BAB 3
ANALISIS DAN PERANCANGAN SISTEM

3.1 Analisis Sistem

Analisis Sistem adalah penguraian dari suatu masalah yang utuh ke dalam bagian-bagian komponennya dengan maksud untuk mengidentifikasikan dan mengevaluasi permasalahan-permasalahan, kesempatan-kesempatan, hambatan hambatan yang terjadi dan kebutuhan-kebutuhan yang diharapkan sehingga dapat diusulkan perbaikan-perbaikannya.

Layanan *Internet Protocol Television* (IPTV) adalah suatu layanan multimedia dalam bentuk *video, audio, text, graphic* dan data yang disalurkan ke pelanggan melalui jaringan *Internet* Protokol (IP), dimana dari sisi penyedia layanan jaringan menjamin kualitasnya (*Quality of Service*), keamanannya (*Security*), keandalannya (*realibility*). Hal ini juga memungkinkan komunikasi dengan pelanggan secara dua arah atau interaktif (*interactivity*) secara *real time*.

Untuk itu dibutuhkan jaminan kuailtas yang harus dipenuhi provider penyedia infrastruktur layanan IPTV yang mengacu pada standar layanan yang berlaku, berupa kebutuhan *bandwidth, delay, jitter, packet loss* dan *throughput* yang sesuai dengan kebutuhan minimal dari sebuah layanan multimedia berbasis *audio-video*, agar layanan IPTV dapat berjalan dengan baik.

3.1.1 Analisis Masalah

Saat ini, internet tak lagi hanya dapat melayani komunikasi data teks atau gambar saja, melainkan dapat melayani komunikasi multimedia seperti *video streaming*. Banyak aplikasi di internet yang menawarkan layanan multimedia, seperti TV internet, *Teleconference, Telepresence, Video On Demand* dan layanan *video streaming* lain, baik itu yang besifat *real-time* maupun *non real-time*. Nyatanya, komunikasi ini sangat rakus akan sumber daya pada internet, terutama *bandwidth*, dan membutuhkan *Quality of Service* yang baik dan konsisten dibandingkan dengan komunikasi data teks dan gambar. Oleh karena itu, dibutuhkan teknologi jaringan yang dapat mengatur ketersediaan sumber daya

internet agar pertukaran informasi multimedia dapat berlangsung secara lancar dan berkelanjutan.

Secara umum, komunikasi data yang terjadi di internet saat ini masih menggunakan konsep TCP/IP. Namun, teknologi TCP/IP saat ini masih memiliki kelemahan dalam pelayanan untuk komunikasi data multimedia, seperti : belum mendukung layanan QoS, keputusaan perutean masih hanya berbasis alamat IP (belum ada system klasifikasi data), belum dapat melakukan *traffic engineering*, *delay* yang dihasilkan masih cukup besar akibat analisa *header* yang terjadi di setiap node jaringan dan kemungkinan terjadinya *packet loss* dan *congestion* pada lalu lintas data masih cukup besar. Kondisi internet seperti ini, tentunya sangat menggaggu komunikasi data multimedia, terutama yang bersifat real-time.

Konsep teknologi TCP/IP memiliki beberapa karakteristik seperti *unreliable, connectionless,* dan *datagram delivery service.*

1. *Unreliable*

   Unreliable berarti IP tidak menjamin datagram yang dikirim pasti sampai ke tempat tujuan. Protokol IP hanya akan melakukan usaha sebaik-baiknya (*best effort delivery service*), agar paket yang dikirim sampai ke tujuan. Jika di perjalanan paket tersebut terjadi hal-hal yang tidak diinginkan (contoh salah satu jalur putus, router mengalami *congestion*, atau *host* tujuan sedang *down*). maka IP hanya akan memberitahukan ke protokol *Internet Control Message Protokol* (ICMP) bahwa terjadi masalah dalam pengiriman paket IP ke tujuan.

2. *Connectionless*

   Connectionless berarti dalam mengirim paket dari tempat asal ke tujuan, pihak pengirim dan penerima paket IP sama sekali tidak mengadakan perjanjian (*handshake*) terlebih dahulu. Pengirim dapat mengirimkan paket kapan pun tanpa mengecek *host* tujuan dalam keadaan hidup atau tidak. Oleh karena itu, bisa saja paket yang terkirim tidak akan diterima.

3. *Datagram delivery service*

Datagram delivery service berarti setiap paket data yang dikirim adalah *independent* terhadap paket data yang lain. Akibatnya, jalur yang ditempuh oleh masing-masing paket data IP ke tujuannya bisa jadi berbeda satu sama lain, sehingga kedatangan paket pun bisa jadi tidak berurutan. Trafik-trafik seperti *voice* dan *video* tidak dapat berkompromi dengan masalah-masalah ini.

Melihat permasalahan tersebut, jika menyediakan sebuah layanan IPTV melalui infrastruktur berbasis TCP/IP pada umumnya, akan menghadapi banyak hambatan, mengingat kompleksnya persyaratan teknis yang sangat jauh berbeda dengan layanan *voice* dan layanan internet berupa kebutuhan *bandwidth* yang besar dan perlu dijamin kehandalan dalam proses pengiriman datanya agar tidak terjadi *packet loss* yang terlalu besar.

3.1.2 Analisis IPTV

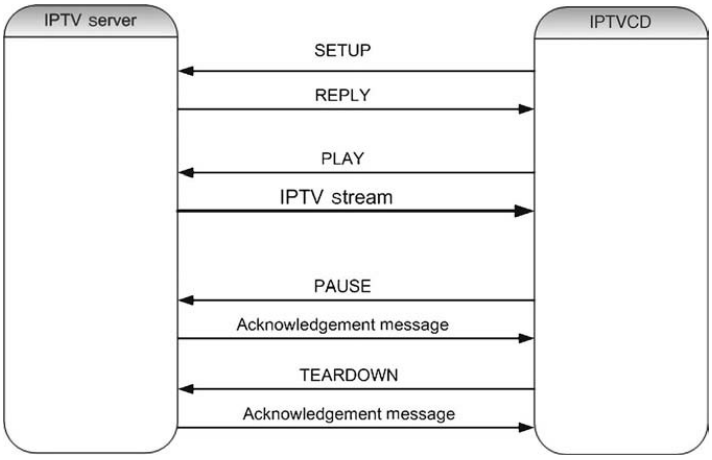
Internet Protocol Television (IPTV) adalah layanan multimedia (*video, audio, text,* grafik dan data) yang di kirim di atas jaringan *Internet Protocol* (IP) dan dijamin kualitas layanannya. IPTV tidak seperti program televisi *broadcast* biasa yang menggunakan internet, tetapi lebih dari itu dimana IPTV merupakan sistem yang tertutup serta siaran atau tayangannya berhak paten yang mirip dengan layanan TV kabel. Namun perbedaannya pengiriman IPTV dibuat lewat kanal-kanal berbasis IP yang cukup aman. Pada layanan IPTV terdapat Minimal 4 tipe layanan yang harus didukung, yaitu: *live TV*, *Video on Demand* (VoD), *time-shifted TV* (TSTV) dan *Personal Video Recording* (PVR) [5]. Layanan IPTV yang bersifat *real time* sangat sensitif terhadap *delay* dan bergantung sepenuhnya pada *bandwidth* yang konstan

3.1.2.1 Protokol Standar IPTV

Protokol standar yang digunakan dalam sistem berbasis IPTV adalah *Real Time Streaming Protokol* (RTSP) Sebuah protokol level aplikasi untuk kontrol atas pengiriman data dengan sifat real-time. Pada jaringan IPTV protkol RTSP digunakan untuk layanan *Video on Demand* (VoD). Pada sebagian server RTSP

menggunakan protokol *Real-Time Transport Protocol* (RTP) yang saling melengkapi dengan protokol *Real-Time Control Protocol* (RTCP) dan juga *User Datagram Protcol* (UDP) adapun fungsi dari protokol tersebut sebagai berikut :

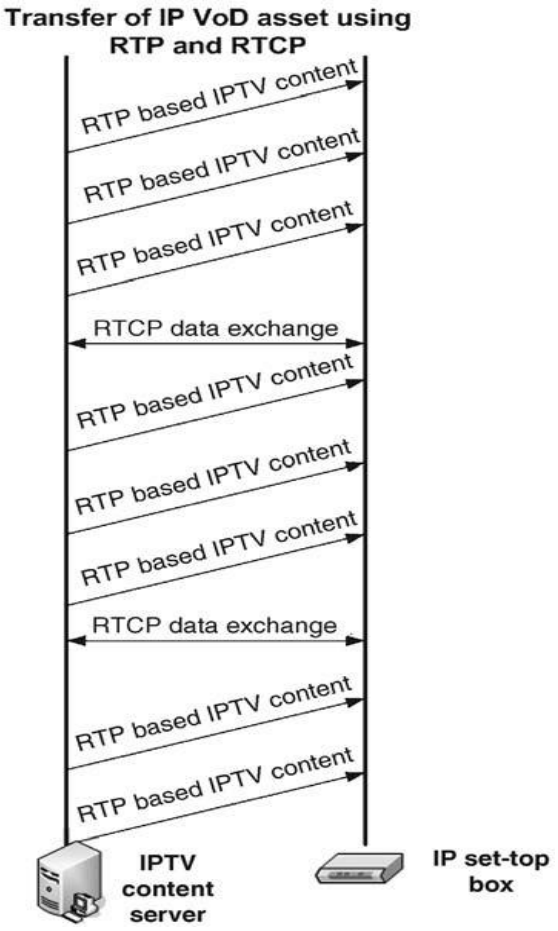
1. *Real-Time Transport Protocol* (RTP) Berfungsi sebagai transport protocol yang mengirimkan data-data video dan audio secara *real time*. Dalam melakukan pengiriman video. System IPTV menggunakan protokol RTP sebagai pembawanya. Informasi RTP dienkapsulasi dalam paket UDP. Jika packet RTP hilang (*lost*) atau di *drop* pada jaringan, maka RTP tidak akan melakukan *retransmission* (sesuai standard protocol UDP). Hal ini agar user tidak terlalu lama menunggu (*long pause*) atau *delay*, dikarenakan permintaan *retransmission*. Jaringan harus didesain sebaik mungkin agar *loss packet* tidak terjadi.



Gambar 3.1 Perintah Pada RTSP *Session*

2. *Real-Time Control Protocol* (RTCP) memberikan informasi kontrol *out-of-band* atas aliran RTP. RTCP memberikan informasi tentang kualitas penerimaan yang digunakan oleh aplikasi untuk melakukan penyesuaian secara lokal. Misalnya, apabila terjadi kongesti, maka aplikasi dapat memutuskan untuk menurunkan kecepatan data (*data rate*). RTCP bekerja sama dengan RTP dalam pengiriman dan pembungkusan (*packaging*) data multimedia, tetapi tidak mentransportasikan data. RTCP digunakan secara periodik untuk mentransmisikan paket kontrol dalam sesi

streaming multimedia. Sehingga fungsi utama RTCP adalah memberikan umpan balik tentang QoS yang diberikan oleh RTP.

