UNIVERSITAS INDONESIA

IMPLEMENTASI DAN ANALISA UNJUK KERJA SECURE VOIP PADA JARINGAN VPN BERBASIS MPLS DENGAN MENGGUNAKAN TUNNELING IPSEC

SKRIPSI

ANDI TAUFIK SAPUTRA
0806365356

FAKULTAS TEKNIK
PROGRAM STUDI TEKNIK ELEKTRO
UNIVERSITAS INDONESIA
DEPOK
JULI 2010

UNIVERSITAS INDONESIA

IMPLEMENTASI DAN ANALISA UNJUK KERJA SECURE VOIP PADA JARINGAN VPN BERBASIS MPLS DENGAN MENGGUNAKAN TUNNELING IPSEC

SKRIPSI

Diajukan sebagai salah satu syarat untuk memperoleh gelar Sarjana Teknik

ANDI TAUFIK SAPUTRA
0806365356

FAKULTAS TEKNIK
PROGRAM STUDI TEKNIK ELEKTRO
UNIVERSITAS INDONESIA
DEPOK
JULI 2010

HALAMAN PERNYATAAN ORISINALITAS

Skripsi ini adalah hasil karya saya sendiri,

dan semua sumber baik yang dikutip maupun dirujuk

telah saya nyatakan dengan benar.

Nama : Andi Taufik Saputra

NPM : 08 06 36 53 56

Tanda Tangan :

Tanggal : 1 Juli 2010

HALAMAN PENGESAHAN

Skripsi ini diajukan oleh :
Nama : Andi Taufik Saputra
NPM : 0806365356
Program Studi : Teknik Elektro
Judul Skripsi : Implementasi dan Analisa Unjuk Kerja *Secure VoIP* pada Jaringan *VPN* berbasis *MPLS* dengan Menggunakan *Tunneling IPSec.*

Telah berhasil dipertahankan di hadapan Dewan Penguji dan diterima sebagai bagian persyaratan yang diperlukan untuk memperoleh gelar Sarjana Teknik pada Program Studi Teknik Elektro, Fakultas Teknik, Universitas Indonesia

DEWAN PENGUJI

Pembimbing : Muhammad Salman ST, MIT ( )

Penguji : Prof. Dr. Ir. Riri Fitri Sari, M.Sc, MM ( )

Penguji : Prof. Dr. Ing. Ir. Kalamullah Ramli, M.Eng ( )

Ditetapkan di : Depok

Tanggal : 1 Juli 2010

KATA PENGANTAR

Puji syukur saya panjatkan kepada Tuhan Yang Maha Esa, karena atas berkat dan rahmat-Nya, saya dapat menyelesaikan skripsi ini. Penulisan skripsi ini dilakukan dalam rangka memenuhi salah satu syarat untuk mencapai gelar Sarjana Teknik Jurusan Teknik Elektro pada Fakultas Teknik Universitas Indonesia. Saya menyadari bahwa, tanpa bantuan dan bimbingan dari berbagai pihak, dari masa perkuliahan sampai pada penyusunan skripsi ini, sangatlah sulit bagi saya untuk menyelesaikan skripsi ini. Oleh karena itu, saya mengucapkan terima kasih kepada:

1. Muhammad Salman ST, MIT selaku dosen pembimbing yang telah menyediakan waktu, tenaga, dan pikiran untuk mengarahkan saya dalam penyusunan skripsi ini.
2. Orang tua dan keluarga saya yang telah memberikan bantuan dukungan secara moral,
3. Sahabat yang telah banyak membantu saya dalam menyelesaikan skripsi ini.

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

Depok, 1 Juli 2010

Penulis

HALAMAN PERNYATAAN PERSETUJUAN PUBLIKASI TUGAS AKHIR UNTUK KEPENTINGAN AKADEMIS

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

Nama	: Andi Taufik Saputra
NPM	: 0806365356
Program Studi	: Teknik Elektro
Departemen	: Teknik Elektro
Fakultas	: Teknik
Jenis karya	: Skripsi

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 :

IMPLEMENTASI DAN ANALISA UNJUK KERJA *SECURE VOIP* PADA JARINGAN *VPN* BERBASIS *MPLS* DENGAN MENGGUNAKAN *TUNNELING IPSEC*

beserta perangkat yang ada (jika diperlukan). Dengan Hak Bebas Royalti Noneksklusif ini Universitas Indonesia berhak menyimpan, mengalihmedia/ formatkan, mengelola dalam bentuk pangkalan data (*database*), merawat, dan mempublikasikan skripsi 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 : 1 Juli 2010

Yang menyatakan

(Andi Taufik Saputra)

ABSTRAK

Nama : Andi Taufik Saputra
Program Studi : Teknik Elektro
Judul : Implementasi dan Analisa Unjuk Kerja *Secure VoIP* pada Jaringan *VPN* berbasis *MPLS* dengan Menggunakan *Tunneling IPSec*

Seiring dengan perkembangan teknologi, banyak layanan multimedia telah dikembangkan di internet. Salah satu dari layanan itu adalah *VoIP*. Teknologi *VoIP* sangat menguntungkan karena menggunakan jaringan berbasis IP, sehingga biaya untuk melakukan panggilan jauh lebih efisien daripada menggunakan telepon analog. Masalah keamanan menjadi kebutuhan yang mendasar karena *VoIP* dikirimkan melewati jaringan publik yang tidak aman, dimana banyak kemungkinan terjadi penyalahgunaan seperti *hacking* dan *data-sniffing*. Salah satu cara untuk membangun keamanan dalam jaringan internet adalah dengan menggunakan jaringan *Virtual Private Network (VPN)*. *VPN* merupakan sebuah jaringan *private* yang menghubungkan satu *node* jaringan ke *node* jaringan lainnya dengan menggunakan jaringan publik. Data akan dienkapsulasi dan dienkripsi agar terjamin kerahasiaannya. Pada tugas akhir ini, data *VoIP* akan dilewatkan pada jaringan *MPLS* dengan *tunnel IPSec* untuk meningkatkan unjuk kerja dan keamanan. Pengujian sistem melibatkan penggunaan 3 jenis *codec* G.711 , G.729 dan *GSM* dengan berbagai variasi pengujian untuk mengukur dan menganalisa unjuk kerjanya. Dari hasil pengujian, diketahui *codec* G.711 memiliki kualitas suara yang paling bagus (*MOS* = 4.4) tetapi membutuhkan *bandwidth* yang paling besar (128 Kbps). *Codec* G.729, membutuhkan *bandwidth* yang lebih kecil (64 Kbps), dengan nilai *MOS* 4.1 menghasilkan kualitas suara yang hampir sama bagusnya dengan G.711. Sedangkan *codec GSM* mempunyai nilai *MOS* 3.4 kualitas suara tidak terlalu bagus, tetapi membutuhkan *bandwidth* yang relatif kecil sama seperti G.729 dan *codec* ini bersifat *open source*. Untuk keamanan data *VoIP*, *VPN* dapat mengamankan data dari ancaman penyalahgunaan. Sebelum menggunakan *VPN* data *VoIP* dapat direkam dan di-*playback*, data *payload*-nya juga dapat di-*capture* dan dilihat. Tetapi setelah menggunakan *VPN*, data *VoIP* tidak dapat direkam dan *payload*-nya pun tidak dapat dilihat. Sehingga dapat disimpulkan bahwa pada *bandwidth* yang terbatas (dibawah 128 Kbps) G.729 adalah *codec* yang paling bagus dan efisien daripada G.711 dan *GSM* untuk diimplementasikan pada jaringan *MPLS* dengan *tunneling IPSec*.

Kata Kunci :
VoIP, MPLS, VPN, IPSec, Codec, MOS

ABSTRACT

Name : Andi Taufik Saputra
Study Program: Electrical Engineering
Title : Implementation and Performance Analysis of Secure VoIP over MPLS based VPN Network using IPSec Tunneling.

Along with development of technology, many multimedia services have been developed on the Internet. One of these services is VoIP. VoIP technology is very advantageous because it uses IP-based network, so that the cost to make calls much more efficient than using an analog phone. But security problem becomes a fundamental need, because it is transmitted through the Internet as public network that not secure, there're a lot of possibilities of abuses such as hacking and data-sniffing occurred. One of ways to build security over the Internet is by using a Virtual Private Network (VPN). It is a private network that connects one network node to other network nodes using a public network. The data will be encapsulated and encrypted to assure confidentiality. In this final project, VoIP data will be run over MPLS network with IPSec tunneling to increase performance and security. System testing involves the use of three types of codec G.711, G.729 and GSM, with a variety of tests to measure and analyze their performances. From the test results, known that G.711 codec has the best voice quality (MOS = 4.4), but it takes the most bandwidth 128 Kbps. G.729 codec requires a smaller bandwidth 64 Kbps, it has MOS value 4.1 but voice quality almost as good as G.711. Whereas, GSM codec has the smallest MOS value (MOS = 3.4), voice quality is not too good but required bandwidth relatively small as G.729, and it's open source. VPN can secure the VoIP data from the threat of data misuse. Before using VPN, VoIP data can be recorded and played back, it can also be captured its payload and views. But after using VPN, VoIP data cannot be recorded and the payload cannot be seen. Then, it can be concluded that in limited bandwidth (under 128 Kbps) G.729 is the best codec and most efficient than G.711 and GSM to implement in MPLS network with IPSec tunneling.

Key Word :
VoIP, MPLS, VPN, IPSec, Codec, MOS

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I

PENDAHULUAN

1.1 LATAR BELAKANG

Teknologi komunikasi berbasis IP berkembang dengan begitu cepatnya seiring dengan kemajuan teknologi. Saat ini jaringan internet tidak hanya terfokus pada layananan paket data dan aplikasi standar seperti *WWW (World Wide Web), http, smtp, ftp*, atau layanan data lainnya yang bersifat non *real time* dan tidak memiliki *QoS*. Saat ini kebutuhan akan layanan atau aplikasi berbasis multimedia melewati jaringan IP telah menjadi sesuatu yang mungkin. Pada dasarnya jaringan IP dibuat untuk tidak dilewati data yang bersifat *real time*. Tetapi dengan ditemukannya teknologi penunjang *QoS* jaringan seperti *RTP*, *streaming* via internet, *RSVP*, dan *MPLS* membuat jaringan IP menjadi *reliable* untuk mengirim data yang bersifat *real time* seperti *voice* ataupun video.

Kemajuan–kemajuan inilah yang membuat berbagai layanan multimedia berbasis IP muncul di masyarakat. *VoIP* adalah salah satunya. Teknologi ini melewatkan suara (*speech*) ke dalam jaringan. Dengan teknologi *VoIP*, biaya untuk melakukan telekomunikasi antara satu *user* ke *user* lainnya menjadi lebih efisien. Hal ini disebabkan karena *VoIP* tidak tergantung pada jarak. Sehingga membuat layanan bertelekomunikasi menggunakan PC menjadi lebih murah. Skype, Yahoo Messanger *with Voice* dan masih banyak lagi *provider* layanan *VoIP* menawarkan jasa pelayanan *VoIP* ini.

Berkembangnya layanan *voice* ini bukan berarti bahwa tidak akan ada masalah yang muncul di masa yang akan datang. Salah satu kelemahan jaringan internet adalah bahwa data yang terkirim tidak terjamin kerahasiaannya sehingga siapapun dapat menangkap dan memanipulasi data tersebut. Jika data yang di tangkap ternyata rahasia maka akan menjadi kerugian bagi kita jika data tersebut diketahui orang lain atau bahkan digunakan untuk hal yang dapat merugikan. Oleh sebab itu munculah berbagai macam cara untuk mengamankan paket yang dilewatkan pada suatu jaringan, diantaranya adalah *VPN*, *tunneling* dan lain sebagainya.

Virtual Private Network (VPN) merupakan suatu cara untuk membuat sebuah jaringan yang bersifat *private* dan aman dengan menggunakan jaringan publik misalnya internet. Sebuah jaringan *private* haruslah berada dalam kondisi *VIP*, dan *top secret*. Masalah keamanan data, ketertutupan transfer data dari akses ilegal yang tidak diharapkan serta skalabilitas jaringan menjadi standar utama sebuah *private network*. Pembangunan *private network* secara fisik, akan lebih mahal dari pada pembangunan sebuah *VPN* karena banyaknya perubahan atau penambahan jalur-jalur fisik baru pada sebuah *private network*.

Dalam skripsi ini dianalisis mengenai keamanan aplikasi *VoIP* di jaringan. Seberapa amankah telekomunikasi menggunakan *VoIP*, apakah perlu untuk mengamankan jaringan *VoIP*, dan bagaimana perubahan performansi dari jaringan *VoIP* jika data tersebut kita amankan dengan suatu metode keamanan, serta apakah perubahan tersebut masih sesuai dengan standar *VoIP* yang telah di tetapkan oleh *ITU-T*. Dimana pada skripsi kali ini *VoIP* yang akan diuji diimplementasikan pada jaringan *VPN* berbasis *MPLS* dengan *tunneling IPSec*.

1.2 PERUMUSAN MASALAH

Rumusan masalah pada skripsi ini adalah mengetahui seberapa amankah suatu pembicaraan *VoIP* yang melalui jaringan IP, mengingat sifat dari data *packet switch* yang dapat di-*taping*, dan dilihat isi datanya. Untuk selanjutnya diimplementasikan salah satu metode keamanan pada jaringan yang dibangun yaitu menggunakan *VPN* (*IPSec tunnel*), kemudian dianalisa performansinya apakah masih dapat memenuhi syarat sesuai standar *ITU-T* untuk komunikasi suara melalui jaringan IP serta seberapa besar pengaruhnya terhadap keamanan *VoIP*. Selain itu dengan diterapkannya *IPSec* pada jaringan *MPLS*, dapat diketahui jenis *codec* mana yang paling optimal, mempunyai kinerja paling baik dan paling stabil dilihat dari parameter-parameter jaringan yang terukur dan nilai *MOS* yang diperoleh serta *bandwidth* minimum yang dibutuhkan

1.3 TUJUAN

Penyusunan Skripsi ini dimaksudkan untuk memenuhi salah satu syarat kelulusan pendidikan Strata 1 di Universitas Indonesia Jurusan Teknik Elektro. Sedangkan tujuan dari penyusunan skripsi ini adalah:

1. Mengetahui celah keamanan pada *VoIP* dan unjuk kerja *VoIP* dalam jaringan
2. Mengetahui kualitas suara dan keamanan yang dihasilkan dari konfigurasi *VoIP over VPN* dengan melakukan perekamanan dan di-*playback*
3. Mengetahui bagaimanakah perubahan unjuk kerja dari *VoIP* sebelum dan sesudah diamankan dengan *VPN* dengan menganalisa *delay, packet loss* dan *jitter*
4. Mengetahui *bandwidth* minimum tiap-tiap *codec* yang sesuai untuk aplikasi *VoIP* over *MPLS-VPN*
5. Mengetahui jenis *codec* yang mempunyai performansi terbaik untuk aplikasi *VoIP* pada jaringan *MPLS-VPN* dengan menggunakan *tunneling IPSec*

1.4 BATASAN MASALAH

1. Sistem *VPN* yang digunakan adalah *MPLS-VPN* dengan menggunakan *tunneling IPSec*
2. Paket yang dianalisa adalah paket RTP, paket lain yang tertangkap bersama paket RTP akan dibuang dan tidak masuk perhitungan analisa
3. *Codec* yang digunakan untuk analisa *VoIP* ada 3 buah yakni *GSM*, G711, dan G729, sedangkan untuk analisa *VoIP over VPN* akan digunakan *codec* yang memiliki performansi terbaik diantara ketiganya
4. Parameter *QoS* yang digunakan adalah *delay, jitter,* dan *paket loss*
5. Analisis performansi tidak menentukan *MOS*. Untuk *MOS* diukur dengan menggunakan *software* VQManager
6. *Range Bandwidth* yang digunakan dari 16 Kbps-128 Kbps, terbagi menjadi 4 bagian 16 Kbps, 32 Kbps, 64 Kbps, dan 128 Kbps
7. Menggunakan *IPv4* sebagai pengalamatannya.

1.5 METODOLOGI

1. Studi literatur

   Mengumpulkan dan mempelajari referensi tentang jaringan *MPLS-VPN*, *IPSec*, *VoIP*, *software* Trixbox, VQManager dan EyeBeam

2. Perancangan system

   Pada skripsi ini dirancang sistem *VoIP* pada *VPN* dengan menggabungkan teknologi *IPSec* dan *MPLS*.

3. Implementasi sistem

   Implementasi dilakukan dengan serangkaian router yang dibangun dengan skenario "*Site to site VPN Business*", terdiri atas 3 buah router cisco tipe 2600 dan 2 buah router cisco tipe 800 yang membentuk jaringan *backbone MPLS* disisi penyedia jasa dan *IPSec tunnel* yang menghubungkan router *head office* dan router *branch office*.

4. Pengambilan dan analisa data

   Setelah dilakukan implementasi, akan dicatat data-data yang berhubungan dengan keamanan data, seperti format paket data sebelum dan sesudah enkripsi, dan format paket data pada saat melintasi jaringan *backbone MPLS*. Kemudian parameter *QoS* yang diukur meliputi *: delay, jitter,* dan *packet loss*. Pengambilan data tersebut memanfaatkan *software* Wireshark, ditambah dengan VQManager untuk memonitor *MOS* dari *VoIP*. Untuk pengujian dari sisi keamanan digunakan *software* Wireshark untuk merekam pembicaraan

5. Penarikan kesimpulan

   Selanjutnya dari hasil analisa tersebut akan ditarik kesimpulan mengenai *codec* yang paling optimal untuk aplikasi *VoIP* pada jaringan *VPN* yang berbasis *MPLS* dengan menggunakan *tunneling IPSec*, selain itu dapat diketahui pula *bandwidth* minimum yang dibutuhkan untuk komunikasi *VoIP*

6. Penulisan buku laporan

   Dalam penulisan laporan ini mengacu pada pedoman penulisan ilmiah, dalam hal ini penulisan skripsi yang bentuk bakunya telah diatur oleh pihak Universitas Indonesia.

1.6 SISTEMATIKA PEMBAHASAN

Sistematika penulisan skripsi ini adalah sebagai berikut :

Bab 1 Pendahuluan

Bab ini berisi penjelasan singkat mengenai Latar Belakang, Tujuan Penulisan, Batasan Masalah, dan Sistematika Penulisan.

Bab 2 *Voice over Internet Protocol (VoIP)* dan *Virtual Private Network(VPN)*

Bab ini berisi penjelasan mengenai konsep *VoIP*, *VPN*, *IPSec* dan *MPLS*

Bab 3 Perancangan dan Implementasi *VoIP* pada Jaringan *VPN*

Bab ini berisi penjelasan mengenai perancangan simulasi *VoIP* pada jaringan *VPN* berbasis *MPLS* dengan menggunakan *tunneling IPSec*. Simulasi tersebut dibangun dengan skenario "*Site to site VPN Business*".

Bab 4 Pengujian dan Analisa Sistem

Bab ini berisi proses pengujian dan analisa data hasil simulasi *VoIP* pada jaringan *VPN* berbasis *MPLS* dengan menggunakan *tunneling IPSec* sesuai dengan skenario yang telah dibuat.

Bab 5 Kesimpulan

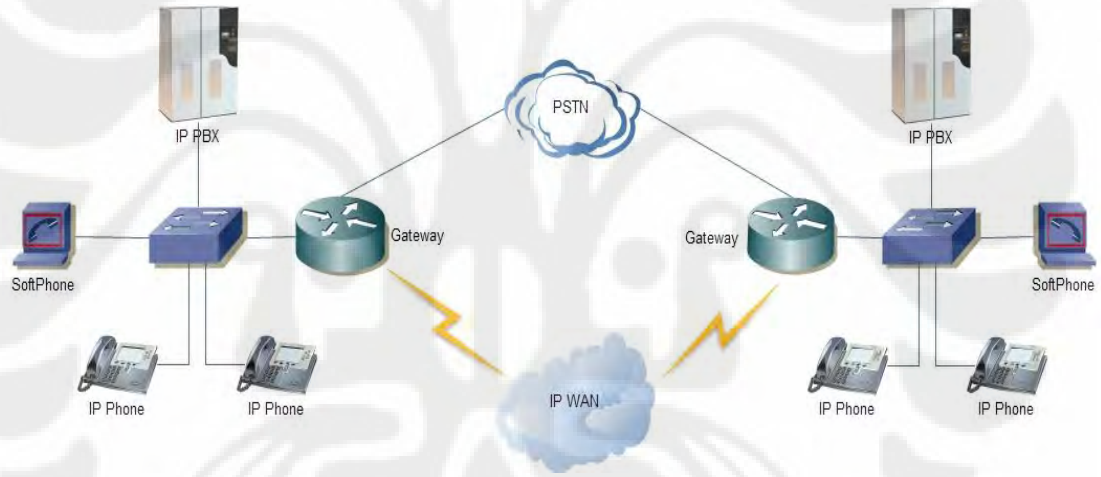
Bab ini berisi Kesimpulan dari hasil pengujian yang sudah dilakukan.

BAB 2

VOICE OVER INTERNET PROTOCOL (VOIP) DAN *VIRTUAL PRIVATE NETWORK (VPN)*

2.1 *Voice Over Internet Protocol (VoIP)*

Voice over Internet Protocol (VoIP) dikenal juga dengan sebutan *IP Telephony* didefinisikan sebagai suatu sistem yang menggunakan jaringan internet untuk mengirimkan paket data suara dari suatu tempat ke tempat lainnya menggunakan perantara protokol IP [3]. Dengan kata lain teknologi ini mampu melewatkan trafik suara yang berbentuk paket melalui jaringan IP. Jaringan IP sendiri adalah merupakan jaringan komunikasi data yang berbasis *packet-switch*.



Gambar 2.1 Diagram *VoIP* [2]

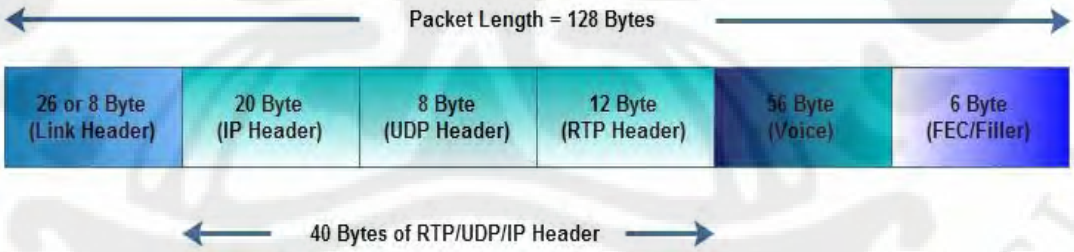
Pada awalnya, perkembangan *VoIP* hanya dapat dipakai antar PC multimedia dengan kualitas rendah. Sesuai dengan perkembangan teknologi, kini *VoIP* memungkinan komunikasi antar PC ke telepon dan komunikasi antar telepon dengan kualitas yang baik sehingga layanan *VoIP* mulai banyak dijual oleh operator-operator telekomunikasi di dunia. Oleh karena itu jaringan IP harus didesain agar memenuhi persyaratan *delay* dan *packet loss*. *Packet loss* (kehilangan paket data pada proses transmisi) dan *delay* bukan hanya merupakan masalah yang berhubungan dengan kebutuhan *bandwidth*, namun lebih dipengaruhi oleh stabilitas rute yang dilewati data pada jaringan, metode antrian

yang efisien, pengaturan pada router, dan penggunaan kontrol terhadap kongesti (kelebihan beban data) pada jaringan.

Packet loss terjadi ketika terdapat penumpukan data pada jalur yang dilewati. Hal ini mendorong agar arsitektur *VoIP* menyediakan infrastruktur yang memiliki kemampuan dan fitur seperti halnya *Signaling System No 7* (*SS7)* di *PSTN*. Panggilan *VoIP* memiliki dua jenis komunikasi yang menempati jaringan IP antara pemanggil (*calling party*) dan pihak yang dipanggil (*called party*), yaitu aliran informasi pembicaraan dan *message-message signaling* yang mengontrol hubungan dan karateristik aliran media. Untuk membawa informasi digunakan *Real-time Transport Protocol* (*RTP*). Sedangkan untuk pensinyalan terdapat dua standar yang dikeluarkan oleh dua badan dunia, yaitu H.323 yang dikembangkan oleh *ITU-T* dan *Session Intitation Protocol* (*SIP*) oleh *Internet Engineering Task Force* (*IETF*)

2.2 Format Paket *VoIP*

Tiap paket *VoIP* terdiri dari dua bagian, yakni *header* dan *payload* (beban). *Header* terdiri atas *IP Header*, *Real-time Transport Protocol* (*RTP*), *User Datagram Protocol* (*UDP*) *header*, dan *link header*. Format paket *VoIP* dapat dilihat pada Gambar 2.2 di bawah



Gambar 2.2 Format Paket *VoIP* [3]

IP header bertugas menyimpan informasi *routing* untuk mengirimkan paket-paket ke tujuan. Pada tiap *header* IP disertakan tipe layanan atau *Type of Service* (*ToS*) yang memungkinkan paket tertentu seperti paket suara yang *non real-time*.

UDP header memiliki ciri tertentu yaitu tidak menjamin paket akan mencapai tujuan sehingga *UDP* cocok digunakan pada aplikasi *voice real-time* yang sangat peka terhadap *delay* dan *latency*.

RTP header adalah *header* yang dapat dimanfaatkan untuk melakukan *framing*, dan segmentasi data *real-time*. Seperti *UDP*, *RTP* juga tidak mendukung realibilitas paket untuk sampai ke tujuan. *RTP* menggunakan protokol kendali yang disebut *Real-time Transport Control Protocol* (*RTCP*) yang mengendalikan *QoS* dan sinkronisasi media stream yang berbeda. Untuk *link header*, besarnya sangat tergantung pada media yang digunakan. Tabel 2.1 menunjukkan perbedaan ukuran *header* untuk media yang berbeda.

