BAB II
DASAR TEORI

2.1 Video *Streaming*

Video *streaming* merupakan salah satu teknologi telekomunikasi yang bersifat *real time* serta dapat menyalurkan informasi berupa audio maupun video dengan menggunakan jaringan *Internet Protocol* (IP). Dengan teknologi video *streaming* ini, *user* tidak perlu menunggu hingga *file* selesai di-*download* secara keseluruhan untuk memainkannya. Sebaliknya, *user* dapat memainkan media dengan menunggu beberapa detik saja. Akan tetapi, ada beberapa permasalahan yang mempengaruhi performansi dari video *streaming* diantaranya adalah video berhenti berjalan atau bergerak lambat saat kita sedang menonton video sehingga kita harus menunggu beberapa waktu hingga video kembali berjalan. Selain itu, kualitas video yang dihasilkan juga seringkali buruk dimana video *streaming* yang kita amati kurang jernih dan cenderung tidak jelas.

Salah satu hal yang sangat mempengaruhi dalam performansi video *streaming* tersebut adalah *bandwidth*. *Bandwidth* adalah jumlah data yang dapat melalui suatu jaringan, atau bagian dari jaringan, untuk setiap waktu tertentu (Behrouz A. Forouzan, 2000). *Bandwidth* pada jaringan bersifat terbagi, terbatas, dan berubah terhadap waktu. Semakin besar *bandwidth* jaringan, maka kualitas video *streaming* yang dihasilkan juga akan semakin baik (Michael Gough, 2006). Selain itu, agar layanan aplikasi video *streaming* ini dapat berjalan dengan baik, lebih efisien, dan cepat dalam penyampaian informasinya maka perlu diterapkan pada suatu teknologi jaringan yang mempunyai kecepatan akses data yang tinggi untuk menghasilkan delay yang seminimal mungkin. Dengan *delay* yang seminimal mungkin diharapkan *user* dapat menikmati video *streaming* tanpa harus menunggu lama.

Awalnya telekomunikasi dua arah langsung yang semula hanya berupa audio saja, kini telah berkembang menjadi audio sekaligus video yang tercakup dalam dua arah percakapan jarak jauh. *Teleconference* dan video *streaming* internet menjadi contoh generasi terbaru sistem komunikasi global.

Dalam dunia internet, *streaming* lebih mengacu kepada teknologi yang mampu mengkompresi atau menyusutkan ukuran *file* audio dan video agar mudah dikirimkan melalui jaringan internet. Pengiriman *file* audio dan video tersebut dilakukan secara

”*stream*”, alias terus-menerus. Dari sudut pandang prosesnya, *streaming* berarti sebuah teknologi pengiriman *file* dari *server* ke *client* melalui jaringan *packet-based* semisal internet. Sedangkan dari sudut pandang *user*, *streaming* adalah teknologi yang memungkinkan suatu *file* dapat segera dijalankan tanpa harus menunggu selesai di*download* seluruhnya dan terus mengalir tanpa ada interupsi.

Tabel 2.1 Jenis-jenis Video *Streaming*

No.	Jenis Video	Penjelasan
1.	*Live streaming*	Terjadi ketika setiap *user* melihat video *streaming* yang sama secara bersamaan
2.	*On-demand streaming*	Terjadi saat pengguna dapat meminta *prerecorded* video dialirkan kepada pengguna ketika pengguna ingin melihat video tersebut. Dalam hal ini, setiap user dapat memilih kapan saat untuk memulai dan mengakhiri *streaming* video.
3.	*Download and play*	Menggunakan memori atau hard disk di dalam perangkat *user* untuk menerima file audio/ video yang kemudian dapat ditampilkan (dimainkan) oleh perangkat *user*. Salah satu keuntungan utama dari *download and play* adalah bahwa file audio/video dapat dikirimkan dengan HTTP, sehingga kemungkinan dapat melewati firewall lebih besar. Beberapa *server* tidak setuju dengan *download and play*, karena hasil salinan video tersimpan di perangkat penyimpanan setiap user sebelum video dilihat/ditampilkan.
4.	*Progressive download and play*	*Progressive download and play* adalah salah satu metode yang paling populer untuk pengiriman file video/audio melalui Internet, dan digunakan oleh situs-situs seperti *YouTube*.

		Teknologi ini pada dasarnya sama seperti *download and play*, kecuali *file* video/audio dipecah menjadi *file-file* kecil yang dikirimkan sebagai video *playback. Progressive download and play* memiliki keuntungan, yaitu dapat melewati *firewall* dengan mudah ketika mentransmisikan kecepatan dan kebutuhan *storage* yang lebih kecil.