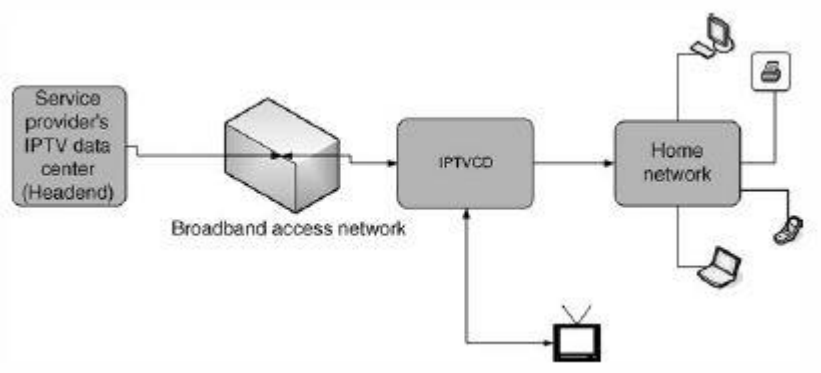


Gambar 3.2 Contoh RTP dan RTCP session

3. *User Datagram Protcol* (UDP) digunakan untuk situasi yang tidak mementingkan mekanisme reliabilitas. UDP digunakan pada IPTV pada pengiriman audio/video *streaming* yang berlangsung terus menerus dan lebih mementingkan kecepatan pengiriman data agar tiba di tujuan. Karena UDP mampu mengirimkan data *streaming* dengan cepat. Untuk mengurangi jumlah paket yang hilang saat pengiriman data (karena tidak terdapat mekanisme pengiriman ulang) maka pada teknologi IPTV pengiriman data banyak dilakukan pada *private network* atau menggunakan jaringan *broadband*.

3.1.2.2 Arsitektur IPTV

Arsitektur standar IPTV terdiri dari *headend*, *network* dan *home network*. Pada penlitian ini akan dibangun system iptv yang sesuai dengan standar yang ada.



Gambar 3.3. arsitektur IPTV

Teknologi yang telibat pada layanan IPTV diklasifikasikan menjadi beberapa bagian utama. Dalam peneltian ini akan dibangun layan arsitektur yang sesuai dengan arsitektur IPTV pada umumnya. Beberpa bagian utama tersebut sebagai berikut :

1. *Head-end*

   Merupakan komponen utama IPTV dimana server penyedia layanan IPTV terletak pada bagian ini. Pada penelitian ini akan dibangun sebuah server sebagai penyedia dari layanan IPTV.

2. *Network*

   Merupakan pengubung dari head-end dan home network. Didalam jaringan IPTV terjadi proses perutean yang biasa disebut routing. Pada penelitian ini akan dibangu dua model jaringan yaitu jaringan TCP/IP tradisonal dan jaringan MPLS.

3. *Home Network*

   Merupakan perangakat antarmuka jaringan yang ditempatkan pada sisi pelanggan. Pada penelitian sisi pelanggan yang disebut client akan mengakses layanan IPTV yang telah dibuat.

3.1.3 Analisis *Code-Decoder* (Codec)

Codec adalah singkatan dari *Code-Decoder* (sebagian menyebutnya *Compressor-Decompressor*) dan digunakan untuk menjelaskan segala sesuatu yang mengubah data kedalam bentuk lain untuk disimpan atau ditransimisi, dan mengubahnya kembali agar dapat digunakan. Dapat juga berupa sebuah *device* atau program yang mampu mengubah atau mentransform sinyal dan aliran data. *codec* dapat merubah *stream* atau sinyal kedalam bentuk yang ter-*encode* (sering dipakai pada transmisi, *storage*, *enkripsi*) kemudian diterima, atau dapat men-*decode* bentuk tersebut agar dapat dilihat atau dimanipulasi ke bentuk yang lain. *codec* sering digunakan dalam *videoconference* dan teknologi *streaming*.

Dalam komputer, *codec* adalah cara mengkompres *video*, gambar dan *audio* kedalam ukuran yang mudah diatur. Kebanyakan *codec* menggunakan metode kompresi *lossy* (hilang), namun ada beberapa yang *lossless* (tidak hilang). *Codec* yang *lossless* seperti MSU atau *Huffyuv*, menghasilkan kembali *video* yang asli tanpa ada data yang hilang ketika dikodifikasi kembali. Kebanyakan *codec lossy* menghilangkan beberapa informasi yang tidak perlu tapi dapat menghemat *space*.

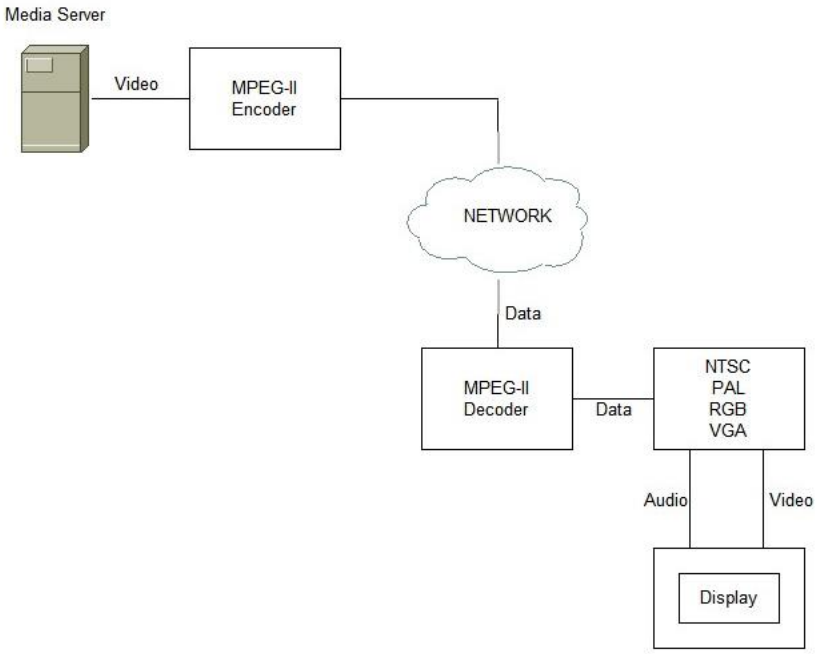
Terdapat banyak jenis *codec* yang tersedia tapi pada penelitian ini menggunakan jenis codec *Moving Picture Experts Group* –I/II (MPEG-I/II). *Codec* MPEG-1 digunakan untuk VCD dan berisi juga standar untuk MP3, *codec audio* yang paling banyak dipakai. Support untuk *codec*-1 sangat besar baik dari komputer maupun peralatan film. Codec MPEG-2 adalah standar kualitas yang tinggi yang digunakan untuk DVD. Sementara *codec* MPEG-1 hanya memungkinkan *scanning progressive*. MPEG-2 juga mendukung *interlacing* yang memungkinkan menampilkan gambar yang lebih bagus. Msekipun *codec* ini bukanlah *codec* yang paling advance diantara *codec* - *codec* lainnya, MPEG-2 banyak digunakan karena ini adalah standar untuk DVD komerisial.

3.1.3.1 Proses *Encoding* dan *decoding*

Proses *encoding* terjadi pada pengirim, *encoding* merupakan suatu program yang digunakan untuk mengubah *source* video ke dalam format yang

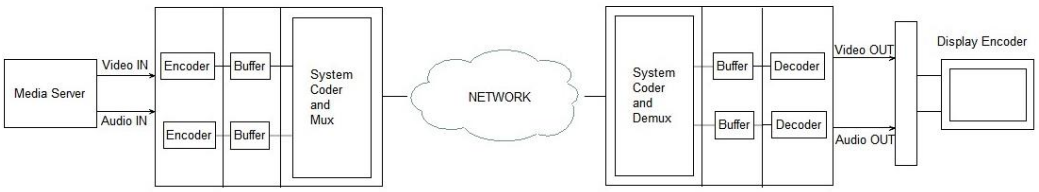
sesuai untuk video *streaming.* Cara kerjanya *file* video yang akan di*encoding* resolusinya diturunkan ke dalam ukuran 320 x 240, berikutnya sebuah fungsi lapisan kompresi MPEG akan menyandikan sikuen *audio* ataupun *video* terdigitalisasi yang masuk tersebut, yang terbentuk dari kumpulan *frame – frame* yang dikodekan ke dalam *bit-bit stream* untuk dikirimkan melalui jaringan, setelah sampai di penerima *bitstream* tersebut akan dikumpulkan lagi pada *buffer* dan *decoder* akan menjadikan *bit-bit stream* tersebut menjadi *frame-frame*. Setelah *frame-frame* video telah penuh di dalam kapasitas *buffer*, maka *frame-frame* tersebut akan ditampilkan dengan kecepatan *frame* per detik Untuk NTSC 30 fps dan 25 fps untuk PAL, jadi selama proses berlangsung, prinsip kerja dari video *streaming* adalah adalah ambil data, simpan,dan tampilkan. Sehingga pada waktu ditampilkan terdapat waktu *delay.* Tetapi ini juga tergantung dari kondisi jaringan, karena tiap jaringan tentu ada kesibukan proses ambil dan menerima data.

Proses *Decoding* adalah proses yang dilakukan setelah *encoding* selesai maka resolusi yang 320 x 240 , akan dikembalikan lagi ke dalam resolusi yang aslinya, yaitu 720x480 (NTSC) dan 720 x 576 (PAL) untuk ditampilkan di layar peraga. Aliran digital yang sudah tidak dikompresi ini akhirnya ditransfer ke sistem peraga (*display*) yang memadai seperti misalnya dengan standar televisi NTSC, PAL, RGB, VGA dan format audio. Proses ini ditunjukkan pada gambar 3.4.



Gambar 3.4 Video *Streaming*

Dalam proses *decoding* ini data yang dikirimkan disimpan di dalam *buffer* sebagai tempat penyimpanan sementara. Karena *bit stream* audio dan video yang dikirimkan dari pengirim akan dibentuk lagi menjadi *frame-frame*, sehingga ketika *frame* telah berada di dalam *buffer* maka proses penampilan di layar siap proses, seperti yang ditunjukkan pada gambar 3.5.