Tabel 2.1 *Link Layer Header Size* [3]

Media	Link Layer Header Size	Bit Rate
Ethernet	14 byte	29.6 kbps
PPP	6 byte	26.4 kbps
Frame Relay	4 byte	25.6 kbps
ATM	5 byte tiap cell	42.4 kbps

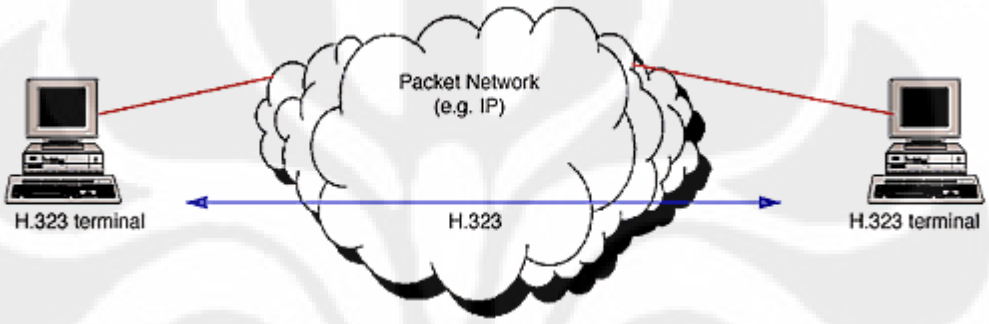
2.3 Protokol *Signaling* dalam Jaringan *VoIP*

Protokol *signaling* dalam *VoIP* diperlukan agar pemakai layanan *VoIP* dapat saling berkomunikasi dengan pesawat telepon. Beberapa protokol *signaling* yang ada saat ini adalah *H.323, SIP, SCCP, MGCP, MEGACO* dan *SIGTRAN*. Tetapi yang paling populer dan banyak digunakan adalah H.323 dan *SIP*. H.323 merupakan teknologi yang dikembangkan oleh *International Telecommunication Union* (*ITU-T*) sedangkan *Session Initiation Protocol* (*SIP*) merupakan teknologi yang dikembangkan *Internet Engineering Task Force* (*IETF*).

2.3.1 H.323

VoIP dapat berkomunikasi dengan sistem lain yang beroperasi pada jaringan *packet-switch*. Untuk dapat berkomunikasi dibutuhkan suatu standar sistem komunikasi yang kompatibel satu sama lain. Salah satu standar komunikasi

pada *VoIP* menurut rekomendasi *ITU-T* adalah H.323 (1995 – 1996). Standar H.323 terdiri dari komponen, protokol, dan prosedur yang menyediakan komunikasi multimedia melalui jaringan *packet-based*. Bentuk jaringan *packet - based* yang dapat dilalui antara lain jaringan internet, *Internet Packet Exchange (IPX) based*, *Local Area Network* (*LAN*), dan *Wide Area Network* (*WAN*). H.323 dapat digunakan untuk layanan-layanan multimedia seperti komunikasi suara (*IP Telephony*), komunikasi video dengan suara (*Video Telephony*), dan gabungan suara, video, dan data.



Gambar 2.3 Terminal pada jaringan paket [2]

Tujuan desain dan pengembangan H.323 adalah untuk memungkinkan interoperabilitas dengan tipe terminal multimedia lainnya. Terminal dengan standar H.323 dapat berkomunikasi dengan terminal H.320 pada N-ISDN, terminal H.321 pada ATM, dan terminal H.324 pada *Public Switched Telephone Network* (PSTN). Terminal H.323 memungkinkan komunikasi *real time* dua arah berupa suara, video, dan data.

Arsitektur H.323

Standar H.323 terdiri dari 4 komponen fisik yang digunakan saat menghubungkan komunikasi multimedia *point-to-point* dan *point-to-multipoint* pada beberapa macam jaringan :

A. Terminal

Digunakan untuk komunikasi multimedia *real time* dua arah. Terminal H.323 dapat berupa *personal computer* (*PC*) atau alat lain yang berdiri sendiri yang dapat menjalankan aplikasi multimedia

B. *Gateway*

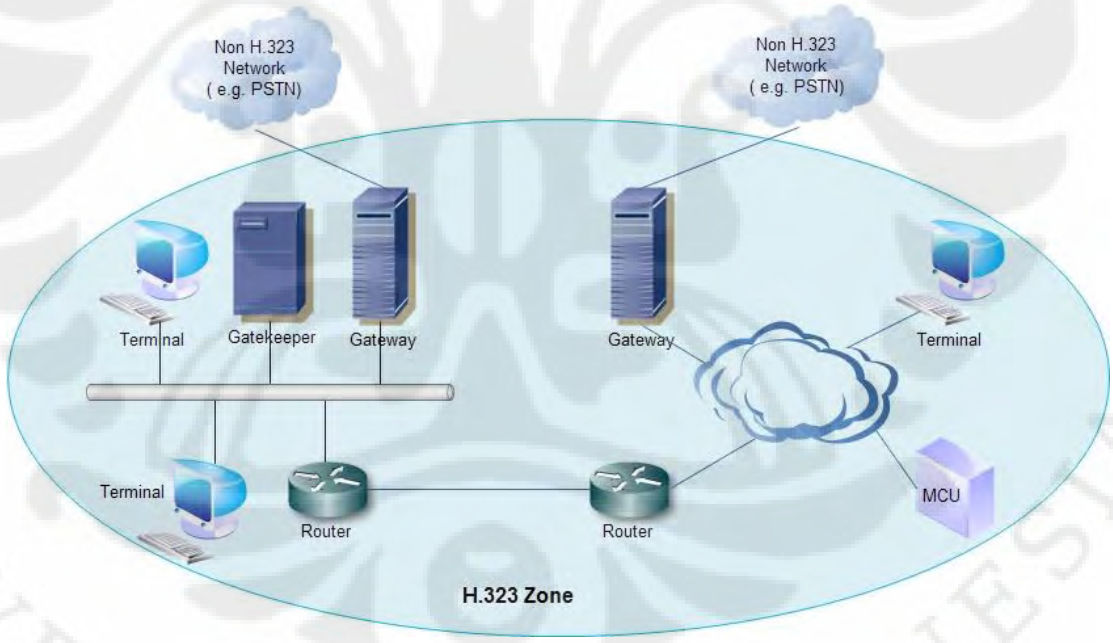
Digunakan untuk menghubungkan dua jaringan yang berbeda yaitu antara jaringan H.323 dan jaringan *non* H.323, sebagai contoh *gateway* dapat menghubungkan dan menyediakan komunikasi antara terminal H.323 dengan jaringan telepon, misalnya *PSTN*.

C. *Gatekeeper*

Dapat dianggap sebagai otak pada jaringan H.323 karena merupakan titik yang penting pada jaringan H.323

D. *Multipoint Control Unit (MCU)*

Digunakan untuk layanan konferensi tiga terminal H.323 atau lebih. Semua terminal yang ingin berpartisipasi dalam konferensi dapat membangun hubungan dengan *MCU* yang mengatur bahan-bahan untuk konferensi, negoisasi antara terminal-terminal untuk memastikan audio atau video *coder/decoder* (*CODEC*).



Gambar 2.4 Arsitektur H.323 [2]

2.3.2 *Session Initiation Protokol* (*SIP*)

Session Initiation Protocol atau *SIP* merupakan standar *IETF* untuk suara atau layanan multimedia melalui jaringan internet. *SIP* [RFC 2543] diajukan pada tahun 1999. Pencipta standar ini adalah Henning Schulzrinne. *SIP* merupakan

protokol layer aplikasi yang digunakan untuk manajemen pengaturan panggilan dan pemutusan panggilan. *SIP* digunakan bersamaan dengan protokol *IETF* lain seperti *SAP, SDP, MGCP (MEGACO)* untuk menyediakan layanan *VoIP* yang lebih luas.

Arsitektur *SIP* mirip dengan arsitektur *HTTP* (*protocol client-server*). Arsitekturnya terdiri dari *request* yang dikirim dari *user SIP* ke *server SIP*. *Server* itu memproses *request* yang masuk dan memberikan respon kepada *client*. Permintaan *request* itu, bersama dengan komponen respon pesan yang lain membuat suatu komunikasi *SIP*.

Komponen *SIP*

Arsitektur *SIP* terdiri dari dua buah komponen seperti di bawah ini :

- *User Agent*

  SIP User Agent merupakan akhir dari sistem (terminal akhir) yang bertindak berdasarkan kehendak dari pemakai. Terdiri dari dua bagian yaitu :

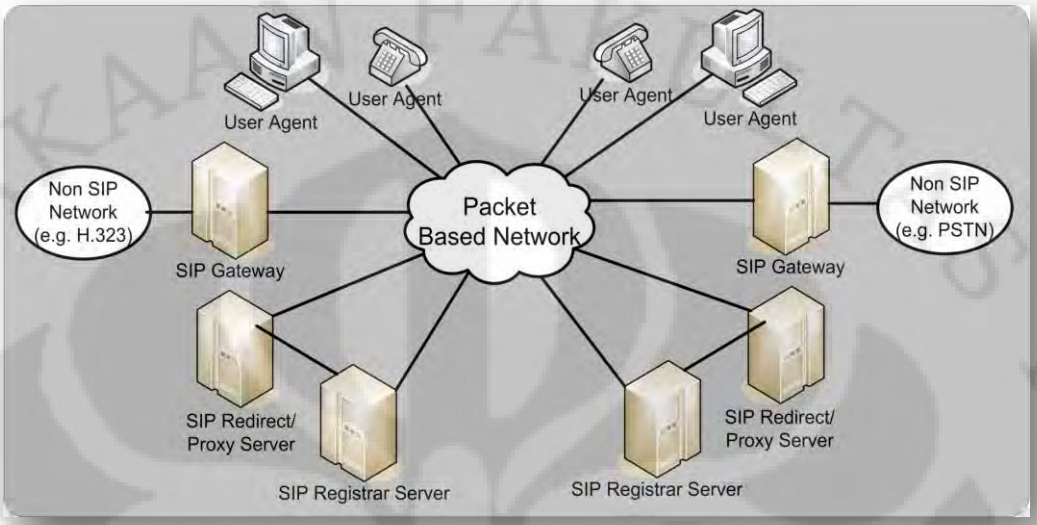
  - *User Agent Client* (*UAC*) : bagian ini terdapat pada pemakai (*client*) yang digunakan untuk melakukan inisiasi *request* dari *server SIP* ke *UAS*
  - *User Agent Server* (*UAS*) : bagian ini berfungsi untuk mendengar dan merespon terhadap *request SIP*

- *SIP Server*

  Arsitektur *SIP* sendiri menjelaskan jenis-jenis *server* pada jaringan untuk membantu layanan dan pengaturan panggilan *SIP*.

  - *Registration Server* : *server* ini menerima *request* dari *user SIP* dan melakukan *update* terhadap lokasi *user* dengan *server* ini
  - *Proxy Server* : *server* ini menerima *request SIP* dan meneruskan ke *server* yang dituju yang memiliki informasi tentang *user* yang dipanggil
  - *Redirect Server* : *server* ini setelah menerima *request SIP*, menentukan *server* yang dituju selanjutnya dan mengembalikan

alamat *server* yang dituju selanjutnya kepada *client* untuk kemudian meneruskan *request* ke *server* yang di tuju tersebut



Gambar 2.5 Operasi *SIP* [18]

2.4 Standar Kompresi Data Suara

Sebuah *codec* (*compressor/decompressor* atau *coder/decoder*) adalah suatu *hardware* atau *software* yang melakukan *sampling* terhadap sinyal suara analog, kemudian mengkonversi ke dalam bit-bit digital dan mengeluarkannya. Beberapa jenis *codec* melakukan kompresi agar dapat menghemat *bandwidth*. *International Telecommunication Union-Telecommunication Sector* (*ITU-T*) membuat beberapa standar untuk *voice coding* yang direkomendasikan untuk implementasi *VoIP*. Beberapa standar yang sering dikenal antara lain :

2.4.1 *Codec* G.711

Sebelum mengetahui lebih jauh apa itu G.711, akan diberikan sedikit gambaran singkat fungsi dari kompresi. Sebuah kanal video yang baik tanpa dikompresi akan mengambil *bandwidth* sekitar 9 Mbps. Sedangkan sebuah kanal suara (*audio*) yang baik tanpa dikompresi akan mengambil *bandwidth* sekitar 64 Kbps. Dengan adanya teknik kompresi, kita dapat menghemat sebuah kanal video menjadi sekitar 30 Kbps dan kanal suara menjadi 6 Kbps (*half-duplex*), artinya

sebuah saluran internet yang tidak terlalu cepat sebetulnya dapat digunakan untuk menyalurkan video dan *audio* sekaligus. Tentunya untuk kebutuhan konferensi dua arah dibutuhkan *double bandwidth*, artinya minimal harus menggunakan kanal 64 Kbps ke internet. Dengan begitu suara/*audio* akan membutuhkan *bandwidth* jauh lebih sedikit dibanding pengiriman gambar / video

G.711 adalah suatu standar internasional untuk kompresi *audio* dengan menggunakan teknik *Pulse Code Modulation* (*PCM*) dalam pengiriman suara. Standar ini banyak digunakan oleh operator Telekomunikasi di seluruh dunia.

PCM mengkonversikan sinyal analog ke bentuk digital dengan melakukan *sampling* sinyal analog tersebut 8000 kali/detik dan dikodekan dalam kode angka. Jarak antar sampel adalah 125 μdetik. Sinyal analog pada suatu percakapan diasumsikan berfrekuensi 300 Hz – 3400 Hz. Sinyal tersampel lalu dikonversikan ke bentuk diskrit. Sinyal diskrit ini direpresentasikan dengan kode yang disesuaikan dengan amplitudo dari sinyal sampel. Format *PCM* menggunakan 8 bit untuk pengkodeannya. Laju transmisi diperoleh dengan mengkalikan 8000 sample/detik dengan 8 bit/sample, menghasilkan 64.000 bit/detik. *Bit-rate* 64 Kbps ini merupakan standar transmisi untuk satu kanal telepon digital.

2.4.2 *Codec* G.729

Codec ini adalah salah satu *codec* berkualitas lebih baik. G.729 merupakan pengkodean suara jenis *CELP* dengan hasil kompresi pada 8 Kbps. Terdapat 2 versi yaitu G.729 dan G.729a. G.729a memiliki algoritma yang lebih sederhana dan membutuhkan *processing power* lebih sedikit dibandingkan G.729.

2.4.3 *Codec GSM*

GSM-AMR merupakan jenis *codec Adaptive Multi-Rate* yang telah di standarisasi untuk digunakan pada telepon selular generasi ke 3. Codec jenis ini mendukung 8 jenis *bit-rate* yaitu : 12.2, 10.2, 7.95, 7.40, 6.7, 5.9, 5.15 dan 4.75 kbps. *GSM* menggunakan metode kompresi *Algebraic Code Excited Linear Prediction (ACELP)*. *GSM* dikembangkan untuk menjaga kualitas suara pada berbagai kondisi transmisi.

Tabel. 2.2 Perbandingan teknik – teknik kompresi standar *ITU –T* [9]

Teknik Kompresi	*Bandwidth* (Kbps)	*Frame Size* (mdetik)	*MOS*
G.711 PCM	64	0.125	4.1
G.726 ADPCM	32	0.125	3.85
G.728 LD – CELP	16	0.625	3.61
G.729 CS - ACELP	8	10	3.92
G.729A CS - ACELP	8	10	3.7
G.723.1 MPMLQ	6.3	30	3.9
G.723.1 ACELP	5.3	30	3.65

2.5 Kualitas Layanan *VoIP*

Quality of Service (*QoS*) adalah kemampuan suatu jaringan untuk menyediakan layanan yang lebih baik pada trafik data tertentu untuk berbagai jenis *platform* teknologi. *QoS* tidak diperoleh langsung dari infrastruktur yang ada, melainkan diperoleh langsung dengan mengimplementasikan pada jaringan bersangkutan [3].

QoS pada *IP Telephony* adalah parameter-parameter yang menunjukkan kualitas paket data jaringan, agar didapatkan hasil suara sama dengan menggunakan telepon tradisional (*PSTN*). Beberapa parameter yang mempengaruhi *QoS* antara lain *latency* (keterlambatan data), *delay, jitter, packet loss* dan *sequence error* pada jaringan internet. Selain itu *QoS* juga dipengaruhi oleh pemenuhan kebutuhan *bandwidth*, jenis kompresi data, interoperabilitas peralatan (vendor yang berbeda) dan jenis standar multimedia yang digunakan (H.323/*SIP*/*MGCP*).

2.5.1 *Delay*

Delay merupakan faktor yang penting dalam menentukan kualitas *VoIP*. Semakin besar *delay* yang terjadi maka akan semakin rendah kualitas *VoIP* yang dihasilkan. Adapun beberapa sumber *delay* antara lain :

- *Delay* Algoritmik

  Merupakan *delay* yang dihasilkan oleh *codec* dalam melakukan pengkodean.

- *Delay* Paketisasi

  Delay paketisasi merupakan *delay* yang terjadi antara satu paket *RTP* dengan paket lainnya. RFC 1890 menspesifikasikan bahwa *delay* paketisasi sebaiknya 20 ms.

- *Delay* Serialisasi

  Merupakan waktu yang dibutuhkan untuk mengirimkan paket dari pengirim. *Delay* serialisasi juga terjadi pada router atau switch

- *Delay* Propagasi

  Merupakan waktu yang diperlukan untuk sinyal listrik atau sinyal optik untuk melewati media transmisi dari pengirim ke penerima. *Delay* ini tergantung dari media transmisi dan jarak yang harus ditempuh sinyal.

- *Delay* Komponen

  Merupakan *delay* yang terjadi yang disebabkan proses yang terjadi pada komponen contohnya router. Dalam router terdapat *delay* yang terjadi jika paket masuk dan diproses kemudian dikeluarkan ke *port output*.

Agar tidak terjadi suara *overlap* dalam komunikasi *VoIP*, G114 menyediakan pedoman untuk batasan *delay* satu arah :

0 s/d 150 ms	*delay* ini masih dapat diterima
150 s/d 400 ms	masih dapat diterima jika *administrator* waspada terhadap waktu yang dapat mempengaruhi kualitas transmisi
Di atas 400 ms	Tidak dapat diterima untuk layanan *VoIP*

2.5.2 *Jitter*

Jitter merupakan variasi *delay* yang terjadi akibat adanya selisih waktu atau *interval* antar kedatangan paket di penerima. Parameter ini dapat ditangani dengan mengatur metode antrian pada router saat terjadi kongesti atau saat perubahan kecepatan. Paket data yang datang dikumpulkan dulu dalam *jitter buffer* selama waktu yang telah ditentukan sampai paket dapat diterima pada sisi penerima dengan urutan yang benar. Hanya saja *jitter* tidak mungkin dihilangkan sebab metode antrian yang paling baik tetap saja tidak dapat mengatasi semua kasus antrian. Untuk meminimalisasi *jitter* ini, diusahakan agar pengiriman tiap-

tiap paket data melalui jalur yang sama dan jangan sampai terjadi *paket loss* atau kongesti jaringan.

2.5.3 *Packet Loss*

Loss packet (kehilangan paket data pada proses transmisi) terjadi ketika terdapat penumpukan data pada jalur yang dilewati pada saat beban puncak (*peak load*) yang menyebabkan kemacetan transmisi paket akibat padatnya trafik yang harus dilayani dalam batas waktu tertentu. Sehingga *frame* (gabungan data *payload* dan *header* yang di transmisikan) akan dibuang sebagaimana perlakuan terhadap *frame* data lainnya pada jaringan berbasis IP. Paket akan di-*drop* di bawah beban puncak dan selama periode kongesti yang disebabkan oleh beberapa faktor seperti kegagalan *link* transmisi atau kapasitas yang tidak mencukupi. Salah satu alternatif solusi permasalahan di atas adalah dengan membangun *link* antar *node* pada jaringan *VoIP* dengan spesifikasi dan dimensi dengan *QoS* yang baik dan dapat mengantisipasi perubahan lonjakan trafik hingga pada suatu batas tertentu.

Packet loss pada jaringan untuk *IP Telephony* sangat besar pengaruhnya, di mana bila terjadi *packet loss* dalam jumlah tertentu, akan menyebabkan terjadi interkoneksi *TCP* melambat (terlalu banyak pengulangan proses *handshake*). Biasanya *packet loss* sebesar 10 % tidak bisa ditolerir [3]

2.6 Metode Pengukuran Kualitas *VoIP*

Untuk menentukan kualitas layanan suara dalam jaringan IP dapat digunakan beberapa metode di bawah ini :

2.6.1 *Mean Opinion Score (MOS)*

Metode ini merupakan metode yang digunakan untuk menentukan kualitas suara dalam jaringan IP yang berdasarkan standar *ITU-T* P.800. Metode ini bersifat subjektif, karena berdasarkan pendapat orang perorang. Untuk menentukan nilai *MOS* terdapat dua cara pengetesan yaitu, *conversation opinion test* dan *listening test*. Rekomendasi nilai *ITU-T* P.800 untuk nilai *MOS* adalah seperti pada Tabel 2.3 di bawah.

Tabel 2.3 Penilaian *MOS* terhadap kualitas layanan *VoIP* [4]

Nilai *MOS*	*Opinion Score* untuk tes pembicaraan	*Opinion Score* untuk tes pendengaran
5	*Excellent*	*Excellent* (Jauh lebih keras daripada yang diisyaratkan)
4	*Good*	*Good* (Lebih keras daripada yang diisyaratkan)
3	*Fair*	*Fair* (Kejelasan suara yang diisyaratkan)
2	*Poor*	*Poor* (Lebih pelan dari yang diisyaratkan)
1	*Bad*	*Bad* (Jauh lebih pelan dari yang diisyaratkan)

Metode *MOS* dirasakan kurang efektif untuk mengestimasi kualitas layanan suara untuk *VoIP*, hal ini dikarenakan :

- Tidak terdapatnya nilai yang pasti terhadap parameter yang mempengaruhi kualitas layanan suara dalam *VoIP*
- Setiap orang memiliki standar yang berbeda-beda terhadap suara yang mereka dengar dengan hanya melalui percakapan.

2.6.2 Estimasi *MOS* dengan Metode *E-Model* (*ITU-T* G.107)

Di dalam jaringan *VoIP*, tingkat penurunan kualitas yang diakibatkan oleh transmisi data memegang peranan penting terhadap kualitas suara yang dihasilkan, hal yang menjadi penyebab penurunan kualitas suara ini diantaranya adalah *delay, paket loss*, dan *echo*. Pendekatan matematis yang digunakan untuk menentukan kualitas suara berdasarkan penyebab menurunnya kualitas suara dalam jaringan *VoIP* dimodelkan dengan *E-Model* yang distandarkan kepada *ITU-T* G.107.

Nilai akhir estimasi *E-Model* disebut dengan *R.Factor*. *R.Factor* didefinisikan sebagai faktor kualitas transmisi yang dipengaruhi oleh beberapa parameter seperti *signal to noise ratio* dan *echo* perangkat, *codec* dan kompresi, *packet loss*, dan *delay*.

R.Factor ini didefinisikan sebagai berikut :

$$R = 94.2 - Id - Ie \quad [5]$$

Dimana :

Id = Faktor penurunan kualitas yang disebabkan oleh pengaruh *delay* satu arah

Ie = Faktor penurunan kualitas yang disebabkan oleh teknik kompresi dan *packet loss* yang terjadi

Nilai Id ditentukan dari persamaan di bawah ini

$$Id = 0.024d + 0.11(d-177.3)\,H(d-177.3)$$ [5]

Nilai Ie tergantung pada metode kompresi yang digunakan. Nilai *R.Factor* secara keseluruhan dihitung dari persamaan di bawah

$$R = 94.2 - [0.024d + 0.11(d-177.3)\,H(d-177.3)] - Ie$$ [5]

Dimana :

R = Faktor kualitas transmisi

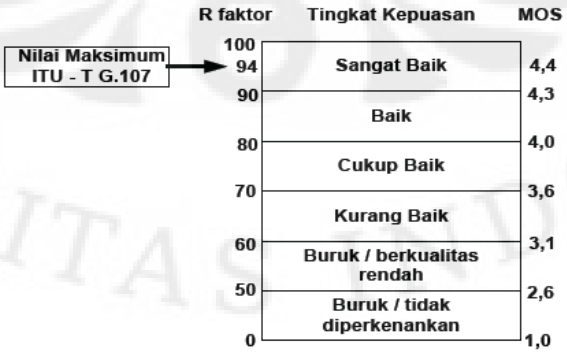
d = *Delay* satu arah (ms)

H = fungsi tangga; dengan ketentuan

H(x) = 0 jika x<0, lainnya

H(x) = 1 untuk x>=0

Nilai *R.Factor* mengacu pada standar *MOS*, hubungannya dapat dilihat pada Gambar 2.6 di bawah ini



Gambar 2.6 Korelasi antara *E-Model* (*ITU-T* G.107) dengan *MOS* (*ITU-T* P.800) [5]

2.7 Celah Keamanan *VoIP*

Sistem *VoIP* memiliki beberapa potongan informasi yang harus di proteksi. Percakapan itu sendiri, *voice mail*, rekaman aktivitas telepon, dan nomor telepon adalah beberapa contoh dari informasi yang harus dapat dirahasiakan. Tetapi bagaimanapun juga ada masalah lain yang jauh lebih besar yang ada di dalam keamanan layanan *VoIP*, yaitu informasi telepon yang seharusnya rahasia, *private* dan penting berada dalam jaringan internet berupa paket-paket data *VoIP*.

Dalam sistem *VoIP*, data suara dikirimkan dari pengirim ke penerima menggunakan protokol *RTP*. *Header* dari paket *RTP* memiliki standar format tertentu, di mana semua orang tahu bagaimana *payload* dapat di-*encoding* dengan hanya melihat isi dari *RTP payload*. *VoIP payload* menggunakan standar *codec* seperti G.711 dan G.729. Paket *RTP* dapat ditangkap, direkonstruksi dan di-*playback*.

Mungkin area dari keamanan *VoIP* dan internet yang paling kurang diperhatikan adalah bahwa faktanya para penyerang melakukan berbagai cara untuk merusak sistem yang kita miliki dengan berbagai cara antara lain :

2.7.1 *Denying Service*

Suatu cara untuk melakukan kerusakan pada *service VoIP* adalah dengan melakukan *Denial of Service* (*DoS*) ke komputer *server VoIP*. Serangan *DoS* biasanya merupakan *request* ke suatu *server* dalam jumlah besar baik dari satu komputer atau beberapa buah komputer secara bersamaan. Hal ini dapat menyebabkan *server* akan kesusahan untuk menangani *request* yang masuk. Jika *server* tidak dapat lagi menangani *request* maka *server* akan *crash/down* dan servis yang dijalankan tidak akan dapat digunakan lagi.

2.7.2 *Call Hijacking*

Hijacking (Pembajakan) berarti seseorang mengatur sistem yang memungkinkan pelaku untuk dapat memanipulasi layanan tersebut. Dalam layanan *VoIP*, hal ini dapat menyebabkan celah keamanan yang cukup beresiko.

2.8. *VPN*

Virtual Private Network (*VPN*) merupakan sebuah jaringan *private* yang menghubungkan satu *node* jaringan ke *node* jaringan lainnya dengan menggunakan jaringan publik (internet). Data yang dilewatkan akan dibungkus (*encapsulation*) dan dienkripsi agar terjamin kerahasiaannya.