(Sumber: Wes Simpson, 2008)

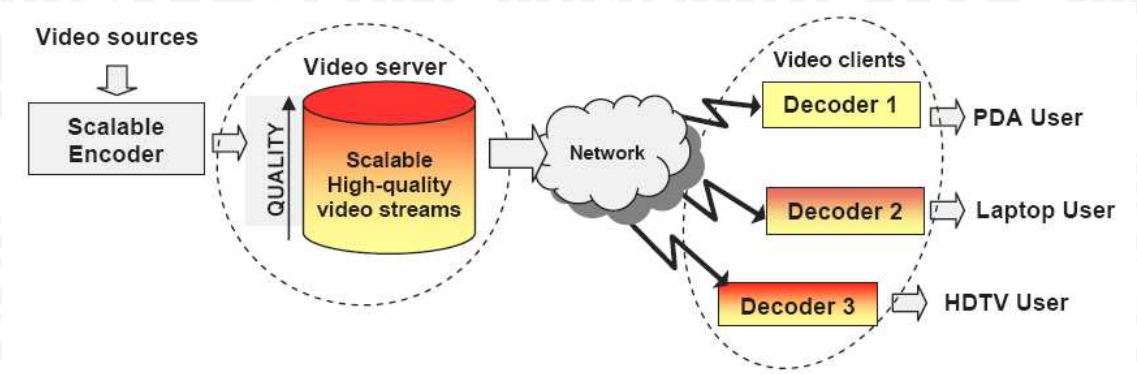
File yang dapat ditransmisikan oleh video *streaming* adalah *file* audio, video, *image*, *text*, data 3D, *software*, dan sebagainya. Tetapi *streaming* lebih mengacu kepada *time based* media, khususnya audio dan video, yang harus dapat dinikmati sesegera mungkin dan berdasarkan pada pewaktuan yang tepat, karena untuk dapat menikmati lagu atau film, haruslah dimainkan secara beruruan dari awal hingga akhir (*sequential*) tanpa terputus-putus (*uninterrupted*). Ada beberapa jenis video *streaming* seperti ditunjukkan pada Tabel 2.1.

2.2 CODEC (Coder/DECoder)

Pada aplikasi multimedia baik untuk *voice*, video, maupun data, besarnya *bandwidth* tergantung dari CODEC yang digunakan. CODEC terdiri dari *Encoder* yang digunakan untuk melakukan *encoding* pada *file* audio/video sehingga memiliki ukuran yang lebih kecil dengan memperkecil *pixel* dan *frame* serta *Decoder* yang digunakan untuk membaca *file* yang telah di-*encode* dan memainkannya di sisi *client*. CODEC sangat diperlukan dalam proses *streaming* video.

Pada proses *streaming* video, video dari *source* akan di-*capture* dan disimpan pada sebuah *buffer* yang berada pada memori komputer (bukan media penyimpanan seperti *harddisk*) dan kemudian di-*encode* sesuai dengan format yang diinginkan. Dalam proses *encode* ini, *server* dapat mengkompresi data sehingga ukurannya tidak terlalu besar (bersifat *optional*). Hal tersebut dilakukan karena keterbatasan *bandwitdh* jaringan. Setelah di-*encode*, data akan di-*stream* ke *user* yang lain. *User* akan melakukan *decode*

data dan menampilkan hasilnya ke layar *user* untuk menikmati video *streaming*. Proses *streaming* video dapat ditunjukkan pada dibawah ini.



Gambar 2.1 Proses *streaming* video melalui jaringan
(**Sumber**: Ashraf M.A. Ahmad and Ismail Khalil Ibrahim, 2009: 94)

Terdapat dua jenis CODEC yang dapat digunakan dalam video *streaming*, yaitu (Andreas Handojo, et al., 2009):

a. *Lossy* CODEC: CODEC ini akan mengurangi kualitas data dengan mengurangi ukuran data (kompresi). Pada umumnya, CODEC ini digunakan untuk menyimpan data pada media penyimpanan yang berukuran terbatas seperti CD-ROM dan DVD. *Lossy* CODEC ini biasanya digunakan untuk *streaming*, karena *bandwdith* jaringan yang terbatas. Contoh: Windows Media Video, H.264.
b. *Lossless* CODEC: Pada *lossless* CODEC ini, kualitas data yang dihasilkan tidak akan berkurang. Tetapi, ukuran data yang dihasilkan oleh *lossless* CODEC ini akan lebih besar dibandingkan dengan *lossy codec*. Pada umumnya, *lossless* CODEC ini digunakan pada video yang masih memerlukan *editing*, karena dalam proses *editing* dilakukan *encode-decode* berulang kali, sehingga jika menggunakan *lossy* CODEC, kualitas video akan jauh menurun dibandingkan dengan video aslinya. Contoh: CorePNG, huffyuv, Apple Lossless Audio Codec.

Ada beberapa jenis audio dan video CODEC menurut ITU.T, seperti ditunjukkan pada tabel dibawah ini.

