dikirim tiap satuan waktu pada OpenVPN lebih besar maka *throughput* pada OpenVPN akan lebih besar.

4.4. Throughput Tunnel GRE dan OpenVPN pada *network* dengan kelas

Pengambilan data dilakukan pada saat GRE/OpenVPN melakukan *streaming video* (paket UDP) dari *Network* A, yang akan di analisa dari pengambilan data ini adalah besarnya *throughput* pada masing-masing jaringan. Adapun kelas trafik yang terpasang pada protokol tunel ini adalah :

- Trafik UDP : CIR = 512 Kbps; MIR = 1024 Kbps
- Trafik RDP : CIR = 256 Kbps; MIR = 1024 Kbps
- Trafik TCP : CIR = 192 Kbps; MIR = 1024 Kbps

Dari data hasil *sniffing* pada simulasi ini di ambil sampel data kedua model dengan transfer time selama 60 s



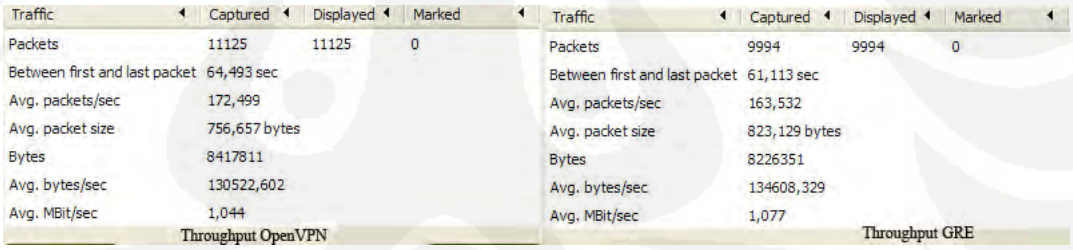
Traffic	Captured	Displayed	Marked
Packets	10054	10054	0
Between first and last packet	61,006 sec		
Avg. packets/sec	164,802		
Avg. packet size	810,229 bytes		
Bytes	8146041		
Avg. bytes/sec	133527,535		
Avg. MBit/sec	1,068		

Throughput GRE

Traffic	Captured	Displayed	Marked
Packets	9802	9802	0
Between first and last packet	61,315 sec		
Avg. packets/sec	159,862		
Avg. packet size	784,730 bytes		
Bytes	7691928		
Avg. bytes/sec	125448,940		
Avg. MBit/sec	1,004		

Throughput OpenVPN

Gambar4.9. Trafik *throughput* GRE dan OpenVPN percobaan 1



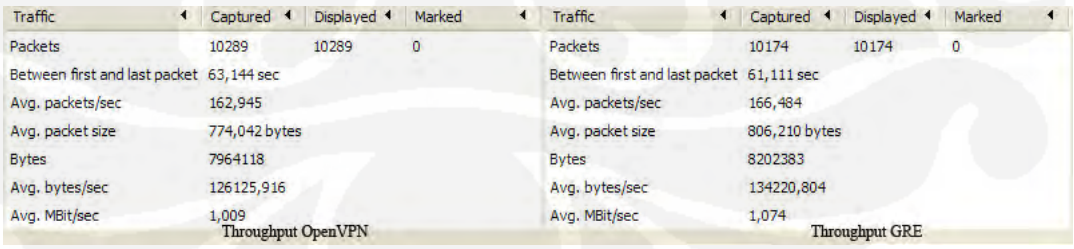
Traffic	Captured	Displayed	Marked
Packets	11125	11125	0
Between first and last packet	64,493 sec		
Avg. packets/sec	172,499		
Avg. packet size	756,657 bytes		
Bytes	8417811		
Avg. bytes/sec	130522,602		
Avg. MBit/sec	1,044		

Throughput OpenVPN

Traffic	Captured	Displayed	Marked
Packets	9994	9994	0
Between first and last packet	61,113 sec		
Avg. packets/sec	163,532		
Avg. packet size	823,129 bytes		
Bytes	8226351		
Avg. bytes/sec	134608,329		
Avg. MBit/sec	1,077		

Throughput GRE

Gambar4.10. Trafik *throughput* GRE dan OpenVPN percobaan 2



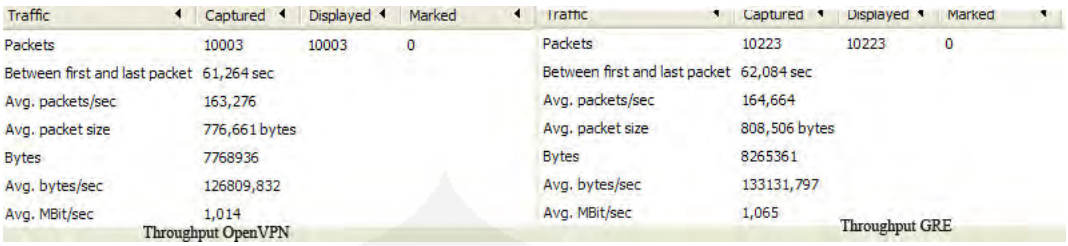
Traffic	Captured	Displayed	Marked
Packets	10289	10289	0
Between first and last packet	63,144 sec		
Avg. packets/sec	162,945		
Avg. packet size	774,042 bytes		
Bytes	7964118		
Avg. bytes/sec	126125,916		
Avg. MBit/sec	1,009		

Throughput OpenVPN

Traffic	Captured	Displayed	Marked
Packets	10174	10174	0
Between first and last packet	61,111 sec		
Avg. packets/sec	166,484		
Avg. packet size	806,210 bytes		
Bytes	8202383		
Avg. bytes/sec	134220,804		
Avg. MBit/sec	1,074		