Jaringan *VPN* dikoneksikan oleh penyedia jasa komunikasi (*Service Provider*) melalui routernya ke router-router lain dengan menggunakan jalur internet yang telah dienkripsi diantara dua titik.

Sistem keamanan di *VPN* menggunakan beberapa lapisan, yaitu :

a) Metode *tunneling* (terowongan)

   Membuat terowongan *virtual* di atas jaringan publik menggunakan protokol seperti *Point to Point Protocol* (*PPTP*), *Layer 2 Tunneling Protocol* (*L2TP*), *Generic Routing Encapsulation* (*GRE*) atau *IPSec*. *PPTP* dan *L2TP* adalah *Layer* 2 *Tunneling Protocol*, keduanya melakukan pembungkusan *payload* pada *frame PPTP* untuk dilewatkan pada jaringan. Sedangkan *IPSec* berada di layer 3, menggunakan paket, dan akan melakukan pembungkusan *IP header* sebelum dikirim ke jaringan.

b) Metode enkripsi

   Untuk membungkus paket data yang lewat di dalam *tunneling*, data yang dilewatkan pada pembungkusan tersebut akan dirubah dengan algoritma kriptografi tertentu seperti *DES, 3DES* dan *AES*.

c) Metode autentikasi *user*

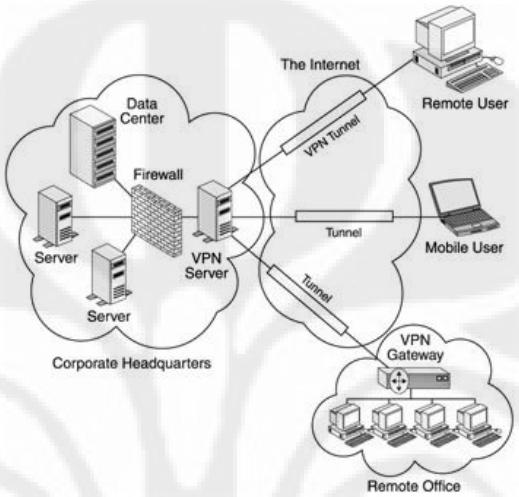
   Karena banyak *user* yang akan mengakses biasanya digunakan beberapa metode autentikasi *user* seperti *Remote Acces Dial in user Services* (*RADIUS*) dan *Digital Certificates.*

d) Integritas data

   Paket data yang dilewatkan pada jaringan publik perlu penjaminan integritas (keutuhan) data, apakah terjadi perubahan atau tidak. Metode *VPN* menggunakan *HMA C-MD5* atau *HMA C-SHA1* agar paket data tidak berubah pada saat pengiriman.

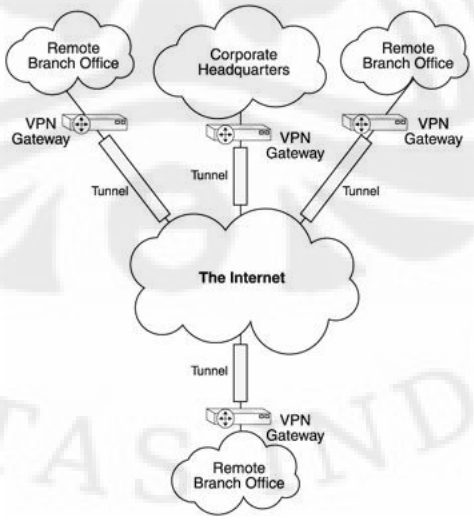
Ada 3 jenis implementasi jaringan *VPN*, antara lain :

- *Remote acces VPN* - menyediakan *remote acces* ke jaringan *intranet* atau *extranet* perusahaan yang memiliki kebijakan yang sama sebagai jaringan *private*. *Acces VPN* memungkinkan *user* untuk dapat mengakses data perusahaan kapanpun dan di manapun mereka inginkan.



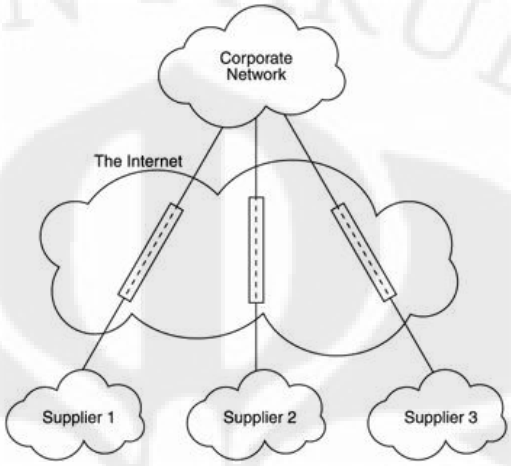
Gambar 2.7 *Remote acces VPN* [20]

- *Intranet VPN* – menghubungkan antara kantor pusat suatu perusahaan dengan kantor cabang, kantor pembantu melalui *shared network* menggunakan koneksi yang permanen (*dedicated*).



Gambar 2.8 *Intranet VPN* [20]

- *Extranet VPN* – menghubungkan konsumen, *suppliers*, mitra bisnis, dan beberapa komunitas dengan kepentingan yang sama ke jaringan intranet perusahaan melalui infrastruktur yang terbagi menggunakan koneksi *dedicated*.



Gambar 2.9 *Extranet VPN* [20]

2.9 *MPLS*

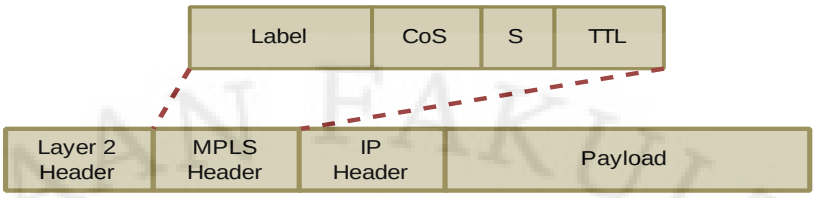
Multiprotocol Label Switching (*MPLS*) adalah teknologi penyampaian paket pada jaringan *backbone* (jaringan utama) berkecepatan tinggi yang menggabungkan beberapa kelebihan dari sistem komunikasi *circuit-switched* dan *packet-switched* yang melahirkan teknologi yang lebih baik dari keduanya.

MPLS adalah arsitektur jaringan yang didefinisikan oleh *IETF* untuk memadukan mekanisme *label swapping* di layer 2 dengan *routing* di *layer* 3 untuk mempercepat pengiriman paket.

Paket-paket pada *MPLS* diteruskan dengan protokol *routing* seperti *OSPF, BGP* atau *EGP*, Protokol *routing* pada layer 3 sistem *OSI*, sedangkan *MPLS* berada diantara layer 2 dan 3.

2.9.1 *MPLS Header*

MPLS bekerja pada paket dengan *MPLS header* yang berisi satu atau lebih label yang disebut dengan label *stack*. *MPLS Header* dapat dilihat pada Gambar 2.10.



Gambar 2.10*MPLS Header* [21]

MPLS header meliputi ;

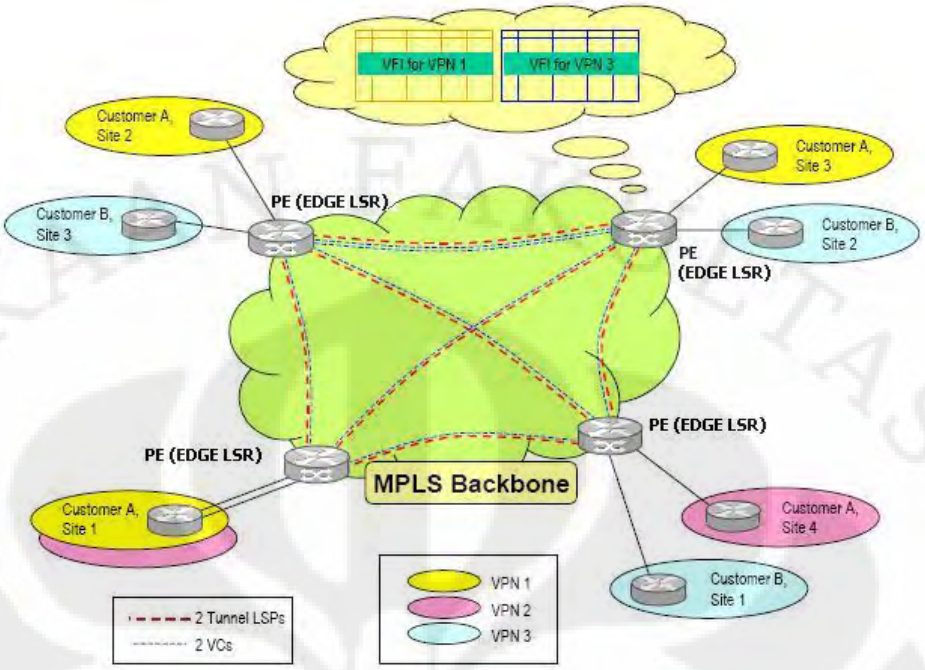
- 20-bit label *value*, suatu bidang label yang berisi nilai yang nyata dari *MPLS* label.
- 3-bit *field CoS*, suatu bidang *CoS* yang dapat digunakan untuk mempengaruhi antrian paket data dan algoritma paket data yang tidak diperlukan.
- 1-bit *bottom of stack flag*, jika 1-bit diset, maka ini menandakan label yang sekarang adalah label yang terakhir. Suatu bidang yang mendukung hirarki label *stack*.
- 8-bit *Time to Live* (*TTL*) *field*. Untuk 8 bit data yang bekerja

2.9.2 Struktur Jaringan *MPLS*

Struktur jaringan *MPLS* terdiri dari *Edge Label Switching Routers* atau *Edge LSRs* yang mengelilingi sebuah *Core Label Switching Routers* (*LSRs*). Adapun elemen-elemen dasar penyusun jaringan *MPLS* adalah :

- *Edge Label Switching Routers* (*ELSR*)

  ELSR ini terletak pada perbatasan jaringan *MPLS*, berfungsi untuk mengaplikasikan label ke dalam paket-paket yang masuk ke dalam jaringan *MPLS*. Sebuah *MPLS Edge Router* akan menganalisa *header* IP, dan akan menentukan label yang tepat untuk dienkapsulasi ke dalam paket tersebut ketika sebuah paket IP masuk ke dalam jaringan *MPLS*. Dan ketika paket yang berlabel meninggalkan jaringan *MPLS*, maka *edge* router yang lain akan menghilangkan label tersebut (*label switches*).



Gambar 2.11 Struktur jaringan *MPLS* [22]

- *Label Distribution Protocol* (*LDP*)

  LDP merupakan suatu prosedur yang digunakan untuk menginformasikan ikatan label yang telah dibuat dari satu *LSR* ke *LSR* lainnya dalam satu jaringan *MPLS*. Dalam arsitektur jaringan *MPLS*, sebuah *LSR* yang merupakan tujuan atau *hop* selanjutnya akan mengirimkan informasi tentang ikatan sebuah label ke *LSR* yang sebelumnya mengirimkan pesan untuk mengikat label tersebut bagi rute paketnya. Teknik ini biasa disebut distribusi label *downstream on demand*.

  Jaringan baru ini memiliki beberapa keuntungan, diantaranya:

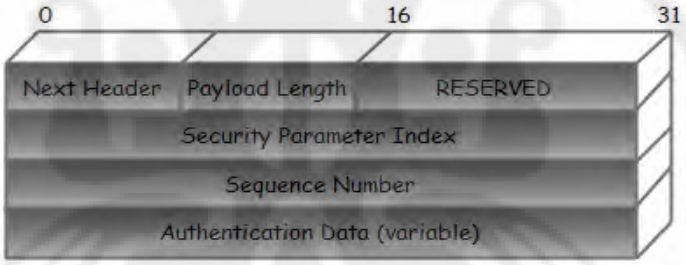
  - *MPLS* mengurangi banyaknya proses pengolahan yang terjadi di IP router, serta memperbaiki kinerja pengiriman suatu paket data.
  - *MPLS* juga bisa menyediakan *Quality of Service* (*QoS*) dalam jaringan *backbone*, dan menghitung parameter *QoS* menggunakan teknik *Differentiated services* (*Diffserv*) sehingga setiap layanan paket yang dikirimkan akan mendapat perlakuan yang berbeda sesuai dengan skala prioritasnya.

2.10. *IPSec Tunnel*

2.10.1. Arsitektur *IPSec*

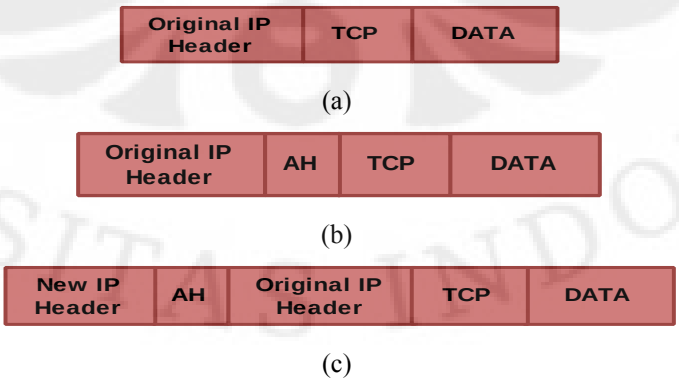
IPSec merupakan jenis protokol yang mengintegrasikan fitur *security* yang meliputi proses authentifikasi, integritas, dan kepastian ke dalam *Internet Protocol (IP)*. Dimana proses tersebut dilakukan pada *network layer* atau layer ketiga dalam model *OSI*. Dengan menggunakan *IPSec Tunnel*, kita dapat melakukan enkripsi dan atau membuat media komunikasi (*tunnel*) terautentifikasi tergantung pada kondisi protokol yang diinginkan oleh dua *peer* tersebut. Protokol tersebut antara lain:

- *Authentication Header* (*AH*), autentifikasi sumber data dan proteksi terhadap pencurian data. Protokol *AH* dibuat dengan melakukan enkapsulasi paket IP asli ke dalam paket baru yang mengandung *IP Header* yang baru yaitu *AH Header* disertai dengan *header* yang asli. Isi data yang dikirimkan melalui protokol *AH* bersifat *clear text* sehingga *tunnel* yang berdasar pada protokol *AH* ini tidak menyediakan kepastian data [RFC 2402].



Gambar 2.12 Paket *header AH* [23]

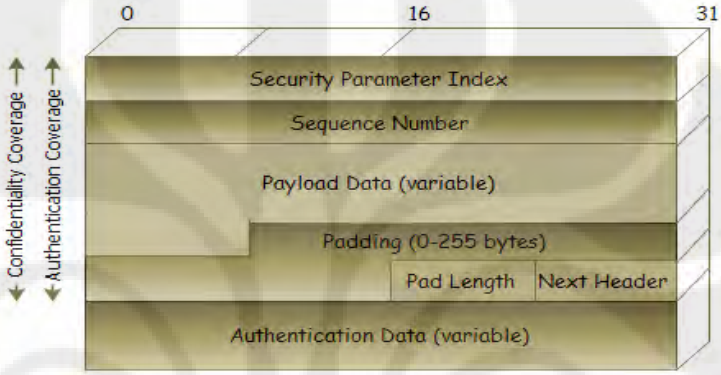
Proses implementasi *AH* pada paket:



Gambar 2.13 (a) Paket IP sebelum diimplementasikan *AH*, (b) *Transport Mode* dan *AH*, (c) *Tunnel Mode* dan *AH* [23]

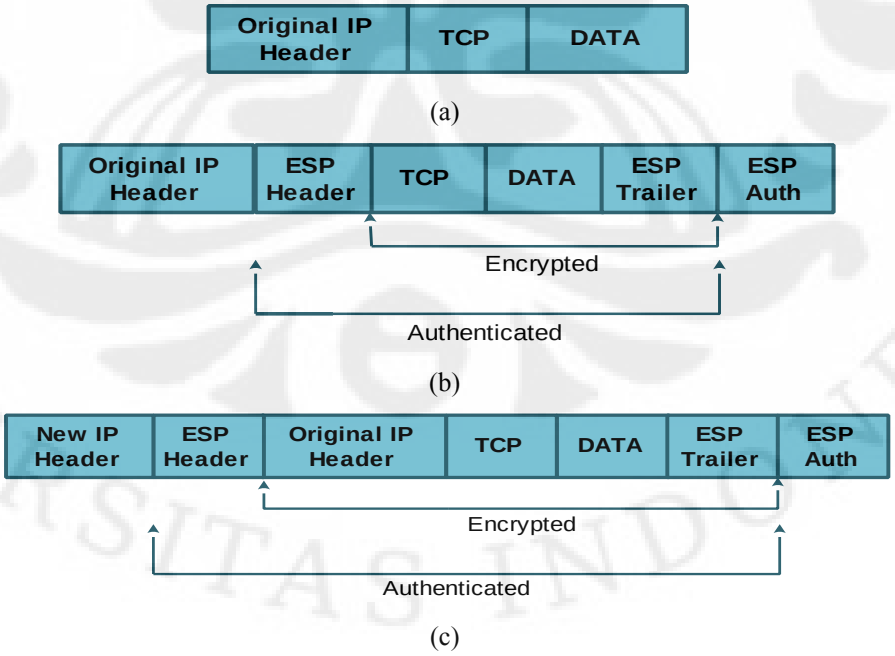
- *Encapsulated Security Payload* (*ESP*) dapat menyediakan kepastian data, autentifikasi sumber data dan proteksi terhadap gangguan pada data. Protokol *ESP* dibuat dengan melakukan enkripsi pada paket IP dan membuat paket IP lain yang mengandung *header* IP asli dan *header ESP*. Data yang terenkripsi (yang mengandung *header* IP asli) dan *trailer ESP*, separuhnya terenkripsi dan sebagian tidak [RFC 2406].

  Format paket data *ESP*:



Gambar 2.14 Paket *header ESP* [23]

Proses implementasi *ESP* pada paket:



Gambar 2.15 (a) Paket IP sebelum diimplementasikan *ESP*, (b) *Transport Mode* dan *ESP*, (c) *Tunnel Mode* dan *ESP* [23]

Salah satu contoh penggunaan *IPSec Tunnel* ialah pada *VPN*. Dimana *VPN* memperbolehkan komunikasi data yang aman melalui jaringan publik atau jaringan yang tidak dipercayai.

IPSec menyediakan dua metode untuk menentukan kunci yang akan digunakan pada algoritma yaitu:

- Metode Manual

  IPSec Tunnel dengan metode ini dikenal sebagai *IPSec Tunnel* manual di mana *shared key* atau kunci yang akan dipakai, protokol (*AH, ESP* atau keduanya) dan algoritma yang akan digunakan ditetapkan sebelum *tunnel* tersebut terbentuk.
- *Internet Key Exchange* (*IKE*)

  IKE tunnel melakukan negoisasi kunci yang akan dipergunakan diantara *peer* serta melakukan negoisasi dalam penentuan algoritma protokol (*AH* dan atau *ESP*) yang akan dipakai. Selain itu, *IKE tunnel* menggunakan algoritma Diffie-Hellman untuk menciptakan kunci yang simetris antar *peer* dalam membentuk *tunnel*. Kunci ini hanya akan berlaku sepanjang nilai *time to live*, setelah itu kunci baru akan dinegoisasikan kembali. *Tunnel* yang terbentuk disebut *IKE Tunnel* atau *Tunnel* Negoisasi.

2.10.2 *Security Association*

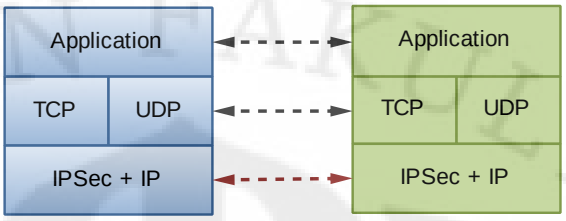
Kombinasi tentang bagaimana melindungi data (*ESP* dan atau *AH* termasuk algoritma dan kunci), apa data yang dilindungi dan pada saat apa data dilindungi disebut dengan *Security Association* (*SA*).

SA merupakan identifikasi unik berbasiskan *Security Parameter Index* (*SPI*), alamat tujuan IP dan protokol keamanan (*AH* dan atau *ESP*) yang diimplementasikan dalam trafik jaringan *IPSec*. Dua tipe *SA* yang didefinisikan yaitu:

- *Transport Mode*

  Pada mode ini, *SA* diimplementasikan pada trafik antara dua *host*. Pada kasus *ESP*, *transport mode SA* memberikan pelayanan keamanan hanya pada *layer* tersebut tidak untuk *IP Header* atau *header* lain yang tidak

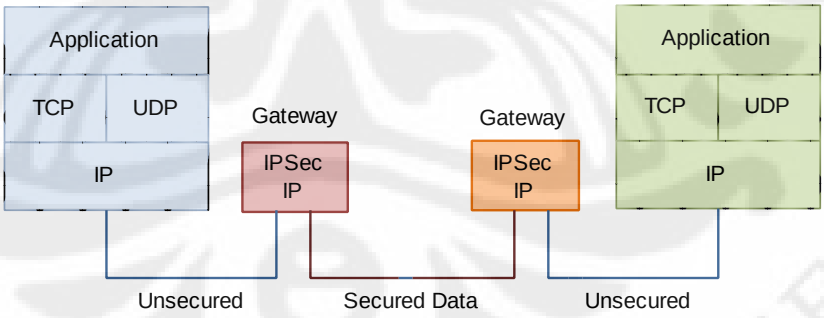
mendahului *header ESP*. Pada kasus *AH*, perlindungan juga diberikan pada *IP Header* atau *header* tambahan lainnya.



Gambar 2.16 *Transport modes IPSec* [23]

- *Tunnel Mode*

  Merupakan *SA* yang diimplementasikan pada dua *gateway IPSec Tunnel*. Pada mode ini, terdapat *IP Header* tambahan di luar yang menspesifikasikan tujuan pemrosesan *IPSec* ditambah *IP Header* tambahan di dalam yang menunjukkan alamat tujuan paket yang sebenarnya. *Header* protokol keamanan akan tampak pada bagian setelah *IP Header* tambahan di luar dan sebelum *IP Header* tambahan di dalam. Pada kasus *AH*, bagian *IP Header* tambahan di luar diberikan perlindungan seperti paket IP yang di-*tunnel*. Pada kasus *ESP*, proteksi hanya diberikan pada paket yang di-*tunnel* saja.



Gambar 2.17 *Tunnel mode IPSec* [23]

2.10.3 Enkripsi Pada *IPSec*

Tujuan penggunaan enkripsi ialah mengijinkan jaringan *peer* untuk saling berkomunikasi melalui jaringan yang tidak aman dan tidak terlindungi. Pada enkripsi terdapat dua komponen utama yaitu *Cipher* dan *Secret key*. *Cipher* merupakan algoritma matematika yang mengkonversi *string* atau sumber data

yang diketahui atau *plain text* menjadi random data yang disebut dengan *Ciphertext*. Hal ini disebut dengan enkripsi.

Dengan menggunakan *cipher*, kita juga dapat melakukan dekripsi data atau mengubah *ciphertext* menjadi *plaintext. Cipher* secara unik tergantung pada variable kriptografi yang disebut dengan kunci. Jika kita tidak memiliki kunci maka kita tidak dapat melakukan enkripsi atau dekripsi data. Kunci rahasia ini disebut dengan "kunci simetris". Ketika kita menggunakan *cipher* yang sama untuk melakukan enkripsi dan dekripsi data, pengirim dan penerima harus berbagi informasi yang rahasia. Tipe *cipher* ini dikenal dengan "*secret key cipher*"

Ada dua jenis mode enkripsi yaitu :

- *Electronic Code Book* (*ECB*) *mode*, merupakan mode dasar dan kurang aman. Dimana setiap blok *plaintext* yang diberikan kunci selalu dienkripsi dengan menggunakan blok yang sama dengan *ciphertext.*
- *Cipher Block Chaining* (*CBC*) *mode*, merupakan mode yang mirip dengan *ECB* namun enkripsi pada blok dengan menggunakan *plaintext*, kunci dan input ketiga dihasilkan dari *ciphertext* blok sebelumnya. Secara spesifik, proses blok *ciphertext* dengan melakukan XOR pada blok *plaintext* saat ini sebelum enkripsi normal dengan kunci. Blok *ciphertext* tersebut dirangkaikan dengan sebelumnya untuk menyembunyikan *pattern* yang diulang pada *plaintext.*

BAB 3

PERANCANGAN DAN IMPLEMENTASI *VOIP* PADA JARINGAN *VPN*

Telah dijelaskan dalam bab 2 tentang dasar teori yang mendukung pembuatan skripsi ini. Pada bab 3 akan dijelaskan dengan lebih spesifik mengenai rancangan pemodelan sistem yang dibuat, *tools* yang digunakan, *input* yang dimasukkan serta *output* yang diinginkan. Topologi jaringan yang dibuat disesuaikan agar dapat mendukung pengukuran terhadap performansi *VoIP* pada jaringan *VPN* berbasis *MPLS* dengan mengggunakan *tunneling IPSec*.

3.1 Spesifikasi dan Perancangan Sistem

Pada skripsi ini akan dibangun *simple VPN network* menggunakan teknologi *MPLS* dan *IPSec mode Tunnel*. Topologi jaringan dibangun dengan skenario *Site-to-site VPN Business*.