Tabel 2.2 Jenis CODEC

Audio CODEC	**Bit Rate (kbps)**	***Maximum Payload* (byte)**	***Delay* CODEC (ms)**

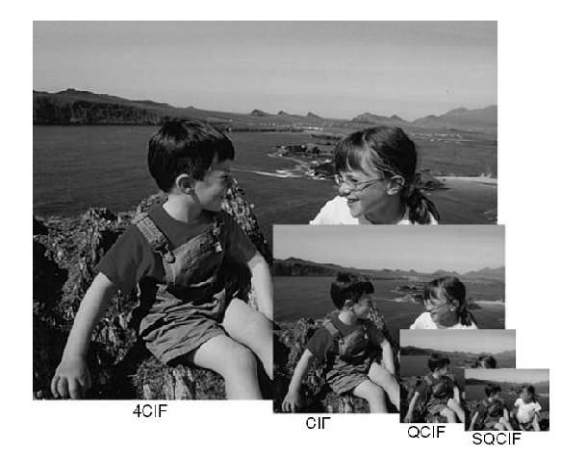
AMR-WB	6,6 – 23,85	35	10 – 20
AMR-WB+	5,2 – 48	46	20 – 40
HE-AAC v2	128 – 320	80	40 – 80
Video CODEC	**Bit Rate (kbps)**	***Maximum Payload* (byte)**	***Delay* CODEC (ms)**
H.264/AVC	64 – 384	254	150 – 300

(**Sumber**: RFC 4352 and RFC 3984 RTP *Payload* Format for H.264 Video, 2005)

2.2.1 Kualitas Video

Video dapat juga disebut sebagai gambar-gambar yang bergerak. Dalam video tampilkan sejumlah gambar atau frame dengan kecepatan tertentu yang disebut dengan istilah *frame rate*, yang dihitung dalam skala *frame per second* (fps). Seperti jenis data yang lain, data video juga dapat disimpan, diedit, ataupun dikirim melalui jaringan.

Ada beberapa format gambar yang digunakan dalam aplikasi video *streaming* diantaranya adalah format CIF, QCIF, SQCIF dan 4CIF. Perbandingan untuk setiap format gambar dapat dilihat pada Gambar 2.



Gambar 2.2 Format Gambar

(**Sumber**: Iain E. G. Richardson, 2003: 20)

Pilihan resolusi frame tergantung pada aplikasi dan kapasitas yang tersedia atau kapasitas transmisi. Sebagai contoh, 4CIF sesuai untuk digunakan pada televisi standar dan DVD-video. Format CIF dan QCIF sangat populer digunakan pada aplikasi video conferencing. Sedangkan format QCIF atau SQCIF sesuai untuk digunakan pada aplikasi *mobile multimedia* dimana resolusi layar dan bit rate format ini terbatas.

Kualitas video yang baik dapat dinilai dari tiga elemen utama, yaitu:

a. Frame Rate, jumlah gambar yang ditampilkan per detik pada video. Minimum frame yang dibutuhkan untuk mendapatkan ilusi gambar yang bergerak adalah sekitar 15 fps.
b. Color Depth, Jumlah bit pada setiap pixel yang menunjukkan informasi warna. Misalnya: 24 bit menunjukkan 16.7 juta warna, 16 bit sekitar 65,535 warna, atau 8 bit hanya 256 warna.
c. Frame Resolution, biasanya ditunjukkan dengan width dan height pada pixel. Misalnya: full screen PC mempunyai frame resolution sebesar 640x480.

Jika kita menggunakan kecepatan pengiriman kecepatan pengiriman frame per second (fps) video yang rendah, akan memakan *bandwidth* yang lebih rendah dibandingkan frame per second (fps) yang tinggi. Video yang cukup baik biasanya dikirim dengan kecepatan frame per second (fps) sekitar 30 fps

2.3 Internet Protocol Version 6 (Ipv6)

Usaha untuk membangun IPv6 sebagai protokol pengganti IPv4 mulai dilaksanakan oleh *Internet Engineering Task Force* (IETF) pada awal tahun 1990-an. IPv6 ini dirancang untuk memecahkan masalah keterbatasan ruang alamat alamat IPv4. Rekomendasi mengenai IPv6 yang terdapat dalam RFC 1752, "*The Recommendation for the IP Next Generation Protocol*" ditetapkan menjadi IETF *Draft Standard* pada 10 Agustus 1998. IP versi 6 (IPv6) adalah protokol internet versi baru yang didesain sebagai pengganti dari internet protokol versi 4 (IPv4).

Perubahan dari IPv4 ke IPv6 pada dasarnya terjadi karena beberapa hal yang dikelompokkan dalam kategori berikut (Sritrusta Sukaridhoto, 2008)

- Kapasitas Perluasan Alamat

  IPv6 meningkatkan ukuran dan jumlah alamat yang mampu didukung oleh IPv4 dari 32 bit menjadi 128 bit. Peningkatan kapasitas alamat ini digunakan untuk mendukung peningkatan hirarki atau kelompok pengalamatan, peningkatan jumlah atau kapasitas alamat yang dapat dialokasikan dan diberikan pada *node*, dan mempermudah konfigurasi alamat pada *node* sehingga dapat dilakukan secara otomatis. Peningkatan skalabilitas juga dilakukan pada *routing multicast* dengan meningkatkan cakupan dan jumlah pada alamat *multicast*. IPv6 ini selain

meningkatkan jumlah kapasitas alamat yang dapat dialokasikan pada *node*, juga mengenalkan jenis atau tipe alamat baru, yaitu alamat *anycast*. Tipe alamat *anycast* ini didefinisikan dan digunakan untuk mengirimkan paket ke salah satu dari kumpulan *node*.