Throughput GRE

Gambar4.11. Trafik *throughput* GRE dan OpenVPN percobaan 3



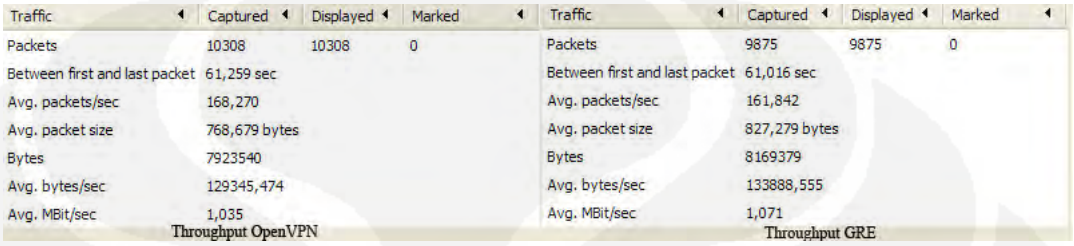
Traffic	Captured	Displayed	Marked
Packets	10003	10003	0
Between first and last packet	61,264 sec		
Avg. packets/sec	163,276		
Avg. packet size	776,661 bytes		
Bytes	7768936		
Avg. bytes/sec	126809,832		
Avg. MBit/sec	1,014		

Throughput OpenVPN

Traffic	Captured	Displayed	Marked
Packets	10223	10223	0
Between first and last packet	62,084 sec		
Avg. packets/sec	164,664		
Avg. packet size	808,506 bytes		
Bytes	8265361		
Avg. bytes/sec	133131,797		
Avg. MBit/sec	1,065		

Throughput GRE

Gambar4.12. Trafik *throughput* GRE dan OpenVPN percobaan 4



Traffic	Captured	Displayed	Marked
Packets	10308	10308	0
Between first and last packet	61,259 sec		
Avg. packets/sec	168,270		
Avg. packet size	768,679 bytes		
Bytes	7923540		
Avg. bytes/sec	129345,474		
Avg. MBit/sec	1,035		

Throughput OpenVPN

Traffic	Captured	Displayed	Marked
Packets	9875	9875	0
Between first and last packet	61,016 sec		
Avg. packets/sec	161,842		
Avg. packet size	827,279 bytes		
Bytes	8169379		
Avg. bytes/sec	133888,555		
Avg. MBit/sec	1,071		

Throughput GRE

Gambar4.13. Trafik *throughput* GRE dan OpenVPN percobaan 5

Tabel 4.1. Tabel trafik *throughput* berkelas GRE dan OpenVPN

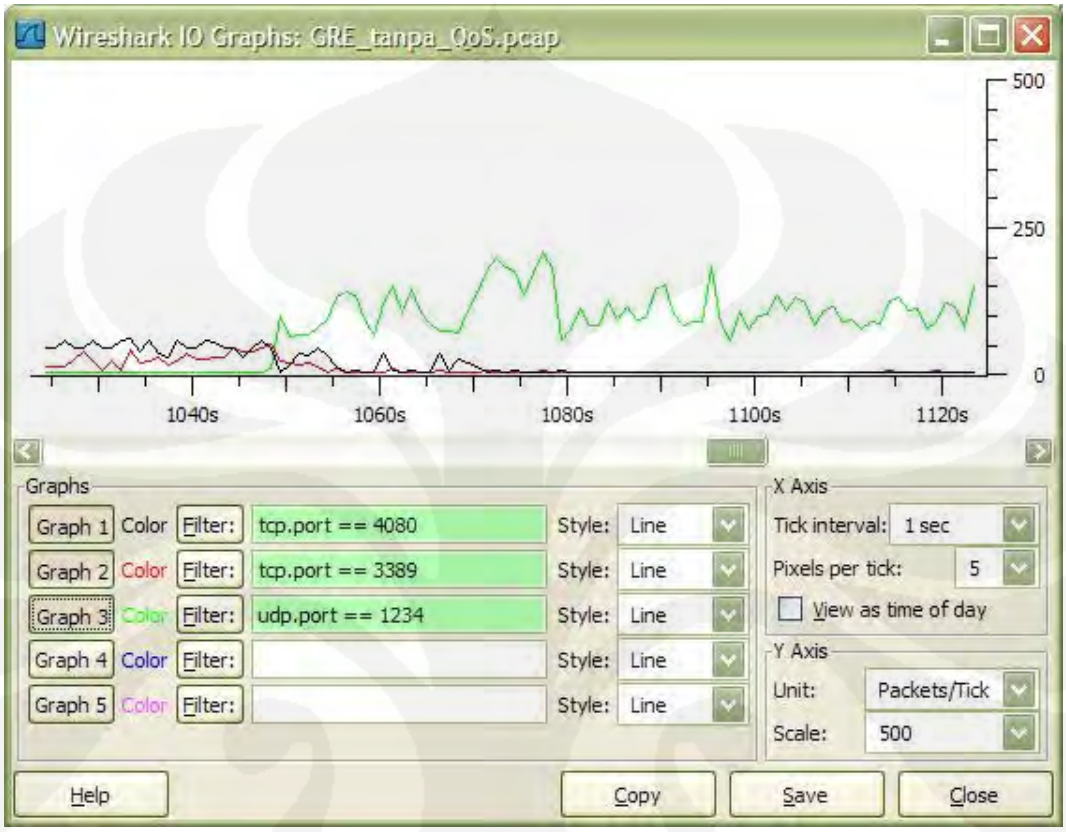
No	*Throughput* Protokol	
	GRE	OpenVPN
1	1.068	1.004
2	1.044	1.077
3	1.009	1.074
4	1.014	1.065
5	1.035	1.071

Jika dilihat dari table diatas terlihat bahwa terjadi *overlimit* pada bandwidth yang tersedia. Dimana total bandwidth yang teredia adalah 1024 Kbps tetapi pada table adalah 1077 Kbps. Hal ini disebabkan oleh pengaruh traffic shaping pada network ISP yang memotong transafer data dari Network A ketika transfer data melebihi bandwidth yang dialokasikan.