3.1.1 Kebutuhan *Hardware*

Dalam pembuatan skripsi ini sistem yang akan dibangun berupa *testbed*, jaringan dalam skala kecil, terdiri atas 5 buah router dan 3 buah *PC* dengan spesifikasi sebagai berikut :

- 3 buah Router Cisco tipe 2600 series, *2 PE* dan *1 Core*
  - ✓ Tipe router : Cisco 2621 (MPC860)
  - ✓ *IOS version* : 12.3(24a)
- 2 buah Router Cisco tipe 800 series, untuk *2 CE*
  - ✓ Tipe router : Cisco 871 (MPC8272)
  - ✓ *IOS version* : 12.4(4)T7
- 3 buah *PC* dengan rincian sebagai sebagai berikut :
  - 1 *PC server*, spesifikasi :
    - ✓ *Processor* : Intel Core 2 Duo T8100 @2.10 GHz, 2094 MHz
    - ✓ *Memory* : DDR2 SDRAM 2 GB
    - ✓ *NIC* : Broadcom Netlink (TM) Fast Ethernet
    - ✓ Sistem Operasi : Linux CentOS 5.8

- 2 *PC client*, spesifikasi sebagai berikut :

  PC 1:

  - ✓ *Processor* : Intel(R) Atom(TM) CPU N270 @1.60GHz 797 MHz
  - ✓ *Memory* : 1.99 GB
  - ✓ *NIC* : Broadcom Netlink (TM) Fast Ethernet
  - ✓ Sistem Operasi : Microsoft Windows XP Home Edition SP 3

  PC 2:

  - ✓ *Processor* : Intel(R) Atom(TM) CPU N280 @1.66GHz 1.66 GHz
  - ✓ *Memory* : 0.99 GHz
  - ✓ *NIC* : Atheros AR8132 PCI-E Fast Ethernet Controller
  - ✓ Sistem Operasi : Microsoft Windows XP Home Edition SP 3

- 1 buah kabel UTP tipe *crossover*, untuk menghubungkan router *PE-Core* dalam *cloud MPLS*
- 4 buah kabel UTP tipe *straight*, untuk menghubungkan router *PE-CE*
- Kabel *console* serial to USB
- Webcam
- *Headset*

3.1.2 Kebutuhan *Software*

Kebutuhan *software* yang digunakan untuk melakukan pengujian sistem dan analisa data, meliputi:

3.1.2.1 Trixbox

Trixbox (sebelumnya dikenal dengan nama Asterisk@home) adalah sebuah *VoIP phonesystem* yang mudah diinstalasi memanfaatkan *software* Asterisk PBX sebagai basis. Didesain baik untuk kalangan instansi perkantoran ataupun pengguna pribadi dengan menyertakan CentOS Linux, Sugar CRM,

MySQL, HUDlite, *free* PBX yang menjadi satu *bundle* aplikasi. Trixbox muncul karena ada keluhan dari banyak *user* yang merasa untuk mengaplikasi teknologi *VoIP* ini harus dengan usaha yang besar bahkan harus menjadi seorang *programmer*, maka ada suatu hal yang menghalangi teknologi ini yaitu masalah *user interface* yang dirasa tidak *user friendly*, oleh karena itu Trixbox diluncurkan guna mengatasi masalah ini. Trixbox dapat dikonfigurasi untuk menangani mulai dari satu sambungan telepon pribadi, puluhan atau ratusan sambungan telepon untuk perkantoran, dapat disambungkan ke beberapa saluran T1 di sebuah *call center* yang menangani jutaan menit percakapan per bulan. Sebuah web *GUI* (*web based*) memungkinkan konfigurasi dan pengoperasian menjadi mudah.

3.1.2.2 EyeBeam

EyeBeam merupakan *softphone* yang berfungsi sebagai *User Agent Client* dan *User Agent Server* pada protokol *SIP*. EyeBeam digunakan karena memiliki beberapa kelebihan. Selain mendukung protokol *SIP* dan H.323, EyeBeam juga mendukung berbagai jenis kompresi *Codec*, *Video Conference*, metode keamanan *SRTP* dan *TLS*.

Gambar 3.1 *Software* EyeBeam

3.1.2.3 VQ Manager

Manage Engine VQ Manager adalah sebuah *software* monitoring jaringan yang merupakan produk dari Zoho Corp. *Software* ini bisa memonitor semua

perangkat atau *user-agent* yang mendukung protokol *SIP* [RFC 3261] dan *RTP/RTCP* [RFC 3550]. VQ Manager memonitor secara *real time VoIP* QoS (*Quality of Service*) dan dapat dikonfigurasi sendiri karena merupakan *software* yang berbasis *web interface* sehingga dapat memantau berbagai parameter *QoS* yang memungkinkan identifikasi kesalahan dan kemerosotan kualitas *VoIP* dengan cepat. *Web user interface* untuk remote akses dapat menggunakan *javascript-enabled web browser* standar seperti Internet Explorer (v6.0 keatas) atau Firefox (v1.5 keatas). VQ Manager juga menyediakan grafik lalu lintas *VoIP* selama jangka waktu yang berbeda-beda dengan menunjukkan trend penggunaan *bandwidth* dan pola lalu lintas *VoIP*.

3.1.2.4 Wireshark

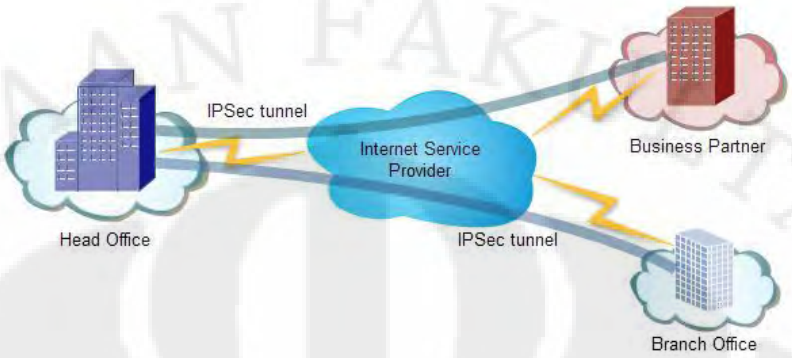
Wireshark merupakan *software* yang digunakan untuk melakukan analisa jaringan komputer. Wireshark dapat menganalisa beberapa *QoS* seperti *jitter, delay, throughput,* dan *packet loss* dan lain lain serta dapat meng-*capture* protokol yang sedang berjalan dalam jaringan tersebut. Versi Wireshark yang digunakan untuk pengujian adalah Wireshark 1.2.3 dan dapat di-*download* secara gratis pada *website* www.wireshark.org

3.2 Pemodelan Sistem

Dianalogikan terdapat sebuah perusahaan A yang mempunyai banyak kantor cabang dan mitra bisnis, juga terdapat sebuah *provider* yang menyewakan layanan komunikasi data via internet menggunakan teknologi *MPLS*. Perusahaan A adalah salah satu *customer* dari *provider* tersebut. Setiap kantor cabang atau mitra bisnis dari perusahaan A diharapkan dapat mengakses sumber daya yang ada di kantor pusat dengan aman dan cepat, bahkan dapat melakukan *meeting* atau diskusi jarak jauh antara *user* di kantor pusat dengan *user* di kantor cabang atau kantor pusat, atau bahkan dengan mitra bisnisnya dengan cara melakukan *video conference*, sehingga dapat menghemat tenaga, waktu dan biaya.

Selain teknologi *MPLS* di sisi *provider*, *IPSec* diimplementasikan pada masing-masing router *CE* untuk menjamin keamanan koneksi *end-to-end router*, dengan asumsi bahwa *Local Area Network* perusahaan A adalah jaringan yang

secure/terpercaya. Perancangan sistem ini dibuat dengan skenario "*Site-to-site VPN Business*" seperti yang terlihat pada Gambar 3.2.

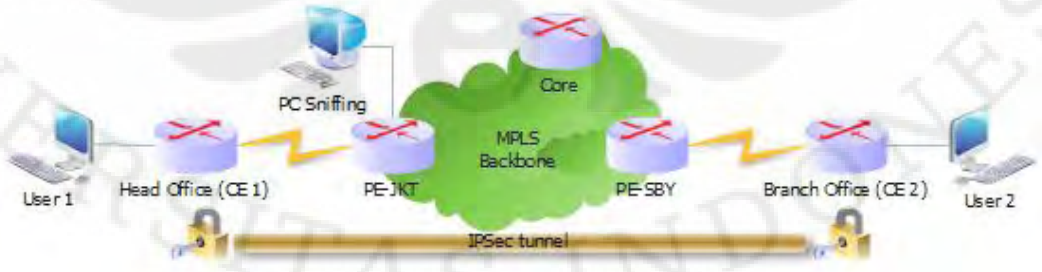


Gambar 3.2 Gambaran *Site-to-site VPN Business*

Pada dasarnya sebuah *cloud MPLS* yang dibangun oleh *service provider* terdiri atas dua lapisan, yaitu:

- Lapisan inti (terdiri atas interkoneksi sesama router *core*)
- Lapisan luar (terdiri atas router *PE* yang mengitari lapisan inti)

Pada skripsi ini, jaringan *VPN* yang dibangun terdiri atas 1 buah router *core*, 2 buah router *PE* dan 2 buah router *CE*. Salah satu router *PE* akan diasumsikan berlokasi di kota Jakarta dan lainnya di Surabaya, kedua router *PE* tersebut menghubungkan router *CE* yang juga berlokasi di Jakarta untuk *Head Office* (kantor pusat) dan di Surabaya untuk *Branch Office* (kantor cabang) dari perusahaan A. Topologi jaringan *VPN* yang akan dibangun dapat dilihat pada Gambar 3.3.



Gambar 3.3 Topologi jaringan *VPN* berbasis *MPLS* dan *IPSec*

Untuk membangun jaringan *VPN* ini dibagi menjadi dua tahap, yaitu :

1) Membangun jaringan *MPLS backbone* di sisi *provider*, dengan melakukan konfigurasi router PE-JKT , *Core* dan PE-SBY. Protokol routing yang digunakan yaitu *MP-BGP* untuk *peering* antar *PE*, dan *RIPv2* untuk koneksi antara *PE* - *Core*.
2) Membentuk koneksi *VPN* dengan *tunnel IPSec*, melakukan konfigurasi pada kedua router *CE* atau *end-to-end* router *Head Office* dan *Branch Office* dengan menggunakan routing protokol *RIPv2*

Daftar *interface* serta *IP address* yang akan digunakan adalah sebagai berikut:

Tabel 3.1 Konfigurasi IP pada masing-masing *interface* router *CE*

Device	Loopback0	Fastethernet0 (Fa0)	Fastethernet4 (Fa4)
Head Office	10.10.12.1/32	10.10.10.1/24	10.10.11.2/30
Branch Office	10.10.15.1/32	10.10.14.1/24	10.10.13.2/30

Tabel 3.2 Konfigurasi IP pada masing-masing *interface* router *service provider*

Device	Loopback0	Fastethernet0/0 (Fa0/0)	Fastethernet0/1 (Fa0/1)
PE-JKT	200.20.20.100/32	200.20.20.1/30	10.10.11.1/24
Core	200.20.20.101/32	200.20.20.2/30	200.20.20.5/30
PE-SBY	200.20.20.102/32	10.10.13.1/24	200.20.20.6/30

3.3 Implementasi Sistem

3.3.1. Implementasi Jaringan *Backbone MPLS*

Langkah-langkah implementasi jaringan *MPLS backbone* ini, antara lain:

1) Memberikan *IP address* pada *interface* router

   IP address dikonfigurasi pada *interface* ke arah *Core* di router PE-JKT dan PE-SBY, sedangkan *IP address* pada *interface* yang ke arah *CE* (*Head Office* dan *Branch Office*) akan dikonfigurasi setelah membuat konfigurasi router virtual di *interface* tersebut.

Konfigurasi di router PE-JKT :

int f0/0
ip address 200.20.20.1 255.255.255.252

Konfigurasi di router *Core* :

int f0/0
ip address 200.20.20.2 255.255.255.252
int f0/1
ip address 200.20.20.9 255.255.255.252

Konfigurasi di router PE-SBY :

int f0/1
ip address 200.20.20.10 255.255.255.252

Langkah ini dimaksudkan untuk memastikan kondisi *link* antara *PE* dan *Core*, yaitu dengan cara melakukan tes ping ke masing-masing *interface* yang berada dalam satu *network*, jika berhasil maka artinya *link* antara *PE* dan *Core* sudah bagus.

2) Membuat router virtual

Provider yang mempunyai banyak *customer* dari berbagai perusahaan memerlukan router-router virtual di setiap router *PE* yang terhubung langsung dengan router *customer* (*CE*), dimana router virtual tersebut terdiri atas *Route Distinguisher* (*RD*) dan *Virtual Routing Forwarding* (*VRF*). *Route distinguisher* merupakan proses pemberian label dari setiap *VPN customer* yang nantinya akan dipertukarkan antar router *PE*, sedangkan nama dari Router virtual ini diasumsikan sebagai mini router yang *dedicated* untuk mengatur tabel routing setiap customer. Karena konsepnya *dedicated*, maka jumlah router virtual ini akan sebanyak jumlah *VPN customer* yang terhubung dengan *PE* tersebut.

Dalam skripsi ini terdapat satu buah *VPN customer* yaitu company-a, sehingga hanya terdapat satu buah router virtual di dalam fisik router PE-JKT dan PE-SBY.

Konfigurasi router virtual untuk PE-JKT dan PE-SBY:

ip vrf company-a
rd 65000:1
route-target export 65000:1
route-target import 65000:1

Setelah terbentuk router virtual yang ber-*ID* company-a, selanjutnya mengimplementasikan router virtual tersebut ke *interface* di router *PE* yang ke arah router *CE* dan memberikan *IP address* pada *interface* tersebut setelahnya.

Konfigurasi di router PE-JKT:

int fa0/1
ip vrf forwarding company-a
ip address 10.10.11.1 255.255.255.252

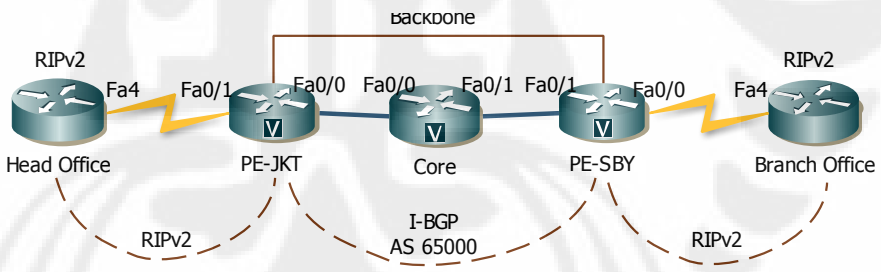
Konfigurasi di router PE-SBY:

int fa0/0
ip vrf forwarding company-a
ip address 10.10.13.1 255.255.255.252

3) Mengaktifkan routing dinamik

Untuk membuat *cloud MPLS*, semua router dalam *cloud* tersebut harus menggunakan routing dinamik, agar routing di masing-masing router dapat mengembang atau terpopulasi.

a) Mengaktifkan *RIPv2* (*Routing Information Protocol*)



Gambar 3.4 Jaringan *VPN* menggunakan routing protokol *RIP*

Konfigurasi *RIPv2* pada router PE-JKT, PE-SBY dan Core:

router rip
version 2
network 200.20.20.0

Network yang akan di-*advertise* adalah *interface* loopback0 dan *network* dari Ethernet yang *directly connected.* Dalam konteks PE-JKT, *interface* yang perlu di-*advertise* adalah *network* dari fastethernet0/0 yang mengarah ke *Core*. Sedangkan *interface* fastethernet0/1 yang mengarah ke *CE Head-Office* tidak perlu di-

*advertis*e. Hal ini dimaksudkan agar di dalam *cloud MPLS*, *IP prefix* yang di-*advertise* hanyalah yang benar-benar berada di dalam *cloud MPLS*. Router *Core* meng-*advertise interface* yang ke arah router PE-JKT dan PE-SBY. Semua *interface* yang akan di-*advertise* dalam *cloud MPLS* ini menggunakan kelas IP yang sama yaitu kelas C, sedangkan karakteristik *RIPv2* salah satunya adalah mendukung *Variable Length Subnet Mask* (*VLSM*) yaitu teknik untuk menghemat penggunaan *IP address*, yang menjadikan routing ini sebagai salah satu *classless routing protocol*, sehingga dalam konfigurasi router *RIP network* yang akan di-*advertise* cukup di-set subnet-nya saja "200.20.20.0".

b) Mengaktifkan *BGP*

Untuk mengaktifkan *BGP* maka status *BGP connection* antara PE-JKT dan PE-SBY harus *establish* terlebih dahulu.

Konfigurasi di PE-JKT:

```
router bgp 65000
no synchronization
bgp log-neighbor-changes
neighbor 200.20.20.102 remote-as 65000
neighbor 200.20.20.102 update-source Loopback0
neighbor 200.20.20.102 next-hop-self
no auto-summary
```

Konfigurasi di PE-SBY:

```
router bgp 65000
no synchronization
bgp log-neighbor-changes
neighbor 200.20.20.100 remote-as 65000
neighbor 200.20.20.100 update-source Loopback0
neighbor 200.20.20.100 next-hop-self
no auto-summary
```

Untuk memastikan *TCP connection* antara *BGP* yang ada di router PE-JKT dan PE-SBY sudah *established* atau belum, dapat diperiksa di salah satu router *PE* seperti di bawah ini:

```
PE-JKT#sh ip bgp sum
BGP router identifier 200.20.20.100, local AS number 65000
BGP table version is 1, main routing table version 1

Neighbor        V    AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  State/PfxRcd
200.20.20.102   4 65000     132     132        1    0    0 02:05:02        0
```

Dari hasil di atas terlihat bahwa status *BGP* sudah *establish*, tertulis "02:05:02" artinya *BGP* sudah *establish* selama 2 jam 5 menit 2 detik.

c) Mengaktifkan *MP-BGP*

Koneksi *MP-BGP* antara PE-JKT dan PE-SBY disebut sebagai *Backbone MP-BGP*. Setelah mengaktifkan *BGP* antara PE-JKT dan PE-SBY, *peering* (bisa disebut sebagai *tunnel*) *BGP* yang telah terbentuk akan berisi *sub-tunnel MP-BGP* untuk membawa *routing table vrf* company-a dari PE-JKT ke PE-SBY dan sebaliknya.

Konfigurasi di PE-JKT:

```
router bgp 65000
...
address-family VPNv4
neighbor 200.20.20.102 activate
neighbor 200.20.20.102 send-community both
exit-address-family
```

Konfigurasi di PE-SBY:

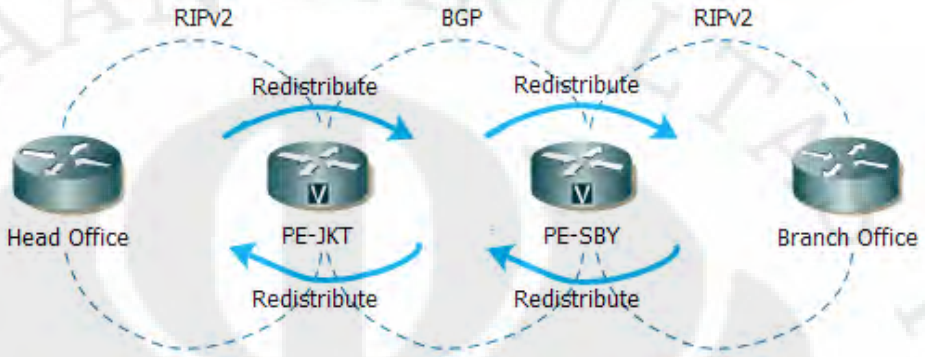
```
router bgp 65000
…
address-family VPNv4
neighbor 200.20.20.100 activate
neighbor 200.20.20.100 send-community both
exit-address-family
```

Konfigurasi di atas seolah-olah membuka sebuah *sub-tunnel* di *tunnel BGP* AS *number* 65000 yang sebelumnya sudah ada, yaitu *sub-tunnel* untuk melewatkan *routing table vrf* company-a. *MP-BGP* memanfaatkan fasilitas *community* pada standar *BGP*, baik yang standar maupun *extended*, di mana keduanya (*both*) akan saling dipertukarkan.

d) Mengaktifkan *RIPv2* pada router virtual dan me-*redistribute* routing *RIPv2* ke dalam *MP-BGP* dan *MP-BGP* ke dalam *RIPv2*

Routing protokol yang digunakan pada jaringan ini dapat digambarkan sebagai *RIPv2-BGP-RIPv2* seperti yang terlihat pada Gambar 3.5. Dengan demikian diperlukan suatu proses *redistribution*, yaitu proses

pada router untuk meng-*import* atau meng-*export* routing protokol satu ke routing protokol yang lain agar kedua routing protokol yang berbeda tersebut dapat saling berkomunikasi.



Gambar 3.5 Ilustrasi *redistribute network RIP* dan *BGP*

Konfigurasi di PE-JKT dan PE-SBY:

Redistribute dari *RIP* ke *MP-BGP*

address-family ipv4 vrf company-a
redistribute bgp 65000 metric transparent
network 10.0.0.0
no auto-summary
version 2
exit-address-family

Redistribute dari *MP-BGP* ke *RIPv2*

address-family ipv4 vrf company-a
redistribute rip
no auto-summary
no synchronization
exit-address-family

4) Mengaktifkan *MPLS*

Agar *MPLS* aktif di PE-JKT, Core dan PE-SBY, maka *interface* di router tersebut yang menerapkan label harus diaktifkan *MPLS*-nya terlebih dahulu.

Mengaktifkan *MPLS* di PE-JKT:

interface FastEthernet0/0
description "Connection to CORE"
...
tag-switching ip

Mengaktifkan *MPLS* di Core:

interface FastEthernet0/0
description "Connection to PE-JKT"
...
tag-switching ip

Mengaktifkan *MPLS* di PE-SBY:

interface FastEthernet0/1
description "Connection to CORE"
...
tag-switching ip

Untuk memastikan bahwa *MPLS* sudah bekerja, dapat di periksa di salah satu router tersebut seperti di bawah ini:

CORE#sh MPLS ldp neighbor
Peer TDP Ident: 200.20.20.100:0; Local TDP Ident 200.20.20.101:0
TCP connection: 200.20.20.100.711 - 200.20.20.101.50374
State: Oper; PIEs sent/rcvd: 0/14; Downstream
Up time: 00:08:30
TDP discovery sources:
FastEthernet0/0, Src IP addr: 200.20.20.1
Addresses bound to peer TDP Ident:
200.20.20.1 200.20.20.100
Peer TDP Ident: 200.20.20.102:0; Local TDP Ident 200.20.20.101:0
TCP connection: 200.20.20.102.55541 - 200.20.20.101.711
State: Oper; PIEs sent/rcvd: 0/13; Downstream
Up time: 00:08:23
TDP discovery sources:
FastEthernet0/1, Src IP addr: 200.20.20.10
Addresses bound to peer TDP Ident:
200.20.20.10 200.20.20.102

Dari hasil pengecekan di atas terlihat bahwa router *Core* sudah mengenali router PE-JKT dan PE-SBY sebagai router *MPLS*, artinya *MPLS* sudah aktif di ketiga router tersebut.

3.3.2. Implementasi Router *CE*

Langkah-langkah konfigurasi router *CE*:

1) Memberikan *IP address* pada semua *interface* router

   Konfigurasi router *Head-Office*:

   interface Loopback0
   ip address 10.10.12.1 255.255.255.255

   interface FastEthernet0
   description "Connection to PC-User1"
   switchport access vlan 2

```
interface FastEthernet1
switchport access vlan 2

interface FastEthernet4
description "Connetion to PE-JKT"
ip address 10.10.11.2 255.255.255.252

interface Vlan2
ip address 10.10.10.1 255.255.255.0
```

PC-User1 yang terhubung ke fastethernet0 bertindak sebagai *server* Tribox dan juga *client*.

Konfigurasi router *Branch-Office*:

```
interface Loopback0
ip address 10.10.15.1 255.255.255.255

interface FastEthernet0
description “Connection to PC-User2”
switchport access vlan 2

interface FastEthernet1
description “Connection to PC-User3”
switchport access vlan 2

interface FastEthernet2
description “Connection to PC-Wireshark”
switchport access vlan 2

interface FastEthernet4
description "Connetion to PE-JKT"
ip address 10.10.13.2 255.255.255.252

interface Vlan2
ip address 10.10.14.1 255.255.255.0
```

PC-User2 yang terhubung ke fastethernet0 dan *PC-User3* yang terhubung ke fastethernet1 bertindak sebagai *client*, sedangkan *PC*-Wireshark yang terhubung ke fastethernet2 bertindak sebagai *PC sniffing* untuk menangkap paket *RTP* yang melintasi jaringan.

2) Mengaktifkan routing *RIPv2*

Untuk mengaktifkan routing *RIPv2*, konfigurasi di router *Head Office* dan *Branch Office* adalah:

```
router rip
version 2
network 10.0.0.0
```

Network yang akan di-*advertise* pada router *Head Office* dan *Branch Office* berasal dari kelas IP yang sama, yaitu kelas A, sehingga *network* yang di-set pada router *RIP* adalah subnet dari IP tersebut "10.0.0.0".

3.3.3. Pengecekan Routing Protokol *RIPv2*

Untuk memastikan konfigurasi yang diimplementasikan sudah sesuai dengan yang diharapkan, maka dilakukan pengecekan routing pada masing-masing router, baik di router *CE* maupun di dalam *cloud MPLS*.

Head-Office#sh ip route
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
E1 - OSPF external type 1, E2 - OSPF external type 2
i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
*ia - IS-IS inter area, * - candidate default, U - per-user static route*
o - ODR, P - periodic downloaded static route

Gateway of last resort is not set

10.0.0.0/8 is variably subnetted, 6 subnets, 3 masks
C 10.10.10.0/24 is directly connected, Vlan2
C 10.10.11.0/30 is directly connected, FastEthernet4
R 10.10.13.0/30 [120/1] via 10.10.11.1, 00:00:02, FastEthernet4
C 10.10.12.1/32 is directly connected, Loopback0
R 10.10.15.1/32 [120/2] via 10.10.11.1, 00:00:02, FastEthernet4
R 10.10.14.0/24 [120/2] via 10.10.11.1, 00:00:02, FastEthernet4

PE-JKT#sh ip route
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
E1 - OSPF external type 1, E2 - OSPF external type 2
i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
*ia - IS-IS inter area, * - candidate default, U - per-user static route*
o - ODR, P - periodic downloaded static route

Gateway of last resort is not set

200.20.20.0/24 is variably subnetted, 5 subnets, 3 masks
C 200.20.20.100/32 is directly connected, Loopback0
R 200.20.20.101/32 [120/1] via 200.20.20.2, 00:00:27, FastEthernet0/0
R 200.20.20.102/32 [120/2] via 200.20.20.2, 00:00:27, FastEthernet0/0
R 200.20.20.8/30 [120/1] via 200.20.20.2, 00:00:27, FastEthernet0/0
C 200.20.20.0/29 is directly connected, FastEthernet0/0

CORE#sh ip rou
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
E1 - OSPF external type 1, E2 - OSPF external type 2
i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2

```
ia - IS-IS inter area, * - candidate default, U - per-user static route
     o - ODR, P - periodic downloaded static route

Gateway of last resort is not set

   200.20.20.0/24 is variably subnetted, 5 subnets, 3 masks
R      200.20.20.100/32 [120/1] via 200.20.20.1, 00:00:03, FastEthernet0/0
C      200.20.20.101/32 is directly connected, Loopback0
R      200.20.20.102/32 [120/1] via 200.20.20.10, 00:00:07, FastEthernet0/1
C      200.20.20.8/30 is directly connected, FastEthernet0/1
C      200.20.20.0/29 is directly connected, FastEthernet0/0
```

Dari hasil pengecekan di atas terlihat bahwa *routing protocol RIPv2* sudah aktif pada semua *interface* router. Untuk memastikan routing *RIPv2* sudah aktif pada router virtual yang diimplementasikan pada PE-JKT dan PE-SBY dapat dicek pada salah satu *PE* sebagai berikut:

```
PE-JKT#sh ip rou vrf company-a

Routing Table: company-a
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is not set

     10.0.0.0/8 is variably subnetted, 6 subnets, 3 masks
R       10.10.10.0/24 [120/1] via 10.10.11.2, 00:00:09, FastEthernet0/1
C       10.10.11.0/30 is directly connected, FastEthernet0/1
B       10.10.13.0/30 [200/0] via 200.20.20.102, 02:04:07
R       10.10.12.1/32 [120/1] via 10.10.11.2, 00:00:09, FastEthernet0/1
B       10.10.15.1/32 [200/1] via 200.20.20.102, 02:04:38
B       10.10.14.0/24 [200/1] via 200.20.20.102, 02:04:38
```

Dari hasil pengecekan di atas dapat dilihat bahwa terdapat dua routing protokol yang aktif untuk *vrf* company-a pada PE-JKT yaitu *RIPv2* dan *MP-BGP*, di mana *MP-BGP* bertugas membawa tabel routing *RIPv2* company-a melintasi *MPLS backbone*.