Gambar 3.5 *Encoder* dan *Decoder*

3.1.4 Analisis Jaringan TCP/IP

TCP/IP merupakan sekumpulan protokol yang dikembangkan untuk mengijinkan komputer-komputer agar dapat saling membagi sumber daya yang dimiliki masing-masing melalui media jaringan. Pada penelitian ini akan dibangun sebuah jaringan TCP/IP menggunakan beberapa router. Sebagai pengubung agar

antar router dapat berkomunikasi maka akan dikonfigurasi menggunakan routing dinamis *Open Shortest Path First* (OSPF). OSPF adalah sebuah *routing* protokol standar terbuka yang telah di implementasikan oleh sejumlah besar vendor jaringan. Dalam konfigurasi OSPF ini *network* yang *diadvertise* adalah *interface loopback* 0 dan *network* dari *fast ethernet* yang *directly connected*. Dalam konteks router, *interface* yang *diadvertise* hanya *network* dari *fast ethernet* yang mengarah ke router tetangganya.

Routing Protocol OSPF dipilih karena berjenis *link-state routing protocol*. OSPF berfungsi untuk mengirimkan keseluruhan informasi mengenai topologi ke semua *router* pada area yang sama. Tahap ini dikatakan berhasil jika *routing table router-router* yang berada di dalam jaringan sudah saling bertukar informasi dan dapat melakukan *ping* antara *router* yang jaringannya bukan *directly connected.* Routing protokol ini memiliki keuntungan dan kerugian dalam mengimplementasikannya. Keuntungan utama dari *link state routing protocol* seperti OSPF adalah bahwa pengetahuan lengkap tentang topologi memungkinkan router untuk menghitung rute yang memenuhi kriteria tertentu. Hal ini dapat berguna untuk tujuan rekaya lalu lintas, dimana rute dapat dibatasi untuk memenuhi kualitas tertentu lari sebuah layanan. Adapun kerugian dari routing protokol ini adalah routing ini tidak cocok untuk pemakain dalam skala besar dengan jumlah router yang banyak. Peningkatan jumlah router meningkatkan ukuran dan frekuensi update topologi, seta lamanya waktu yang dibutuhkan untuk menghitung rute *end-to-end*. Kurangnya skalabilitas berarti *link state routing procol* tidak cocok untuk routing internet pada umumnya.

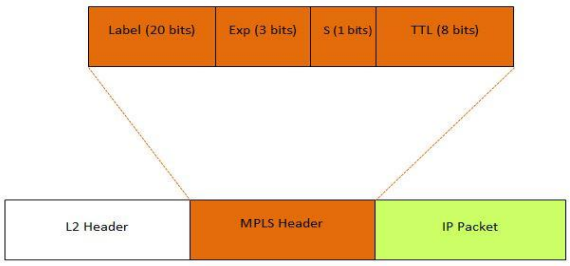
3.1.5 Analisis Jaringan MPLS

Jaringan backbone pada infrastruktur jaringan IPTV diperlukan untuk membawa paket yang berisi konten *video* pada kecepatan tinggi antara IPTV data center dan jaringan distribusi *broadband*. Ada beberapa jenis jaringan *Backbone* standar yang memberikan perlindungan *multipath* dan link. Setiap standar memiliki sejumlah fitur tertentu termasuk kecepatan transfer data dan skalabilitas.

Tiga jenis teknologi jaringan backbone yang digunakan dalam infrastruktur jaringan IPTV yaitu ATM SONET SDH, IP MPLS dan *Metro Ethernet*[9].

Pada penelitian ini digunakan salah satu jenis jaringan *backbone* yang digunakan sebagai infrastruktur jaringan IPTV yaitu jaringan *Multi Protocol Label Switching* (MPLS). *Multi Protocol Label Switching* (MPLS) adalah suatu metode *forwarding* (meneruskan data melalui suatu jaringan dengan menggunakan informasi dalam label yang dilekatkan pada IP), sehingga memungkinkan router untuk meneruskan paket dengan hanya melihat label dari paket itu, tidak perlu melihat IP alamat tujuannya.

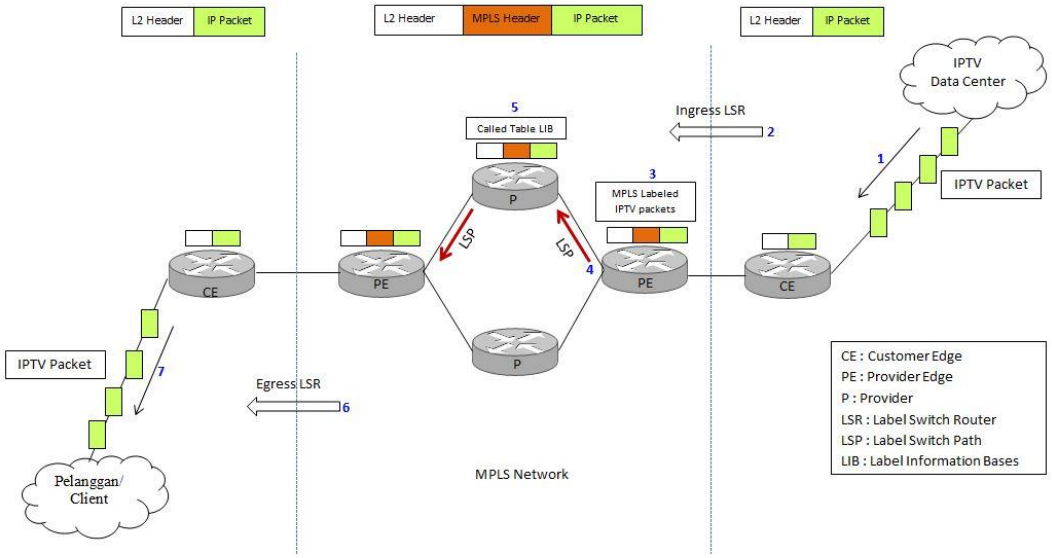
MPLS melakukan enkapsulasi paket IP dengan memasang *header* MPLS pada paket yang dilewati pada jaringan tersebut. *Header* MPLS terdiri atas 32 bit data, termasuk 20 bit *label*, 3 bit *eksperimen*tal, dan 1 bit identifikasi *stack*, serta 8 bit *Time To Live* (TTL). *Label* memiliki panjang yang bersifat tetap dan berisi spesifikasi *hop routing* selanjutnya, khusus pada paket yang dikirim. *Eksperimen*tal Disediakan untuk penggunaan lain, misalnya mekanisme MPLS *Diffserv*. *Stack Header* dapat berisi satu atau lebih label. Setelah disusun dan diatur kesalah satu *header*, LSR akan mengidentifikasi label terakhir di paket. *Time To Live* (TTL) Nilai ini disalin dari *field* TTL di IP header. Detail enkapsulasi bisa dilhat pada gambar 3.6.



Gambar 3.6 *Enkapsulasi* MPLS

Cara kerja MPLS, pertama *Label Distribution Protocol* (LDP) memberikan label pada rute-rute di *routing table*, nilai label tersebut ditentukan dari *Forwarding Equivalence Class* (FEC) yang merupakan informasi mengenai *grouping* dari paket-paket data. Setiap perangkat yang terhubung dengan jaringan MPLS harus membentuk LDP *neighbour* untuk membangun koneksi, kemudian

siap untuk pertukaran label. Jadi bedanya dengan jaringan IP, yang dipertukarkan dengan perangkat lain adalah label, bukan rutenya. MPLS yang dirancang dan dibangun dengan menggunakan *Label Switch router* (LSRs). LSRs ini bertanggung jawab untuk membangun sambungan berorientasi *route* untuk tujuan spesifik di jaringan. Untuk lebih detail bisa dilihat ilustrasi pengiriman paket IPTV pada gambar 3.7 berikut.



Gambar 3.7 Trafik IPTV Jaringan MPLS

Penjelasan gambar 3.7 diatas sebagai berikut :

1. IPTV data center mengirim paket kepada pelanggan/client berupa IPTV packet kedalam jaringan.
2. *Ingress* LSR mengidentifikasi jenis lalu lintas jaringan dengan menambahkan sebuah header MPLS ke awal setiap packet IPTV.
3. Label MPLS ditambahkan kedalam paket IPTV.
4. *Label Switched path* (LSP) Jalur virtual ini dikonfigurasi dengan cukup sumber daya untuk memastikan kelancaran transisi IPTV lalu lintas melalui jaringan MPLS. Menyederhanakan dan mempercepat routing paket melalui jaringan karena paket mendalam inspeksi hanya terjadi pada jalan masuk ke jaringan dan tidak diperlukan di setiap router hop.
5. Sementara IPTV lalu lintas melintasi seluruh MPLS *router* sejumlah tabel lokal yang disebut *Label informasi basis* (LIBs) berkonsultasi untuk menentukan rincian tentang hop berikutnya sepanjang rute.

6. Pada *Egress* LSR proses penghapusan MPLS header.
7. Tahap akhir ini paket IPTV sampai pada pelanggan/client.

Jaringan yang berbasiskan MPLS menggunakan label-label yang berisi informasi dalam mengirimkan paket tersebut, dimana label – label tersebut diletakkan di dalam paket oleh router yang berada paling ujung dari suatu jaringan.

3.1.6 Solusi Penyelesain Masalah

Dari analisa permasalahan dan sistem di atas, terlihat bahwa sebuah layanan IPTV memerlukan sebuah solusi yang tepat, yang dapat digunakan sebagai infrastruktur layananannya. Sehingga layanan IPTV dapat berjalan dengan baik pada media berbasiskan *Internet Protokol* (IP), oleh karena itu dibutuhkanlah suatu teknologi IP yang dapat menjamin kualitas layanannya.

Teknologi MPLS salah satu usulan perbaikan untuk jaringan TCP/IP biasa (tradisional) karena meningkatkan flesibilitas, efisiensi dan skalabilitas sehingga cocok untuk diterapkan pada layanan multimedia[8]. Pada penelitian ini akan dibangun jaringan TCP/IP dan jaringan MPLS sebagai infrastruktur layanan IPTV juga yang didukung dengan jaminan *Quality of Service* (QoS) menggunkan metode *Differentiated Services* (*Diffserv*). Dengan adanya jaminan QoS, maka akan diberikan *higher priority* pada protokol IP yang mengalirkan trafik untuk layanan IPTV pada jaringan untuk sampai pada pelanggan.

3.1.7 Analisis Pendukung Infrastruktur

Kebutuhan akan infrastruktur untuk membangun infrstruktur jaringan IPTV terbagi menjadi dua macam, yaitu perangkat keras dan perangkat lunak. Kebutuhan ini dibutuhkan untuk pengujian jaringan TCP/IP biasa maupun Jaringan MPLS.