4.5. Trafik TCP, UDP dan RDP Pada GRE dan OpenVPN

Pengambilan data dilakukan pada saat *client* melakukan transfer data FTP, *streaming video* dan melakukan *remote desktop* ke *server*, yang akan di analisa dari pengambilan data ini adalah bagaimana karakter TCP (*Transmission Control Protocol*), UDP (*User Datagram Protocol*) dan RDP (*Remote Desktop Protocol*) di dalam tunnel jika ketiga trafik tersebut dilewatkan secara bersamaan.

Sniffing data pada saat GRE melakukan transfer data FTP, *streaming video* dan melakukan *remote desktop* ke *server*:



Gambar 4.14. Grafik *Throughtput* GRE

Sniffing data pada saat OpenVPN melakukan transfer data FTP, *streaming video* dan melakukan *remote desktop* ke *server*:

Gambar 4.15. Grafik *Throughtput* OpenVPN

Dapat dilihat dari kedua grafik diatas ketika dilakukkan transfer TCP dan UDP secara bersamaan, Pada tunnel GRE paket UDP mendapatkan *throughput* yang paling besar dibandingkan dua buah protocol TCP lainnya.hal ini dikarenakan karakteristik dari paket UDP itu sendiri yang merupakan protokol *connectionless* artinya paket-paket tersebut dikirim begitu saja oleh *server* tanpa melakukkan *handshaking* terlebih dahulu.Berbeda dengan protokol TCP yang memerlukan *handshaking*, sehingga berdasarkan karakter tersebut paket UDP menggunakan hampir semua alokasi *bandwidth* yang tersedia di *network* yang menyebabkan ketika *client* mengirimkan ACK ke *server* paket ACK tersebut tidak diterima oleh *client* sehingga *client* berasumsi bahwa *network* terputus dan kembali mengirimkan SYN.Berbeda dengan GRE perilaku pengiriman paket pada OpenVPN diberlakukkan layaknya sebuah paket UDP biasa, dan juga paket UDP

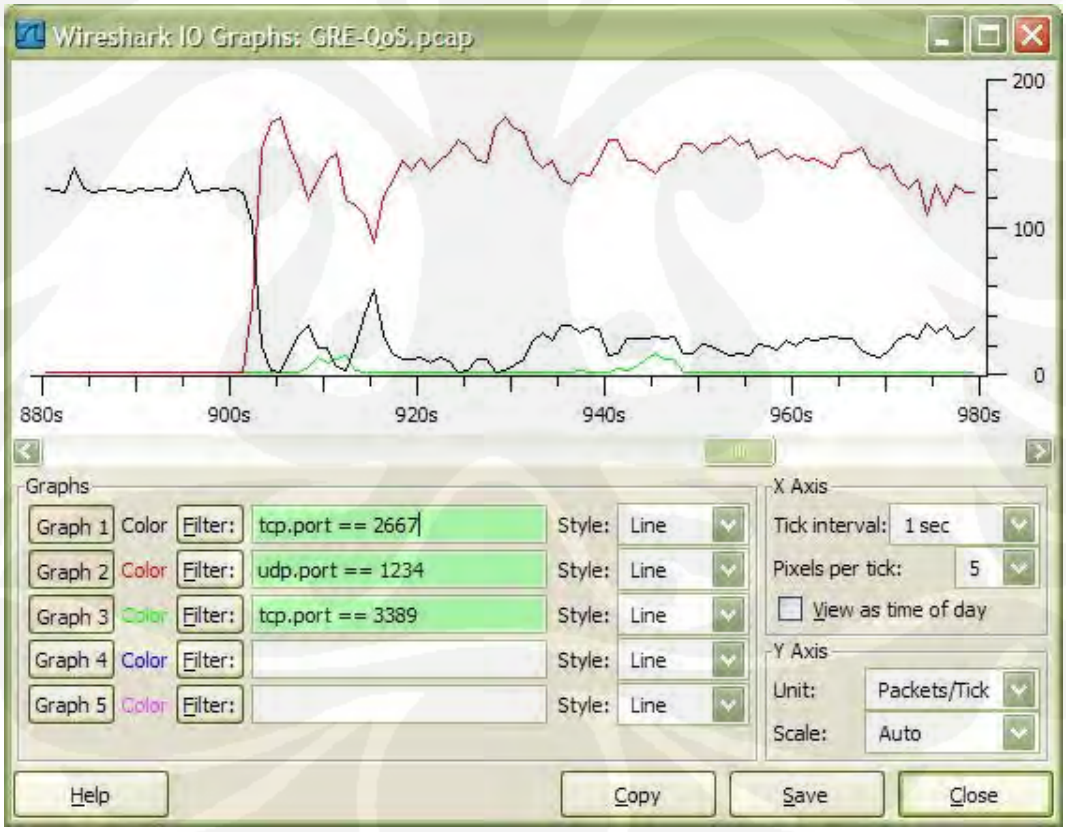
tersebut dikompres terlebih dahulu oleh OpenVPN sehingga paket yang dikirim lebih kecil dan tidak membebani *network*.

Untuk mengatasi hal tersebut diperlukan adanya pembagian *Class of Service* untuk membedakan *Type of Service* dari masing-masing protokol.

Trafik-trafik yang akan dilewatkan dibedakan atas *priority, Commited Information Rate* (CIR) dan *Maximum Information Rate* (MIR).

- Trafik UDP : CIR = 512 Kbps; MIR = 1024 Kbps; *Priority* = 7
- Trafik RDP : CIR = 256 Kbps; MIR = 1024 Kbps; *Priority* = 6
- Trafik TCP : CIR = 192 Kbps; MIR = 1024 Kbps; *Priority* = 4

QoS pada GRE *tunnel* akan di pasang pada *FastEthernet* 0/1 yang terkoneksi pada LAN (*Local Area Network*). Konfigurasi *class of service* GRE dan OpenVPN dapat dilihat pada lampiran 7 halaman 39.