3.3.4 Implementasi *IPSec Tunnel Mode* di Router *CE*

Untuk mengaktifkan enkripsi dan *tunnel IPSec* harus memenuhi langkah-langkah sebagai berikut:

1) Mengkonfigurasi *IKE Policies*

Internet Key Exchange (*IKE*) aktif secara *default* dan tidak harus diaktifkan pada masing-masing *interface* tetapi aktif secara global pada semua *interface* router. *IKE policies* harus dipasang pada setiap *peer*, dalam hal ini router *CE*. *IKE policies* didefinisikan sebagai sebuah kombinasi dari parameter-parameter *security* yang digunakan selama negosiasi *IKE*. *IKE policies* dapat dibuat lebih dari 1, di mana setiap *policies* memiliki kombinasi nilai parameter yang berbeda. Masing-masing *policies* yang dibuat diberi nilai prioritas yang unik yaitu dari 1 sampai 10000 (1 adalah prioritas tertinggi). Setiap *peer* dapat memiliki banyak *policies*, tetapi sedikitnya 1 *policies* harus mempunyai nilai parameter enkripsi, *hash*, autentikasi dan Diffie-Helman yang sama dengan salah satu *policies* yang ada pada *remote peer*.

Konfigurasi di router *CE Head-Office* dan *Branch-Office*:

```
crypto isakmp policy 1
encr 3des
authentication pre-share
group 2
crypto isakmp key p4ssw0rd address 10.10.11.2
```

Sesuai dengan konfigurasi di atas, pada skripsi ini algoritma enkripsi yang digunakan adalah 168-bit *Tripple DES* (*3DES*), algoritma *hash* adalah *sha*, autentikasi menggunakan *pre-shared* dan Diffie-Hellman *Identifier* adalah 1024 bit (*group* 2).

2) Memeriksa *IKE Policies*

Pengecekan *IKE policies* dilakukan di kedua router *CE* di mana *IPSec* diimplementasikan. Kedua router *CE* harus mempunyai parameter enkripsi yang sama, yaitu algoritma enkripsi, algoritma *hash*, metode autentikasi dan Diffie-Hellman *identifier/group*.

Pengecekan *IKE Policies* di router *Head-Office*:

```
Head-Office#sh crypto isakmp policy

Global IKE policy
Protection suite of priority 1
encryption algorithm:   Three key triple DES
hash algorithm:         Secure Hash Standard
```

authentication method: Pre-Shared Key
Diffie-Hellman group: #2 (1024 bit)
lifetime: 86400 seconds, no volume limit
Default protection suite
encryption algorithm: DES - Data Encryption Standard (56 bit keys).
hash algorithm: Secure Hash Standard
authentication method: Rivest-Shamir-Adleman Signature
Diffie-Hellman group: #1 (768 bit)
lifetime: 86400 seconds, no volume limit

Pengecekan *IKE Policies* di router *Branch-Office*:

Branch-Office#sh crypto isakmp policy

Global IKE policy
Protection suite of priority 1
encryption algorithm: Three key triple DES
hash algorithm: Secure Hash Standard
authentication method: Pre-Shared Key
Diffie-Hellman group: #2 (1024 bit)
lifetime: 86400 seconds, no volume limit
Default protection suite
encryption algorithm: DES - Data Encryption Standard (56 bit keys).
hash algorithm: Secure Hash Standard
authentication method: Rivest-Shamir-Adleman Signature
Diffie-Hellman group: #1 (768 bit)
lifetime: 86400 seconds, no volume limit

IPSec menggunakan metode *Internet key Exchange* (*IKE*) untuk melakukan negosiasi kunci dan menentukan algoritma protokol (*AH* dan atau *ESP*) yang digunakan. Seperti yang terlihat dari hasil pengecekan *IKE policies* di atas, penentuan algoritma *IKE* meliputi :

- Algoritma enkripsi : *Three key triple DES (3DES)*
  3DES menyediakan kepastian data. Kecepatan yang dibutuhkan lebih lambat dan membutuhkan *resource* yang lebih banyak untuk melakukan tiga kali perhitungan, namun algoritma ini lebih aman dibandingkan dengan *DES*.
- Algortima *hash* : *Secure Hash Standard (SHA)*
  Integritas data didukung dengan penentuan *hash algorithm* yang akan digunakan pada proses negosiasi. *SHA* memproduksi 160 bit *digest*, di mana dibutuhkan waktu pemrosesan yang lebih lama dan *resource* yang lebih banyak daripada *MD5* dan tentunya *SHA* lebih kuat daripada *MD5*.

- Metode autentikasi : *Pre-Shared-Key*

  Proses autentifikasi dalam sistem ini menggunakan *shared secret* yang membutuhkan *pre-shared-key* daripada *certificate*, karena konfigurasinya lebih mudah untuk jaringan yang berskala kecil.

- Diffie-Helman *group* : 2 (1024 bit)

  IKE tunnel menggunakan algoritma Diffie-Hellman untuk menciptakan kunci yang simetris antar *peer* dalam bentuk *tunnel*. Penurunan kunci merupakan pembentukan awal kunci berdasarkan *Group* Diffie-Hellman *(dh-group)*. Pada skripsi ini, kedua *peer* (router *Head Office* dan *Branch Office*) menggunakan group 2 dengan modulus 1024 (mod1024).

3) Mengkonfigurasi *IPSec* dan *IPSec* mode *tunnel*

   Setelah melakukan konfigurasi *IKE Policies* pada masing-masing *peer*, selanjutnya adalah melakukan konfigurasi *IPSec* di masing-masing *peer* dengan langkah-langkah seperti berikut:

   - Membuat *crypto access lists*

     Crypto access lists digunakan untuk mendefinisikan trafik IP yang mana yang akan dilindungi atau tidak dilindungi oleh *crypto*. *Access lists* ini tidak sama dengan *access lists* biasa yang menentukan apakah trafik akan diteruskan atau diblok di sebuah *interface*.

     Konfigurasi di router *Head Office*:

     ip access-list extended JKT-to-SBY
     permit ip 10.10.10.0 0.0.0.255 10.10.14.0 0.0.0.255

     Konfigurasi di router *Branch Office*:

     ip access-list extended SBY-to-JKT
     permit ip 10.10.14.0 0.0.0.255 10.10.10.0 0.0.0.255

     Extended access list yang diberi nama JKT-to-SBY di router *Head Office* dan SBY-to-JKT di router *Branch Office* nantinya akan dipasang pada *interface* yang akan dilewati trafik *IPSec*.

   - Memeriksa *crypto access lists*

     ***Branch Office*#sh access-lists**

Extended IP access list SBY-to-JKT
10 permit ip 10.10.14.0 0.0.0.255 10.10.10.0 0.0.0.255
***Head Office*#sh access-lists**
Extended IP access list JKT-to-SBY
10 permit ip 10.10.10.0 0.0.0.255 10.10.14.0 0.0.0.255

Trafik data akan diproteksi dari router *Branch Office* hingga *Head Office,* demikian sebaliknya.

- Mendefinisikan *transform set* dan mengkonfigurasi *IPSec tunnel mode*

  Transform-set harus didefinisikan sesuai dengan protokol yang digunakan.

  Konfigurasi di router *Head Office* dan *Branch Office*:

  crypto IPSec transform-set finalproject esp-3des esp-sha-hmac

  Dilihat dari konfigurasi di atas, sesuai dengan *IKE policies* yang telah diimplementasikan, *transform-set* diberi nama "finalproject" dan terdiri atas kombinasi *esp-sha-hmac*, sedangkan *encryption set* terdiri atas kombinasi *esp-3des*, *ESP* diimplementasikan pada *Security Association* (*SA*) mode *tunnel*, di mana proteksi hanya diberikan pada paket yang berada di dalam *tunnel*. *ESP* dibuat dengan melakukan enkripsi pada paket IP dan membuat paket IP lain yang mengandung *header* IP asli dan *header* ESP.

- Memeriksa *transform set* dan *IPSec tunnel mode*

  Pengecekan di salah satu router *Branch Office* maupun *Head Office* diperoleh hasil yang sama sebagai berikut:

  Branch Office*#sh *crypto IPSec transform-set
  Transform-set finalproject: { esp-3des esp-sha-hmac }
  will negotiate = { Tunnel, },

  IPSec diimplementasikan dalam mode *Tunnel*, dengan *transform-set* yang telah didefinisikan dengan nama "finalproject".

4) Mengkonfigurasi *crypto maps*

Ketika kedua *peer* mencoba membangun sebuah *Security Association* (*SA*), maka masing-masing dari *peer* tersebut harus mempunyai sedikitnya satu daftar *crypto map* yang sesuai dengan salah satu daftar *crypto map* pada *peer* yang lainnya. Langkah-langkah untuk melakukan konfigurasi *crypto maps* antara lain:

- Membuat daftar *crypto map*

  Daftar *crypto map* ini menggunakan *IKE* untuk membangun *SA*. Konfigurasi *crypto map* di router *Head-Office*:

  ```
  crypto map skripsi 1 IPSec-isakmp
  set peer 10.10.13.2
  set transform-set finalproject
  match address JKT-to-SBY
  ```

  Konfigurasi *crypto maps* di router *Branch-Office*:

  ```
  crypto map skripsi 1 IPSec-isakmp
  set peer 10.10.11.2
  set transform-set finalproject
  match address SBY-to-JKT
  ```

  Crypto map yang diimplementasikan diberi nama "skripsi", set *peer* adalah *interface peer* lawan yang juga mengimplementasikan *IPSec*, *transform-set* "finalproject" dan *extended access lists* yang telah dibuat diimplementasikan dalam daftar *crypto map*.

- Memeriksa daftar *crypto map*

  Pengecekan daftar *crypto maps* di salah satu router *CE*:

  ```
  Branch-Office#sh crypto map
  Crypto Map "skripsi" 1 IPSec-isakmp
        Peer = 10.10.11.2
        Extended IP access list SBY-to-JKT
            access-list SBY-to-JKT permit ip 10.10.14.0 0.0.0.255 10.10.10.0
  0.0.0.255
        Current peer: 1
  0.10.11.2
        Security association lifetime: 4608000 kilobytes/3600 seconds
        PFS (Y/N): N
        Transform sets={
              finalproject,
        }
        Interfaces using crypto map skripsi:
              FastEthernet4
  ```

Crypto map dengan nama "skripsi" mempunyai *peer* dengan *IP address* 10.10.11.2 yaitu router *Head Office* (*interface* kearah PE-JKT atau diasumsikan sebagai *interface* WAN).

- Mengimplementasikan *crypto map* ke *interface*

  Crypto map diimplementasikan pada masing-masing *interface* yang dilewati trafik *IPSec*. Implementasi *crypto map* pada sebuah *interface* router membuat router mengevaluasi semua trafik pada *interface* sesuai dengan daftar *crypto map* dan menggunakan *policies* tertentu selama negosiasi *SA* agar trafik mendapatkan proteksi dari *crypto*.

  Konfigurasi *crypto map* di router *Head Office*:

```
interface FastEthernet4
description "Connetion to PE-JKT"
...
crypto map skripsi
```

  Konfigurasi *crypto map* di router *Head Office*:

```
interface FastEthernet4
description "Connetion to PE-SBY"
...
crypto map skripsi
```

- Memeriksa *crypto map interface associations*

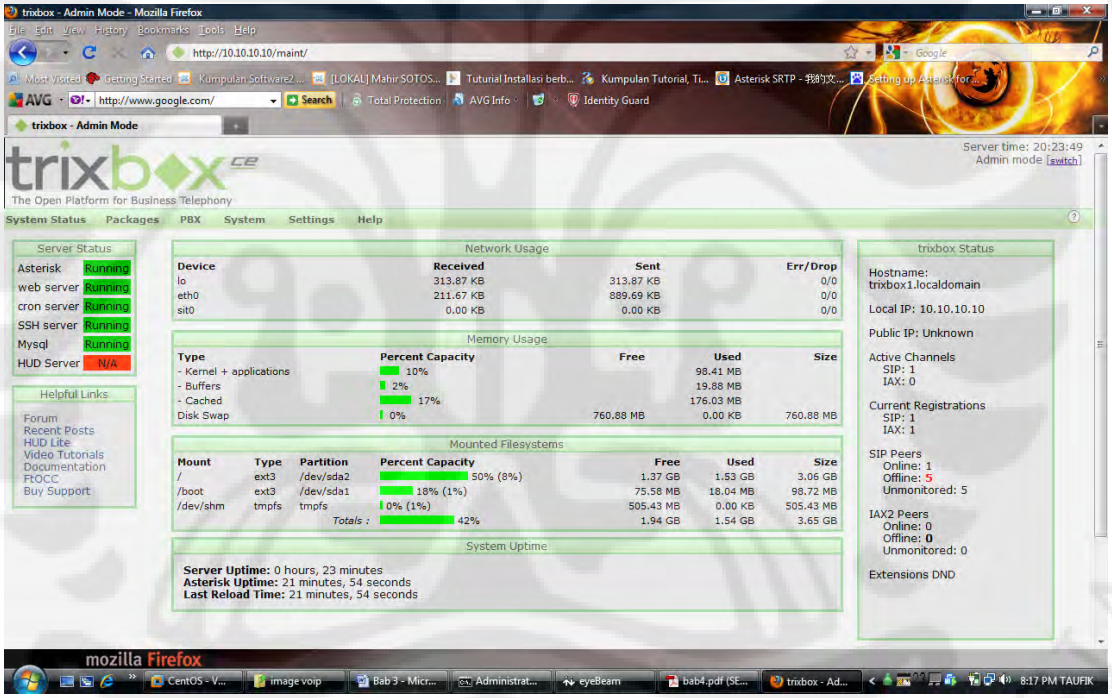
```
Branch-Office#sh crypto map interface fastEthernet 4
Crypto Map "skripsi" 1 IPSec-isakmp
      Peer = 10.10.11.2
      Extended IP access list SBY-to-JKT
          access-list SBY-to-JKT permit ip 10.10.14.0 0.0.0.255 10.10.10.0
0.0.0.255
      Current peer: 10.10.11.2
      Security association lifetime: 4608000 kilobytes/3600 seconds
      PFS (Y/N): N
      Transform sets={
            finalproject,
      }
      Interfaces using crypto map skripsi:
            FastEthernet4
```

  Crypto maps diimplementasikan pada *interface* fastethernet4, yaitu *interface* WAN di router *CE*.

3.4 Implementasi *Software* Pendukung

3.4.1 Implementasi *Server VoIP* (Trixbox)

Pada skripsi ini, *server VoIP* dibangun dengan menggunakan *software* Trixbox. Trixbox distro dengan CentOs Linux dapat di-*download* di *website* resmi Trixbox yaitu http://www.trixbox.org/downloads. Saat ini *release* terbaru dari Trixbox adalah versi 2.6, berbasis CentOS 5.4, Asterisk 1.6 dan FreePBX 2.6. Proses instalasi Trixbox ini dilakukan sama dengan instalasi pada sistem operasi linux lainnya. Yang perlu diperhatikan ketika melakukan instalasi Trixbox, *hardisk* akan terformat secara otomatis oleh Linux Trixbox ini. Jika proses instalasi selesai dilakukan, maka langkah selanjutnya mengkonfigurasi Trixbox



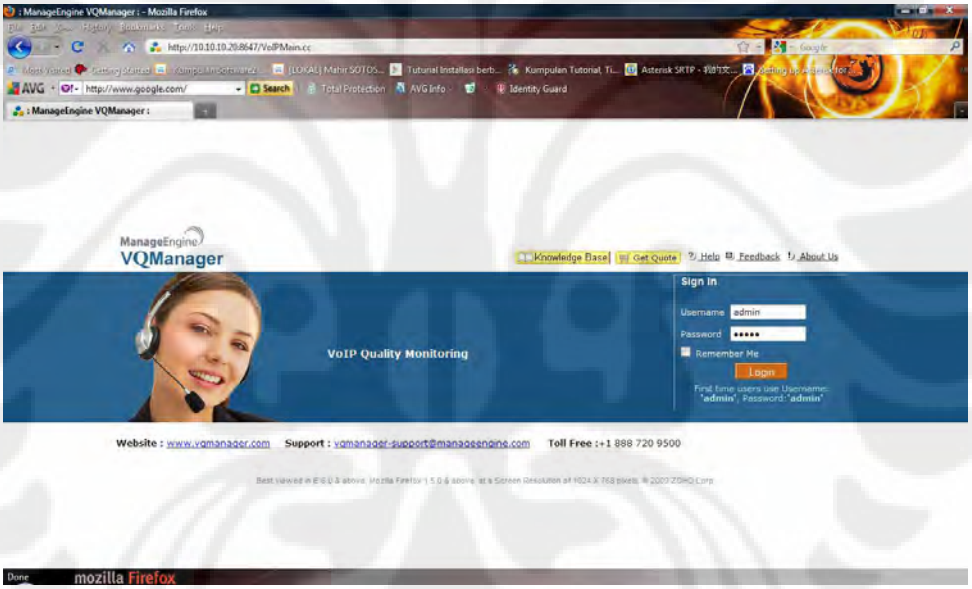
Gambar 3.6 Tampilan awal setelah *login* ke Trixbox

Dalam tampilan ini terdapat beberapa menu utama yaitu : *System Status* (tampilan awal status trixbox), *Packages* (berisi paket-paket yang mendukung perangkat lunak Trixbox), *PBX* (menu yang penting di dalam mengkonfigurasi *setting VoIP* yang dibuat, berisi sub menu antara lain : *PBX Setting*, *PBX* Status, dan lain-lain), *System* (berisi informasi mengenai seluruh sistem yang telah

diinstalasi, *Settings* (mengubah jaringan seperti nama *host*, alamat IP dan alamat *DNS*, selain itu terdapat menu registrasi).

3.4.2 Konfigurasi VQManager

Voice Quality Manager merupakan *freeware* yang dikembangkan oleh ZOHO corp. Pada skripsi ini menggunakan VQManager versi 6.2 yang dapat di-*download* secara bebas di http://www.manageengine.com/products/vqmanager. VQManager versi terbaru ini sudah di-*bundle* dengan WinPcap 4.0 (untuk *sniffer*), JRE 1.5.0_11 (*application server*) dan MySQL 5.0.44 (*database*). *Software* ini dapat digunakan untuk memonitoring protokol *SIP*, H.323, *SCCP* dan *RTP/RTCP*. VQManager dapat di-*install* baik di windows maupun linux.



Gambar 3.7 Tampilan menu *login* VQManager

3.4.3 Konfigurasi *Softphone* (EyeBeam)

Konfigurasi perangkat lunak telepon (*softphone*) pada komputer *client* bertujuan agar *account/extension* yang telah dibuat oleh admin dapat digunakan untuk melakukan koneksi ke FreePBX *server*. Hal pertama yang perlu dilakukan pada *softphone* yang ada di *client* adalah men-set *user* sesuai dengan *user* yang telah didaftarkan pada Asterisk. Akan ada beberapa kotak yang harus diisi diantaranya *display name, username, password, authentication user* dan *domain*.

Untuk *username* dan *password* harus sesuai dengan data pada *SIP server* yakni Trixbox, sedangkan untuk domain adalah *IP address* dari *Server VoIP*

Setelah semua konfigurasi dilakukan, secara otomatis *softphone* akan mendeteksi terhadap perubahan parameter-parameter yang telah dilakukan. Apabila dari tahap awal konfigurasi tidak menemui masalah, dan *softphone* sudah dapat terhubung dengan PBX *Server* / Trixbox maka pada *display* EyeBeam akan muncul tulisan *ready* yang menandakan bahwa *softphone* tersebut sudah siap untuk menerima ataupun melakukan panggilan dengan *user* lainnya yang berada dalam satu *domain*. Hal yang sama kita lakukan jika ingin menambahkan *account* di komputer *client* yang lain.

Gambar 3.8 Tampilan Proses Registrasi *User*

3.5 Metode Pengambilan Data

Setelah semua hal yang diperlukan untuk membangun sistem *VoIP* selesai, maka ada beberapa skenario yang akan dilakukan untuk melakukan pengujian terhadap celah keamanan dan unjuk kerja *VoIP* pada jaringan *VPN* berbasis *MPLS* dengan menggunakan *tunneling IPSec*.

- Skenario Pertama

  Skenario pertama ini dilakukan bertujuan untuk menentukan besarnya *bandwidth* minimum yang dibutuhkan oleh masing-masing *codec* agar unjuk kerjanya optimum. Pada pengujian ini, melibatkan 2 *client* di mana *user* yang berada di *Branch Office* SBY melakukan panggilan ke *user*

yang berada di *Head Office JKT*. Untuk *codec* digunakan beberapa *codec* sebagai perbandingan yaitu *codec* G.711 dengan *bit-rate* 64 Kbps, *codec* G.729 yang memiliki *bit-rate* 8 Kbps serta *codec GSM* yang memiliki *bit-rate* 13 Kbps. *Bandwidth* jaringan diubah-ubah mulai dari 16 Kbps, 32 Kbps, 64 Kbps sampai dengan 128 Kbps. Setelah terjadi percakapan antara *client* yang berada di *Branch Office* SBY dengan *client* yang berada di *Head Office* JKT maka pembicaraan yang terjadi akan di-*capture* dengan menggunakan Wireshark yang dipasang di sisi *client* yang berada di *Branch Office* SBY. Data yang di-*capture* tersebut kemudian dianalisa untuk mengetahui besar *bandwidth* minimum yang dibutuhkan oleh ketiga jenis *codec* tersebut agar dapat mencapai unjuk kerja yang optimum

- Skenario Kedua

  Jika pada skenario pertama, *bandwidth* yang digunakan diubah-ubah mulai dari 16 Kbps sampai dengan 128 Kbps serta semuanya dialokasikan hanya untuk aplikasi *VoIP* saja. Pada skenario kedua ini, dipilih salah satu *bandwidth* di mana ketiga *codec* dapat bekerja secara optimum selain itu *bandwidth* yang ada tidak hanya digunakan oleh aplikasi *VoIP* saja akan tetapi digunakan juga untuk aplikasi yang lain. Pada pengujian ini digunakan *software* TfGen untuk membangkitkan *traffic* yang diinginkan. Ada 3 jenis pengujian yang dilakukan yaitu percakapan mulai dilakukan ketika 25 % dari *bandwidth* yang ada sudah digunakan oleh aplikasi lain, ketika 50 % dari *bandwidth* yang ada sudah digunakan serta yang terakhir ketika 75 % *bandwidth* sudah digunakan oleh aplikasi yang lain. Metode pengujian ini dilakukan bertujuan untuk mengetahui seberapa besar pengaruh penggunaan *bandwidth* bersama-sama dengan aplikasi lain terhadap unjuk kerja *VoIP* yang dibangun.

- Skenario Ketiga

  Berbeda dengan 2 pengujian sebelumnya, pada skenario yang ketiga ini melibatkan 3 *user* dalam proses percakapan (*conference*). Pengujian ini dilakukan dengan tujuan untuk mengetahui seberapa besar pengaruh penambahan *user* terhadap unjuk kerja masing-masing dari 3 jenis *codec* yang digunakan. Dari ketiga skenario yang sudah dilakukan diharapkan

dapat memberikan gambaran secara jelas mana dari ketiga jenis *codec* yang digunakan yaitu G.711, G.729 dan *GSM* yang paling baik dilihat dari segi besarnya konsumsi *bandwidth*, perfomansi serta kualitas suara yang dihasilkan pada komunikasi *VoIP*.

- Skenario Keempat

  Skenario terakhir ini dilakukan untuk menguji keamanan dari sistem yang sudah dibangun. Tidak seperti pada skenario sebelum-sebelumnya, pada skenario ini hanya menggunakan 1 *codec* saja yang dianggap mempunyai unjuk kerja terbaik. Untuk menyadap pembicaraan, digunakan *software* Wireshark yang dipasang pada router *PE*. Dari data yang di-*capture*, selanjutnya coba di-*decode*-kan dan di-*playback*. Jika data tersebut tidak dapat di-*playback*, berarti sistem yang dibangun sudah aman. Selain coba untuk di-*playback*, data yang disadap tersebut juga dianalisa untuk di bandingkan dengan data *VoIP* sebelum menggunakan *VPN*. Hal ini untuk melihat seberapa besar pengaruh implementasi *VPN* pada performansi *VoIP*.

BAB 4

PENGUJIAN DAN ANALISA SISTEM

Seperti yang telah dijelaskan pada bab sebelumnya bahwa terdapat beberapa skenario yang akan dijalankan. Dari skenario tersebut akan dilakukan analisis terhadap aspek unjuk kerja, aspek keamanan serta pengaruh *VoIP* ketika ditambahkan aplikasi *VPN*. Untuk membantu analisis unjuk kerja dan keamanan, maka digunakan salah satu *software sniffing* yaitu Wireshark. *Software* yang dapat diimplementasikan baik di Linux maupun di Windows ini akan meng-*capture* semua paket yang ditransmisikan dan melakukan analisis keamanan terhadap data *VoIP* (apakah data *VoIP* yang ditransmisikan dapat direkam dan di-*playback*). Selain itu, dari data yang direkam dengan menggunakan *software* ini dapat dianalisis unjuk kerja dari *VoIP* dengan cara menghitung *delay, jitter,* dan *packet loss*. Data yang akan dianalisis adalah data dengan paket *RTP*.

