

BAB II

LANDASAN TEORI

2.1. Pengiriman Suara

Suara manusia yang dikirimkan lewat media kabel, berbentuk sinyal analog. Pada awalnya, jaringan telepon dibangun untuk infrastruktur analog. Berkomunikasi secara analog menimbulkan banyak permasalahan saat penerimaan karena adanya derau (*noise*). Untuk mengatasinya, sinyal analog tersebut dilewatkan penguat (*amplifier*).

Dalam pengiriman secara digital, pengulang (*repeater*) tidak hanya menguatkan sinyal suara tetapi juga dapat membersihkan *noise*. Keunggulan inilah yang kemudian dimanfaatkan untuk jaringan telepon, yaitu dengan membuat sistem PCM (*Pulse Code Modulation*).

2.1.1. *Pulse Code Modulation*

PCM akan mengubah isyarat analog menjadi isyarat 0 atau 1 yang terurut. Proses PCM meliputi:

1. *Filtering*

Pada tahap ini, sinyal suara dibersihkan dari sinyal gangguan.

2. *Sampling*

Sinyal suara yang telah difilter kemudian di-*sampling* (dicuplik) 8000 *sample* per detik.

3. *Conversion*

Sinyal analog lalu diubah menjadi isyarat digital.

2.1.2. Telephony Signaling

Metode pensinyalan telepon yang berjalan diatas berbagai media, pada umumnya dikelompokkan ke dalam:

a. *User-to-network signaling (local loop)*

Infrastruktur sebuah telepon dimulai dengan sepasang kabel tembaga yang dihubungkan ke telepon rumah. Kabel fisik ini dikenal dengan istilah *local loop*. *Local loop* menghubungkan telepon rumah dengan *central office (CO) switch*.

b. *Network-to-network signaling (trunk)*

Jalur komunikasi yang menghubungkan antar beberapa CO switch dikenal sebagai *trunk*.

2.2. Voice over Internet Protocol

Voice over Internet Protocol (VoIP) adalah teknologi yang mampu melewati trafik suara, video dan data yang berbentuk paket melalui jaringan IP.

2.2.1. Kebutuhan Jaringan

Jaringan IP didesain untuk membawa (melewatkan) data. *Data networks* tidak didesain untuk membawa trafik suara. Meskipun *traffic data* bersifat *best-effort* dan dapat menahan *delay*, *jitter*, dan *packet loss*, namun *voice traffic* bersifat *real-time*.

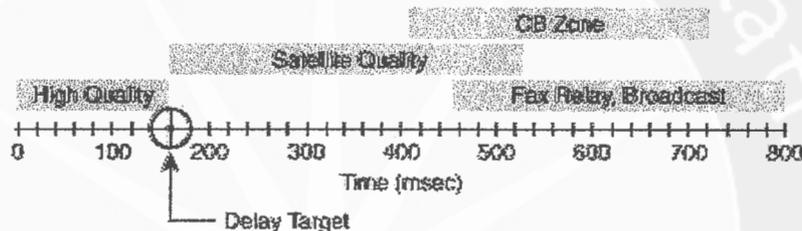
Contoh permasalahan, misalnya pihak pengirim berkata "Selamat pagi, apa kabar?". Dengan adanya *end-to-end delay* maka pihak penerima akan mendengar ".....Selamat pagi, apa kabar?". Jika terdapat *jitter*, pihak penerima akan mendengar "Selamat.....pagi, apa.....kabar?". Jika ada paket yang hilang (*packet loss*), maka pihak penerima akan mendengar "Sela... pagi,a...bar?".

a. *Packet loss*

Packet loss timbul ketika terjadi *peak load* dan *congestion* (kemacetan transmisi paket akibat padatnya saluran transmisi) dalam batas waktu tertentu, maka *frame* suara (gabungan data *payload* dan *header* yang ditransmisikan) akan dibuang sebagaimana perlakuan terhadap *frame* data lainnya pada jaringan berbasis IP. Salah satu solusi alternatif permasalahan di atas adalah 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.

b. *Delay*

Ukuran kualitas suara dapat dilihat pada gambar 2.1.



Gambar 2.1. Kualitas suara dengan *delay* (Davidson, 2006)

Delay bisa terjadi karena banyak faktor, misalnya *delay* yang terjadi karena perambatan dari sisi pengirim ke sisi penerima membutuhkan waktu, apalagi jika trafik jaringan padat, akan sangat membutuhkan banyak waktu. Ini akan mengakibatkan suara terdengar patah-patah.

Jenis *delay* yang mengganggu kualitas VoIP dapat antara lain:

1. *Propagation delay* (*delay* yang terjadi akibat transmisi melalui jarak antar pengirim dan penerima)
2. *Serialization delay* (*delay* pada saat proses peletakan bit ke dalam *circuit*)
3. *Processing delay* (*delay* yang terjadi saat proses *coding*, *compression*, *decompression* dan *decoding*)

4. *Packetization delay* (*delay* yang terjadi saat proses paketisasi *digital voice sample*)
5. *Queuing delay* (*delay* akibat waktu tunggu paket sampai dilayani)
6. *Jitter buffer* (*delay* akibat adanya *buffer* untuk mengatasi *jitter*)

c. *Jitter* (*delay* yang bervariasi)

Jitter adalah variasi diantara keadaan kedatangan paket yang diperkirakan dan waktu sebenarnya ketika paket tersebut benar-benar diterima. Dengan kata lain, *jitter* merupakan variasi *delay* yang terjadi akibat adanya selisih waktu atau interval antar kedatangan paket di penerima. Untuk mengatasi *jitter* maka 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.

d. Throughput

Throughput adalah jumlah data aktual (tanpa *header*) yang dikirim antara dua titik dalam selang waktu tertentu. *Throughput* dipengaruhi oleh *bandwidth* (semakin besar *bandwidth*, maka akan semakin besar *throughput*-nya), *error performance*, kepadatan trafik, dan faktor lainnya. Cara mempertinggi *throughput* bisa dengan:

1. *Queuing*

Queuing memungkinkan *router* dan *switch* menangani trafik yang terlalu penuh (*overload*), mengukur kepadatan jaringan, memberi prioritas pada trafik, dan alokasi *bandwidth*.

2. *Congestion avoidance*

Teknik *congestion avoidance* dapat memonitor beban pada trafik jaringan. Tujuannya adalah untuk mengantisipasi dan menghindari kepadatan pada suatu jaringan dan kemacetan antar jaringan sebelum hal tersebut menjadi penyebab permasalahan. Pada saat yang sama, teknik ini memaksimalkan *throughput* jaringan dan kapasitas penggunaan dan meminimalkan *packet loss* serta *delay*.