3.1.7.1 Kebutuhan Perangkat Keras

Kebutuhan perangkat keras (*hardware*) untuk dapat membangun sebuah infrastruktur jaringan layanan IPTV. Pada penelitian ini beberapa perangkat keras yang disediakan adalah sebagai berikut:

Tabel 3.1. Spesifikasi Kebutuhan Perangkat Keras

No.	Perangkat Keras	Spesifikasi
1	Kompter Utama Sebagai PC Router	- AMD XII 3.0 Ghz - Hardisk 500 GB - VGA Gforce GT9500 - RAM 8 GB - Sistem operasi Windows 8 64 bit - 2x FastEthernet
2	Komputer Sever	- Intel Core Duo 2.0 Ghz - Hardisk 250 GB - RAM 2 GB - Sistem Operasi Linux Ubuntu 10.04 - FastEthernet
3	Komputer Client	- Intel Core 2 Duo 2.93 Ghz - Hardisk 360 GB - RAM 2 GB - Sistem Operasi Linux Ubuntu 10.10 - FastEthernet

3.1.7.2 Kebutuhan Perangkat lunak

Beberapa perangkat lunak yang digunakan untuk membangun sebuah infrastruktur jaringan layanan IPTV tentu memiliki beberapa perangkat lunak yang menjadi keharusan. Pada penelitian ini beberapa perangkat lunak yang disediakan adalah sebagai berikut :

Tabel 3.2. Spesifikasi Perangkat Lunak

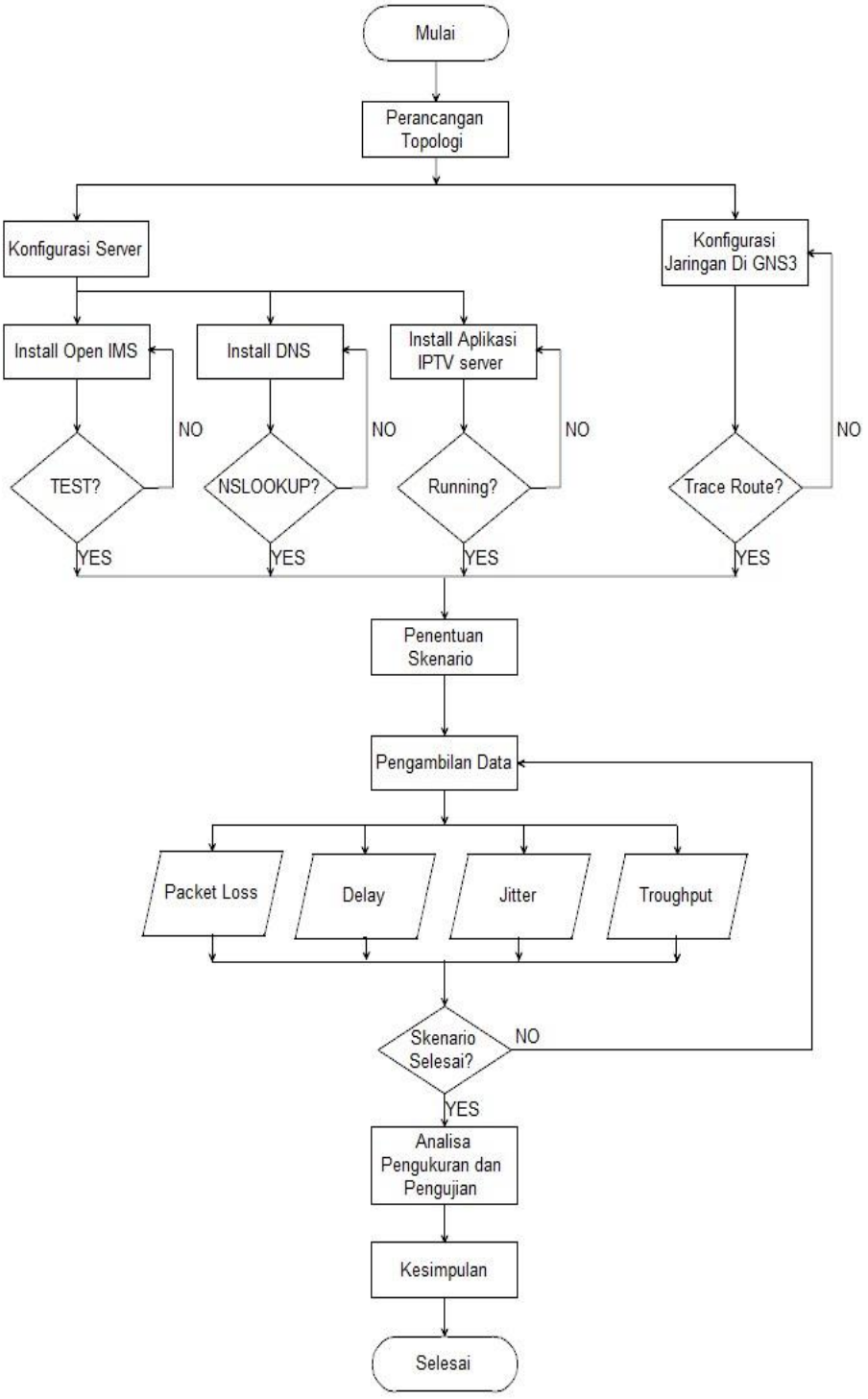
No.	Perangkat Lunak	Keterangan
1	GNS3	program *graphical network simulator* yang berguna mensimulasikan topologi jaringan, dari yang sederhana hingga yang sangat kompleks.
2	*Linux Ubuntu 10.04*	Sistem Operasi dari OpenIMSCore dan sistem operasi client
3	UCT IPTV Advanced	Sebuah aplikasi yang digunkan untuk membuat server IPTV.
4	UCT IMSClient	Ssebuah aplikasi yang digunkan untuk mengakses layanan IPTV
5	Putty/SeperPutty	Perangkat lunak untuk melakukan konfigurasi *router* dan *switch*; baik

		melalui *telnet*, SSH, maupun *line console*.
6	Wireshark	Perangkat lunak untuk melakukan *traffic capture* paketpaket yang melewati jaringan.
7	VLC	Aplikasi *open source* yang digunakkan untuk *media server* VoD
8	IOS c3640-jk9o3s-mz.124-16.bin	IOS CISCO router yang dipakai pada GNS3.

3.2 Perancangan Sistem

Perancangan sistem bertujuan untuk menspesifikasikan aspek-aspek teknik yang menjadi solusi dalam perencanaan. Pada tahap ini perancangan akan didefinisikan secara detail untuk mengatasi masalah-masalah yang lebih teknis, berkaitan dengan kegiatan implementasi seperti perancangan topologi, perancangan *Quality of Service* (QoS) dan skenario pengujian performansi *Quality of service* (QoS).

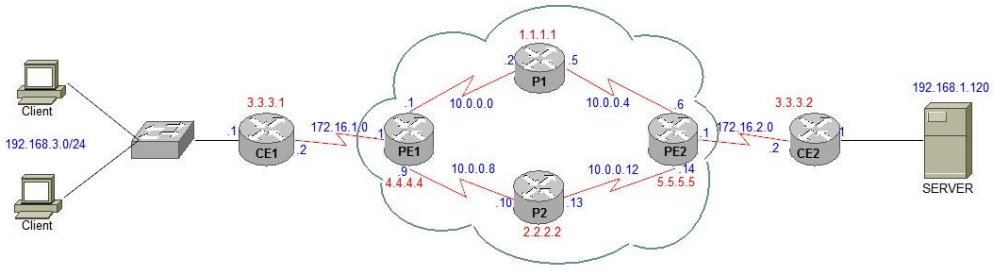
Untuk memudahkan proses implementasi diperlukan *flowcahart* yang membantu dalam memahami proses perancangan yang akan dibuat. *Flowchart* yang dibuat akan menjelaskan garis besar proses yang dilakukan pada saat pelaksanaan penelitian.



Gambar 3.8 *Flowchart* Perancangan Sistem

3.2.1 Perancangan Topologi

Dalam pelaksaan penelitian ini akan dirancang sebuah topologi yang akan digunakan sebagai infrastruktur jaringan IPTV. Adapun rancangan topologi dapat dilihat pada gambar 3.7 berikut.



Gambar 3.9 Topologi Jaringan

Gambar 3.7 merupakan topologi jaringan untuk implementasi layanan IPTV berbasis jaringan TCP/IP biasa dan juga jaringan MPLS. Untuk jaringan sendiri diterapkan pada pc router menggunakan GNS3. Penjelasan Pada gambar 3.7 sebagai berikut:

1. Bagian server, bagian ini merupakan pusat penyedia layanan, dimana dalam konfigurasi di atas Open IMS Core yang bertindak sebagai pusat penyedia layanan VoD dan IPTV.
2. Jaringan tersebut menggunakan 6 router cisco dengan IOS c3640-jk9o3s-mz.124-17.

3.2.2 Proses Instalasi Software dan Konfigurasi Jaringan

Setelah melakukan perancangn topologi, maka selanjutnya melakukan instalasi software dan konfigurasi router-router yang dipakai pada jaringan. Adapun software yang digunakan pada penletian ini adalah Open IMS Core yang bertindak sebagai server dari IPTV. Sedangkan jaringan akan dikonfigurasi pada simulator GNS3.

3.2.2.1 Instalasi Open IMS Core

OpenIMS merupakan *core IP Multimedia Subsystem* (IMS) yang terdiri dari FHoSS (HSS), dan CSCF (P-CSCF, I-CSCF dan S-CSCF). Keempat

komponen ini saling berhubungan dengan menggunakan alamat domain, sehingga untuk menjalankan *server* ini DNS *server* harus aktif terlebih dahulu. Pada penelitian ini Open IMS diinstal pada OS linux ubuntu 10.04. langkah-langkah untuk instalasi *software* dan konfigurasi OpenIMS adalah sebagai berikut :

1. Buat *direktori* OpenIMS di /opt dan masuk ke *direktori* tersebut

```
root@tama:~$ mkdir /opt/OpenIMSCore
root@tama:~$ cd /opt/OpenIMSCore/
```

2. *Download* paket ser_ims dan FHoSS dan disimpan dalam *file* /opt/OpenIMSCore. Pada penelitian ini digunakan paket FHoSS2008223.r0532.tgz dan ser_ims2008223.r0532.tgz. Ekstrak paket – paket tersebut.

```
root@tama:~/opt/OpenIMSCore$ tar -xvf FHoSS2008223.r0532.tgz
root@tama:~/opt/OpenIMSCore$ tar -xvf ser_ims2008223.r0532.tgz
```

3. Install FhoSS

```
root@tama:~/opt/OpenIMSCore$ cd FHoSS
root@tama:~/opt/OpenIMSCore/FHoSS$ ant compile
root@tama:~/opt/OpenIMSCore/FHoSS$ ant deploy
```

4. Install ser_ims

```
root@tama:~/opt/OpenIMSCore/FHoSS$ cd ../ser_ims
root@tama:~/opt/OpenIMSCore/ser_ims$ make install-libs all
```

5. Running program dan pengetesan

   Untuk menjalankan *software* OpenIMS diperlukan 4 terminal yang berbeda. Masing – masing shell dijalankan dengan perintah berikut.

```
root@tama:~/opt/OpenIMSCore$ ./pcscf.sh
root@tama:~/opt/OpenIMSCore$ ./icscf.sh
root@tama:~/opt/OpenIMSCore$ ./scscf.sh
root@tama:~/opt/OpenIMSCore$ ./fhoss.sh
```

3.2.2.2 Instalasi dan Konfigurasi *Domain Name System* (DNS)

Software OpenIMS baru bisa jalan jika DNS telah selesai diinstall dan aktif. Untuk itu maka lebih baik dilakukan instalasi dan konfigurasi DNS terlebih

dahulu. *Software* DNS yang digunakan adalah bind yang diinstall pada OS Ubuntu 10.04. Bind harus dapat menjadi DNS *server* dan juga Enum *server* sekaligus, maka ada dua tahap konfigurasi, yaitu konfigurasi DNS dan Enum.