4.1 Pengujian Unjuk Kerja *Codec*

Pada pengujian di bab ini, digunakan 3 jenis *codec* yakni G.711 ulaw, G729, dan *GSM*. G.711 dan G729 merupakan *codec* yang berbayar sedangkan *GSM* merupakan *codec* yang sifatnya *open source*. Pemilihan *codec* ini dilakukan dengan beberapa pertimbangan seperti perbedaan *bit-rate* di mana G.711 ulaw mempunyai *bit-rate* 64 Kbps, G.729 dengan *bit-rate* 8 Kbps sedangkan *GSM* mempunyai *bit-rate* 13 Kbps. Selain *bit-rate*, ketiga *codec* tersebut masing-masing menggunakan algoritma yang berbeda dalam kompresi datanya yaitu G.711 dengan *Pulse Code Modulation* (*PCM*), G729 dengan *Conjugate Structure Algebraic-Code Excited Linear Prediction* (*CS-ACELP*) sedang *GSM* menggunakan *Regular Pulse Excitation Long-Term Prediction* (*RPE-LTP*). Analisa unjuk kerja akan dilakukan untuk menentukan *bandwidth* minimum yang dibutuhkan oleh masing-masing *codec* tersebut, untuk selanjutnya dipilih salah satu dari ketiga *codec* tersebut yang mempunyai unjuk kerja terbaik pada jaringan *VPN* berbasis *MPLS* dengan menggunakan *tunneling IPSec*. Selain itu akan dilakukan analisis keamanan untuk melihat seberapa amankah percakapan dengan *VoIP* setelah ditransmisikan pada jaringan yang menggunakan *VPN*.

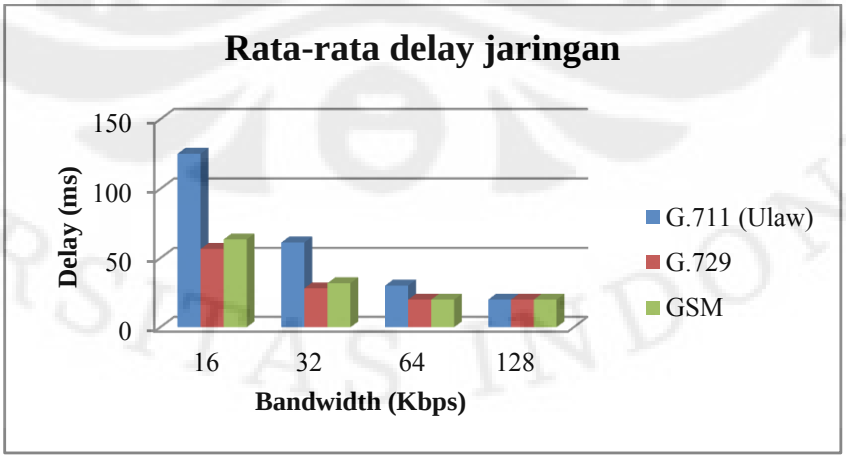
4.1.1 Pengujian dan Analisis *Delay*

Delay merupakan waktu yang diperlukan oleh paket dari terminal pengirim hingga sampai ke terminal penerima. *Delay* merupakan parameter penting untuk menentukan kualitas jaringan *VoIP*. Berdasarkan standar dari *ITU-T* G.104 untuk kualitas *VoIP* yang baik, *delay* harus < 150 ms agar tidak terjadi *overlap* pada komunikasi.

Pada skenario pertama ini, seperti telah dijelaskan pada bab sebelumnya, dilakukan panggilan dari *client* yang ada di *Branch Office* ke *client* yang ada di *Head Office* dimana *bandwidth* diubah-ubah mulai dari 16 Kbps, 32 Kbps, 64 Kbps sampai dengan 128 Kbps. Untuk meng-*capture* paket yang masuk ke *client* di gunakan Wireshark. Kemudian paket tersebut dihitung *latency*-nya untuk mengetahui besarnya *latency* yang muncul dan dibandingkan dengan standar yang ditetapkan oleh *ITU-T*. Adapun hasil dari pengukuran *delay* tersebut dapat dilihat dari Tabel 4.1 di bawah ini

Tabel 4.1 Rata-rata *delay* jaringan

Bandwidth (Kbps)	G.711 (Ulaw) *Delay* (ms)	G.729 *Delay* (ms)	*GSM* *Delay* (ms)
16	125.232	56.7	63.616
32	61.38	28.19	31.886
64	30.134	19.99	19.99
128	19.99	19.99	19.99



Gambar 4.1 Grafik rata-rata *delay* jaringan

Dari Gambar 4.1 di atas dapat diketahui bahwa *delay* berbanding terbalik dengan *bandwidth*. Semakin besar *bandwidth* maka *delay* akan semakin kecil. Grafik menunjukkan bahwa untuk *bandwidth* yang makin kecil, peningkatan *delay* yang paling besar dimiliki oleh *Codec* G.711. Hal ini disebabkan karena G.711 memiliki *payload* yang paling besar yaitu 160 bytes. Perlu diperhatikan bahwa semakin besar *payload*, maka *delay* paketisasi, routing, transmisi dan *switching* akan semakin besar. *Delay* terkecil untuk *Codec* G.711 diperoleh ketika *bandwidth*-nya 128 Kbps, sedangkan untuk G.729 dan *GSM*, *delay* terkecil (19 ms) mulai dicapai ketika *bandwidth* di-set 64 Kbps

Pengukuran *Delay* Total

Dalam teknologi *VoIP*, parameter *delay* disebabkan oleh beberapa komponen *delay* yaitu *delay coder* (*processing*), *delay packetization*, dan *delay network*.

- *Coder (Processing) Delay*

  Coder (Processing) = (Waktu kompresi) + (Waktu dekompresi) + *algorithmic delay* [1]

  - Untuk G.711 :

    Waktu kompresi = 3 x *frame size* + *look ahead* [6]
    = 3 x 0.125 ms + 0 ms
    = 0.375

    Waktu dekompresi = 10 % x waktu kompresi
    = 0.1 x 0.375 ms
    = 0.0375 ms

    Algoritmic delay (G.711) = 0 ms [1]

    Jadi, *Coder (Processing) Delay* = 0.4125 ms

  - Untuk G.729

    Waktu kompresi = 3 x *frame size* + *look ahead*
    = 3 x 10 ms + 5 ms
    = 35 ms

Waktu dekompresi = 10 % x waktu kompresi
= 0.1 x 35 ms
= 3.5 ms

Algoritmic delay (G.729) = 5 ms

Jadi, *Coder (Processing) Delay* = 43.5 ms

- Untuk *GSM* :

Waktu kompresi = 3 x *frame size* + *look ahead*
= 3 x 20 ms + 0 ms
= 60 ms

Waktu dekompresi = 10 % x waktu kompresi
= 0.1 x 60 ms
= 6 ms

Algoritmic delay (*GSM*) = 7.5 ms

Jadi, *Coder (Processing) Delay* = 73.5 ms

➢ *Packetization Delay*

Mengacu pada hasil pengukuran *network analyzer* (Wireshark) diketahui panjang paket *VoIP*, besar data informasi paket *VoIP* dapat diperoleh dengan cara sebagai berikut:

Voice payload size = Panjang paket IP – (*Ethernet header* + *IP Header* + *UDP header* + *RTP header*)

- Untuk G.711 :

  Payload = 214 byte – (14+20+8+12) byte
  = 160 byte

  Untuk teknik kompresi G.711 dengan besar *payload* 160 byte maka *delay* paketisasinya adalah 20 ms [1]

- Untuk G.729

  Payload = 74 byte – (14+20+8+12) byte
  = 20 byte

Untuk teknik kompresi G.729 dengan besar *payload* 20 byte maka *delay* paketisasinya adalah 20 ms [1]

- Untuk *GSM*

  Payload = 87 – (14+20+18+12) byte

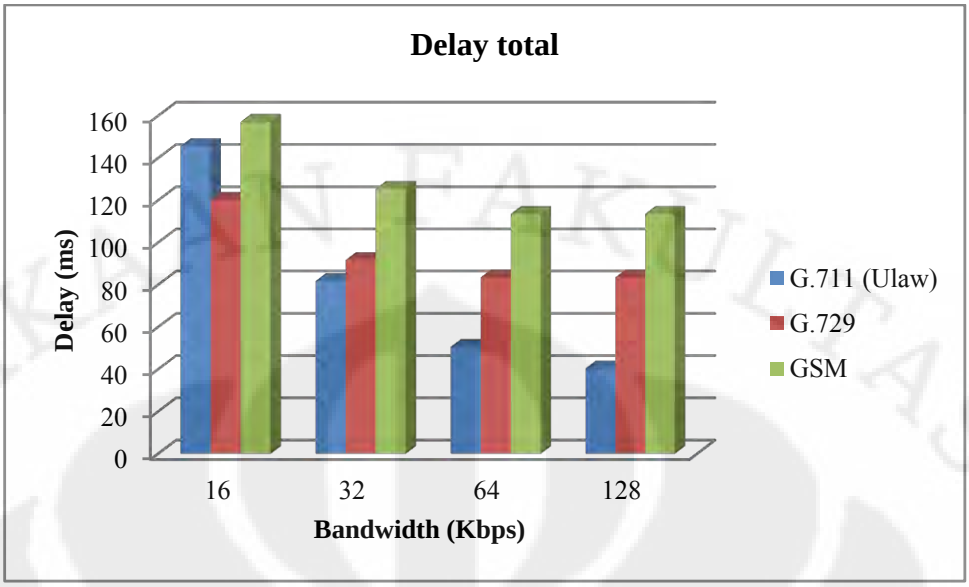
  = 33 byte

  Untuk teknik kompresi *GSM* dengan *payload* 33 byte maka *delay* paketisasi adalah 20 ms [1]

Berdasarkan data yang didapat dari hasil pengukuran tersebut maka *one way delay* dapat dihitung dengan menjumlahkan *coder processing delay, packetization delay* dan *network delay*

Tabel 4.2 *Delay* total masing-masing *codec*

Jenis *Codec*	*Bandwidth* (Kbps)	*Delay* Jaringan (ms)	*Delay* Prosesing (ms)	*Delay* Paketisasi (%)	*Delay* Total (ms)
G.711 (Ulaw)	16	125.232	0.4125	20	145.64
	32	61.38	0.4125	20	81.79
	64	30.134	0.4125	20	50.55
	128	19.99	0.4125	20	40.4
G.729	16	56.7	43.5	20	120.2
	32	28.19	43.5	20	91.69
	64	19.99	43.5	20	83.49
	128	19.99	43.5	20	83.49
GSM	16	63.616	73.5	20	157.12
	32	31.886	73.5	20	125.39
	64	19.99	73.5	20	113.49
	128	19.99	73.5	20	113.49



Gambar 4.2 Grafik *delay* total masing-masing *codec*

Setelah ditambahkan dengan *delay* lainnya, ternyata *delay* total masih bisa diterima (*acceptable for most application*) berdasarkan standar G.114 *ITU-T*.

4.1.2 Pengujian dan Analisis *Jitter*

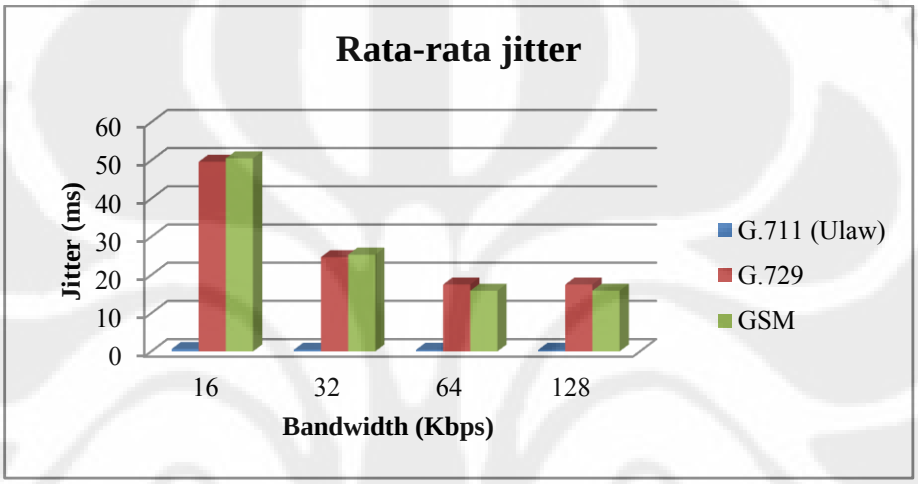
Jitter merupakan variasi *delay* yang terjadi karena waktu kedatangan paket yang berbeda-beda. Secara sederhana bisa dikatakan bahwa *jitter* adalah perbedaan waktu kedatangan antara 1 paket dengan paket setelahnya. Parameter *jitter* perlu untuk dianalisis untuk mengetahui *delay* kedatangan antar satu paket dengan paket lainnya. Semakin besar *jitter* maka perbedaan waktu antara suara asli dengan suara yang terdengar akan semakin besar. Hal ini dapat menyebabkan besarnya *collision* antara paket bahkan dapat menyebabkan *echo cancelation*

Skenario Pengukuran *Jitter*

Pengukuran *jitter* dilakukan bersamaan dengan pengukuran *delay* dan *packet loss*. Paket *VoIP* yang lewat di-*capture* dan dianalisa. Adapun hasil pengukuran tampak pada Tabel 4.3 di bawah ini

Tabel 4.3 Rata-rata *jitter*

Bandwidth (Kbps)	G.711 (Ulaw) *Jitter* (ms)	G.729 *Jitter* (ms)	*GSM* *Jitter* (ms)
16	0.628	49.624	50.544
32	0.408	24.67	25.322
64	0.386	17.498	15.87
128	0.386	17.5	15.874



Gambar 4.3 Grafik rata-rata *jitter*

Analisa Pengujian *Jitter*

ITU-T O.172 merekomendasikan *jitter* yang baik adalah <30 ms. Sedangkan pada hasil pengujian terhadap 3 *codec*, untuk *codec* G.711 mempunyai nilai *jitter* yang sangat kecil hal ini dikarena *codec* ini mempunyai nilai *frame size* yang kecil yaitu 0.125 ms jika dibandingkan dengan 2 *codec* yang lainnya yaitu G.729 (10 ms) dan *GSM* (20 ms) sehingga meskipun *bandwidth* jaringan hanya 16 Kbps nilainya *jitter*-nya tetap memenuhi standar *ITU-T* yaitu 0.628. Untuk *codec* G.729 dan *codec GSM* nilai *jitter*-nya baru memenuhi standar *ITU-T* ketika digunakan pada jaringan dengan *bandwidth* 32 Kbps ke atas, sedangkan untuk *bandwidth* di bawah itu tidak memenuhi. *Jitter* merupakan salah satu faktor yang mempengaruhi kualitas suara. Semakin besar *jitter* maka suara yang dihasilkan akan semakin tidak jelas (terputus-putus). Nilai *jitter* berpengaruh ketika paket *RTP* yang datang akan diproses menjadi suara. Ketika nilai *jitter* lebih kecil dari waktu pemrosesan paket data, maka sebelum paket selesai diproses paket

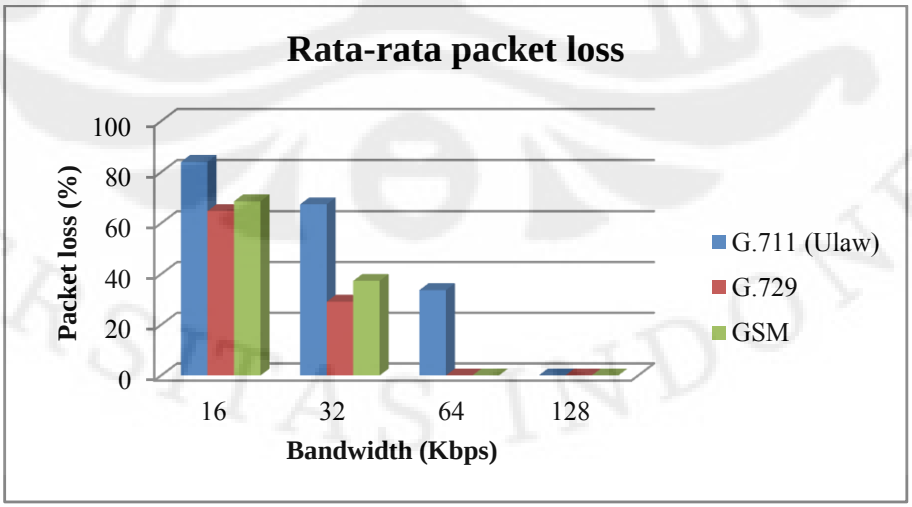
selanjutnya telah datang untuk menunggu diproses. Sehingga suara yang dihasilkan pun bagus. Ketika nilai *jitter* lebih besar dari *delay processing* maka suara akan terdengar terputus-putus. Hal ini dapat diantisipasi dengan menggunakan *jitter buffer*. Dengan demikian paket yang datang akan di-*buffer* terlebih dahulu sebelum diproses. Tetapi ketika *jitter* dari paket jauh lebih besar dari *buffer jitter*, maka kualitas suara akan menjadi buruk. Hal ini terjadi ketika menggunakan *codec* G.711 dengan *bandwidth* 16 Kbps, 32 Kbps dan 64 Kbps

4.1.3 Pengujian dan Analisis *Packet loss*

Packet loss menentukan besarnya paket yang hilang di dalam perjalanannya dari *source address* ke *destination address*. Semakin besar *packet loss* menyebabkan suara yang dikirim tidak akan bisa didengarkan (hilang). Berikut *packet loss* yang diperoleh dengan menggunakan Wireshark untuk 3 jenis *codec* yang berbeda

Tabel 4.4 Rata-rata *packet loss*

Bandwidth (Kbps)	G.711 (Ulaw)	G.729	*GSM*
	Packet loss (%)	*Packet loss* (%)	*Packet loss* (%)
16	84.034	64.756	68.57
32	67.41	29.084	37.3
64	33.658	0	0
128	0	0	0



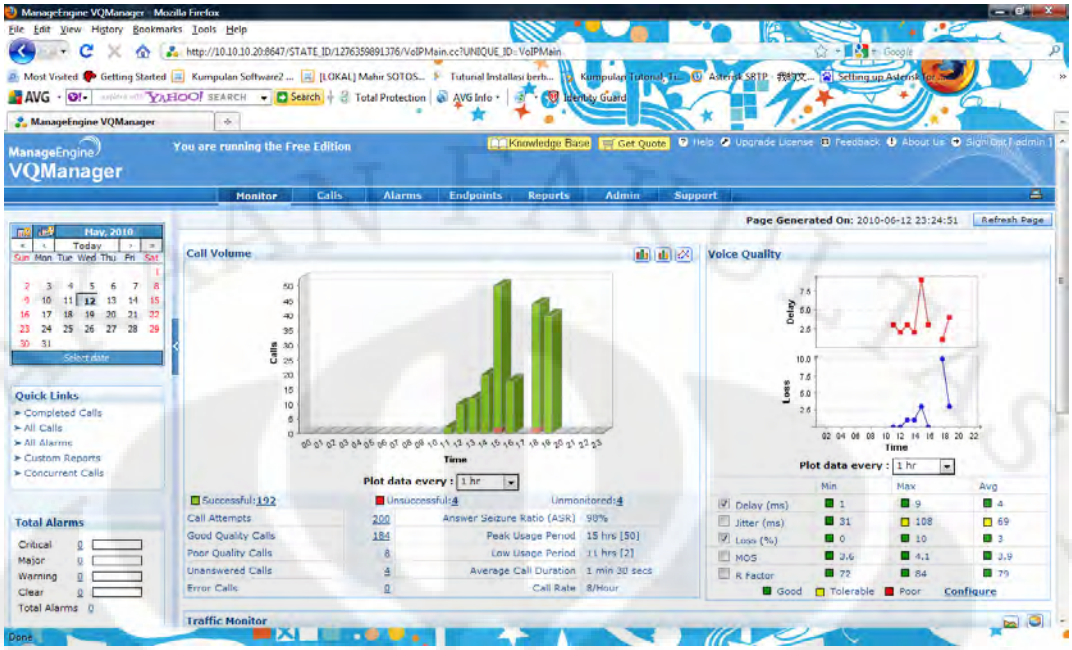
Gambar 4.4 Grafik rata-rata *packet loss*

Analisa *Packet loss*

Menurut standar *ITU-T packet loss* yang masih dapat diterima yaitu 10 % sampai 30% [3]. Untuk *codec* G.711 standar tersebut baru terpenuhi ketika diberi *bandwidth* 128 Kbps, sedangkan untuk *codec* G.729 dan *GSM* dengan *bandwidth* 64 Kbps *paket loss* yang terjadi sudah dibawah 30 %. *VoIP* menggunakan protokol *UDP* untuk mengirimkan data. Berbeda dengan *TCP* yang bersifat *connection oriented*, protokol *UDP* tidak mengirim ulang data yang hilang atau tidak sesuai. Hal ini menyebabkan kemungkinan data yang hilang tidak dapat diganti (*recovery*). Apalagi bila data dengan ukuran yang besar dikirimkan dengan *bit-rate* yang rendah. Hal ini terjadi pada *codec* G.711 dengan *bandwidth* rendah menyebabkan kemungkinan *collision* antar data menjadi lebih tinggi. Ini jelas akan menyebabkan *packet loss* meningkat. Dengan *packet loss* yang tinggi, maka suara yang terkirim tidak akan diterima dengan baik di sisi penerima.

4.1.4 Menentukan *MOS* dan *R.Factor*

Pada skripsi ini digunakan *software* Voice Quality Manager (VQManager) untuk menentukan nilai *MOS* dan *R.Factor* dari 3 jenis *codec* yang digunakan. *R.Factor* ditentukan dengan melihat beberapa parameter seperti *signal to noise ratio* dan *echo* perangkat, *codec* dan kompresi, *packet loss* dan *delay*. Nilai *R.Factor* yang diperoleh selanjutnya dikonversi ke dalam nilai *MOS*. Pada gambar 4.5 yang merupakan tampilan awal dari *software* VQManager terlihat grafik yang menggambarkan semua panggilan yang sudah dilakukan baik yang berhasil (berwarna hijau) maupun panggilan yang gagal (warna merah). Di sebelah kanan tampak grafik rata-rata dari *delay, jitter, packet loss* dan *MOS* dari semua panggilan yang dilakukan dalam 1 hari



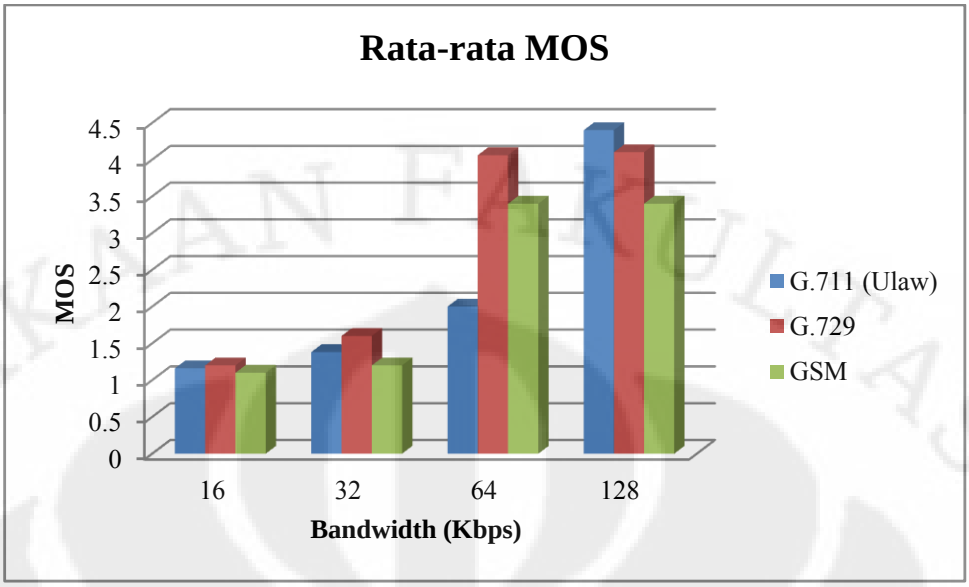
Gambar 4.5 Tampilan awal dari VQ Manager

Hasil Pengukuran Nilai *MOS* dan *R.Factor*

Pengukuran nilai *MOS* dan *R.Factor* ini dilakukan bersamaan dengan pengambilan data untuk *delay, jitter* dan *packet loss*. Setelah 1 sesi percakapan selesai, VQManager akan menghitung besarnya nilai *MOS* dan *R.Factor* dari *user* pemanggil ke *server* dan dari *user* penerima ke *server*. Jadi untuk 1 kali sesi percakapan antara 2 *user* akan diperoleh 2 nilai *MOS* dan *R.Factor*.

Tabel 4.5 Nilai *MOS* dan *R.Factor* masing-masing *Codec*

Bandwidth (Kbps)	G.711 (Ulaw)		G.729		*GSM*	
	MOS	*R.Factor*	*MOS*	*R.Factor*	*MOS*	*R.Factor*
16	1.16	15.8	1.2	20	1.1	14
32	1.38	24.4	1.6	29	1.2	20
64	2	36	4.06	81	3.4	66.8
128	4.4	93	4.1	82	3.4	66.8



Gambar 4.6 Grafik nilai *MOS* masing-masing *Codec*

Menurut rekomendasi *ITU-T* P.800 nilai *MOS* yang masih dapat diterima adalah > 3. Dari Gambar 4.6 di atas, tampak bahwa untuk *codec* G.711 standar itu baru terpenuhi ketika digunakan pada jaringan yang mempunyai *bandwidth* 128 Kbps ke atas hal ini dikarena *bit-rate* yang cukup besar untuk *codec* ini yaitu 64 Kbps. Adapun untuk *codec* G.729 dan *GSM*, karena mempunyai *bit-rate* yang kecil maka hanya dengan menggunakan *bandwidth* sebesar 64 Kbps sudah dapat diperoleh nilai *MOS* yang maksimal. Dapat diketahui pula bahwa dari ketiga jenis *codec* yang digunakan, ternyata G.711 merupakan jenis *codec* yang mempunyai kualitas yang paling bagus (nilai *MOS* 4.4) hal ini dikarenakan jenis *codec* ini mempunyai *payload* yang paling besar jika dibandingkan dengan 2 jenis *codec* lainnya (G.729 dan *GSM*) yaitu sebesar 160 bytes akan tetapi *codec* ini mempunyai masalah di sisi konsumsi *bandwidth*. Agar diperoleh unjuk kerja yang optimal, *codec* G.711 ini minimal membutuhkan *bandwidth* sebesar 128 Kbps. Salah satu cara untuk mengatasi problem ini yaitu dengan menggunakan *codec* G.729. Hanya dengan menggunakan setengah dari *bandwidth* yang dibutuhkan oleh *codec* G.711 yaitu 64 Kbps, sudah dapat diperoleh nilai *MOS* yang optimum untuk *codec* G.729 yaitu sebesar 4.1. Adapun untuk jenis *codec GSM* nilai *MOS* optimum yang dimiliki tidaklah telalu baik tapi masih memenuhi rekomendasi dari ITU-T yaitu 3.4, akan tetapi jenis *codec* ini mempunyai beberapa kelebihan yaitu selain hanya butuh *bandwidth* yang kecil (64 Kbps) untuk mencapai unjuk

kerja yang maksimum, *codec* ini juga bersifat *open source* sehingga dapat dipergunakan secara gratis.