3. Mengkompres *header*

Suara dibawa menggunakan RTP (*Real-Time Transport Protocol*), yang dibungkus dalam UDP, kemudian dimasukkan dalam paket IP untuk dilewatkan jaringan IP. Oleh karena itu terdapat 40 *bytes* RTP/UDP/IP *header*. Ukuran *header* ini terlalu besar bila dibandingkan dengan data suara yang hanya 20 *bytes*. CRTP (*Compressed RTP*) dapat mengurangi *header* sampai 2 *bytes* pada kasus-kasus tertentu, jadi menghemat *bandwidth* dan menjadikan *throughput* lebih baik.

4. RSVP (*Resource Reservation Protocol*)

RSVP adalah protokol di lapisan transport yang memungkinkan penyediaan *bandwidth* dalam jumlah tertentu, sesuai dengan kebutuhan.

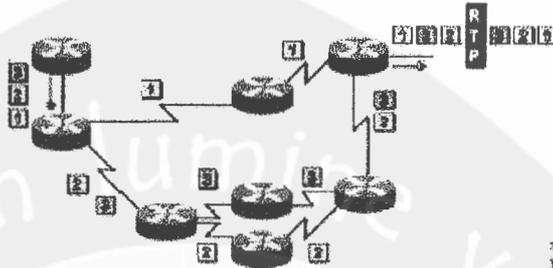
RSVP digunakan untuk memesan *bandwidth* agar data suara yang akan dikirimkan tidak mengalami *delay* ataupun kerusakan saat mencapai alamat tujuan.

5. Fragmentasi

Fragmentasi mendefinisikan ukuran maksimal dari sebuah paket data dan digunakan di lingkungan suara untuk mencegah *serialization delay* yang berlebihan. *Serialization delay* merupakan waktu yang dibutuhkan untuk menempatkan bit-bit pada *interface*. Contohnya, paket berukuran 1500 byte membutuhkan waktu 187 ms untuk keluar dari router diatas 64 kbps link. Jika sebuah data paket yang bersifat *best-effort* yang berukuran 1500 *bytes* dikirimkan, paket suara yang bersifat *real-time* akan diantri hingga paket data yang besar tersebut terkirim. *Delay* ini tidak dapat ditolerir dalam trafik suara. Bagaimanapun juga, jika sebuah data paket yang bersifat *best-effort* difragmentasikan ke dalam potongan-potongan yang lebih kecil, maka pengiriman paket data dapat disela oleh paket suara yang bersifat *real-time*. Keduanya dapat dikirim secara bersama-sama pada link atau saluran yang berkecepatan rendah tanpa menimbulkan *delay* yang berlebihan pada pengiriman paket suara.

e. Pengurutan Paket Suara

Pada gambar 2.2, RTP mengurutkan paket dengan menggunakan nomor urut.



Gambar 2.2. Pengurutan Paket (Cisco, 2004)

Ketika paket suara dikirimkan melalui jaringan, muncul permasalahan untuk pengurutan paket oleh IP. RTP akan mengurutkan paket dengan menggunakan nomor urut yang ada pada masing-masing paket. Saat RTP mengurutkan paket, paket suara tersebut diberi jarak berdasarkan *time stamp* yang ada di setiap *header* RTP. Penerima mendengar suara secara terurut dengan *timing* yang sama seperti saat aliran suara dikirimkan dari sumbernya.

f. Reliability dan Availability

Sistem telepon tradisional menuntut *uptime* sebesar 99,999%. Ketika menggabungkan suara dengan data pada jaringan IP, kebutuhan akan *reliability* dan *availability* harus dipertimbangkan.

Jika *availability* mencapai 99,999% maka *downtime* yang ada hanya 5,3 menit dalam satu tahun. *Downtime* adalah kebalikan dari *uptime*. *Uptime* sendiri adalah waktu sistem berjalan dengan normal.

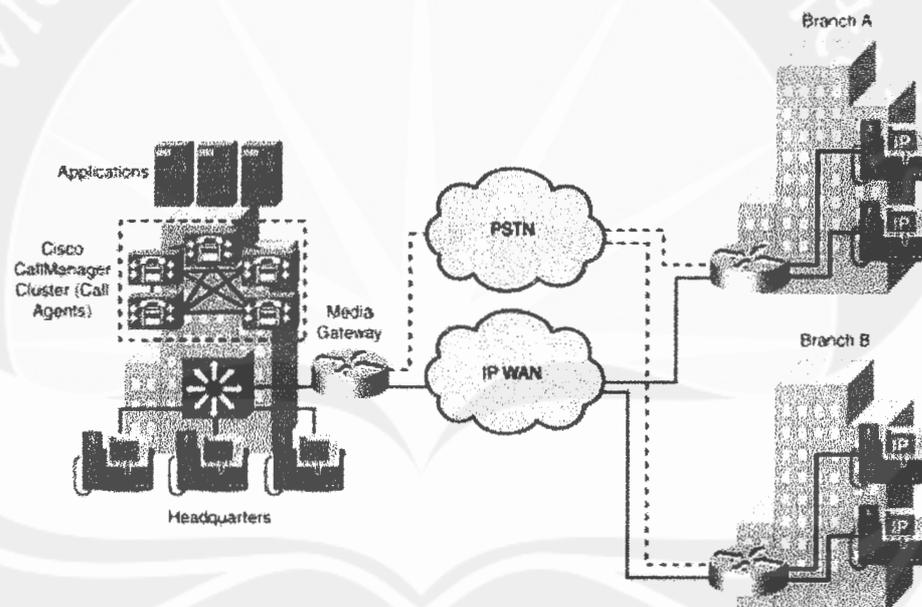
Reliability merupakan ukuran bagaimana kehandalan sebuah jaringan. Sedangkan *availability* merupakan ukuran bagaimana jaringan dapat diakses oleh user.

Untuk memperbaiki *availability* dan *reliability* dapat digunakan metode:

- a. *redundant hardware*
- b. *redundant links*
- c. menggunakan UPS (jika perlu ditambah generator listrik)
- d. dengan melakukan manajemen jaringan secara proaktif