- Penyederhanaan Format *Header*

  Beberapa kolom pada *header* IPv4 telah dihilangkan atau dapat dibuat sebagai *header* pilihan. Hal ini digunakan untuk mengurangi biaya pemrosesan hal-hal yang umum pada penanganan paket IPv6 dan membatasi biaya *bandwidth* pada *header* IPv6. Dengan demikian, pemrosesan *header* pada paket IPv6 dapat dilakukan secara efisiien.

- Peningkatan Dukungan untuk *Header* Pilihan dan *Header* Tambahan (*option and extension header*)

  Perubahan yang terjadi pada *header-header* IP yaitu dengan adanya pengkodean *header opions* (pilihan) pada IP dimaksudkan agar lebih efisien dalam penerusan paket (*packet forwarding*), agar tidak terlalu ketat dalam pembatasan panjang *header* pilihan yang terdapat dalam paket IPv6 dan sangat fleksibel dimungkinkan untuk mengenalkan *header* pilihan baru pada masa akan datang.

- Kemampuan pelabelan aliran paket

  Kemampuan atau fitur baru yang ditambahkan pada IPv6 ini adalah memungkinkan pelabelan paket atau pengklasifikasian paket yang merupakan milik dari aliran trafik tertentu dan aliran trafik ini meminta penanganan spesial, seperti kualitas mutu layanan tertentu (QoS) atau layanan *real-time*.

- Autentifikasi dan Kemampuan Privasi

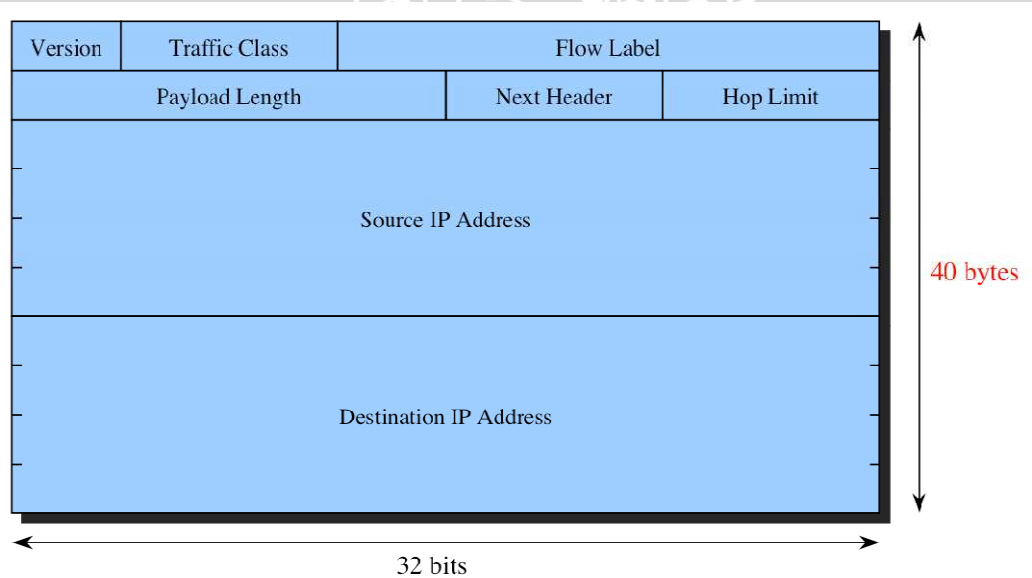
  Kemampuan tambahan untuk mendukung autentifikasi, integritas data dan data penting juga dispesifikasian dengan alamat IPv6.

2.3.1 Format Header Ipv6

Format *header* alamat IPv6 menyederhanakan format *header* pada alamat IPv4. Setiap paket IPv6 membawa data yang terdiri dari (Silvia Hagen, 2003):

- *Version* (4 bit), berisi nomor versi protokol internet, yaitu 6.

- *Traffic Class* (8 bit), berisi kolom kelas atau klasifikasi trafik. Tersedia untuk digunakan oleh permulaan *node* atau *router forwarding* untuk_mengidentifikasian dan membedakan antara kelas yang berbeda atau prioritas paket IPv6.
- *Flow Label* (20 bit), berisi label aliran dari trafik. Digunakan oleh *host* untuk melabeli paket-paket yang mana meminta penanganan khusus oleh *router* dalam suatu jaringan.
- *Payload Length* (16 bit), merupakan panjang *payload*/muatan IPv6. *Payload* ini adalah sisa paket setelah *header* IPv6 dalam bentuk oktet (di luar *header* IPv6). Catatan bahwa *header* tambahan/*extension header* yang ada dihitung sebagai bagian dari *payload* termasuk panjang *header* tambahan tersebut.
- *Next header* (8 bit), merupakan identifikasi tipe *header* yang akan ada setelah *header* IPv6. Nilai *heeder* ini menggunakan tipe *header* yang sama dengan yang ada pada protokol IPv4 (RFC 1700).
- *Hop Limit* (8 bit), nilai pada kolom ini akan dikurangi satu jika paket ini melewati *node* yang berfungsi melewatkan/mem*forward* paket (melewati *router*). Paket ini akan dibuang jika batas *hop* ini berubah menjadi nol.
- *Source Address* (128 bit), merupakan alamat IPv6 asal dari paket.
- *Destination Address* (128 bit), merupakan alamat IPv6 tujuan paket.