1. Buat *file zone forward* untuk *domain* open-ims.dnszone *File* ini akan menerjemahkan akan menerjemahkan alamat domain menjadi alamat IP.

```
root@tama:~$ sudo gedit /etc/bind/open-ims.dnszone
```

Untuk konfigursi *file* open-ims.dnszone semua alamat domain diganti dengan IP address server seperti berikut.

```
$ORIGIN open-ims.test.
$TTL 1W
@                       1D  IN  SOA     open-ims.test.  root.ns.  open-
ims.test. (
                                        2008101001              ;
serial
                                        3H                      ;
refresh
                                        15M             ; retry
                                        1W                      ;
expiry
                                        1D )                    ;
minimum

                        1D IN NS        ns
ns                      1D IN A         192.168.1.120

trg                     1D IN A         192.168.1.120

pcscf                   1D IN A         192.168.1.120

open-ims.test.          1D IN A         192.168.1.120
icscf                   1D IN A         192.168.1.120
_sip                    1D SRV 0 0 5060 icscf
_sip._udp               1D SRV 0 0 5060 icscf
_sip._tcp               1D SRV 0 0 5060 icscf

open-ims.test.              1D IN NAPTR 10 50 "s" "SIP+D2U" ""
_sip._udp. open-ims.test.
open-ims.test.              1D IN NAPTR 20 50 "s" "SIP+D2T" ""
_sip._tcp. open-ims.test.


scscf                   1D IN A         192.168.1.120

hss                     1D IN A         192.168.1.120
ue                      1D IN A         192.168.1.120
presence                1D IN A         192.168.1.120
iptv                    1D IN A         192.168.1.120
```

2. Membuat *file zone statement*. *File* ini akan merutekan *query* ke *file zone* yang dibutuhkan. Biasanya *zone statement* di buat di *file* named.conf,

```
root@tama:~$ sudo gedit /etc/bind/named.conf
```

karena *file* ini yang akan pertama kali dilihat. Tetapi untuk membuat data DNS dan Enum yang teratur, maka *zone statement* dibuat di *file* yang berbeda.

Untuk konfigurasi dari *file* named.conf seperti yang tertera dibawah ini.

```
zone "open-ims.test" IN {
        type master;
        file "/etc/bind/open-ims.dnszone";
        notify no;
};
```

3. *Running Program* dan pengetesan

   Untuk menjalankan *program* dan pengetesan dilakukan dengan perintah berikut:

```
root@tama:~$ /etc/init.d/bind9 start
* Starting domain name service... bind
[ OK ]
root@tama:~$ nslookup open-ims.test
Server:         192.168.1.120
Address:        192.168.1.120#53

Name:   open-ims.test
Address: 192.168.1.120
```

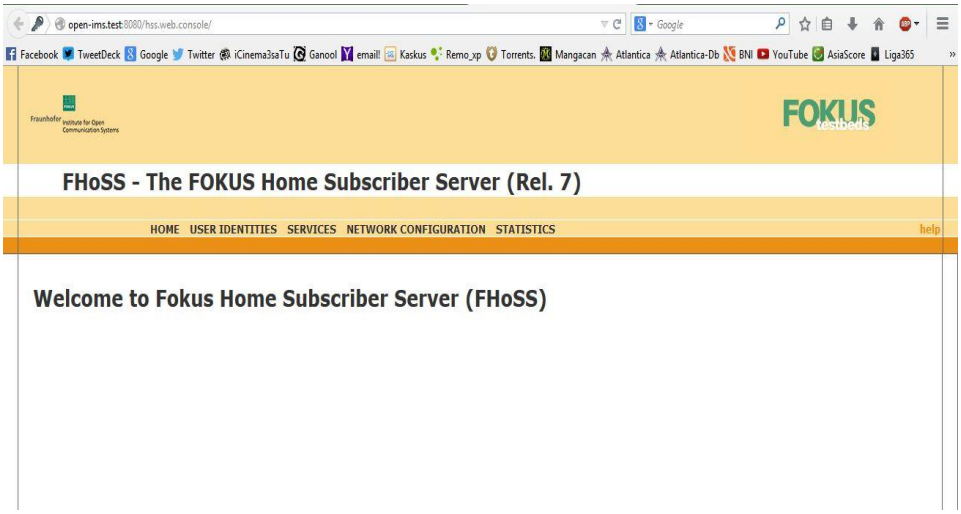
3.2.2.3 Instalasi dan Konfigurasi Aplication Server

Aplication Server (AS) menyimpan dan mengeksekusi layanan dan interface dengan CSCF menggunakan SIP (*Session Initiation Protocol*). AS mengizinkan provider pihak ketiga kemudahan integrasi dan penyebaran *added value service* ke system IMS. Contoh-contoh layanan (service) antara lain: *Conference call service,Voice Mail, Text-to-speech*, SMS, MMS, IPTV, Push to talk, Presence Information dan Instant messaging. Langkah-langkah menginstall UCT IPTV aplication server. Paket yang digunakan adalah uctiptv_advanced1.0.0.deb

```
root@tama:~$ sudo dpkg –I uctiptv_advanced1.0.0.deb
```

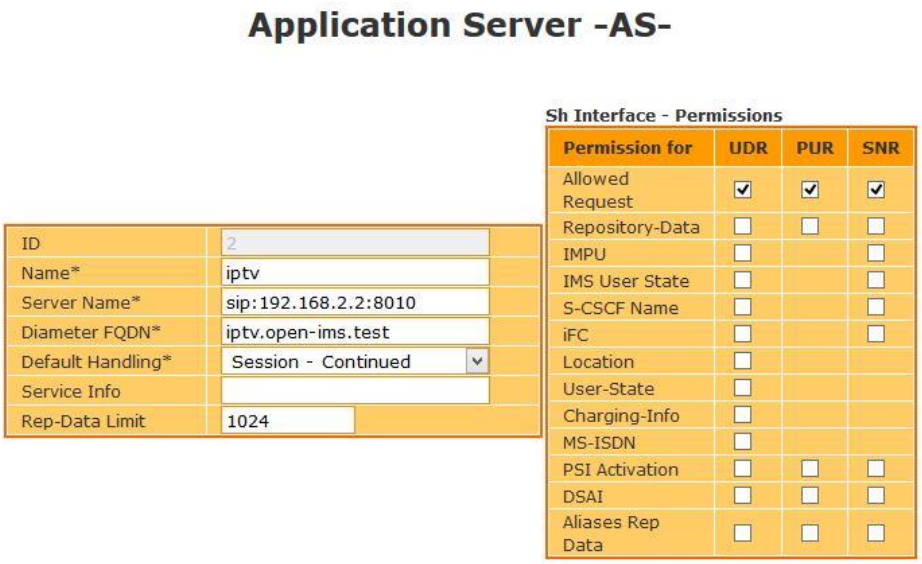
Selanjutnya melakukan konfigurasi FHoSS untuk meneruskan permintaan IPTV ke mesin yang menjalankan *application server*. Langkah-langkah konfigurasi FHoSS yaitu:

a. Buka open-ims.test:8080 pada web browser dengan user : hssAdmin dan password : hss



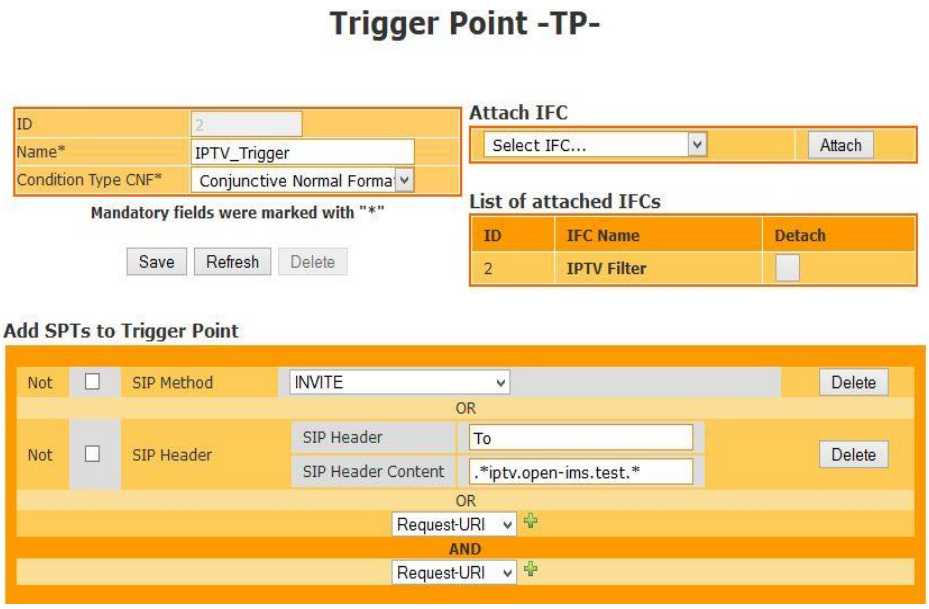
Gambar 3.10 Halaman Awal FHoSS

b. Buat *application server* (AS) untuk IPTV baru yang mana server *running* pada port 8010