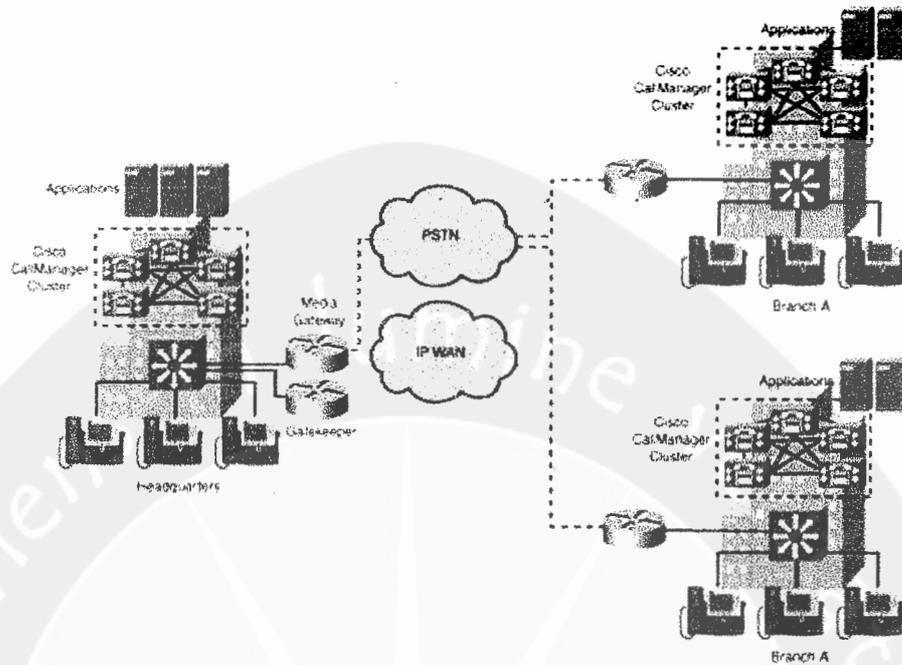
2.2.2. Arsitektur Jaringan VoIP

Salah satu keuntungan teknologi VoIP adalah memungkinkan jaringan dibangun dengan arsitektur yang terpusat maupun terdistribusi. Yang menentukan jenis arsitektur jaringan VoIP adalah kebutuhan bisnis perusahaan.



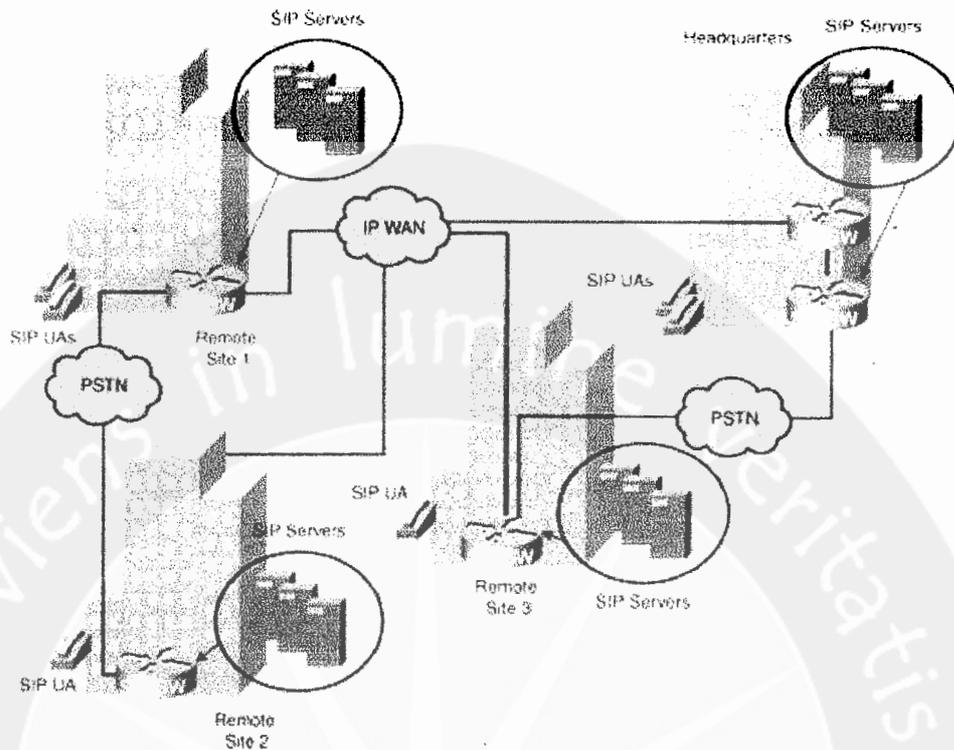
Gambar 2.3. Arsitektur Jaringan yang Terpusat (Wallace, 2006)

Komponen arsitektur yang terpusat, seperti pada gambar 2.3, adalah *Call Agent* (yaitu berupa *Private Branch Exchange* atau PBX), IP WAN (untuk membawa *call control signaling* dan beban suara antara pusat dengan masing-masing cabang), dan *media gateway* yang menyediakan interkoneksi antara jaringan telepon, *individual endpoints* (masing-masing pengguna), dan jaringan IP.



Gambar 2.4. Arsitektur Jaringan yang Terdistribusi dengan Protokol H.323 (Wallace, 2006)

Komponen arsitektur yang terdistribusi dengan protokol H.323, seperti pada gambar 2.4, adalah IP WAN, *media gateway*, dan *gatekeeper*. Fungsi *gateway* adalah sebagai translator, misalnya untuk mengalihkan protocol IP ke PSTN. *Gateway* bisa berupa *router*, *stand alone voice gateway*, atau *network modules*. Fungsi dari *gatekeeper* adalah sebagai *address resolution* yang memetakan nomor atau nama ke dalam bentuk nomor telepon, mengatur jumlah pengguna yang dapat menelpon secara bersama-sama, mengontrol penggunaan *bandwidth*, dan manajemen wilayah yang dapat dijangkau.



Gambar 2.5. SIP Distributed Network Architecture (Wallace, 2006)

Komponen arsitektur terdistribusi dengan protocol SIP, seperti pada gambar 2.5, adalah SIP *user agent*, *proxy server*, *redirect server*, dan *registrar server*. Komponen SIP akan dibahas selanjutnya.

Bila dibandingkan dengan sistem yang terdistribusi, sistem terpusat lebih mudah dikonfigurasi sebab PBX terdapat di pusat. Namun, sistem terpusat kurang handal dan tidak efisien sebab setiap panggilan antar cabang harus melalui pusat sehingga menyebabkan lalu lintas pusat menjadi penuh dan jika sistem di pusat tidak dapat berfungsi atau mengalami kerusakan, maka semua sistem tidak dapat berkomunikasi.

2.2.3. *Voice Packetization*

Hal-hal yang mempengaruhi pemaketan suara adalah:

a. Coding-Decoding (CODEC)

Agar dapat melewati jalur *packet switch*, VoIP membutuhkan coder dan decoder. Coder mengkonversi sinyal audio menjadi paket data digital, lalu dikompres (dimampatkan), setelah itu baru dikirim melalui jalur Internet. Di titik lain, decoder akan membuka data digital yang diterimanya, dan selanjutnya diubah kembali menjadi sinyal audio. Konversi codec bekerja dengan cara melakukan pencuplikan pada sinyal audio (melakukan *sampling*) dalam jumlah tertentu per detiknya. Pemilihan jenis codec sangat berpengaruh pada penggunaan *bandwidth*.

b. Protokol *signaling*

Protokol *signaling* yang sering dipakai oleh VoIP adalah H.323, MGCP, dan SIP. Protokol yang mengangkut paket suara adalah RTP, sedangkan yang mengontrol kualitas suara adalah RTCP.

c. Kompresi paket

Kompresi paket suara dapat dilakukan pada muatannya atau pada *header*-nya. Dibagian *header* akan dikompres, sehingga *header* yang semula besarnya 40 bytes (40 bytes berasal dari 20 bytes IP header + 8 bytes UDP header + 12 bytes RTP header) dikompres menjadi 2 bytes.

d. Ukuran kualitas suara

Kejernihan suara dipengaruhi oleh *fidelity* (dihasilkan oleh *bandwidth* yang konsisten), *echo* (gema), *delay*, dan *jitter*. Untuk mengukur kejernihan suara dapat menggunakan teknik:

1. MOS (*Mean Opinion Score*): berdasarkan tingkat pendengaran manusia dan yang paling sederhana dan mudah dilakukan karena tidak membutuhkan alat khusus
2. PSQM (*Perceptual Speech Quality Measurement*)
3. PAMS (*Perceptual Analysis Measurement System*)
4. PESQ (*Perceptual Evaluation of Speech Quality*): merupakan gabungan dari teknik PSQM dan PAMS.