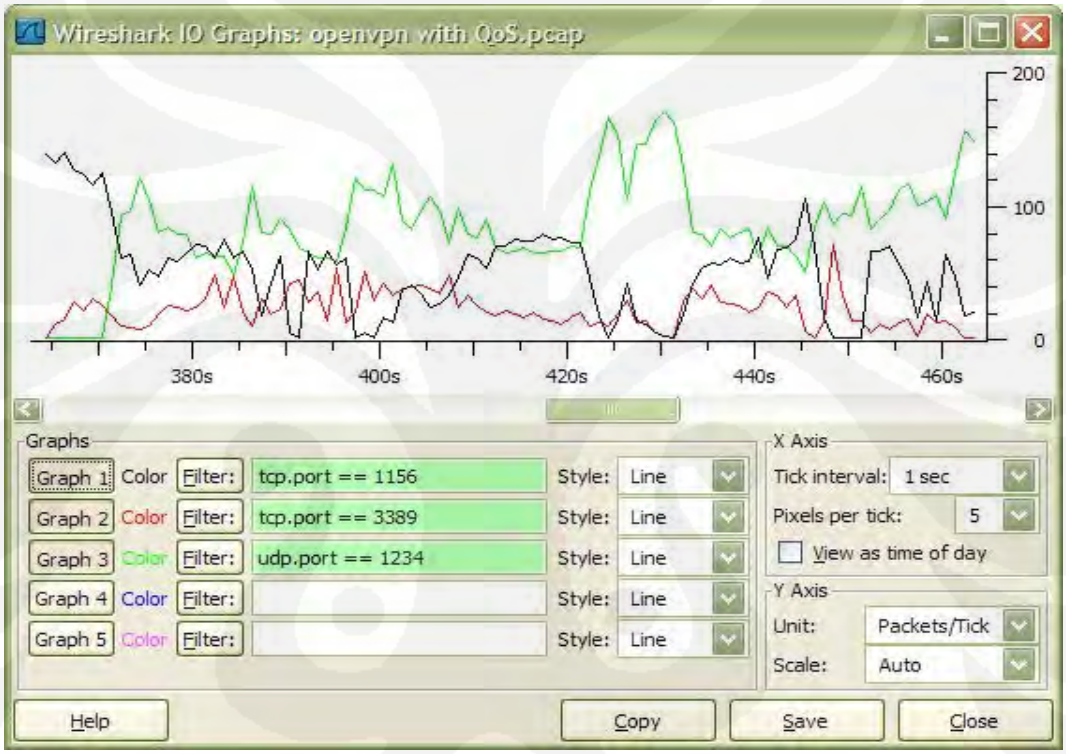


Gambar 4.16. Grafik *Throughtput* GRE dengan *QoS*

Dari grafik diatas dapat dilihat bahwa ketika kondisi *network* tidak terpakai maka paket TCP dalam hal ini FTP dapat menggunakan seluruh alokasi

bandwidth yang tersedia. Dan ketika terdapat paket UDP yang melewati *network* maka paket TCP tersebut akan mengikuti aturan yang diberlakukan sesuai dengan *class of servicenya*.

Pengelompokan kelas trafik dapat membantu menyelesaikan masalah pada *network* dimana pada sebuah *network* tanpa *class of service* paket UDP akan mengambil alih seluruh alokasi *bandwidth* yang tersedia. Sehingga paket-paket lain dalah hal ini paket TCP tidak memiliki mendapatkan *session* sehingga paket TCP tersebut akan terputus koneksinya.



Gambar 4.17. Grafik *Throughtput* OpenVPN dengan *QoS*

Penggunaan *class of service* pada tunnel OpenVPN tidak berpengaruh terhadap isi paket didalam OpenVPN. Hal ini disebabkan karena paket didalam OpenVPN mengalami pengompresan paket sebelum dikirim.

BAB V
KESIMPULAN

Pada tugas akhir ini dirancang sebuah *network corporate* yang terdiri dari dua buah *network* (*network* A dan *network* B) yang dipisahkan oleh cloud *internet*. Kedua buah *network* tersebut dihubungkan oleh tunnel. Setelah melakukan pengambilan data dan menganalisis, didapatkan beberapa buah kesimpulan yaitu :

1. Proses enkapsulasi tunel pada paket akan mengurangi jumlah payload seharusnya pada paket normal (tanpa tunel).Sebuah paket normal seharusnya memiliki besar payload 1500 byte, ketika paket tersebut di enkapsulasi dengan tunnel GRE maka jumlah payload seharusnya akan berkurang sekitar 24 byte.Dimana 24 byte tersebut dipakai oleh sebesar 20 byte untuk outer IP address, dan 4 byte sebagai GRE header.Dimana outer IP address berisikan destination network untuk paket GRE tersebut.

   Sedangkan OpenVPN akan mengurangi payload paket sebesar 28 byte, dimana 20 byte akan digunakan sebagai outer IP address dan 8 byte digunakan sebagai protokol tunelnya.Selain itu OpenVPN juga melakukkan enkripsi paket yang dikirimnya, adapun besarnya enkripsi pada openVPN adalah 48 byte
2. Dari hasil percobaan yang telah dilakukan pada sebuah *network* tanpa kelas didapatkan bahwa tunnel pada OpenVPN memiliki *throughput* lebih besar dibandingkan *throughput* pada tunnel GRE. Pada sebuah *network* tanpa kelas tunnel GRE akan menggunakan sekitar 95% dari total *bandwidth* yang tersedia, sedangkan OpenVPN akan menggunakan seluruh alokasi *bandwidth* yang tersedia. Sehingga jika dilihat dari sisi efisiensi *bandwidth* maka OpenVPN lebih baik dari tunnel GRE.
3. Penggunaan *class of service* pada tunnel GRE dapat lebih mengefisienkan penggunaan *bandwidth* pada protokol-protokol tertentu.Yang jika sebelumnya tunnel GRE diimplementasikan tanpa *class of service* tunnel GRE hanya menggunakan 95% dari total bandwidth yang tersedia menjadi 100%. Sedangkan penggunaan *class of service* pada OpenVPN tidak

berpengaruh pada penggunaan *bandwidth*, hal ini dikarenakan header dari OpenVPN itu

sendiri yang menggunakan header UDP, sehingga perlakuan paket OpenVPN diberlakukan sama dengan layaknya paket UDP biasa.