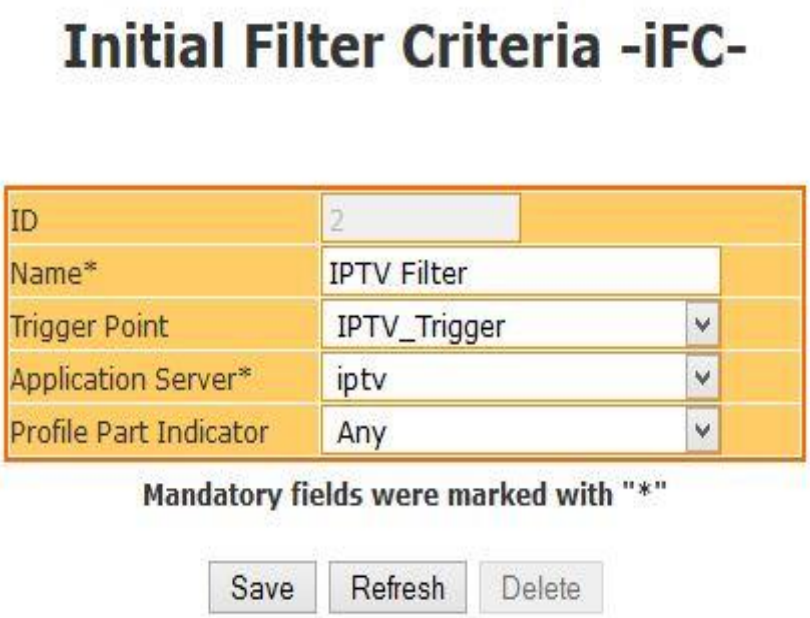
Gambar 3.11 *Aplication Server* (AS)

c. Kemudian Buat *trigger point* (TP) dengan nama IPTV_Trigger



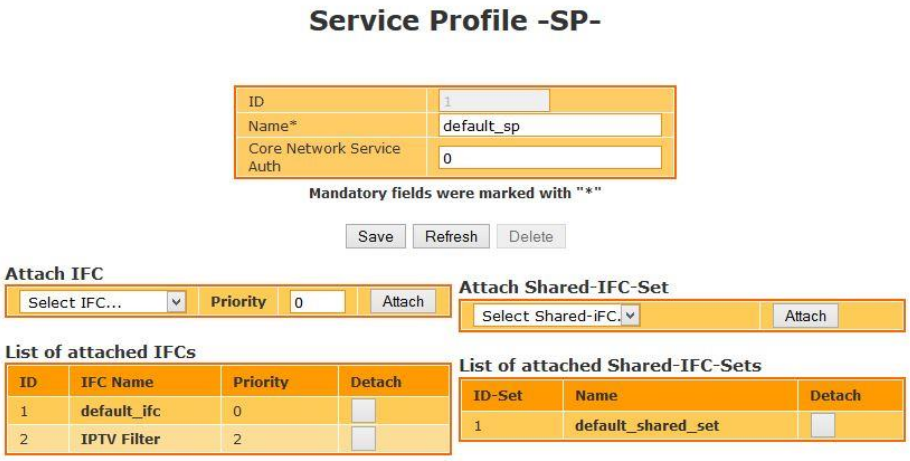
Gambar 3.12 *Trigger Point* (TP)

d. Hubungkan *application server* (AS) dan *trigger point* (TP) dengan *initial filter criteria* (iFC) dengan nama IPTV Filter.



Gambar 3.13 *Initial Filter Criteria*

e. Tambahakan iFC yang telah dibuat ke default *service profile* (SP)



Gambar 3.14 *Sevice Profile*

f. Setelah proses diatas selesai dilakukan maka jalankan iptv streaming server dengan perintah sebagai berikut untuk mengedit **key_value_file**

```
root@tama:~$ sudo gedit /usr/share/uctiptv_advanced/key_value_file
```

Isi dari file key_value_file adalah :

```
<key-value_pair>
<key>channel1</key>
       <value>rtsp://192.168.1.120:5554/channel1</value>
       #merupakan pengaturan alamat untuk media server agar dapat
       diakses di channel 1 pada uctimsclient.
       </key-value_pair>
<key-value_pair>
<key>channel2</key>
       <value>rtsp://192.168.1.120:5554/channel2</value>
       #merupakan pengaturan alamat untuk media server agar dapat
       diakses di channel 2 pada uctimsclient.
       </key-value_pair>
```

3.2.2.4 Installasi dan Konfigurasi IMS Client

Untuk dapat memanggil aplikasi IPTV maka kita memerlukan perangkat lunak yang dapat kita gunakan untuk melakukan registrasi dan memanggil aplikasi IPTV. Pada penelitian ini digunakan UCT IMS Client sebagai client. Langkah-langkah installasi dan konfigurasi sebagai berikut :

1. Periapkan *packages* uctimsclient1.0.14.tar.gz yang bisa di unduh dari http://uctimsclient.berlios.de/

2. Letakan file dalam folder downloads, kemudian dari terminal eksekusi file uctimsclient1.0.14.tar.gz yang berada dalam folder downloads dengan cara mengetikan perintah beikut :

```
root@tama:~$ sudo cd Downloads
root@tama:~/Downloads$ tar -zxvf uctimsclient1.0.14.tar.gz
```

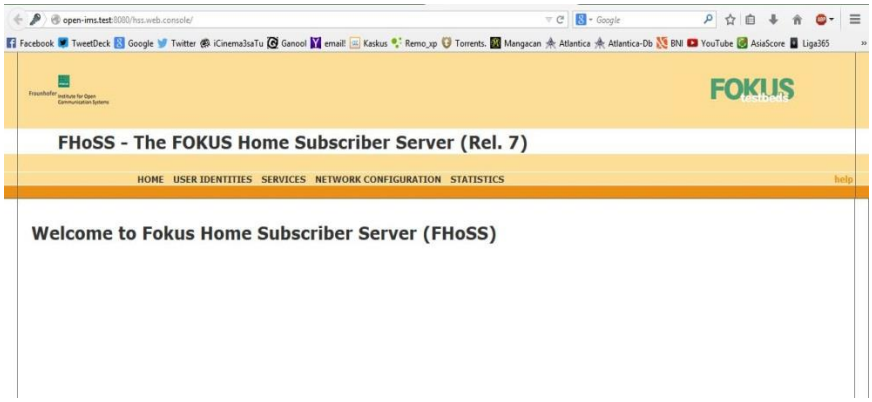
3. Sebelum menjalankan UCT IMS Client install terlebih dahulu *packages library* yang dibutuhkan oleh UCT IMS Client. Berikut perintah yang digunakan :

```
root@tama:~$ sudo apt-get install libexosip2-4 gstreamer0.10-plugins-
bad gstreamer0.10-plugins-ugly gstreamer0.10-ffmpeg vlc libvlc-dev
libexosip2-dev libgtk2.0-dev libxml2-dev libcurl4-openssl-dev
```

4. Jika *packages library* yang dibutuhkan telah diinstall langkah selanjutnya adalah menjalakan UCT IMS Client yang berada pada folder downloads dengan cara mengetikan perintah berikut berikut :

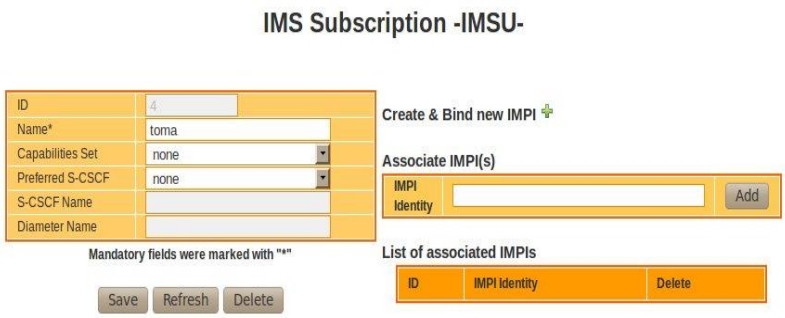
```
root@tama:~/Downloads$ cd uctimsclient
root@tama:~/Downloads/uctimsclient$ make
root@tama:~/Downloads/uctimsclient$ ./uctimsclient
```

5. Proses Installasi UCT IMS Client telah selesai, untuk dapat mengakses channel IPTV yang ada pada *aplication server* perlu konfigurasi hak akses untuk client agar dapat registrasi. Berikut konfigurasi yang dilakukan pada sisi server :
    1. Buka open-ims.test:8080 pada web browser dengan user : hssAdmin dan password : hss.



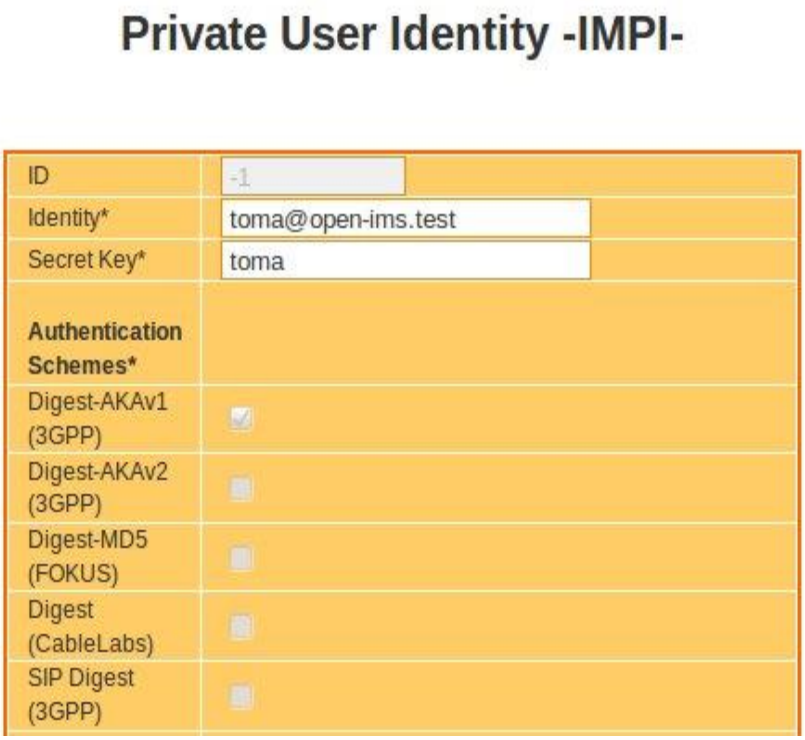
Gambar 3.15 Halaman Awal FHoSS

2. Pada halaman *home* FhoSS, klik tab *USER IDENTITIES* kemudian klik *create* pada bagian *IMS Subcription* seperti tampilan dibawah ini dan kemudian tekan tombol *save*