2.2.4. Kebutuhan *Bandwidth*

Kualitas VoIP dipengaruhi oleh *bandwidth*. Agar suara terdengar jelas dan tidak patah-patah, maka kebutuhan *bandwidth* perlu mendapat perhatian. *Bandwidth* yang dibutuhkan untuk pengiriman paket suara melalui Internet Protokol tergantung pada jenis codec yang digunakan.

Tabel 2.1. Jenis codec dan Kebutuhan *Bandwidth*nya (Wallace, 2006)

Jenis Codec	Codec <i>Bandwidth</i>	Codec Speed
G.711	64 Kbps	8 KHz
G.726	32, 24, 16 Kbps	8 KHz
G.728	16 Kbps	8 KHz
G.729	8 Kbps	8 KHz
G.723	6.3, 5.3 Kbps	8 KHz

Faktor yang mempengaruhi perhitungan *bandwidth* adalah:

- Kebutuhan *bandwidth* tergantung pada jenis codec. Dengan kata lain, semakin tinggi ukuran codec *bandwidth*, semakin lebar pula *bandwidth* jalur yang dibutuhkan.
- Overhead* yang besar pada data link akan membutuhkan *bandwidth* yang besar. Voip sendiri membawa 40-byte IP/UDP/RTP *header*, dengan asumsi tidak dikompres. Berdasarkan pada pemakaian protokol lapisan 2, overhead bisa tumbuh secara nyata. Diperlukan *bandwidth* yang lebih banyak untuk mengangkut frame VoIP dengan overhead lapisan 2 yang lebih besar.
- Ukuran sampling yang lebih besar membutuhkan *bandwidth* yang kecil
- Kompresi *header* RTP akan membutuhkan *bandwidth* yang kecil

Untuk merencanakan *bandwidth* yang diperlukan maka dibutuhkan analisis trafik. Tujuan lain yang hendak dicapai dengan analisis trafik adalah untuk merencanakan keuntungan perusahaan tetapi tidak mengabaikan kepuasan pengguna.

2.2.5. Analisis Trafik

Dua faktor penting yang perlu diperhatikan dalam merancang jaringan adalah servis dan biaya. Servis sangat berkaitan dengan kepuasan pelanggan dan biaya selalu menjadi faktor penting dalam profit yang diperoleh.

Trafik atau lalu lintas digambarkan sebagai jumlah data atau banyaknya pesan pada suatu sirkuit selama periode waktu yang ditentukan. Analisa trafik memungkinkan untuk menentukan jumlah *bandwidth* yang dibutuhkan dalam sirkuit yang ada untuk data dan untuk panggilan suara.

2.3. Dial Plans

Dial plans adalah sekumpulan aturan yang mengatur saluran (*channel*) dan lalu lintas suara sehingga akan membentuk *call flow*. *Dial plans* pada asterisk diatur dalam file konfigurasi yang bernama *extension.conf*.

Contoh *dial plans* sederhana:

```
[internal]
Exten => 101, 1, Dial(SIP/101)
Exten => 101, 2, Answer()
```

2.3.1. Komponen Dial Plans

Dial plan terdiri dari empat bagian, yaitu *context*, *extensions*, *priorities*, dan *applications*.

a. Context

Dial plans dipecah menjadi bagian-bagian yang salah satunya disebut *context*. *Context* adalah nama sekelompok *extension*. Sederhananya, *context* menyimpan

bagian-bagian dialplan yang berbeda dari interaksi antara satu sama lain. Sebuah extension yang didefinisikan dalam satu *context* akan terisolir dari extension pada *context* yang lain. Penulisan sintaksnya biasanya dalam tanda []. Pada contoh dialplan sederhana di atas, [internal] adalah *context*-nya.

b. *Extension*

Extension adalah sebuah instruksi yang akan diikuti dan dilakukan oleh asterisk ketika ada pemicu, seperti panggilan masuk. Penulisan sintaksnya diawali dengan kata *exten*, kemudian diikuti tanda '=', maka akan menjadi seperti:

```
Exten =>
```

Sebuah extension yang lengkap terdiri dari 3 komponen, yaitu:

1. Nama atau nomor extension
2. Prioritas
3. Perintah (aplikasi) yang akan melakukan beberapa aksi pada panggilan

Komponen-komponen tersebut akan membentuk extension dengan urutan sebagai berikut:

```
exten => nama, prioritas, aplikasi()
```

Contoh extension:

```
exten => 101, 1, Dial(SIP/101)
```

```
exten => 101, 2, Answer()
```

c. *Priority*

Masing-masing extension bisa memiliki beberapa langkah-langkah, yang disebut prioritas. Tiap prioritas diberi nomor terurut, dimulai dari 1. Contohnya:

```
exten => 123, 1, Answer()
```

```
exten => 123, 2, Hangup()
```

Pada contoh diatas, extension akan menjawab (answer) panggilan (sebab prioritasnya nomor 1), kemudian akan memutus (hang up) panggilan (prioritasnya nomor 2).

d. Application

Application adalah perintah spesifik kepada asterisk untuk melakukan sebuah tindakan tertentu, seperti dial(), answer(), hangup(), playback(), dan lain sebagainya.

2.3.2. Ekspresi Dial Plans

Ekspresi adalah kombinasi dari variabel, operator, dan suatu nilai yang digunakan untuk menghasilkan sesuatu. Sebuah ekspresi dapat berupa nilai, string, atau operasi matematika. Pada asterisk, penulisan sintaks ekspresi akan menjadi:

`$(expression)`

Contoh ekspresi:

`$(${COUNT} + 1)`

Contoh ekspresi dalam dialplans:

```
exten => 321,1,Set(COUNT=3)
```

```
exten => 321,2,Set(NEWCOUNT=${ ${COUNT} + 1 })
```

```
exten => 321,3,SayNumber( ${NEWCOUNT} )
```

Berdasarkan prioritasnya, maka alur *dial plans* diatas adalah:

1. Mengeset nilai variabel count dengan nilai 3.
2. Mengeset nilai variabel newcount berdasarkan ekspresi yang ada:

```
exten => 321, 2,Set(NEWCOUNT=${3 + 1})
```

Maka setelah sintaks ekspresi diatas dieksekusi, yaitu menambahkan 3 dengan 1, asterisk akan mengerjakan:

exten => 321, 2, Set (NEWCOUNT=4)

3. Asterisk akan mengerjakan perintah SayNumber(), yaitu dengan menyebutkan nilai yang disimpan oleh variabel newcount, yaitu 4.