Gambar 2.3 Format *header* IPv6
(**Sumber**: Rob Blokzijl, 2009)

2.4 *Long Term Evolution* (LTE)

LTE disebut juga sebagai E-UTRA (*Evolved Universal Terrestrial Radio Access*) dan diperkenalkan oleh 3GPP sebagai *release* 8. LTE merupakan teknologi telekomunikasi seluler sebagai evolusi atau pengembangan dari teknologi UMTS/WCDMA/HSPA. LTE menggunakan spesifikasi *air interface* yang baru untuk meningkatkan kecepatan data dibandingkan dengan HSPA. Perbedaan utama antara LTE, WCDMA dan HSPA adalah penggunaan teknologi OFDMA pada sisi *downlink* dan SC-FDMA pada sisi *uplink*-nya. LTE mempunyai kemampuan mengirimkan data dengan kecepatan tinggi mencapai 100 Mbps untuk *downlink* dan 50 Mbps untuk *uplink*-nya. Peningkatan kecepatan ini juga disebabkan oleh kemampuan LTE untuk menggunakan teknologi MIMO (*Multi Input Multi Output*) dan *bandwidth*-nya yang *scalable* mulai dari 1,4 MHz sampai 20 MHz (K. Fazel and S. Kaiser, 2008)

2.4.1 Spesifikasi Teknis LTE

Teknologi radio akses LTE harus dioptimalkan untuk trafik *packet switched* dengan kecepatan data yang tinggi dan *latency* yang rendah. Tabel 2.3 menunjukkan spesifikasi teknis untuk teknologi LTE.

Tabel 2.3 Spesifikasi Teknis LTE

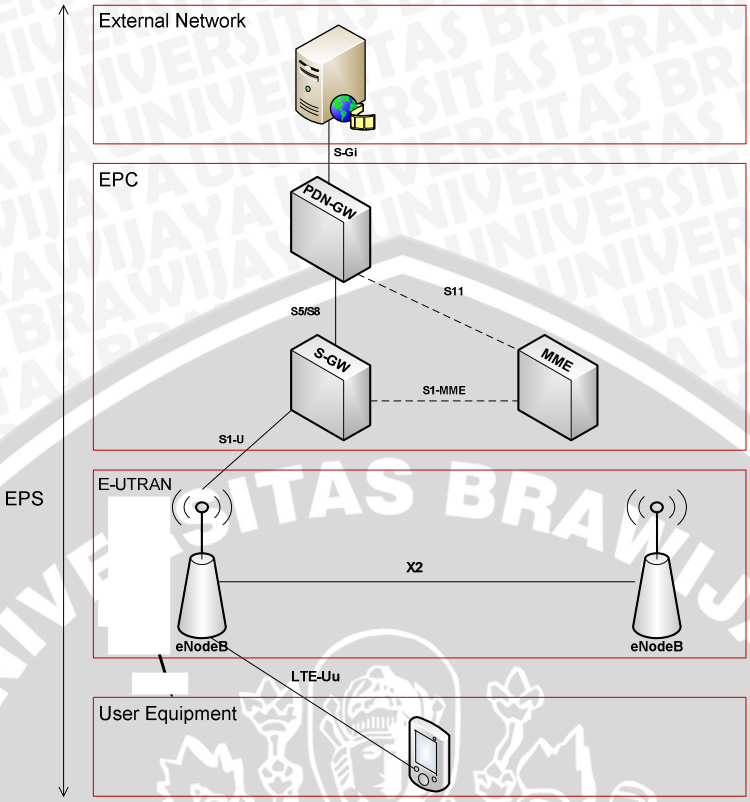
Parameter Spesifikasi Teknis	Jenis/Nilai
Peak data rates	100 Mbit/s *for downlink;* 50 Mbit/s *for uplink*
Average user throughput per MHz than HSPA *Release* 6	3-4 *higher for downlink;*2-3 *higher for uplink*
Mobility	0-15 km/h (optimum);15-120 km/h (*high performance guarranted*); 120-350 km/h (*connection maintained*)
Bandwidth	1.25-20 MHz
Spectrum allocation	*Operation in paired spectrum (FDD) and unpaired spectrum (TDD) should be supported*
Multiple access	OFDMA (*downlink*) SC-FDMA (*uplink*)
MIMO	*Downlink* 2x2, 4x2, 4x4

	Uplink 1x2, 1x4
Modulation	*QPSK, 16-QAM, 64-QAM*
Latency	*5ms user-plane latency at IP layer, for one-way* *100 ms control-plane latency from idle to active state*
User per cell	*At least* 200 *at* 5MHz *bandwidth* *At least* 400 *at bandwidth* > 5MHz

(**Sumber :** K. Fazel dan S. Kaiser, 2008)

2.4.2 *Evolved Packet System* (EPS)

EPS *framework* terdiri dari *Evolved Packet Core* (EPC) dan *Evolved UMTS Terrestrial Radio Access Network* (E-UTRAN) seperti yang ditunjukkan pada Gambar 2.4. EPC berhubungan dengan E-UTRAN dan yang lainnya. EPC berisi *Mobile Management Entity* (MME), *System Architecture Evolution Gateway* (S-GW) dan *Packet Data Network Gateway* (PDN-GW). E-UTRAN hanya berisi *Evolved Universal Terrestrial Radio Access Network Base Stations* (eNodeB) dimana *User Equipment* (UE) berhubungan dengan eNB dan eNB berhubungan dengan EPC dan yang lainnya.