4.1.5 Analisa Trafik dengan *SNMP Traffic Grapher (STG)*

Selain *delay, jitter* dan *packet loss* satu faktor penting yang perlu diperhatikan dalam pemilihan jenis *codec* yang harus digunakan dalam komunikasi dengan *VoIP* adalah *bandwidth*. *Codec* yang baik, adalah *codec* yang tetap dapat bekerja dengan optimum meskipun diberi *bandwidth* yang kecil. Pada pengujian ini, digunakan suatu *freeware* yang berjalan pada *platform* Windows yang bernama *SNMP Traffic Grapher (STG). STG* ini akan menangkap secara *real-time* trafik-trafik yang masuk ataupun keluar pada perangkat yang dimonitor. Pada skripsi ini, *STG* digunakan untuk memonitoring utilisasi yang ada pada router *Regional Office* SBY. Secara teori besarnya *bandwidth* minimum yang dibutuhkan pada aplikasi *VoIP* untuk satu kali panggilan pada masing-masing jenis *codec* yang digunakan dapat dihitung dengan menggunakan rumus sebagai berikut:

Total ukuran paket = *Voice Payload* + *RTP Header* (12 bytes) + *UDP Header* (8 bytes) + *IP Header* (20 bytes) + *Link Header* [9]

Packet Per Second (PPS) = Codec bit rate / *Voice Payload* [9]

Konsumsi *Bandwidth* = Total ukuran paket x *PPS* [9]

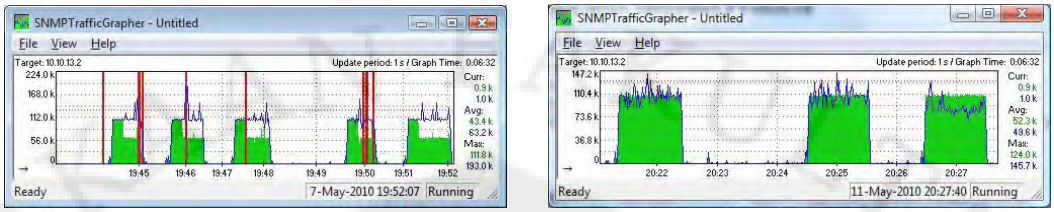
- *Bandwidth* untuk G.711

  Total Ukuran Paket = 160 bytes + 40 bytes + 14 bytes
  = 214 bytes
  = 1712 bit

  Packet Per Second = (64 Kbps) / (1280 bit)
  = 50

  Total *Bandwidth* = (1712 bit) x 50
  = 85.6 Kbps

Utilisasi yang di-*capture* dengan menggunakan *STG* seperti pada Gambar 4.7 di bawah



64 Kbps 128 Kbps

Gambar 4.7 Trafik G.711 dengan *bandwidth* berbeda

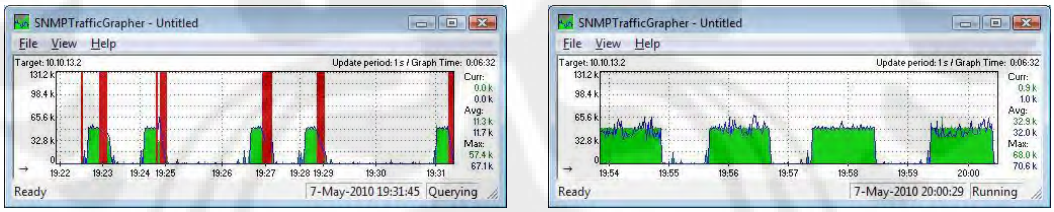
- *Bandwidth* untuk G.729

  Total Ukuran Paket = 20 bytes + 40 bytes + 14 bytes
  = 74 bytes
  = 592 bit

  Packet Per Second = (8 Kbps) / (160 bit)
  = 50

  Total *Bandwidth* = (592 bit) x 50
  = 29.6 Kbps

Utilisasi yang di-*capture* dengan menggunakan *STG* tampak pada Gambar 4.8



32 Kbps 64 Kbps

Gambar 4.8 Traffic G.729 dengan *bandwidth* berbeda

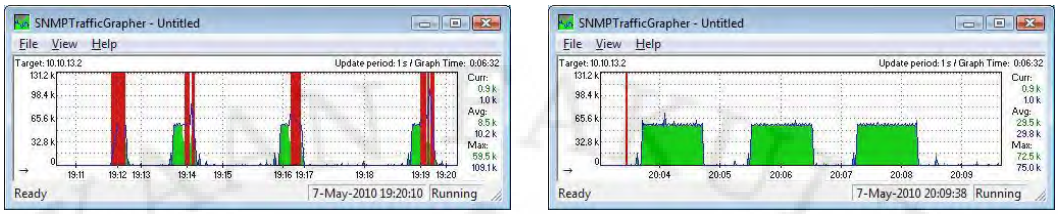
- *Bandwidth* untuk *GSM*

  Total Ukuran Paket = 33 bytes + 40 bytes + 14 bytes
  = 87 bytes
  = 696 bit

  Packet Per Second = (13 Kbps) / (264 bit)
  = 49.2

  Total *Bandwidth* = (696 bit) x 49.2
  = 34.2 Kbps

Utilisasi yang di-*capture* dengan menggunakan *STG* tampak pada Gambar 4.9



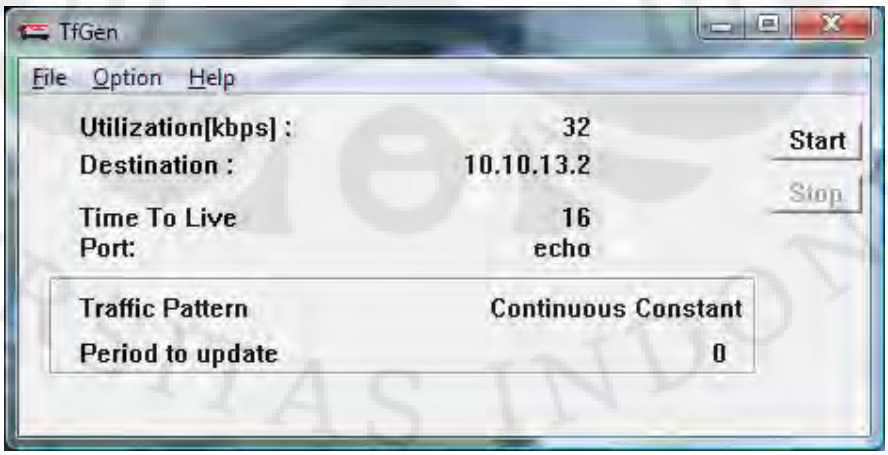
32 Kbps 64 Kbps

Gambar 4.9 Traffic *GSM* dengan *bandwidth* berbeda

Pada grafik utilisasi di atas, tampak bahwa untuk *codec* G.729 yang mempunyai *bitrate* sebesar 8 Kbps dan *codec GSM* dengan *bitrate* sebesar 13 Kbps, minimum dibutuhkan *bandwidth* sebesar 64 Kbps agar tidak terdapat *packet loss*. Adapun *codec* G.711 yang mempunyai *bit-rate* paling besar yaitu 64 Kbps memerlukan *bandwidth* minimum 128 Kbps agar tidak ada *packet loss*.

4.2 Pengujian *VoIP* jika Digunakan Bersamaan dengan Aplikasi Lain

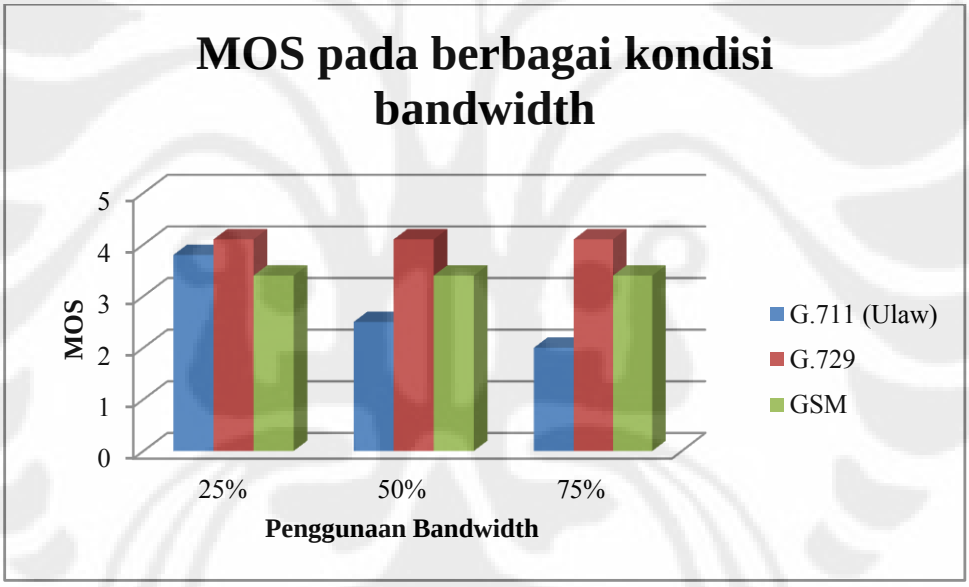
Jika pada skenario pertama, *bandwidth* jaringan diubah-ubah untuk mencari *bandwidth* minimium yang dibutuhkan oleh masing-masing *codec*, pada skenario yang kedua ini, *bandwidth* jaringan diset sebesar 128 Kbps. Hanya saja *bandwidth* tersebut tidak hanya dipergunakan untuk aplikasi *VoIP* saja, tapi juga digunakan bersama-sama dengan aplikasi yang lain. Pada pengujian ini digunakan *software* TFGen (*Traffic Generator*) versi 1.0 untuk membangkitkan trafik yang dikirimkan ke router *Regional Office* SBY.



Gambar 4.10. *Software* yang digunakan untuk membangkitkan trafik

Tabel 4.6 Unjuk kerja *codec* pada berbagai kondisi *bandwidth*

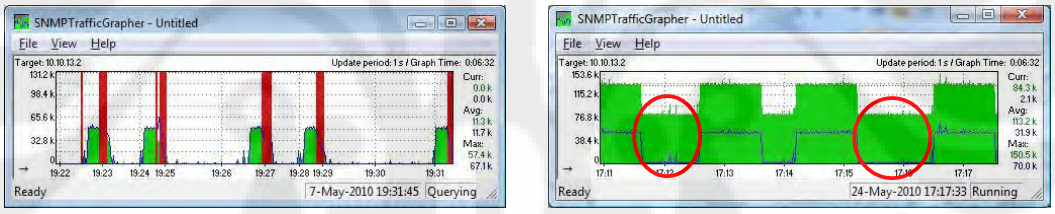
Jenis *Codec*	Penggunaan *Bandwidth* (Kbps)	*Delay* (ms)	*Jitter* (ms)	*Packet loss* (%)	*MOS*	R.Factor
G.711 (Ulaw)	25%	21.206	0.362	5.746	3.8	74
	50%	25.39	1.226	21.276	2.5	48
	75%	29.49	1.176	32.21	2	40
G.729	25%	19.988	17.5	0	4.1	82
	50%	19.99	17.5	0	4.1	82
	75%	19.99	17.51	0	4.1	82
GSM	25%	19.99	15.89	0	3.4	67
	50%	19.986	15.88	0	3.4	67
	75%	19.99	15.88	0	3.4	67



Gambar 4.11. Grafik *MOS* pada berbagai kondisi penggunaan *bandwidth*

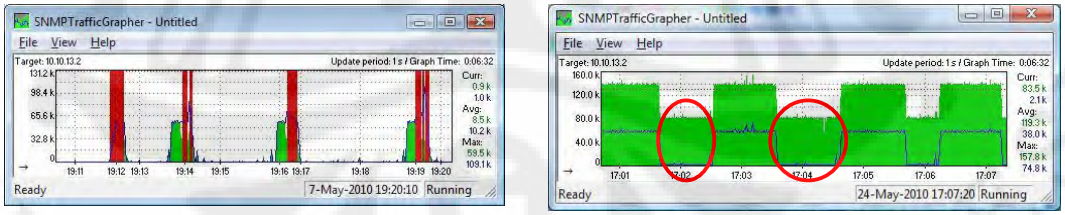
Dari hasil pengujian, terlihat bahwa dengan *bandwidth* sebesar 128 Kbps dimana 50 % dari *bandwidth* tersebut sudah dipergunakan untuk aplikasi satu arah (misalnya *ftp*, *video streaming*, dan lain sebagainya), unjuk kerja dari *codec* G.711 sudah di bawah standar (Nilai *MOS* yang diperoleh hanya sebesar 2.5) hal ini dikarenakan besarnya *bandwidth* yang tersisa untuk digunakan pada aplikasi *VoIP* adalah sebesar 64 Kbps padahal pada pengujian sebelumnya diketahui bahwa minimum *bandwidth* yang dibutuhkan oleh jenis *codec* ini adalah sekitar 85 Kbps. Adapun untuk dua jenis *codec* yang lain (G.729 dan *GSM*) unjuk kerjanya tetap

terjaga meskipun 75 % dari *bandwidth* yang ada sudah digunakan, hal ini dimungkinkan karena kedua *codec* ini hanya membutuhkan minimum *bandwidth* sebesar 30 Kbps. Ada sedikit perbedaan nilai *MOS* yang diperoleh dari hasil percobaan sebelumnya (gambar 4.6) dengan nilai *MOS* yang diperoleh pada pengujian ini, di mana pada percobaan sebelumnya, ketika *bandwidth* jaringan di-set 32 Kbps nilai *MOS* dari *codec GSM* dan G.729 masih di bawah standar *ITU-T* akan tetapi pada pengujian ini, ketika 75 % dari *bandwidth* yang ada sudah digunakan (yang berarti hanya tersisa 32 Kbps untuk aplikasi *VoIP*) kualitas suara dengan menggunakan kedua jenis *codec* ini masih sesuai standar *ITU-T*. Hal ini terjadi karena sebenarnya *bandwidth* yang tersisa sebesar 32 Kbps hanya untuk trafik *incoming* saja sedangkan di sisi *outgoing, bandwidth* yang tersedia masih 128 Kbps sehingga ketika dipergunakan untuk aplikasi *VoIP* kualitas suara masih tetap terjaga. Untuk lebih jelasnya dapat dilihat pada Gambar 4.12 dan Gambar 4.13 di bawah



(a) (b)

Gambar 4.12 (a) Trafik G.729 dengan *bandwidth* 32 Kbps, (b) Trafik G.729 dengan 75% *bandwidth* digunakan oleh aplikasi lain



(a) (b)

Gambar 4.13 (a) Trafik *GSM* dengan *bandwidth* 32 Kbps, (b) Trafik *GSM* dengan 75% *bandwidth* digunakan oleh aplikasi lain

Pada Gambar 4.12 (b) dan 4.13 (b) di atas terlihat bahwa *bandwidth* yang dipergunakan oleh aplikasi lain sebesar 92 Kbps hanya untuk trafik *incoming* saja (di dalam lingkaran merah) sedangkan untuk trafik *outgoing* (grafik yang

berwarna biru) semua *bandwidth* yang ada (128 Kbps) dipergunakan untuk aplikasi *VoIP*. Sehingga dengan komposisi *bandwidth* seperti ini, ketika digunakan untuk aplikasi *VoIP* dengan *codec* G.729 dan *GSM bandwidth* yang ada mencukupi dan tidak terdapat *packet loss*. Berbeda ketika *bandwidth* di-set 32 Kbps baik untuk trafik *incoming* maupun *outgoing*-nya (Gambar 4.12 (a) dan 4.13 (a)) ketika digunakan untuk aplikasi VoIP, terdapat cukup banyak *packet loss*.

4.3 Analisa Pengujian *Codec* untuk *Multi Call* (3 *User*)

Jika sebelumnya, pengujian hanya berfokus pada 2 *user* saja, pada skenario yang ketiga ini dilakukan pengujian untuk proses percakapan yang dilakukan oleh 3 *user* secara bersamaan (*conference*). Di sini alokasi *bandwidth* yang diberikan pada jaringan adalah sebesar 64 Kbps dengan pertimbangan bahwa unjuk kerja untuk *codec* G.729 dan *GSM* sudah bisa optimal jika diberikan *bandwidth* sebesar itu.

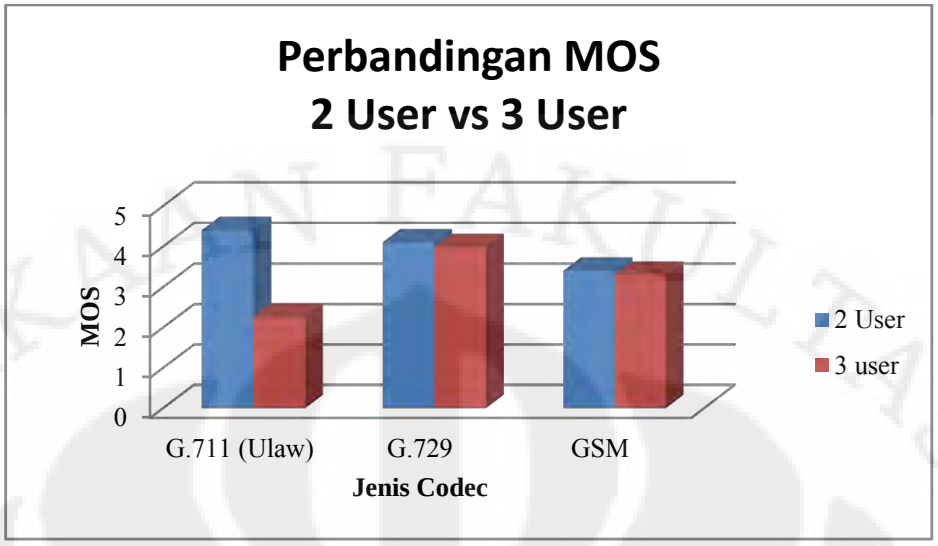
Tabel 4.7 Unjuk kerja *Codec* untuk *conference call* pada *bandwidth* 64 Kbps

Jenis *Codec*	*Delay* (ms)	*Jitter* (ms)	*Packet loss* (%)	*MOS*	*R.Factor*
G.729	56.898	49.83	28.922	1.64	30.8
GSM	56.942	45.29	32.826	1.2	17.8

Dari Tabel 4.7 di atas, tampak bahwa terdapat pengaruh yang cukup signifikan pada saat dilakukan *conference call*. Baik untuk *codec* G.729 maupun *GSM*, kualitas suara sangat buruk terlihat dari nilai *MOS* yang di bawah standar dari ITU-T yaitu 1.64 untuk G.729 dan 1.2 untuk *GSM*. Hal ini terjadi karena *bandwidth* 64 Kbps tidak mencukupi jika digunakan untuk 3 *user* bahkan pada *codec* G.711 *conference call* sama sekali tidak dapat dilakukan

Tabel 4.8 Unjuk kerja *codec* untuk *conference call* pada *bandwidth* 128 Kbps

Jenis *Codec*	*Delay* (ms)	*Jitter* (ms)	*Packet loss* (%)	*MOS*	*R.Factor*
G.711 (Ulaw)	48.85	2.644	27.562	2.26	43
G.729	22.232	19.46	0.5	3.98	79.6
GSM	26.564	21.1	7.958	3.3	64.6



Gambar 4.14 Grafik perbandingan *MOS* 2 *user* dengan 3 *user*

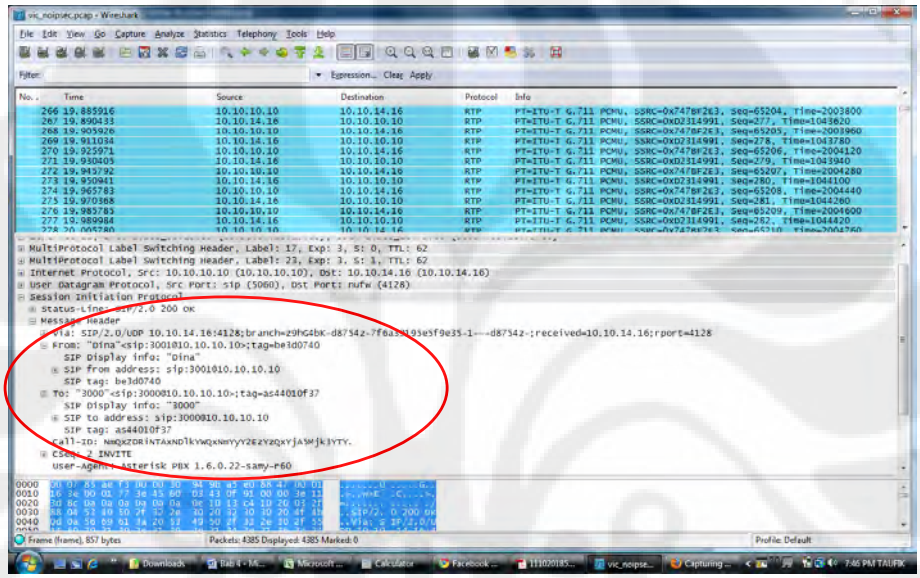
Pada saat *bandwidth* ditambah menjadi 128 Kbps, *conference call* dengan menggunakan *codec* G.711 sudah dapat dilakukan hanya saja kualitasnya masih di bawah standar (nilai *MOS* 2.26), hal ini berarti dibutuhkan *bandwidth* yang lebih besar lagi agar kualitas suara bisa lebih baik. Sedangkan untuk *codec* G.729 dan *GSM*, kualitas suara yang dihasilkan sudah di atas standar, meskipun terjadi penurunan jika dibandingkan pada saat percakapan hanya melibatkan 2 user saja.

4.4 Pengujian dan Analisis Keamanan *VoIP*

Ancaman keamanan pada jaringan *VoIP* dapat dilakukan dengan banyak cara, seperti yang telah dijelaskan dalam Bab 2, mulai dari *hijacking, spoofing, Man in the middle attack, file capturing, tapping*, dan lain sebagainya. Pada skenario kali ini akan dilakukan pengujian keamanan jaringan yang sudah dibangun. *Software* yang digunakan masih tetap sama yaitu Wireshark, karena *software* ini selain bisa digunakan untuk meng-*capture* paket-paket yang lewat dalam jaringan, juga bisa digunakan untuk merekam pembicaran. *Software* ini selanjutnya akan dipasang pada router *PE* dan akan digunakan untuk merekam pembicaraan 2 *user* (*user* di *Head Office* dan *Branch Office*). Apabila rekaman data *VoIP* tersebut dapat di-*playback*, maka dapat disimpulkan bahwa jaringan *VoIP* yang sudah diimplementasikan tidak aman

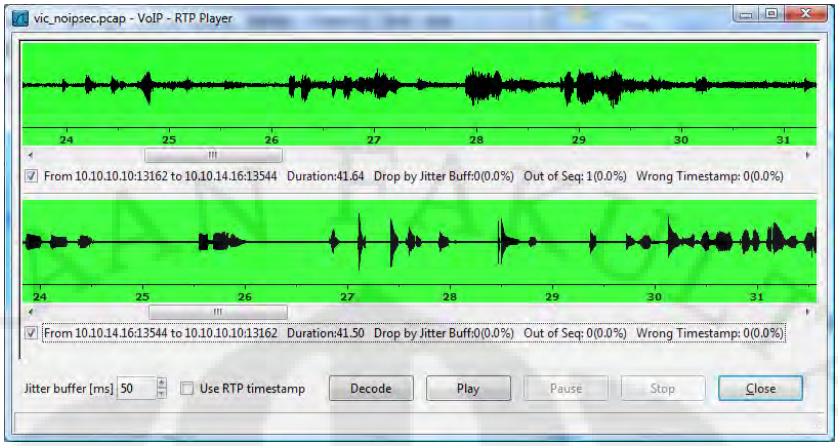
4.4.1 Pengujian Keamanan *VoIP* pada Jaringan *MPLS* tanpa *VPN*

Pada metode yang pertama ini, *tunnel IPSec* belum diimplementasikan pada jaringan. Hal ini dilakukan untuk menguji seberapa amankah komunikasi *VoIP* pada jaringan *MPLS* jika tidak disertai dengan enkripsi data. Paket-paket data dari hasil pembicaraan 2 user yang di-*capture* dengan menggunakan Wireshark dapat dilihat pada Gambar 4.15 di bawah



Gambar 4.15 Data *VoIP* yang di-*capture*

Dari Gambar 4.15 di atas, terlihat secara jelas identitas *user* yang melakukan percakapan. Di mana terlihat bahwa *user* dengan ektensi 3001 atas nama "Dina" melakukan panggilan ke *user* dengan ekstensi 3000. Selain data ekstensi, dengan menggunakan Wireshark ini diketahui pula *IP address* dari *user-user* yang melakukan percakapan yaitu ekstensi 3001 dengan *IP addres* 10.10.14.16 sedangkan *user* dengan ekstensi 3000 mempunyai *IP address* 10.10.10.10. Tidak cukup sampai di sini, *IP Address* dari *server SIP* juga terlihat dengan jelas yaitu 10.10.10.10. Hal ini jelas sangat berbahaya, sebab dengan mengetahui IP *server SIP*, para *hacker* dapat dengan mudah menyerang sistem *VoIP* yang dibangun salah satunya dengan melakukan *Denial of Service* (*DoS*) pada *server* SIP. Selain informasi detail dari *user-user* yang melakukan percakapan, ternyata hasil percakapan yang terjadi diantara kedua *user* tersebut dapat direkam dan di-*playback*.