2.3.3. Operator Ekspresi *Dial Plans*

Operator digunakan untuk memanipulasi variabel yang ada. Operator yang tersedia dalam asterisk antara lain:

a. Operator boolean

1. OR

Sintaksnya : `expr1 | expr2`

2. AND

Sintaksnya : `expr1 & expr2`

3. Operator perbandingan

Sintaksnya : `expr1 {=, <, >, <=, >=, !=} expr2`

b. Operator matematika

1. Penjumlahan dan pengurangan

Sintaksnya : `Expr1 {+, -} expr2`

2. Perkalian dan pembagian

Sintaksnya : `Expr1 {*, /} expr2`

3. Sisa hasil bagi

Sintaksnya : `Expr1 % expr2`

c. Operator ekspresi regular

Sintaksnya ditulis `Expr1 : expr2`

2.3.4. *Dial plans Function*

Fungsi akan memungkinkan untuk membuat suatu ekspresi yang lebih kompleks. Sintaks fungsi akan berbentuk:

```
#{NAMA_FUNGSI(argumen)}
```

Pada dialplans, biasanya fungsi ditulis didalam aplikasi Set(). Contohnya:

```
Exten => 123, 1, Set(TEST=ex)
```

```
Exten => 123, 2, SayNumber($LEN(#{TEST}))
```

Ada tiga jenis variabel yang bisa digunakan dalam fungsi, yaitu:

1. Variabel global

Merupakan variabel yang berlaku bagi semua extension dan semua *context*

2. Variabel *channel*

Merupakan variabel yang dikaitkan dengan satu panggilan tertentu

3. Variabel *environment*

Merupakan variabel untuk mengakses lingkungan unik. Variabel ini jarang digunakan.

Ekspresi dan fungsi bisa digunakan dalam percabangan kondisional pada *dial plans* yang lebih kompleks.

1. GotoIf()

Kunci percabangan kondisional adalah perintah GotoIf() yang akan mengevaluasi sebuah ekspresi dan mengirimnya ke sebuah tujuan tertentu berdasarkan hasil evaluasinya, true atau false. Sintaksnya akan berbentuk:

```
GotoIf(expression?destination1:destination2)
```

Jika hasil evaluasi ekspresi adalah true, maka dikirim ke tujuan pertama. Jika hasilnya false, maka dikirim ke tujuan kedua.

Contoh:

```
exten => 345,1,Set(TEST=1)
exten => 345,2,GotoIf(${TEST} = 1)?10:20)
exten => 345,10,Playback(weasels-eaten-phonesys)
exten => 345,20,Playback(office-iguanas)
```

Pada perintah dengan GotoIf(...), jika nilai TEST adalah 1, maka akan menuju ke prioritas nomor 10 (memutar weasels-eaten-phonesys). Sebaliknya, jika tidak bernilai 1, maka menuju ke prioritas nomor 20 (memutar office-iguanas).

2. GotoIfTime()

Sintaks ini merupakan percabangan kondisional time-based. Urutan penulisan sintaksnya secara lengkap:

```
GotoIfTime(times, days_of_week, days_of_month, months? label)
```

Dimana:

Times = dalam format 24-jam (misalnya 09:00-17.00).

Days_of_week = dalam format mon, tue, wed, thu, fri, sat, sun atau mon-fri.

Days_of_month = merupakan tanggal dalam bulan yang bersangkutan, misalnya 7-12 atau 15,30.

Months = menyatakan bulan, dalam format jan, feb, mar, apr, dan selanjutnya.

Label = ekspresi atau fungsi

Contohnya:

```
Exten => 123, 1, GotoIfTime(09.00-17.59, mon-fri, *,
*?open, s, 1)
```

Artinya, jika terdapat panggilan antara jam 09.00 sampai jam 17.59, hari senin sampai jumat, maka panggilan akan dikirim ke extension s pada *context* yang

bernama open. Jika panggilan diluar waktu yang telah ditetapkan, maka akan dikirim ke extension berikutnya (1).

2.3.5. Voicemail

Voicemail merupakan salah satu fitur telepon yang populer. Sistem *voicemail* dalam asterisk bersifat fleksibel. Konfigurasi *voicemail* terdapat dalam file konfigurasi bernama *voicemail.conf*.

Untuk memanfaatkan fitur ini, terlebih dulu membuat *mailbox*. Sintaksnya adalah:

Mailbox => password, nama[,email[,pager_cellphone[,options]]]

Contoh:

500 => 12345, lia

Dimana:

500 adalah nomor *mailbox*

12345 adalah password

lia adalah username

Untuk email, pager_cellphone bersifat opsional

2.4. Session Initiation Protocol

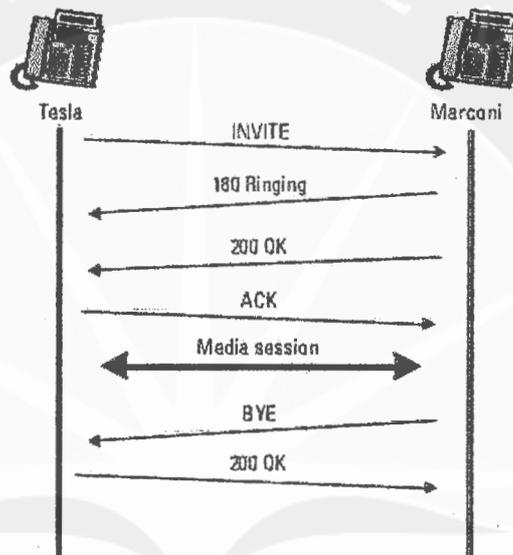
SIP adalah sebuah protokol kontrol (*signaling*) lapisan aplikasi untuk menciptakan, memodifikasi, dan mengakhiri sesi. Fungsi utama SIP adalah memungkinkan dua *end points* membentuk sesi media diantara keduanya. Fungsi ini dapat rinci lebih lanjut sebagai berikut:

1. Letak dari sebuah *end point*
2. Mengontak sebuah *end point* utk membentuk hubungan

3. Bertukar media informasi utk memungkinkan pembentukan hubungan
4. Memodifikasi sesi media yg sudah ada
5. Memumutuskan sesi media yg ada

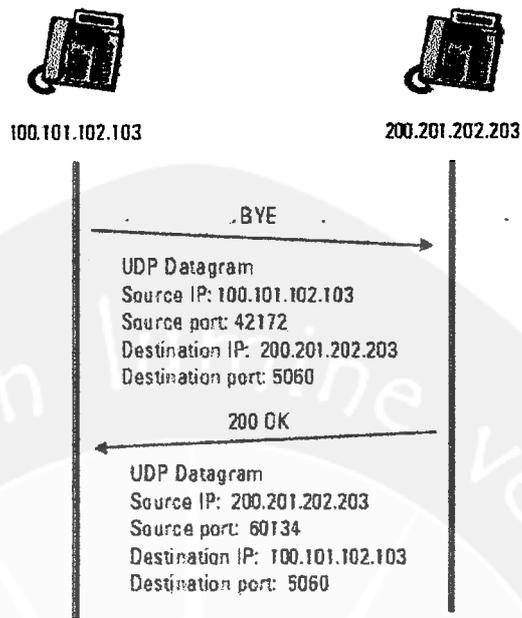
SIP adalah signalling protocol, bukan media transfer protocol, sehingga SIP tidak membawa paket data voice atau video. Dalam implementasi VoIP berbasis protocol SIP, RTP (*Real Time Protocol*) digunakan sebagai media transfer protocol.