DAFTAR ACUAN

[1] "*Introduction of Generic Routing Encapsulation*", www.cisco.com diakses Maret 2008. http://www.cisco.com/en/US/tech/tk827/tk369/tk287/tsd_technology_support_sub-protocol_home.html

[2] Training Module "*IP Tunneling and VPNs*", Cisco System, Copyright 2001

[3] *"OpenVPN 2.1"*, *http://openvpn.net* diakses April 2009 http://openvpn.net/index.php/open-source/documentation/manuals/69-openvpn-21.html

[4] *"OpenVPN and the SSL VPN Revolution"*, SANS institute copyright 2004

[5] Lewis.Chris, *"Cisco TCP/IP Routing Professional Reference"*, (Computing McGraw-Hill : 1999)

[6] Feilner, Markus, *"OpenVPN: Building and Integrating Virtual Private Networks"* (Packt Publishing Ltd : 2006)

- ***Setup* GRE pada *Router* :**

1. *Network* A :

```
interface Tunnel0
 IP address 192.168.100.1 255.255.255.252
 tunnel source 192.168.1.9
 tunnel destination 192.168.1.26
!
interface FastEthernet0/0
 IP address 192.168.1.9 255.255.255.248
 ip nat outside
 ip virtual-reassembly
 speed auto
 full-duplex
!
interface FastEthernet0/1
 IP address 172.16.1.60 255.255.255.0
 ip nat inside
 ip virtual-reassembly
 speed auto
 full-duplex
ip forward-protocol nd
ip route 0.0.0.0 0.0.0.0 192.168.1.10
ip route 192.168.1.32 255.255.255.248 Tunnel0
no ip http server
no ip http secure-server
ip nat inside source list Networks_2A_NATed interface FastEthernet0/0 overload
ip access-list standard Networks_2A_NATed
 permit 172.16.1.0 0.0.0.255
```

Lampiran 2

2. ISP A

```
interface FastEthernet0/0
 description connected to router ISP B
 bandwidth 1024
 IP address 192.168.1.17 255.255.255.248
 rate-limit input 1024000 128000 128000 conform-action transmit exceed-action drop
 rate-limit output 1024000 128000 128000 conform-action transmit exceed-action drop
 speed auto
 full-duplex
interface FastEthernet0/1
 description connected to router network A
 bandwidth 1024
 IP address 192.168.1.10 255.255.255.248
 rate-limit input 1024000 128000 128000 conform-action transmit exceed-action drop
 rate-limit output 1024000 128000 128000 conform-action transmit exceed-action drop
 speed 10
 full-duplex
```

3. ISP B

```
interface FastEthernet0/0
 bandwidth 1024
 IP address 192.168.1.25 255.255.255.248
 rate-limit input 1024000 128000 128000 conform-action transmit exceed-
action drop
 rate-limit output 1024000 128000 128000 conform-action transmit exceed-
action drop
 duplex auto
 speed auto
interface FastEthernet0/1
 bandwidth 1024
 IP address 192.168.1.18 255.255.255.248
 rate-limit input 1024000 128000 128000 conform-action transmit exceed
action drop
 rate-limit output 1024000 128000 128000 conform-action transmit exceed-
action drop
```

Lampiran 3

4. *Sniffing server*

   Untuk dapat melakukkan fungsi *routing* terlebih dahulu *service routing* pada windows XP SP 2 harus dinyalakan.Adapun langkah-langkah yang dilakukkan adalah :

   - Klik Run ➔ services.msc lalu tekan enter sehingga halaman console terbuka
   - Cari ***Routing and Remote Access*** ganti *startup type* dari *Disable* menjadi *Automatic* lalu klik *start*.

5. *Network* B

```
interface Tunnel0
 IP address 192.168.100.2 255.255.255.252
 keepalive 10 3
 tunnel source 192.168.1.26
 tunnel destination 192.168.1.9
!
interface FastEthernet0/0
 IP address 192.168.1.33 255.255.255.248
 ip nat inside
 ip virtual-reassembly
 duplex auto
 speed auto
!
interface FastEthernet0/1
 IP address 192.168.1.26 255.255.255.248
 ip nat outside
 ip virtual-reassembly
 speed auto
 full-duplex
!
interface Serial0/1/0
 no ip address
 shutdown
 no fair-queue
 clock rate 2000000
!
ip forward-protocol nd
ip route 0.0.0.0 0.0.0.0 192.168.1.25
ip route 172.16.1.0 255.255.255.0 Tunnel0
no ip http server
no ip http secure-server
!
!
ip nat inside source list Networks_2B_NATed interface FastEthernet0/1
overload
!
ip access-list standard Networks_2B_NATed
 permit 192.168.1.32 0.0.0.6
```

Lampiran 4

- ***Setup* OpenVPN :**

Program OpenVPN dapat *didownload* dengan gratis pada alamat http://www.OpenVPN.se, setelah proses *download* selesai, kita tinggal menjalankan *program* OpenVPN-2.1_rc22-install.exe dengan mengeklik *double* dan ikuti petunjuk instalasi.