Gambar 2.4. Arsitektur LTE
(**Sumber :** H. Holma dan A. Toskala, 2009)

a. PDN-GW *(Packet Data Network Gateway)*

PDN-GW merupakan jangkar untuk mobilitas antara 3GPP dengan teknologi *non*-3GPP seperti WiMAX, 3GPP2 dan WLAN (*Wireless Local Area Network*) melalui beberapa *interface*. PDN-GW menyediakan konektivitas dengan jaringan paket data eksternal. Selain itu PDN-WG bertanggung jawab untuk mengalokasikan alamat IP UE, pengisian aliran data berdasarkan aturan dari PCRF dan menyaring *downlink* paket IP *user* kedalam *bearer* QoS yang berbeda.

b. S-GW *(Serving Gateway)*

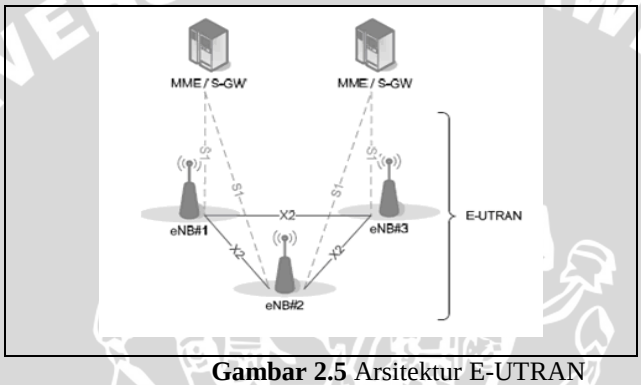
Semua pengguna paket IP dikirimkan melalui S-GW, yang berfungsi sebagai jangkar untuk mobilitas data *bearer* bila UE bergerak di antara eNodeB. S-GW juga bertanggung jawab dalam menetapkan *routing* dan meneruskan paket data *user* dengan *interface* S-1, menangani kompresi *header* IP serta enkripsi data *user*.

c. MME *(Mobility Management Entity)*

MME adalah *node* kontrol yang memproses sinyal antara UE dan CN (*Core Network*)/EPC. Selain itu MME juga berfungsi sebagai autentifikasi dan keamanan serta *mobility management*. Protokol yang berjalan antara UE dan CN dikenal sebagai protokol *Non-Access Stratum* (NAS). (Stefania Sesia, Issam Toufik dan Matthew Baker, 2009 : 25)

2.4.3 E*volved UMTS Terrestrial Radio Access Network* (E-UTRAN)

Jaringan akses LTE, E-UTRAN, hanya terdiri dari eNodeB seperti yang ditunjukkan pada Gambar 2.5.



Gambar 2.5 Arsitektur E-UTRAN

(**Sumber :** Stefania Sesia, Issam Toufik and Matthew Baker,2009)

eNodeB biasanya saling terhubung satu sama lain melalui *interface* yang dikenal sebagai *X2*, dan pada EPC melalui *interface* yang lebih khusus yaitu *S1*, dengan MME melalui *interface* S1-MME dan S-GW melalui *interface* S1-U. Protokol yang bekerja antara eNodeB dan UE disebut protokol *Access Stratum* (AS). (Stefania Sesia, Issam Toufik and Matthew Baker, 2009)

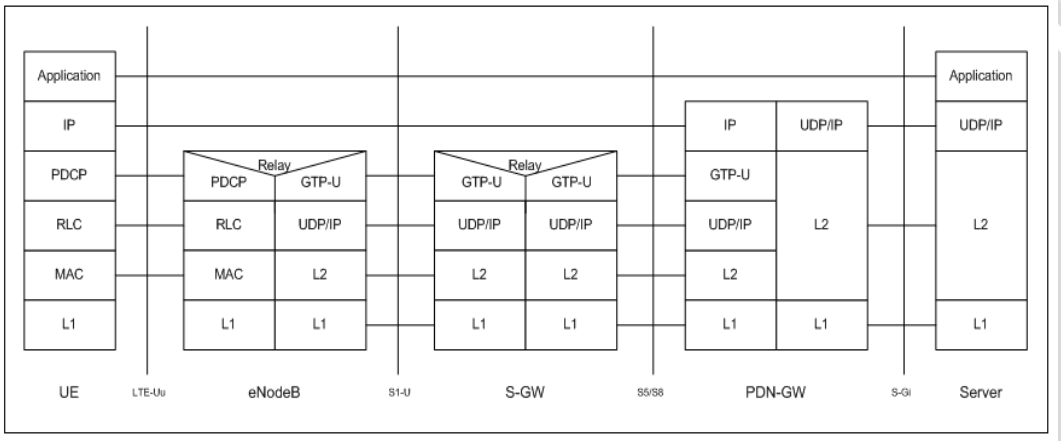
eNB merupakan *entity* yang hanya ada didalam E-UTRAN yang merupakan *interface* dengan UE melalui *interface* LTE-UE. eNB merupakan *hosts* dari layer fisik, MAC, *Radio Link Control* (RLC) dan *Packet Data Control Protocol* (PDCP). eNB bisa mendukung *mode* FDD, TDD atau operasi *dual mode*.