Aliran panggilan dasarnya digambarkan pada gambar 2.6 berikut.

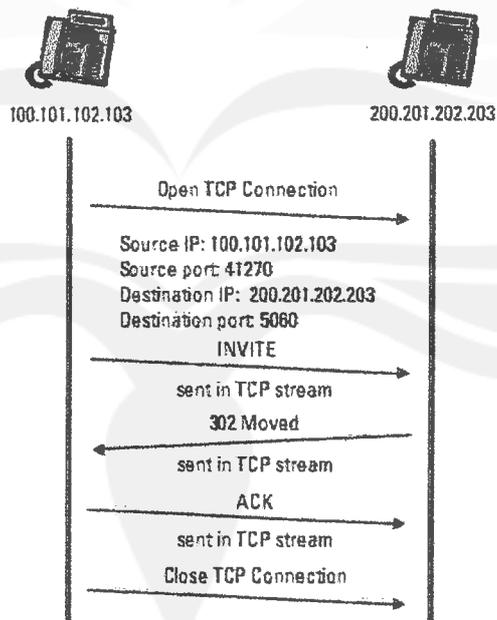


Gambar 2.6. Aliran Panggilan (B. Johnston, 2004)

SIP menggunakan UDP, TCP, dan TLS (*Transport Lapisan Security*) untuk mentransfer pesan. UDP digunakan dalam request dan response, sedangkan TCP digunakan ketika SIP menggunakan pesan INVITE. TLS digunakan untuk meningkatkan keamanan. Fungsi UDP digambarkan pada gambar 2.7 dan TCP digambarkan pada gambar 2.8.



Gambar 2.7. Alur UDP (B. Johnston, 2004)



Gambar 2.8. Alur TCP (B. Johnston, 2004)

2.4.1. Komponen SIP

Komponen yang menggunakan SIP disebut User Agent. User Agent terdiri dari 2 komponen utama yaitu:

a. *User Agent Client (UAC)*

Merupakan komponen yang memulai sesi komunikasi.

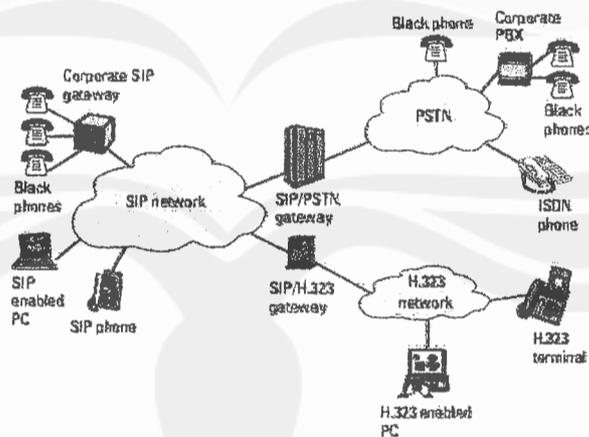
b. *User Agent Server (UAS)*

Merupakan komponen yang menerima atau menanggapi sesi komunikasi.

Baik UAC maupun UAS dapat menutup sesi. *User agent* dapat berupa software (softphone) ataupun hardware (IP-phone).

SIP Gateway

Gateway SIP adalah aplikasi yang berfungsi sebagai perantara antara jaringan yang menggunakan SIP dengan jaringan yang menggunakan protokol signaling jenis lain. *Gateway SIP* digambarkan pada gambar 2.9.



Gambar 2.9. *Gateway SIP* (B. Johnston, 2004)

SIP Server

Server SIP adalah aplikasi yang menerima permintaan (request) dan memberikan tanggapan (response).

Server SIP terdiri dari:

1. Proxy server

Proxy server menerima request dari user agent atau dari proxy server lainnya lalu meneruskannya atau memberi tanggapan (response). Request dapat dilayani sendiri atau diteruskan ke proxy server lain. Sebuah proxy server berbeda dari user agent atau *gateway* dalam tiga hal, yaitu:

- a. Proxy server tidak membangkitkan request, hanya meneruskan saja
- b. Proxy server tidak memiliki kemampuan media
- c. Proxy tidak memperdulikan isi dari pesan, yang diperhatikan hanya *headernya* saja

2. Redirect server

Merupakan komponen yang menerima *request message* dari user agent, memetakan alamat SIP user agent atau proxy server tujuan kemudian menyampaikan hasil pemetaan kembali pada user agent pengirim (UAC). Redirect server memberi tanggapan ke user agent dan menggunakan database untuk mencari user.

3. Registrar server

Digunakan untuk memberi tanggapan terhadap permintaan SIP REGISTER yang berasal dari sebuah UA. Ini digunakan untuk mencegah terjadinya pembajakan terdapat panggilan yang masuk. Registration server dapat menambahkan fungsi otentikasi user untuk validasi.

2.4.2. Request dan Response

Dalam melakukan request dan response, SIP menggunakan SIP URI dan tags.

SIP URI digunakan dalam *header* seperti To, From, dan Contact. SIP URI ditulis dalam format:

```
sip:username@host:port;parameters
```

Contohnya:

```
sip:han2@192.168.1.1:5060
```

Tags adalah nomor acak yang dienkrpsi dengan panjang 32-bit yang ditambahkan pada *header* To dan From untuk memberi identitas unik terhadap sebuah sesi dialog.

SIP Request

SIP Request atau methode adalah perintah tertentu yang digunakan oleh protokol SIP dalam meminta user agent atau server untuk melakukan aksi tertentu. Methode ini meliputi INVITE, REGISTER, BYE, ACK, CANCEL, OPTIONS, dan INFO.

a. INVITE

Digunakan untuk membentuk hubungan antara dua user agent. Dengan kata lain, INVITE akan mengundang user agent lain untuk bergabung dalam sesi komunikasi.

b. REGISTER

Digunakan oleh user agent untuk registrasi di registrar server.

c. BYE

Digunakan untuk mengakhiri sesi komunikasi.

d. ACK

Digunakan untuk memberi tanggapan akhir dari permintaan INVITE.

e. CANCEL

Digunakan untuk membatalkan usaha pencarian atau sebuah panggilan yang sedang ditangguhkan, misalnya untuk membatalkan INVITE.

f. OPTIONS

Digunakan untuk meminta informasi tentang kemampuan server.

g. INFO

Digunakan untuk membawa pesan informasi lainnya, seperti informasi inline DTMF.