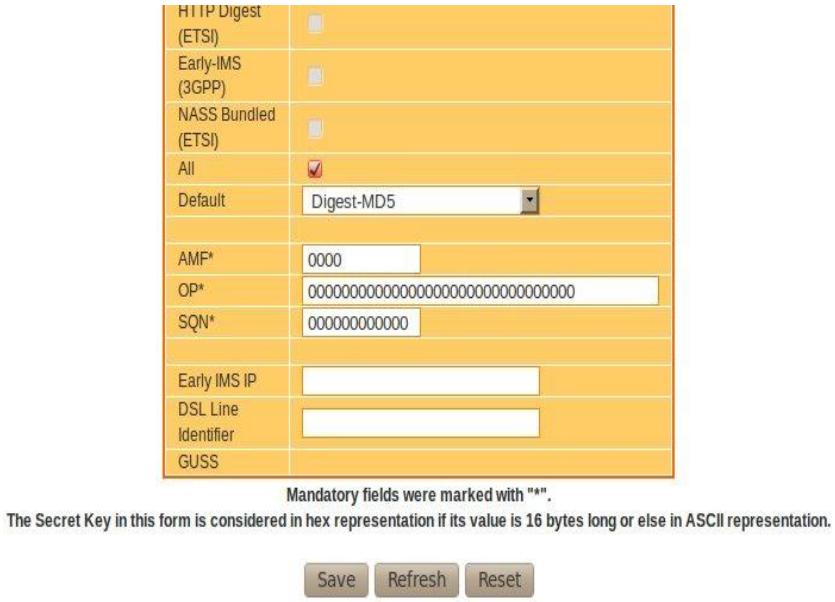


Gambar 3.16 *IMS Subcription* –IMSU-

3. Klik tanda + dibagian *create & Bind new* IMPI pada tampilan diatas, lalu isi bagian *identity, secret key*, centang *tab all, default* seperti tampilan dibawah ini dan kemudian tekan tombol *save*

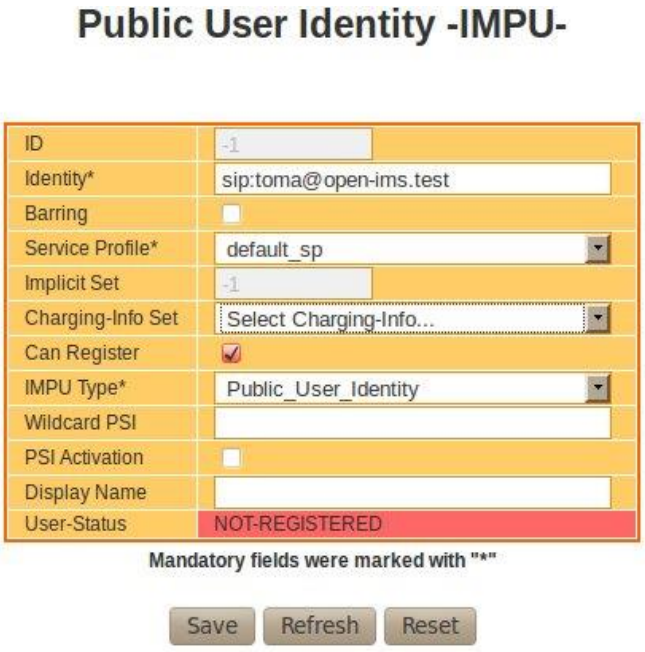


Gambar 3.17 Bagian Atas Private User Identity-IMPI-



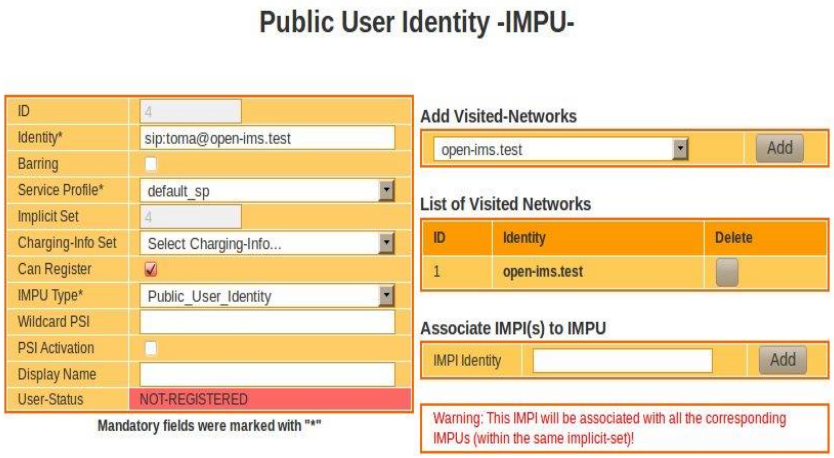
Gambar 3.18 Bagian Bawah Private User Identity-IMPI-

4. Klik tanda + dibagian *create & Bind new* IMPU pada tampilan diatas, lalu isi bagian *identity, service profile, display name* seperti tampilan dibawah ini dan kemudian tekan tombol *save*



Gambar 3.19 Public User Identity-IMPU-

5. tambahkan open-ims.test pada bagian *add visited-networks* seperti tampilan dibawah ini dan kemudian tekan tombol *save*



Gambar 3.20 *Add Visited-Networks*

6. Proses konfigurasi untuk user telah selasai, sekarang user telah dapat melakukan registrasi dan mengakses layanan IPTV.

3.2.2.5 Konfigurasi Jaringan

Konfigurasi jaringan akan implementasikan pada simulator GNS3 menggunakan IOS c3640-jk9o3s-mz.124-16.bin yang bertindak sebagai PC router. Tahap konfigurasi ini dilakukan dalam dua bentuk jaringan. Jaringan pertama merupakan jaringan TCP/IP biasa dimana konfigurasi yang dilkakukan menggunakan routing statis. Jaringan kedua merupakan jaringan MPLS dimana konfigurasi menggunkan beberapa protokol jaringan. Sebelum konfigurasi dilakukan pertama-tama adalah melakukan pembagian alamat IP, dapat dilihat pada tabel berikut.

Tabel 3.3 IP *Address Client dan Server*

Client	IP Address/Subnetting	Interface
Client1	192.168.3.2/24	Fe0/0
Client2	192.168.3.5/24	Fe0/0
Server IMS	192.168.1.120/24	Fe0/0

Alamat IP Address router dapat dilihat pada tabel 3.4 berikut.

Tabel 3.4 IP *Address Router*

Nama Router	Loopback 0	Subnetting	Interface
CE1	3.3.3.1/32	192.168.3.1/24	Fe0/1
		172.16.1.2/30	S0/0/0
CE2	3.3.3.2/32	192.168.1.1/24	Fe0/1
		172.16.2.2/30	S0/0/0
PE1	4.4.4.4/32	10.0.0.1/30	S0/0/0
		10.0.0.9/30	S0/0/1
		172.16.1.1/30	S1/0/0
PE2	5.5.5.5/32	10.0.0.6/30	S0/0/0
		10.0.0.4/30	S0/0/1
		172.16.2.1/30	S1/0/0
P1	1.1.1.1/32	10.0.0.2/30	S0/0/0
		10.0.0.5/30	S0/0/1
P2	2.2.2.2/32	10.0.0.10/30	S0/0/0
		10.0.0.13/30	S0/0/1

Setelah alamat IP Address telah ditentukan langkah selanjutnya adalah konfigurasi jaringan. Untuk konfigurasi kedua jaringan tersebut akan dijelaskan lebih detail pada sebagai berikut.

1. Konfigurasi Jaringan TCP/IP

   Konfigurasi pada jaringan ini menggunakan routing OSPF. Pertama-tama konfigurasi dilakukan dengan memberikan alamat IP yang di tentukan untuk setiap *interface* router.

```
interface Loopback0
  ip address 3.3.3.1 255.255.255.255
!
interface serial0/0/0
  ip address 10.0.0.1 255.255.255.252
!
interface serial0/0/1
  ip address 10.0.0.9 255.255.255.252
!
interface serial1/0/0
  ip address 172.16.1.1 255.255.255.252
```

Konfigurasi tersebut ditentukan untuk setiap *interface* router sesuai alamat ip address yang ada pada tabel 3.4. Setelah semua *interface* telah diberi alamat ip, selanjutnya melakukan konfigurasi routing OSPF pada setiap router agar dapat berkomunikasi satu sama lainnya.

```
CE1

router ospf 1
  network 3.3.3.1 0.0.0.0 area 0
  network 192.168.3.0 0.0.0.255 area 0
  network 172.16.1.0 0.0.0.3 area 0

CE2
!
router ospf 1
  network 3.3.3.2 0.0.0.0 area 0
  network 192.168.1.0 0.0.0.255 area 0
  network 172.16.2.0 0.0.0.3 area 0

P1
!
router ospf 1
  network 1.1.1.1 0.0.0.0 area 0
  network 10.0.0.0 0.0.0.3 area 0
  network 10.0.0.4 0.0.0.3 area 0
P2
!
router ospf 1
  network 2.2.2.2 0.0.0.0 area 0
  network 10.0.0.8 0.0.0.3 area 0
  network 10.0.0.12 0.0.0.3 area 0
```

```
PE1
!
router ospf 1
  network 4.4.4.4 0.0.0.0 area 0
  network 10.0.0.0 0.0.0.3 area 0
  network 10.0.0.8 0.0.0.3 area 0
  network 172.16.1.0 0.0.0.3 area 0

PE2
!
router ospf 1
  network 5.5.5.5 0.0.0.0 area 0
  network 10.0.0.4 0.0.0.3 area 0
  network 10.0.0.12 0.0.0.3 area 0
  network 172.16.2.0 0.0.0.3 area 0
```

Untuk memastikan konfigurasi telah berjalan dengan baik lakukan ping dari router CE1 ke router CE2 begitupun sebaliknya, jika ada balasan maka router telah berjalan dengan baik.

2. Konfigurasi Jaringan MPLS

   pada jaringan ini konfigurasi untuk alamat ip setiap interface sama dengan konfigurasi yang dilakukan pada jaringan sebelumnya. Kemudian, MPLS pada *router* PE1, *router* P1, *router* P2, dan *router* PE2 akan diaktifkan. MPLS harus diaktifkan, agar label dapat dikenali oleh *interface*. Setiap *interface router* yang akan meneruskan paket MPLS harus mengaktifkan fitur CEF (*Cisco Express Forwarding*). Setelah diaktifkan, maka *router* secara otomatis akan melihat paket LDP. Berikut konfiguranya.

```
##PE1

ip cef
mpls label protocol ldp
mpls label router-id loopback0 force
!
interface FastEthernet0/0
  mpls ip
!
interface FastEthernet 0/1
  mpls ip

##PE2
ip cef
mpls label protocol ldp
mpls label router-id loopback0 force
!
!
interface serial0/0/0
  mpls ip
!
interface serial0/0/1
  mpls ip

##P1
interface serial0/0/0
  mpls ip
!
interface serial0/0/1
  mpls ip

##P2
interface serial0/0/0
  mpls ip
!
interface serial0/0/1
  mpls ip
```

Untuk memastikan MPLS telah aktif lakukan verifikasi menggunakan command berikut **#sh mpls ldp neighbor.** Semua router harus membentuk adjacency dengan neighbor yang terhubung langsung, tetapi adjacency bisa terbentuk bila ada routing IGP-nya untuk mendistribusikan LDP session yang akan dipilih sebagai neighbor dan juga ada loopback address sebagai LDP router-id.