Gambar 4.16 Sinyal suara dari 2 *user* yang direkam

Analisa Hasil Pengujian Keamanan *VoIP* pada Jaringan *MPLS* tanpa *VPN*

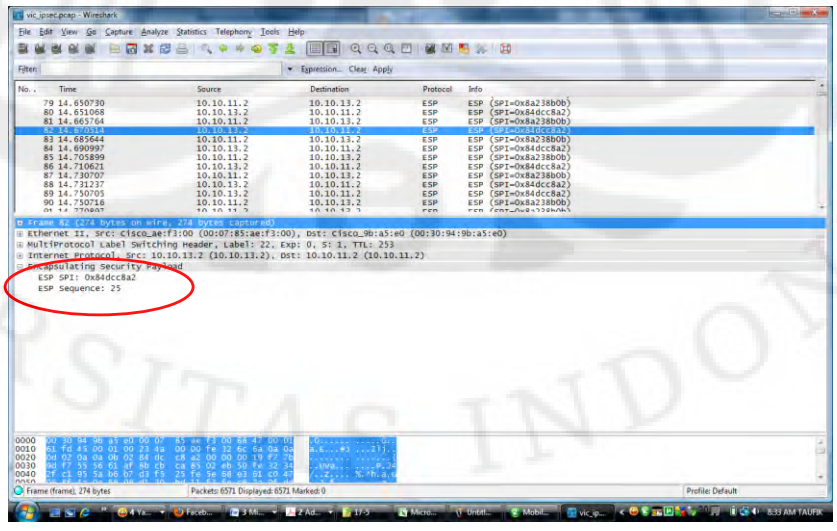
Pada skenario di atas telah dilakukan pengujian keamanan *VoIP* pada jaringan *MPLS* tanpa menggunakan *VPN* yang dibangun. Secara garis besar pengujian dilakukan dengan 2 cara. Pertama yaitu menangkap paket-paket yang lewat dan kemudian menganalisa isi dari paket tersebut untuk mengetahui celah keamanan yang bisa ditembus. Sedangkan cara yang kedua adalah dengan merekam percakapan yang terjadi untuk kemudian di-*playback* sehingga bisa diketahui isi dari percakapan yang terjadi. Hasil dari paket data yang tertangkap dapat diketahui bahwa untuk paket data *VoIP* tidak ada metode keamanan yang digunakan untuk mengamankan paket data *RTP*. Dari paket data tersebut kita dapat melihat secara jelas IP dari *user* yang melakukan panggilan, *user* yang dipanggil bahkan IP dari *server VoIP*. Jika alamat *client* bisa diketahui maka dengan menggunakan *software Network Scanner*, kita dapat dengan mudah mengetahui lokasi dari pembicara. Dengan demikian kita dapat dengan mudah merekam pembicaraan secara lebih spesifik. Selain itu banyak metode *hacking* yang dapat digunakan untuk menyerang *VoIP server* jika alamat *server* itu diketahui, seperti *registration hijacking, call hijacking, Denial of Service, man in the middle attack*, dan masih banyak lagi yang lainnya.

Selain itu, data *VoIP* yang lewat ternyata bisa di-*playback* hanya dengan men-*decode*-kan kembali data tersebut dengan menggunakan *software* Wireshark. Hal ini terjadi karena *payload* dari paket *VoIP* tidak dilindungi. *Codec* yang

digunakan berfungsi untuk melakukan kompresi pada data suara sebelum data tersebut dikirimkan ke dalam jaringan. Jika kita mengetahui algoritma yang digunakan dan cara kerja dari algoritma tersebut, maka kita akan dapat melakukan dekompresi jika kita mendapatkan *payload* dari data tersebut. Kelemahan dari komunikasi menggunakan *VoIP* adalah data *payload* tidak diproteksi sehingga ketika dikirimkan dan ditangkap, maka data tersebut dapat dengan mudah dimanipulasi. Hal ini jelas sangat berbahaya apabila pembicaraan yang dilakukan merupakan sesuatu yang sifatnya pribadi atau rahasia.

4.4.2 Pengujian Keamanan *VoIP* pada Jaringan *MPLS-VPN*

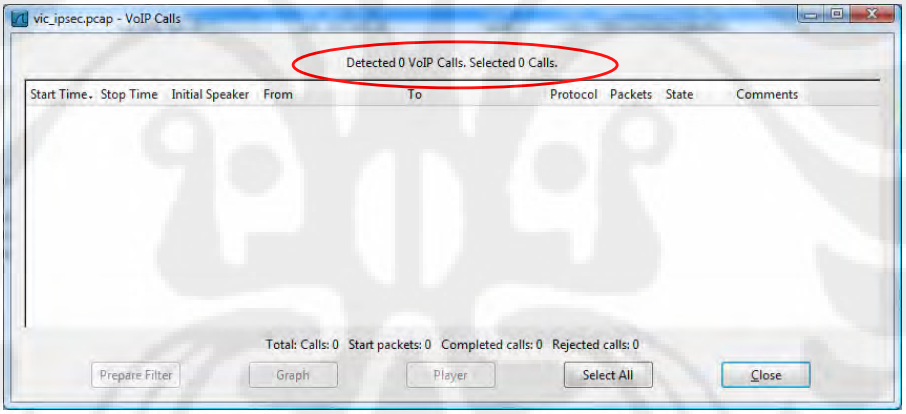
Sama dengan pengujian keamanan *VoIP* tanpa menggunakan *VPN*, pada pengujian ini juga dilakukan melalui 2 cara yaitu merekam pembicaraan *VoIP* dan menangkap paket *VoIP* untuk kemudian dianalisa. Sebelumnya pada jaringan *MPLS* yang sudah dibangun diimplementasikan *IPSec* dengan menggunakan algoritma enkripsi *3DES*. Dari hasil pengujian, data yang melewati router *PE* tidak dapat direkam menggunakan *software* Wireshark. Hal ini dikarenakan pada jaringan *VoIP over VPN* terdapat metode *tunneling*. Di mana data yang dikirim terenkripsi dan ditambahkan *header* baru sehingga baik data pengirim maupun penerima tidak dapat terlihat. Selain itu data yang dikirim dengan protokol *Encapsulation Service Payload (ESP)* dan data *payload* dari *VoIP* tidak terlihat di jaringan



Gambar 4.17 Isi *payload* dari paket *VoIP over VPN* yang di-*capture*

Payload yang di-*capture* hanya berisikan alamat IP dari router *Head Office* dan *Branch Office*. Sedangkan alamat IP dari *VoIP client* dan *VoIP server* tidak terdeteksi, sehingga kecil kemungkinan bagi *hacker* untuk melacak keberadaan *client* dan *server VoIP*.

Sedangkan ketika data *VoIP* coba direkam, ternyata *stream RTP* tidak dapat direkam. Hal ini karena paket telah berganti protokol dari *RTP* menjadi *ESP*. Sedangkan *software* ini tidak dirancang untuk merekam paket dengan protokol *ESP*. Selain itu data yang ditangkap telah terenkripsi menggunakan algoritma *3DES* 168 bit. Sampai saat ini belum ada *brute force attack* yang mampu untuk menerobos enkripsi dari *3DES*. Secara implementasi akan sulit untuk melakukan *brute force attack* pada algoritma ini. Kelemahan dari algoritma ini adalah proses pengenkripsian yang lambat di mana *3DES* menggunakan algoritma *DES* sebanyak 3 kali dengan pengulangan sebanyak 48 kali.



Gambar 4.18 Tampilan Wireshark ketika tidak ada data yang berhasil direkam

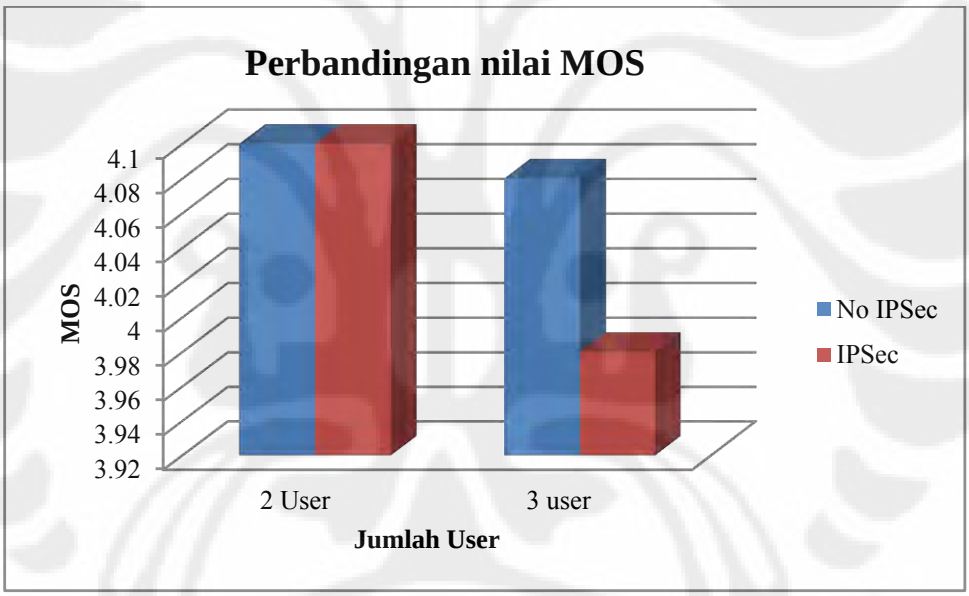
4.4.3 Analisa Pengaruh Penggunaan *VPN* Terhadap Unjuk Kerja *VoIP*

Setelah melakukan pengujian keamanan dari jaringan yang sudah dibangun, selanjutnya pada skenario terakhir ini, akan diuji seberapa besar pengaruh implementasi *IPSec* terhadap unjuk kerja *VoIP* secara keseluruhan, sebab sebagaimana sudah disebutkan sebelumnya, untuk mengamankan data *VoIP* digunakan algoritma enkripsi *3DES* 168 bit. Penambahan metode enkripsi ini jelas akan menambah *header* baru, selain itu ada waktu jeda sejenak pada proses enkripsi data sebelum data tersebut dikirimkan. Hal ini pasti akan sangat

berpengaruh terhadap *delay* dan *jitter* pada jaringan. *Codec* yang digunakan pada pengujian ini adalah G.729 karena diantara ketiga *codec* yang dianalisis pada skenario sebelumnya, *codec* G.729 memiliki unjuk kerja terbaik. Adapun hasil dari pengujian sistem seperti yang tampak pada Tabel 4.9 di bawah

Tabel 4.9 Perbandingan *VoIP* sebelum dan sesudah menggunakan *VPN*

Parameter	2 *User*		3 *User*	
	No IPSec	*IPSec*	*No IPSec*	*IPSec*
Delay (ms)	19.99	19.988	20.104	22.232
Jitter (ms)	17.498	17.496	17.626	19.464
Packet loss (%)	0	0	0	0.5
MOS	4.1	4.1	4.08	3.98



Gambar 4.17. Grafik perbandingan nilai *MOS*

Analisa *Delay, Jitter* dan *Packet Loss* pada Jaringan *VoIP over MPLS-VPN*

Pada *VoIP over MPLS-VPN* terdapat penambahan *header* untuk *Codec* G.729 dari 74 bytes setiap paket menjadi 134 bytes. Jumlah penambahan *header* ini cukup signifikan dan hal ini akan mempengaruhi *bitrate* sehingga akan menyebabkan terjadi peningkatan *bitrate* dan perubahan unjuk kerja pada jaringan

Dari Gambar 4.17 di atas dapat dilihat bahwa dengan *bandwidth* 128 Kbps dan percakapan hanya melibatkan 2 user saja ternyata implementasi *IPSec* yang

dilakukan pada jaringan tidak berpengaruh signifikan terhadap unjuk kerja *VoIP*. Nilai *delay, jitter*, dan *packet loss* bahkan cenderung sama baiknya ketika belum dilakukan implementasi *IPSec*. Nilai *delay, jitter* dan *packet loss* baru bertambah ketika dilakukan *conference call* (3 *user*). Akan tetapi penambahan tersebut tidak terlalu signifikan sehingga tidak mengganggu unjuk kerja *VoIP* secara keseluruhan (tidak merubah nilai *MOS*). Dapat dianalisa bahwa pertambahan *delay, jitter* dan *packet loss* terjadi karena adanya penambahan *header* pada paket *VoIP*. Penambahan *header* ini mengakibatkan penambahan *delay, jitter* dan *packet loss*. Sebelum menggunakan *VPN* besarnya satu paket untuk G.729 adalah 74 byte. Ketika menggunakan *VPN*, besarnya 1 paket menjadi 134 bytes, jumlah ini merupakan 2 kali lipat lebih banyak dari paket awal. Ketika kapasitas *bandwidth* masih lebih besar dari *bitrate* paket, maka *delay, jitter* dan *packet loss* tidak akan terlihat (terjadi pada saat percakapan hanya melibatkan 2 user). Tetapi ketika kapasitas *bandwidth* jaringan menjadi lebih kecil maka penambahan besar *delay, jitter*, dan *packet loss* akan terlihat. *VPN* merupakan sistem yang menjamin *confidentiality, integrity* dan *authentication*.

BAB 5

KESIMPULAN

Dari hasil analisa dan pengujian yang telah dilakukan, dapat disimpulkan sebagai berikut :

1. Dari ketiga jenis *codec* yang digunakan, G.711 merupakan *codec* dengan kualitas suara yang paling baik karena memiliki *MOS* sebesar 4.4 sedangkan G.729 nilai *MOS*-nya adalah 4.1 dan *GSM* hanya sebesar 3.4
2. Dengan *payload* sebesar 160 bytes, minimum *bandwidth* yang dibutuhkan oleh G.711 agar dapat bekerja secara optimum adalah 128 Kbps, sedangkan G.729 dan *GSM* yang memiliki *payload* sebesar 20 bytes dan 33 bytes hanya membutuhkan *bandwidth* sebesar 64 Kbps saja
3. Pada *bandwidth* 128 Kbps, ketika menggunakan G.711 untuk percakapan 2 *user*, *bandwidth* yang tersisa yang dapat digunakan untuk aplikasi satu arah seperti *ftp* dan *video streaming* hanya sebesar 25 % saja, selain itu hal ini mengakibatkan terjadinya penurunan unjuk kerja pada *VoIP* (*MOS*-nya menjadi 3.8). Sedangkan jika menggunakan G729 dan *GSM*, *bandwidth* yang tersisa mencapai 75%
4. Penambahan jumlah *user* dalam percakapan (*conference call*) menyebabkan terjadinya penurunan unjuk kerja dari *VoIP*, di mana penurunan nilai *MOS* terjadi sangat signifikan pada saat menggunakan G.711 yaitu sebesar 48%, sedangkan untuk G.729 dan *GSM* hanya mengalami penurunan sebesar 3%
5. Pada *bandwidth* yang terbatas (di bawah 128 Kbps), G.729 merupakan jenis *codec* yang paling bagus dan efisien daripada G.711 dan *GSM* untuk diimplementasikan pada jaringan *MPLS* dengan *tunneling IPSec*. Sebaliknya pada *bandwidth* yang besar (128 Kbps atau lebih) G.711 merupakan *codec* yang unjuk kerjanya paling bagus dibanding G.729 dan *GSM*
6. Implementasi *IPSec* pada jaringan menyebabkan terjadinya penurunan unjuk kerja dari *VoIP*. Namun penurunan yang terjadi tidak terlalu signifikan yaitu hanya sebesar 2% saja

DAFTAR REFERENSI

[1]. Cisco System "*Understanding Delay in Packet Voice Networks*". USA : Cisco Press. 2006

[2]. Davidson, J. Peters, J. (2007). *Voice Over IP Fundamentals (2nd Edition)*. Indianapolis : Cisco Press.

[3]. Tharom, Tabratas. W. Purbo, Onno. (2001). *Teknologi VoIP (Voice Over Internet Protocol)*. Jakarta : PT. Elex Media Komputindo

[4]. ITU-T Recommendation P.800, "*Methods for subjective determination of transmission quality*", 1996

[5]. ITU-T G.107, "*The E-Model, a computational model for use in transmission planning*", 2003

[6]. ITU-T G.114, "*One-way transmission time*", 2003

[7]. ITU-T Recommendation G.711, "*Pulse Code Modulation (PCM) of Voice Frequencies*", 1998

[8]. ITU-T Recommendation G.729, "*Coding of speech at 8 kbits/s using conjugate-structure algebraic-code excited linear-predition (CS-ACELP)*", 1996

[9]. Cisco System "*Voice Over IP – Per Call Bandwidth Consumption*". USA : Cisco Press. 2004

[10]. W.Purbo, Onno. Rahaja, Anton. (2010). *VoIP Cookbook : Building Your Own Telecommunication Infrastructure*. http://www.scribd.com/doc/27485126/VoIP-Cookbook (Diakses tanggal 24 April 2010)

[11]. Bates, Regis J. Gregory , Donald . (2007). *Voice and Data Communication Handbook* (5th Edition). New York : McGraw-Hill Professional.

[12]. Guillen, Edward Paul. Chacon, Diego Alejandro. *VoIP Networks Performance Analysis with Encryption Systems*. Journal of World Academy of Science, Engineering and Technology, 58. 2009

[13]. Besacier, Laurent. Bergamini. Vaufreydaz. Castelli. *The Effect of Speech and Audio Compression on Speech Recognition Performance*. IEEE Computer Society Journal, 301-306. 2001

[14]. Grah, Melanie. Radcliffe, Peter. *Dynamic QoS for Commercial VoIP in Heterogeneous Networks*. IEEE Computer Society Journal. 2009

[15]. Acharya, G.Prasad. Reddy, B.Veerender. Kumar, P.Naveen. *Analysis of the Encoding Scheme for CS-ACELP Codec for Secured VoIP Communication*. IEEE Computer Society Journal. 2009

[16]. Khuter, Ali Abd. (2008). *Performance Analysis of Voice Codec for VoIP*. Malaysia : Universiti Teknologi Malaysia eprints.utm.my/9553/1/AliAbdKhutherMFSKSM2008.pdf (Diakses tanggal 8 Mei 2010)

[17]. M.Iskandarsyah. (2003). *Dasar-dasar Jaringan VoIP*. http://ilmukomputer.org/2008/11/25/pengantar-jaringan-voip/ (Diakses tanggal 4 Mei 2010)

[18]. Raharja, Anton. (2006). *Materi VoIP Fundamental*. http://www.voiprakyat.or.id/?inc=files&file=materi-voip-fundamental.pdf (Diakses tanggal 12 mei 2010)

[19]. Deris Setiawan. *Membangun Interkoneksi VPN dengan solusi Hardware-Based dan Jaringan IIX*

[20]. Anonymous. Cisco IOS Enterprise VPN Configuration Guide. *Site-to-Site And Extranet VPN Business Scenarioss* (Chapter 3).

[21]. Kuncoro Wastuwibowo. (2003). *Jaringan MPLS Whitepaper*. Versi 1.2. Telkom.info

[22]. Rosen, Eric. et al. "Multiprotocol Label Switching Architecture". RFC-3031 IETF. January 2001 http://www.ietf.org/rfc/rfc3031.txt(09/18/2004) (Diakses tanggal 10 mei 2010)

[23]. Cisco Systems Learning. (2006). Implementing Secure Converged Wide Area Networks (Volume 1). San Jose : Cisco Systems Inc.

LAMPIRAN

1.1 Router *Head Office*

```
Head-Office#sh run
Building configuration...
Current configuration : 1427 bytes
!
version 12.4
no service pad
service timestamps debug datetime msec
service timestamps log datetime msec
no service password-encryption
!
hostname Head-Office
!
boot-start-marker
boot-end-marker
!
enable password password
!
no aaa new-model
!
resource policy
!
ip subnet-zero
ip cef
!
crypto isakmp policy 1
encr 3des
authentication pre-share
group 2
crypto isakmp key p4ssw0rd address 10.10.13.2
!
crypto ipsec transform-set finalproject esp-3des esp-sha-hmac
!
crypto map skripsi 1 ipsec-isakmp
set transform-set finalproject
match address JKT-to-SBY
!
interface Loopback0
ip address 10.10.12.1 255.255.255.255
!
interface FastEthernet0
switchport access vlan 2
!
interface FastEthernet1
switchport access vlan 2
!
interface FastEthernet2
!
interface FastEthernet3
!
interface FastEthernet4
description "Connection to PE-JKT"
ip address 10.10.11.2 255.255.255.252
duplex auto
speed auto
crypto map skripsi
!
interface Vlan1
no ip address
!
interface Vlan2
ip address 10.10.10.1 255.255.255.0
!
router rip
version 2
network 10.0.0.0
!
ip classless
!
no ip http server
no ip http secure-server
!
ip access-list extended JKT-to-SBY
permit ip 10.10.10.0 0.0.0.255 10.10.14.0 0.0.0.255
!
snmp-server community skripsi RW
!
control-plane
!
line con 0
no modem enable
line aux 0
line vty 0 4
password bismillah
login
!
scheduler max-task-time 5000
end
```

1.2 Router PE-JKT

```
PE-JKT#sh run
Building configuration...
Current configuration : 1709 bytes
!
version 12.3
service timestamps debug datetime msec
service timestamps log datetime msec
no service password-encryption
!
hostname PE-JKT
!
```

```
boot-start-marker
boot-end-marker
!
enable password password
!
no aaa new-model
ip subnet-zero
!
ip vrf company-a
rd 65000:1
route-target export 65000:1
route-target import 65000:1
!
ip cef
!
interface Loopback0
ip address 200.20.20.100 255.255.255.255
!
interface FastEthernet0/0
description "Connection to CORE"
ip address 200.20.20.1 255.255.255.248
duplex auto
speed auto
tag-switching ip
!
interface FastEthernet0/1
description "Connection to Head-Office"
ip vrf forwarding company-a
ip address 10.10.11.1 255.255.255.252
rate-limit input 256000 256000 256000 conform-action continue exceed-action drop
rate-limit output 256000 256000 256000 conform-action continue exceed-action drop
duplex auto
speed auto
!
router ospf 100
log-adjacency-changes
network 200.20.20.0 0.0.0.7 area 0
network 200.20.20.100 0.0.0.0 area 0
!
router ospf 101 vrf company-a
log-adjacency-changes
redistribute bgp 65000 subnets
network 10.10.11.0 0.0.0.3 area 0
!
router bgp 65000
no synchronization
bgp log-neighbor-changes
neighbor 200.20.20.102 remote-as 65000
neighbor 200.20.20.102 update-source Loopback0
neighbor 200.20.20.102 next-hop-self
no auto-summary
!
address-family vpnv4
neighbor 200.20.20.102 activate
neighbor 200.20.20.102 send-community both
exit-address-family
!
address-family ipv4 vrf company-a
redistribute ospf 101
no auto-summary
no synchronization
exit-address-family
!
ip http server
ip classless
!
line con 0
line aux 0
line vty 0 4
password bismillah
login
!
end
```

1.3 Router *Core*

```
CORE#sh run
Building configuration...
Current configuration : 904 bytes
!
version 12.3
service timestamps debug datetime msec
service timestamps log datetime msec
no service password-encryption
!
hostname CORE
!
boot-start-marker
boot-end-marker
!
enable password password
!
no aaa new-model
ip subnet-zero
!
ip cef
!
interface Loopback0
ip address 200.20.20.101 255.255.255.255
!
interface FastEthernet0/0
description "Connection to PE-JKT"
ip address 200.20.20.2 255.255.255.248
duplex auto
speed auto
tag-switching ip
!
```

```
interface FastEthernet0/1
description "Connection to PE-SBY"
ip address 200.20.20.9 255.255.255.252
duplex auto
speed auto
tag-switching ip
!
log-adjacency-changes
router ospf 100
network 200.20.20.0 0.0.0.7 area 0
network 200.20.20.8 0.0.0.3 area 0
network 200.20.20.101 0.0.0.0 area 0
!
ip http server
ip classless
!
line con 0
line aux 0
line vty 0 4
 password bismillah
 login
!
end
```

1.4 Router PE-SBY

```
PE-JKT#sh run
Building configuration...
Current configuration : 1495 bytes
!
version 12.3
service timestamps debug datetime msec
service timestamps log datetime msec
no service password-encryption
!
hostname PE-JKT
!
boot-start-marker
boot-end-marker
!
enable password password
!
no aaa new-model
ip subnet-zero
    !
ip vrf company-a
rd 65000:1
route-target export 65000:1
route-target import 65000:1
!
ip cef
!
interface Loopback0
ip address 200.20.20.100 255.255.255.255
!
interface FastEthernet0/0
description "Connection to CORE"
ip address 200.20.20.1 255.255.255.252
duplex auto
speed auto
tag-switching ip
!
interface FastEthernet0/1
description "Connection to Head-Office"
ip vrf forwarding company-a
ip address 10.10.11.1 255.255.255.252
duplex auto
speed auto
!
router rip
version 2
network 200.20.20.0
!
address-family ipv4 vrf company-a
redistribute bgp 65000 metric transparent
network 10.0.0.0
no auto-summary
version 2
exit-address-family
!
router bgp 65000
no synchronization
bgp log-neighbor-changes
neighbor 200.20.20.102 remote-as 65000
neighbor 200.20.20.102 update-source
Loopback0
neighbor 200.20.20.102 next-hop-self
no auto-summary
!
address-family vpnv4
neighbor 200.20.20.102 activate
neighbor 200.20.20.102 send-community
both
exit-address-family
!
address-family ipv4 vrf company-a
redistribute rip
no auto-summary
no synchronization
exit-address-family
!
ip http server
ip classless
!
line con 0
line aux 0
line vty 0 4
password bismillah
login
!
end
```

1.5 Router *Branch Office*

```
Branch-Office#sh run
Building configuration...
Current configuration : 1454 bytes
!
version 12.4
no service pad
service timestamps debug datetime msec
service timestamps log datetime msec
no service password-encryption
!
hostname Branch-Office
!
boot-start-marker
boot-end-marker
!
enable password password
!
no aaa new-model
!
resource policy
!
ip subnet-zero
ip cef
!
crypto isakmp policy 1
encr 3des
authentication pre-share
group 2
crypto isakmp key p4ssw0rd address 10.10.11.2
!
crypto ipsec transform-set finalproject esp-3des esp-sha-hmac
!
crypto map skripsi 1 ipsec-isakmp
set peer 10.10.11.2
set transform-set finalproject
match address SBY-to-JKT
!
interface Loopback0
ip address 10.10.15.1 255.255.255.255
!
interface FastEthernet0
switchport access vlan 2
!
interface FastEthernet1
switchport access vlan 2
!
interface FastEthernet2
switchport access vlan 2
!
interface FastEthernet3
!
interface FastEthernet4
description "Connetion to PE-SBY"
ip address 10.10.13.2 255.255.255.252
duplex auto
speed auto
crypto map skripsi
!
interface Vlan1
no ip address
!
interface Vlan2
ip address 10.10.14.1 255.255.255.0
!
router rip
version 2
network 10.0.0.0
!
ip classless
!
no ip http server
no ip http secure-server
!
ip access-list extended SBY-to-JKT
permit ip 10.10.14.0 0.0.0.255 10.10.10.0 0.0.0.255
!
snmp-server community skripsi RO
!
control-plane
!
line con 0
no modem enable
line aux 0
line vty 0 4
password bismillah
login
!
scheduler max-task-time 5000
end
```