SIP Response

SIP Response terbagi dalam beberapa kelas, seperti yang ditunjukkan pada tabel berikut.

Tabel 2.2. Kelas dalam SIP Response (B. Johnston, 2004)

Class	Fungsi
1xx	Pemberi informasi
2xx	Penanda bahwa request berhasil
3xx	Sebagai <i>redirection</i>
4xx	Menunjukkan adanya error di <i>client</i>
5xx	Menunjukkan bahwa request gagal karena terdapat error pada server
6xx	Menberi tahu bahwa request gagal dan request yang gagal tersebut tidak boleh dikirimkan lagi ke server

2.4.3. Header SIP

Header SIP ditulis dalam format text. Aturan penulisan header SIP adalah nama_header:field, dimana:

- a. Nama_header bersifat *case-sensitive*
- b. Field adalah informasi spesifik dan bersifat *case-sensitive*

Dari cara pengantaran, header fields diklasifikasikan sebagai:

- a. end-to-end
tidak bisa dimodifikasi oleh proxy
- b. hop-by-hop
bisa dimodifikasi oleh proxy

Dari informasi yang dibawa, header field diklasifikasikan sebagai:

- a. Request dan Response header fields
Adalah sekelompok header fields yang muncul di request maupun di response.
- b. Request header fields
Header fields yang muncul di request saja. Contohnya Accept, Accept-Contact, Accept-Encoding, dan lainnya.
- c. Response header fields
Header fields yang muncul di response saja. Contohnya Error-Info, Authentication-Info, dan lain sebagainya.
- d. Message body header fields
Bagian header yang membawa informasi tentang message body. Contohnya Allow, Content-Encoding, dan lainnya.

2.4.4. *Session Description Protocol*

Awalnya dirancang untuk memberi deskripsi pembentukan sesi multicast via Internet. Tetapi sekarang digunakan oleh SIP dlm mendeskripsi pembentukan sesi multimedia secara umum. Ada juga informasi tambahan yang dibawa oleh SDP, seperti waktu mulai dan selesainya, serta informasi kontak tentang sesi. Informasi utama yang dibawa SDP adalah:

1. IP address
2. Nomor port
3. Jenis media (audio, video, dan lainnya)
4. Jenis encoding yang digunakan (PCM alaw, PCM ulaw, MPEG, dan lainnya)

SIP menggunakan koneksi, media dan atribut untuk membentuk sebuah sesi. Karena jenis media dan CODEC yang digunakan adalah bagian dari negosiasi pembentukan sebuah sesi, maka SIP dapat menggunakan SDP. Contoh penggunaan SDP didalam SIP adalah ketika penelpon mengirim pesan INVITE. Pada tahap ini terjadi negosiasi tentang jenis media yang akan digunakan.

Contoh: `m = audio 13792 RTP/AVP 0 8 101`

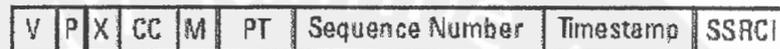
Artinya, media yang akan digunakan adalah audio, port 13792, protokol RTP, jenis CODEC yang dapat digunakan 0 (PCM u-law) atau 8 (PCM a-law) atau 101 (telepon).

2.4.5. *Real-Time Transport Protocol (RTP)*

RTP dikembangkan utk memungkinkan pengiriman secara real-time paket suara, video, dan informasi lainnya di atas porotokol IP. RTP tdk menyediakan quality of service (QoS) dan diperlakukan sama seperti paket lainnya di dlm jaringan IP. Tetapi RTP memiliki kemampuan mendeteksi kelainan dlm pengiriman paket, seperti:

1. Packet loss
2. Variable transport *delay*
3. *Jitter*
4. Routing yang tidak simetris (menyebabkan paket tiba di tujuan dalam keadaan tidak teratur)

RTP memiliki *header* seperti yang digambarkan pada gambar 2.10.



Gambar 2.10. *Header* RTP (B. Johnston, 2004)

Dimana:

- a. *Version* (V): versi RTP, saat ini versi 2
- b. *Padding* (P): Bila bit-nya diset, maka ada penambahan octet pada akhir paket untuk membuat panjang paket tetap.
- c. *Extension* (X): didefinisikan oleh jenis muatan (*payload*) tertentu.
- d. *CSRC count* (CC): hanya dipakai oleh *mixer* yang memerlukan beberapa aliran RTP dan menghasilkan satu aliran RTP.
- e. *Marker* (M): digunakan oleh video untuk menyatakan tanda pada frame baru.
- f. *Payload Type* (PT): mendefinisikan jenis CODEC yang digunakan.
- g. *Sequence Number*: 16-bit, digunakan untuk memberi nomor urut pada paket dan mendeteksi paket yang hilang.
- h. *Timestamp*: 32-bit, mengindikasikan waktu relative ketika paket dicuplik. Ini memungkinkan penerima menghilangkan *jitter* dan memainkan kembali dengan interval yang tepat.
- i. *Synchronization Source Identifier* (SSRCI): Digunakan untuk memberi identifikasi pada paket yang berasal dari pengirim. Pada awal sesi, angka ini dipilih secara acak oleh partisipan (peserta).

2.4.6. Real-Time Control Protocol

Digunakan oleh peserta RTP (piranti yang menggunakan RTP) untuk bertukar informasi tentang kualitas dan laporan statistik paket suara atau video.

2.4.7. Audio/Video Profiles

Penggunaan profile memungkinkan RTP menjadi protokol yang bersifat umum yang mampu membawa berbagai paket multimedia. Profile tentang RTP meliputi:

- a. UDP digunakan sebagai protokol pembawa muatan
- b. RTP menggunakan port genap, RTCP menggunakan port ganjil (satu tingkat di atas port RTP)
- c. Tidak perlu menggunakan *header* tambahan

2.5. Quality of Service (QoS)

QoS adalah kemampuan jaringan untuk memperlakukan trafik secara berbeda menurut kelompoknya. QoS diperlukan karena:

1. Paket data dan suara diperlakukan sama di dalam jaringan IP. Padahal, paket suara bersifat sensitif terhadap waktu, sedangkan paket data tidak
2. Peralatan jaringan (seperti: router, switch, computer, dan lainnya) tdk bisa membedakan jenis trafik
3. Perilaku trafik tidak bisa diprediksi (paket hilang, tundaan, lonjakan, dan lain-lain)
4. Perilaku protokol TCP dan UDP yang berbeda. Aplikasi yang menggunakan TCP memberi tanggapan bila terjadi paket hilang, sedangkan aplikasi suara yang menggunakan UDP hanya bisa diam saja.