3.2.3 Perancangan *Quality of Service* (QoS)

Untuk merancang konfigurasi *Quality of Service* (QoS), pertama-tama perlu di ketahui secara mendetail tentang trafik apa saja yang akan digunakan oleh *customer*. Penulis mengasumsikan layanan IPTV merupakan layanan yang mngelirkan trafik audio, video dimana sangat sensitif tehadap delay, memiliki tingkat packet loss yang kecil dan juga membutuhkan prioritas *bandwidth*. Maka

didalam konfigurasi QoS pada router, paket ini akan diberikan sebuah prioritas dengan cara memberikan nilai *Diffserv Code Point* (DSCP) af31 berdasarkan rekomendasi dari RFC untuk *class* iptv, untuk bentuk umum nilai DSCP dapat dilihat pada tabel 2.2.

3.2.3.1 Konfigurasi *Quality of service* (QoS)

Pada penelitian ini metode yang digunakan untuk optimasi QoS adalah *Diff-Serv (Differentiated Service)* dimana pada metode ini *Field Type of Service (ToS)* disetiap *header IP* ditandai untuk mendefenisikan prioritas atau tingkat layanan yang dibutuhkan oleh sebuah paket. Kemudian nilai dari 3-bit *IP Precedence* dari *field Type of Service* dipetakan ke 3-bit *field EXP* pada *label* MPLS. *Diff-Serv* memberikan mekanisme QoS sedangkan MPLS memberikan kemampuan teknik *routing* dan *forwarding* yang dapat meningkatkan performasi jaringan. Pada model *Diff-Serv*, *router* dikonfigurasi *QoS policy* yang bisa diimplementasikan pada sebuah kelas *traffic*. Dengan mekanisme ini, *router* mengklasifikasikan dan mengimplementasikan *QoS policy* berdasarkan klasifikasi yang sudah ditentukan. Pada topologi jaringan yang sudah ada konfigurasi QoS dilakukan pada semua router yang berada pada jaringan, berikut konfigurasi QoS pada router CE1 dan CE2.

```
##CE1 & CE2
class-map match-any iptv
 match protocol rtsp
 match protocol rtp
 match protocol rtcp
class-map match-any network
 match protocol ospf
 match protocol snmp
!
class iptv
  set ip dscp af31
  priority percent 30
 class network
  set ip dscp cs6
  bandwidth percent 5
 class class-default
  fair-queue
!
interface FastEthernet0/0
 service-policy output IPTV-Link
!
interface FastEthernet0/1
 service-policy output IPTV-Link
```

Setelah konfigurasi pada router CE1 dan CE2 selesai langkah selanjutnya adalah konfigurasi pada router PE1, PE2, P1, dan P2 berikut konfigurasinya.

```
##PE1 & PE2

class-map match-any iptv
 match protocol rtsp
 match protocol rtp
 match protocol rtcp
class-map match-any network
 match protocol ospf
 match protocol snmp
!
class iptv
  set ip dscp af31
  priority percent 30
 class network
  set ip dscp cs6
  bandwidth percent 5
 class class-default
  fair-queue
!
interface serial0/0/0
 service-policy output IPTV-Link
!
interface serial0/0/1
 service-policy output IPTV-Link
!
interface seria1/0/0
 service-policy output IPTV-Link
```

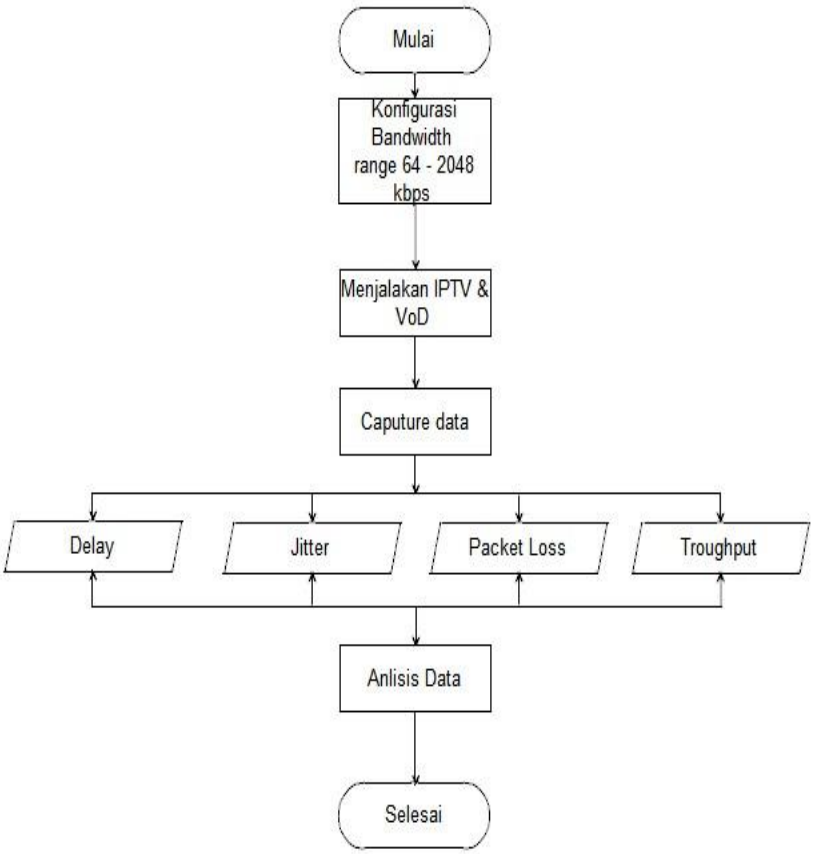
```
##P1 & P2

class-map match-any iptv
 match ip dscp af31
class-map match-any network
 match ip dscp cs6
!
policy-map IPTV-Link
 class iptv
  priority percent 30
 class network
  bandwidth percent 5
 class class-default
  fair-queue
!
interface serial0/0//0
 service-policy output IPTV-Link
!
interface serial0/0/1
 service-policy output IPTV-Link
```

3.2.4 Skenario Pengujian Performansi dari Parameter QoS

Pada penelirian ini akan dilakukan pengujian performansi jaringan dengan melakukan aplikasi *Video on Demand* dan live TV. Pengujian dilakukan dengan membandingkan performansi yang didapat dari jaringan TCP/IP biasa dengan

jaringan yang menggunakan MPLS dengan penerapan QoS. Untuk memudahkan dalam penetuan skenario dan pemahaman dalam skenario maka digambarkan pada flowchart. Berikut flowchart untuk skenario pengujian perfomansi QoS.



Gambar 3.21 Flowhart skenario pengujian perfomansi QoS

Pengujian dilakukan dengan beberapa macam skenario, yaitu:

1. Uji komunikasi layanan IPTV berupa aplikasi VoD melalui jaringan biasa dengan variasi nilai *bandwidth* dari jaringan yang digunakan sebesar 64 Kbps, 128 Kbps, 256 Kbps, 512 Kbps, 768 Kbps, 1024 Kbps, 1280 Kbps, 1536 Kbps, 1792 Kbps dan 2048 Kbps. Pada saat client sedang menjalakan aplikasi VoD, komunikasi tersebut akan di *capture* menggunakan aplikasi *sniiffing wireshark* untuk mengetahui trafik dan paket yang dikirim dari server menuju client. Setalah itu data yang didapat akan dinalisis untuk mengetahui performansi jaringan dan juga untuk

menganalisi parameter QoS berupa *delay, jitter, packet loss,* dan *throughput*.

2. Pada skenario ini tidak jauh berbeda dengan skenario yang pertama. Akan tetapi uji komunikasi layanan IPTV berupa aplikasi VoD dilakukan pada jaringan MPLS yang sudah dikonfigurasi QoS dengan variasi nilai *bandwidth* dari jaringan yang digunakan sebesar 64 Kbps, 128 Kbps, 256 Kbps, 512 Kbps, 768 Kbps, 1024 Kbps, 1280 Kbps, 1536 Kbps, 1792 Kbps dan 2048 Kbps. Pada saat client sedang menjalakan aplikasi VoD, komunikasi tersebut akan di *capture* menggunakan aplikasi *sniiffing wireshark* untuk mengetahui trafik dan paket yang dikirim dari server menuju client. Setalah itu data yang didapat akan dinalisis untuk mengetahui performansi jaringan dan juga untuk menganalisi parameter QoS berupa *delay, jitter, packet loss,* dan *throughput*.
3. Pada skenario ini dilakukan Uji komunikasi layanan IPTV berupa aplikasi VoD melalui jaringan biasa dengan variasi nilai *bandwidth* dari jaringan yang digunakan sebesar 64 Kbps, 128 Kbps, 256 Kbps, 512 Kbps, 768 Kbps, 1024 Kbps, 1280 Kbps, 1536 Kbps, 1792 Kbps dan 2048 Kbps. Jaringan akan dibebani trafik yang bertujuan untuk mengetahui performansi jaringan dalam keadaan sibuk. Adapun penambahan beban tarfik bervariasi mulai dari 10 %, 20 %, 40 % dan 80 % dari *bandwidth* yang disediakan. Pada saat client sedang menjalakan aplikasi VoD, komunikasi tersebut akan di *capture* menggunakan aplikasi *sniiffing wireshark* untuk mengetahui trafik dan paket yang dikirim dari server menuju client. Setalah itu data yang didapat akan dinalisis untuk mengetahui performansi jaringan dan juga untuk menganalisi parameter QoS berupa *delay, jitter, packet loss,* dan *throughput*.
4. Pada skenario ini dilakukan Uji komunikasi layanan IPTV berupa aplikasi VoD melalui jaringan MPLS yang sudah dikonfigurasi QoS dengan variasi nilai *bandwidth* dari jaringan yang digunakan sebesar 64 Kbps, 128 Kbps, 256 Kbps, 512 Kbps, 768 Kbps, 1024 Kbps, 1280 Kbps, 1536 Kbps, 1792 Kbps dan 2048 Kbps. Jaringan akan dibebani trafik yang bertujuan untuk

mengetahui performansi jaringan dalam keadaan sibuk. Adapun penambahan beban tarfik bervariasi mulai dari 10 %, 20 %, 40 % dan 80 % dari *bandwidth* yang disediakan. Pada saat client sedang menjalakan aplikasi VoD, komunikasi tersebut akan di *capture* menggunakan aplikasi *sniiffing wireshark* untuk mengetahui trafik dan paket yang dikirim dari server menuju client. Setalah itu data yang didapat akan dinalisis untuk mengetahui performansi jaringan dan juga untuk menganalisi parameter QoS berupa *delay, jitter, packet loss,* dan *throughput*.

Untuk setiap besar *bandwidth* dilakukan 10 kali pengukuran, untuk setiap pengukuran dilakukan pengamatan selama 30 detik dengan menggunakan *software wireshark* pada sisi client. Sedangkan untuk penambahan beban trafik mengunakan *software iperf*.