1. buka *command prompt*
2. Konfigurasi OpenVPN *server*
   Copy file *server*.ovpn ke c:\program files\OpenVPN\config\
   C:\>copy C:\Program Files\OpenVPN\sample-config C:\Program Files\OpenVPN\config
3. Edit file *server*.opvn [7]

```
## server.ovpn ##
port 1194
proto udp
dev tun
ca ca.crt
cert server.crt
key server.key
dh dh1024.pem
server 10.10.10.0 255.255.255.0
ifconfig-pool-persist ipp.txt
push "route 172.16.1.0 255.255.255.0"
keepalive 10 120
comp-lzo
max-clients 4
persist-key
persist-tun
status OpenVPN-status.log
verb 3
```

4. *Set up a Certificate Authority (CA)*
   - Buka command prompt
     C:\>Program Files\OpenVPN\easy-rsa>**init-config**
   - Edit file vars.bat dan rubah nilai “KEY_” pada file tersebut

```
set KEY_COUNTRY=GB
set KEY_PROVINCE=London
set KEY_CITY=London
set KEY_ORG=Acme
set KEY_EMAIL=hostmaster@acme.com
```

   - Membuat *keys folder* pada *root* OpenVPN

```
C:\Program Files\OpenVPN\easy-rsa> vars
C:\Program Files\OpenVPN\easy-rsa> clean-all
C:\Program Files\OpenVPN\easy-rsa> build-ca
```

Lampiran 5

- Isi *common name*

  Common Name (eg, your name or your *server*'s hostname) []:*Server*

- *Copy* file ca.crt yang di *generate* tadi ke *folder* config

  C:\Program Files\OpenVPN\easy-rsa> copy keys\ca.crt ..\config\

- *Setup server* key dan *certificate*

  C:\Program Files\OpenVPN\easy-rsa> vars

  C:\Program Files\OpenVPN\easy-rsa> build-key-*server* *server*

- Isi *common name*

  Common Name (eg, your name or your *server*'s hostname) []:Fadry

- *Generate Diffie Hellman* parameter

  C:\Program Files\OpenVPN\easy-rsa> build-dh

- Kopi file *key, certificate*, dan *DH* ke *folder config*

  C:\Program Files\OpenVPN\easy-rsa> copy keys\widget.crt ..\config\

  C:\Program Files\OpenVPN\easy-rsa> copy keys\widget.key ..\config\

  C:\Program Files\OpenVPN\easy-rsa> copy keys\dh1024.pem ..\config\

5. *Setup* OpenVPN *client* [7]

- buka command prompt
- Konfigurasi OpenVPN *server*

  Copy file *client*.ovpn ke c:\program files\OpenVPN\config
  C:\>copy C:\Program Files\OpenVPN\sample-config C:\Program Files\OpenVPN\config

Lampiran 6

6. *Edit* file *client*.opvn

```
## acme.ovpn ##
client
proto udp
dev tun
remote 192.168.1.9 1194
resolv-retry infinite
nobind
persist-key
persist-tun
ca ca.crt
cert user.crt
key user.key
comp-lzo
verb 3
```

7. *Generate* file *key* untuk *client*

   C:\Program Files\OpenVPN\easy-rsa> vars

   C:\Program Files\OpenVPN\easy-rsa> build-key user

8. Kopi file user.crt dan user.key ke PC *client* dan masukkan ke *folder config* di sisi *client*
9. Jalankan OpenVPN pada sisi *server* dan pada sisi *client* dengan cara mengklik kanan *icon* OpenVPN GUI yang berada pada *system tray*.

Lampiran 7

- **Konfigurasi QoS**

Pertama yang dilakukan adalah pengelompokan berdasarkan trafik

```
ip cef
ip access-list extended UDP
 remark --- UDP from LAN
 permit udp 192.168.1.40 0.0.0.7 any eq 1234
ip access-list extended RDP
 remark --- RDP traffic from LAN to RDP servers
 permit tcp 192.168.1.40 0.0.0.7 any eq 3389
ip access-list extended FTP
 remark --- FTP traffic from LAN to FTP servers
 permit tcp 192.168.40.0 0.0.0.7 any eq ftp
 permit tcp 192.168.40.0 0.0.0.7 any eq ftp-data
class-map match-any High-Class-Inbound
match access-group name UDP
match ip dscp ef
class-map match-any Med-Class-Inbound
match access-group name RDP
class-map match-any Low-Class-Inbound
match access-group name FTP
policy-map Packet-Tagging
class High-Class-Inbound
police 512000 64000 64000 conform-action set-prec-trans 7 exceed-action set-prec-trans 7
class Med-Class-Inbound
police 256000 32000 32000 conform-action set-prec-trans 6 exceed-action set-prec-trans 5
class Low-Class-Inbound
police 192000 24000 24000 conform-action set-prec-trans 4 exceed-action set-prec-trans 3
class class-default
set ip precedence 1
interface Ethernet1
 service-policy input Packet-Tagging
```

Lampiran 8

Setelah *marking* packet, membuat *queue tree* pada *interface* LAN:

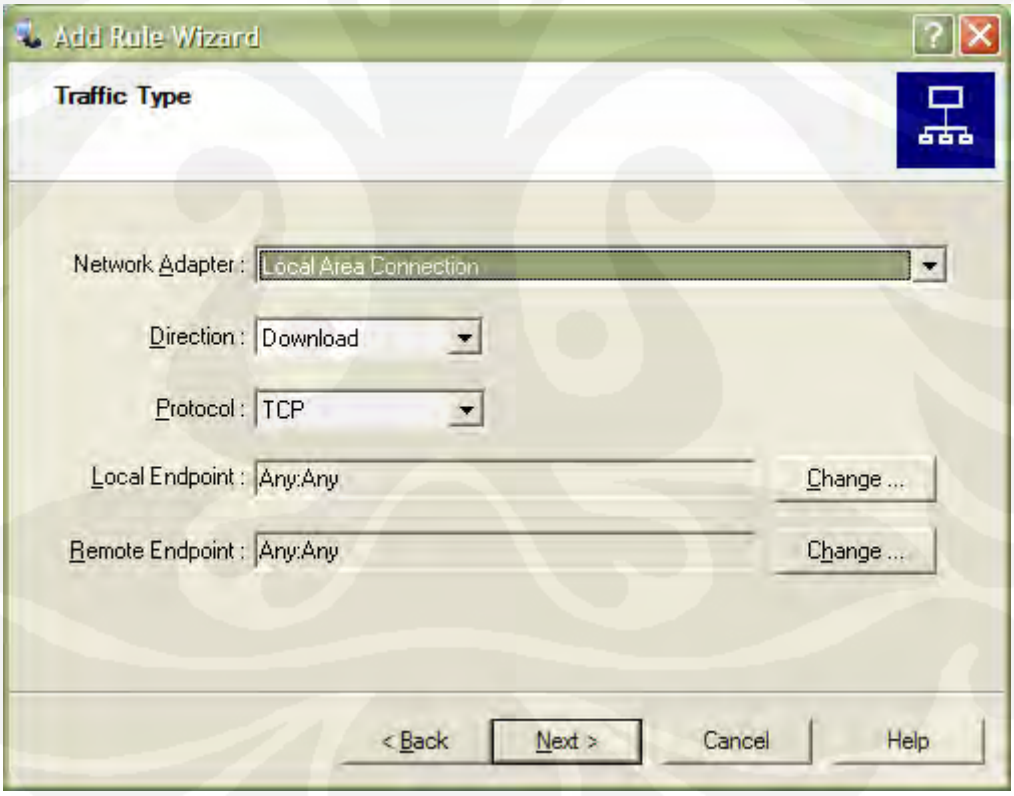
```
lass-map match-any High-Class-Outbound
match ip precedence 7
class-map match-any Med-Class-Outbound
match ip precedence 6
match ip precedence 5
class-map match-any Low-Class-Outbound
match ip precedence 4
match ip precedence 3
policy-map Data-Only-Queueing
class Med-Class-Outbound
bandwidth percent 50
  random-detect prec-based
  random-detect exponential-weighting-constant 8
  random-detect precedence 6   20    60    20
  random-detect precedence 5   6     15    6
class Low-Class-Outbound
bandwidth percent 50
random-detect prec-based
  random-detect exponential-weighting-constant 8
  random-detect precedence 4   15    30    15
  random-detect precedence 3   1     15    3
policy-map Packet-Queueing
class High-Class-Outbound
priority 512
class class-default
shape average 448000
bandwidth 448
service-policy Data-Only-Queueing
interfaceethernet0
bandwidth 1000
max-reserved-bandwidth 95
service-policy output Packet-Queueing
 tx-ring-limit 2
 tx-queue-limit 2
```

Lampiran 9

QoS pada OpenVPN akan di pasang pada *FastEthernet* 0/1 yang terkoneksi pada LAN (*Local Area Network*)

Pertama yang dilakukkan adalah :

1. Buka aplikasi *bandwidth controller standard* lalu pilih *connect*
2. Klik tombol *add rule wizard*
3. Setelah *window wizard* terbuka pilih *netwok adapter* yang akan dipasangkan *QoS*
4. Ubah *Direction download*nya menjadi both
5. Ubah protokolnya sesuai dengan protokol yang akan digunakan
6. Ubah *local end point*nya dan *remote end point* sesuai dengan *port* yang akan digunakan, jika *port* tidak terdata maka kita harus mendaftarkan *port* tersebut.

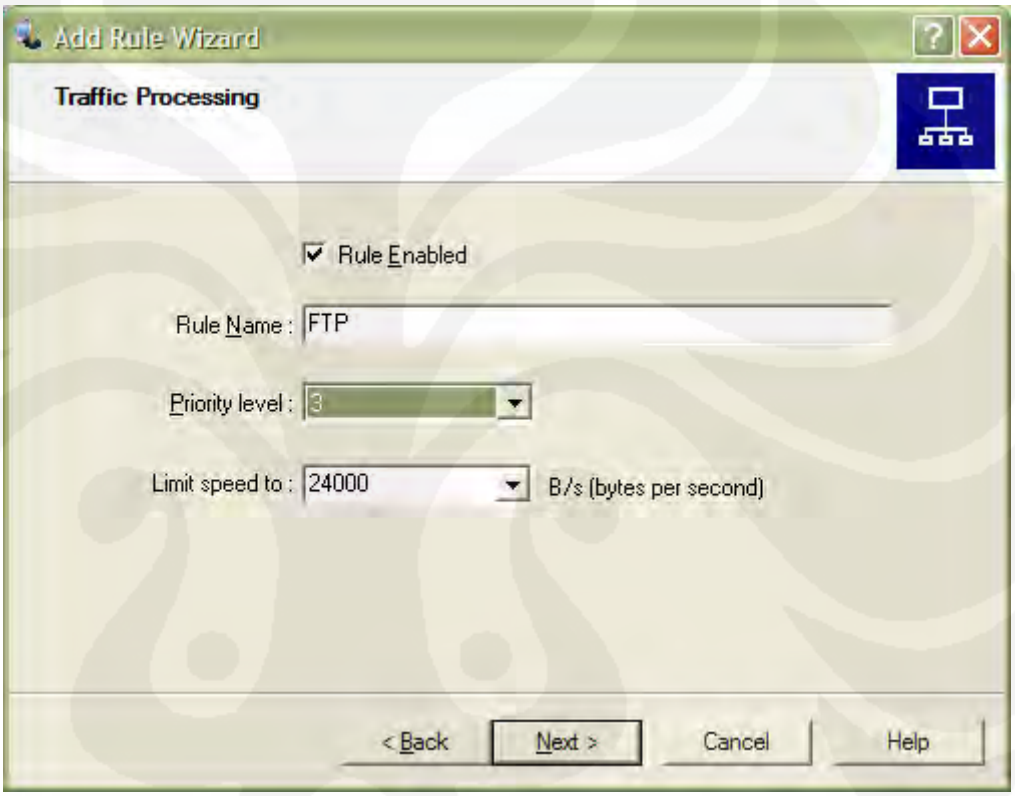


Gambar 4.8. *Traffic Type*

7. *Checklist enable rule*
8. Berikan nama pada rule yang akan kita buat, berikan prioritas pada paket
    - FTP : *priority* 3