2.5 Arsitektur Protokol Radio E-UTRAN

2.5.1 User Plane

Semua paket IP untuk UE dienkapsulasi dalam sebuah protokol khusus EPC dan di jalur antara P-GW dan eNodeB untuk transmisi ke UE. Protokol *tunneling* yang berbeda digunakan pada *interface* yang berbeda. Sebuah protokol *tunneling* khusus disebut *GPRS Tunnelling Protocol* (GTP) yang digunakan melalui *interface core network*.

Protokol *stack user plane* E-UTRAN ditunjukkan pada Gambar 2.6, yang terdiri dari sublayer *Packet Data Convergence Protocol* (PDCP), *Radio Link Control* (RLC) dan *Medium Access Control* (MAC), yang diterminasi di eNodeB pada sisi jaringan. (Sumber : Stefania Sesia, Issam Toufik and Matthew Baker, 2009)



Gambar 2.6 *User Plane Protocol Stack* E-UTRAN
(**Sumber :** Stefania Sesia, Issam Toufik and Matthew Baker, 2009)

1) *Radio Resource Control* (RRC)

RRC adalah bagian dari *control plane* yang bertanggung jawab untuk protokol PDCP, RLC, MAC, dan PHY layer 1 dan layer 2. Layanan dan fungsi utama dari RRC adalah *admission control*, manajemen *handover*, manajemen *Quality of Servise* (QoS), mengatur dan melaporkan pengukuran stasiun terminal serta mengatur MBMS.

2) *Packet Data Convergence Protocol* (PDCP)

Fungsi utama dari PDCP untuk *user plane* adalah mengkompres *header* IP, pengiriman data dari pengguna dan *chipering*. Pada *control plane*, PDCP bertanggung jawab dalam fungsi pengendalian pengiriman data dan *ciphering*.

3) *Radio Link Control* (RLC)

Segmentasi dan *reassembly* paket data dari layer yang lebih tinggi serta perbaikan kesalahan melalui ARQ adalah fungsi utama dari RLC. Selain itu, *flow control* antara eNodeB dan *mobile terminal* juga ditangani oleh RLC.

4) *Medium Access Control* (MAC)

MAC bertanggung jawab untuk penjadwalan *uplink* dan *downlink*, perbaikan kesalahan melalui *hybrid* ARQ (HARQ), modulasi adaptif dan pemetaan antena.

5) Layer Fisik (PHY)

Fungsi utama dari layer fisik adalah *coding*, modulasi dan transmisi *multiple* antena (MIMO). (K. Fazel dan S. Kaiser, 2008)

2.6 Parameter Performansi Aplikasi Video *Streaming*

2.6.1 Paket Data Aplikasi Video *Streaming*

Pada aplikasi video *streaming*, paket yang ditransmisikan dibedakan atas paket audio dan paket video, dimana tiap paket tersebut mempunyai besar *payload* yang berbeda. Aplikasi video *streaming* menggunakan jenis CODEC H.264/AVC untuk video dengan *bit rate* CODEC antara 64 – 384 kbps dan AMR-WB+ untuk audio dengan *bit rate* CODEC antara 5,2 – 48 kbps. Besar *payload* tiap paket audio dan video dapat dinyatakan dengan persamaan (Wes Simpson, 2008):

$$PLa = B_{CODECA} \times f_R \quad (2\text{-}1)$$

$$PLv = B_{CODECv} \times f_R \quad (2\text{-}2)$$

dengan:

PLa = *payload* paket audio (bit)

PLv = *payload* paket video (bit)

B_{CODECA} = *bit rate* CODEC *audio* (bps)

B_{CODECv} = *bit rate* CODEC *video* (bps)

f_R = *frame rate* (s)

Besarnya tiap paket audio dan video aplikasi video *streaming* merupakan penjumlahan dari *payload* paket audio dan video dengan *header* RTP, UDP, dan IP. Sehingga, besar tiap paket audio dan video *streaming* yang dihasilkan dapat dinyatakan dengan (Wes Simpson, 2008):

$$Pa = PLa/PLa_{max} \quad (2\text{-}3)$$

$$Pa\text{-}size = PLa + Pa \times (H_{RTP} + H_{UDP} + H_{IP}) \quad (2\text{-}4)$$

$P_v = PLv/PLv_{max}$ (2-5)

Pv-*size* = PLv + Pv x ($H_{RTP} + H_{UDP} + H_{IP}$) (2-6)

dengan:

Pa = jumlah paket *audio*

PLa_{max} = maksimum *payload* paket audio (bit)

Pv = jumlah paket *video*

PLv_{max} = maksimum *payload* paket video (bit)