2.5.1. Memanipulasi Karakteristik Jaringan dengan QoS

Dengan menggunakan QoS, karakteristik jaringan dapat dimanipulasi untuk mencapai hasil yang dituntut oleh aplikasi VoIP. Ada 4 karakteristik jaringan yang penting, yaitu *bandwidth*, *delay*, *jitter*, dan *packet loss*.

2.5.2. Model QoS

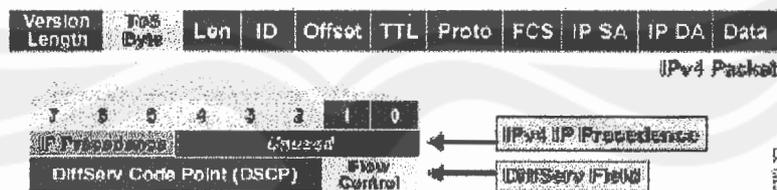
Terdapat tiga model QoS, yaitu:

1. *Best Effort* : tidak ada QoS
2. *Integrated Service*: aplikasi mengirim isyarat agar menggunakan QoS tertentu
3. *Differentiated Service*: trafik jaringan menggunakan kelas-kelas dan menerapkan QoS tertentu. Digunakan dalam VoIP.

2.5.3. Differentiated Services

Differentiated service menggunakan DSCP (*Differentiated Service Code Point*) untuk menandai paket-paket sesuai dengan klasifikasinya.

Pada *header IP*, *ToS (Type of Service)* memuat 8 bit, dimana 6 bit pertama adalah *field Differentiated Service* (disingkat dengan DiffServ) dan 2 bit selanjutnya digunakan untuk *flow control*, seperti yang ditunjukkan dalam gambar 2.11.



Gambar 2.11. Delapan bit ToS (Odom, 2005)

Enam bit DiffServ, defaultnya jika seluruhnya bernilai 0. Enam bit ini menentukan prioritas trafik jaringan. Prioritas tertinggi jika bit DiffServ bernilai 46

(nilai-nilai bit DiffServ seperti gambar 2.12, sedangkan tabel nilai marking dapat dilihat pada tabel 2.3.).

Field and Value (Decimal)	Binary Value	Name	Defined by This RFC
DSCP 0	000000	best effort or default	2475
DSCP 8	001000	CS1	2475
DSCP 16	010000	CS2	2475
DSCP 24	011000	CS3	2475
DSCP 32	100000	CS4	2475
DSCP 40	101000	CS5	2475
DSCP 48	110000	CS6	2475
DSCP 56	111000	CS7	2475
DSCP 10	001010	AF11	2597
DSCP 12	001100	AF12	2597
DSCP 14	001110	AF13	2597
DSCP 18	010010	AF21	2597
DSCP 20	010100	AF22	2597
DSCP 22	010110	AF23	2597
DSCP 26	011010	AF31	2597
DSCP 28	011100	AF32	2597
DSCP 30	011110	AF33	2597
DSCP 34	100010	AF41	2597
DSCP 36	100100	AF42	2597
DSCP 38	100110	AF43	2597
DSCP 46	101110	EF	2598

Gambar 2.12. Tabel Nilai Bit DiffServ (Odom, 2005)

Tabel 2.3. Nilai Marking yang Direkomendasikan oleh RFC (Odom, 2005)

Tipe Trafik	DSCP
Voice Payload	EF
Video Payload	AF41
Voice/Video Signaling	AF31
Best Effort	BE

2.5.4. Klasifikasi dan *Marking*

Klasifikasi adalah proses identifikasi dan penggolongan trafik ke dalam kelas-kelas yang berbeda. Klasifikasi dilakukan supaya QoS dapat diterapkan. Klasifikasi bisa dilakukan pada lapisan data link atau pada network lapisan, tergantung pada topologi jaringan.

Marking adalah proses perubahan bit-bit tertentu di dalam frame atau paket *header* agar dapat dikenali dan diperlakukan oleh perangkat jaringan sesuai dengan maksudnya.

Prosedur klasifikasi dan penandaan adalah sebagai berikut (langkah-langkah penerapannya dapat dilihat pada lampiran F nomor 1):

1. Mengklasifikasikan paket-paket ke dalam kelas-kelas layanan tertentu.

```
# class-map match-all dscp-ef
# match ip dscp ef
```

Sintaks tersebut akan membuat sebuah kelas yang nilai bit DiffServ-nya diset EF (DSCP 46).

2. Menandai setiap kelas layanan.

```
# policy-map lab-skripsi
# class voip-rtp
# set ip dscp ef
```

Sintaks diatas akan menandai kelas layanan. Layanan kelas voip-rtp akan diset nilai bit DiffServ-nya EF (DSCP 46).

2.5.5. Kelas Layanan QoS (*QoS Service Class*)

Sebuah kelas layanan QoS adalah paket-paket yang secara logic dimasukkan dalam satu kelompok dan diperlakukan sama sesuai dengan QoS yang diterapkan. Sebuah kelas layanan QoS bisa berupa:

- a. Pengguna tunggal: MAC address, IP address, dan lain-lain
- b. Departemen: subnet, interface, dan lain-lain
- c. Aplikasi: nomor port, URL, dan lain-lain

2.5.6. Manajemen Antrian

Congestion management atau sering disebut juga *queuing* adalah cara router mengantri, menjadwalkan dan mengirimkan paket-paket yang diterimanya. Paket yang diterima disimpan di memori sementara router memproses paket-paket lainnya. Ketika router selesai, router siap mengirimkan paket-paket yang ada di memori.

Model antrian yang paling dasar adalah FIFO (*First In First Out*). Paket dikirim router sesuai urutan penerimaannya. Tidak ada prioritas bagi trafik suara maupun data.

Ada banyak cara untuk menerapkan antrian. Tetapi router Cisco menggunakan LLQ (*Low Latency Queuing*) untuk mengatasi masalah *delay* dan *jitter* pada trafik suara (seperti pada Lampiran F nomor 2 yang menerapkan LLQ untuk manajemen antrian).

2.5.7. Efisiensi Saluran

Kompresi akan memampatkan paket sehingga ukurannya bisa menjadi lebih kecil. Setelah mengirim untaian paket-paket kecil yang telah dikompresi, algoritma kompresi di ujung lain akan membalikan proses dengan mengembalikan paket-paket menjadi bentuk semula.

Kompresi dapat dilakukan baik pada *header* maupun muatan (*payload*) yang dibawa oleh *frame*. Kompresi muatan dilakukan pada data dan *header* paket. Sedangkan kompresi *header* hanya pada *header* paket.

Pada paket suara, umumnya kompresi akan terasa lebih efisien jika dilakukan pada *header* paket sebab ukuran muatannya lebih kecil dari ukuran *headernya*. Kompresi *header* digunakan ketika *bandwidth* rendah dan setelah dikompres end-to-end *delay* tidak melebihi 200ms. Jika *delaynya* melebihi 200ms, maka *bandwidth* harus dinaikan atau *codecnya* diganti.