- RDP : *priority* 5
- UDP : *priority* 9

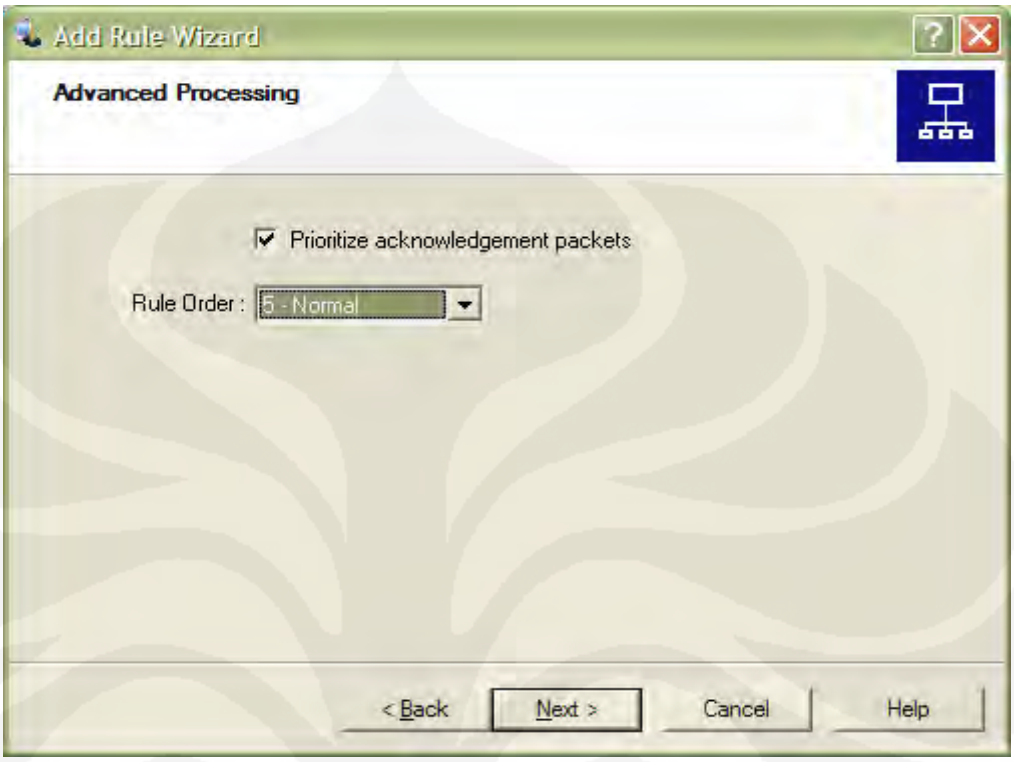
9. Masukkan besarnya *download* dan *upload*
    - RDP : 32.000 B/s (*byte*)
    - FTP : 24.000 B/s
    - UDP : 64.000 B/s



Gambar 4.9. *Traffic Processing*

Lampiran 11

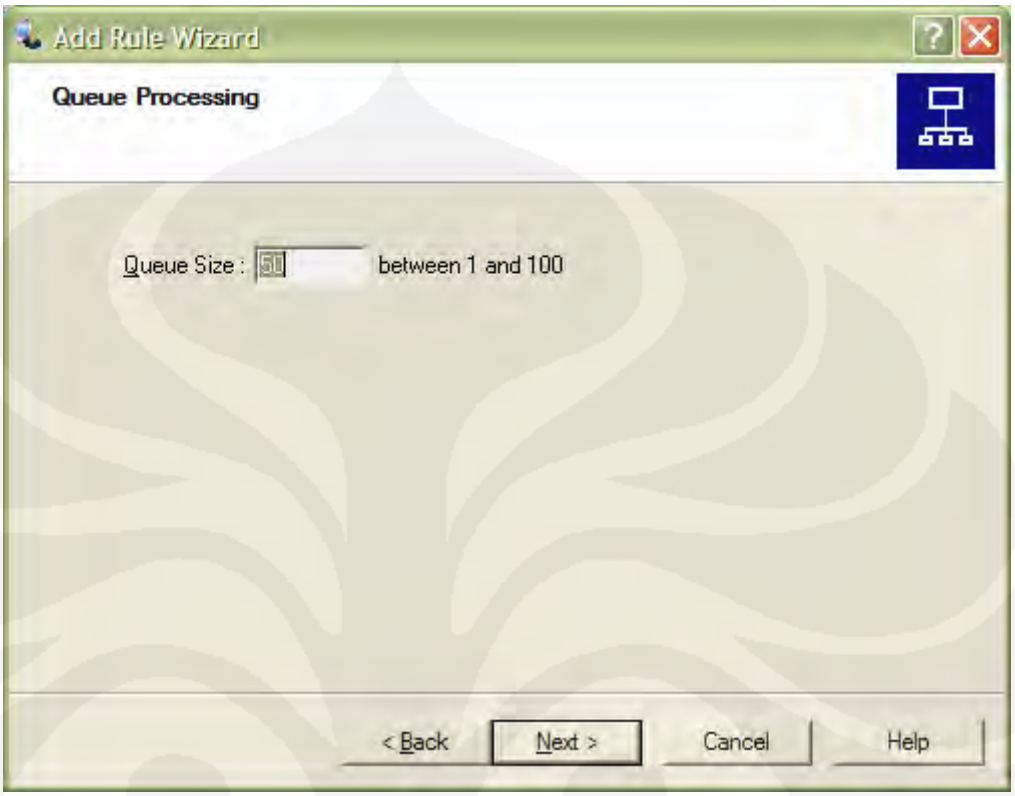
10. Biarkan *rule order default*



Gambar 4.10. *Advanced Processing*

Lampiran 12

11. Masukkan nilai Queue sesuai yang kita inginkan



Gambar 4.11. *Queue Processing*

Dengan melakukkan pembagian *bandwidth* tersebut maka :

UDP akan mendapatkan *bandwidth* sebesar 576 Kbps.

RDP akan mendapatkan *bandwidth* sebesar 256 Kbps.

TCP akan mendapatkan *bandwidth* sebesar 192 Kbps.

Setelah *QoS* terpasang dilakukkan kembali sniffing traffic protocol TCP, RDP dan UDP