Paket-paket video dan audio yang telah di-*encode* akan ditransmisikan melalui RTP (*Real Time Protocol*). Besar paket video *streaming* yang akan ditransmisikan dapat diperoleh menggunakan persamaan (Wes Simpson, 2008):

$P_{vs\text{-}size}$ = Pa-size + Pv-size (2-7)

dengan:

$P_{vs\text{-}size}$ = ukuran paket aplikasi video *streaming* (byte)

Pa-*size* = besar paket *audio* (byte)

Pv-*size* = besar paket *video* (byte)

H_{RTP} = *header* RTP (12 byte)

H_{IP} = panjang *header* IP (40 byte)

H_{UDP} = panjang *header* UDP (8 byte)

2.6.2 Bandwidth Aplikasi Video *Streaming*

Bandwidth aplikasi video *streaming* menyatakan banyaknya data yang dapat ditransmisikan setiap satuan waktu, dengan satuan bps (bit per *second*). *Bandwidth* aplikasi video *streaming* adalah jumlah *bandwidth* audio ditambah dengan video. Sehingga besarnya *bandwidth* aplikasi video *streaming* dapat dinyatakan dengan persamaan (Wes Simpson, 2008):

$$B_{vs} = Ba + Bv$$

$$= \frac{Pa_{size}}{PLa}.B_{CODECa} + \frac{Pv_{size}}{PLv}.B_{CODECv} \quad (2\text{-}8)$$

dengan:

Ba = *bandwidth* audio (bps)

Bv = *bandwidth* video (bps)

B_{vs} = *bandwidth* aplikasi video *streaming* (bps)

2.6.3 *Delay End-to-End* Video *Streaming*

Delay end-to-end aplikasi video *streaming* merupakan jumlah *delay* CODEC aplikasi video *streaming* dengan *delay* jaringan dimana aplikasi tersebut berjalan. *Delay* CODEC aplikasi video *streaming* terdiri dari *delay* CODEC audio dan video untuk menghasilkan satu paket video *streaming*. Besarnya *delay* CODEC aplikasi video *streaming* dapat dihitung dengan menggunakan persamaan:

$$t_{CODEC} = t_a + t_v \quad (2\text{-}9)$$

dengan:

t_{CODEC} = *delay* CODEC aplikasi *video streaming* (detik)

t_a = *delay* CODEC audio (detik)

t_v = *delay* CODEC *video* (detik)

Sehingga, total *delay end-to-end* penerapan aplikasi *video streaming* pada jaringan LTE menggunakan IPv6 dapat dihitung dengan persamaan:

$$t_{end\text{-}to\text{-}end} = t_{CODEC} + t_{net} \quad (2\text{-}10)$$

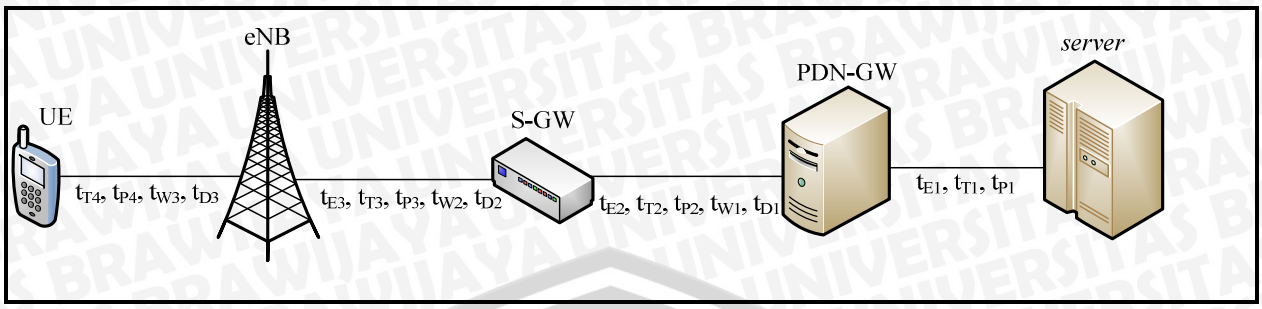
dengan:

t_{total} = *delay end-to-end* untuk aplikasi *video streaming* (detik)

t_{net} = delay jaringan LTE (detik)

2.6.4 *Delay End-to-End* Jaringan LTE

Delay adalah waktu yang dibutuhkan untuk mengirimkan paket data dari sumber sampai ke tujuan. *Delay* pada jaringan LTE merupakan penjumlahan *delay-delay* yang ada dalam perjalanan paket dari *server* hingga ke UE (*User Equipement*). Besarnya *delay end-to-end* pada jaringan LTE untuk aplikasi video *streaming* dapat ditunjukkan dalam Gambar 2.7



Gambar 2.7 *Delay* pada jaringan LTE

(**Sumber:** H. Holma dan A. Toskala, 2009)

Delay End-to-End berdasarkan Gambar dihitung dengan menggunakan persamaan:

$$t_{TOT} = t_E + t_D + t_T + t_P + t_W \quad (2\text{-}11)$$

dimana:

t_{TOT} = *delay end-to-end* pada jaringan LTE (s)

t_E = *delay* enkapsulasi (s)

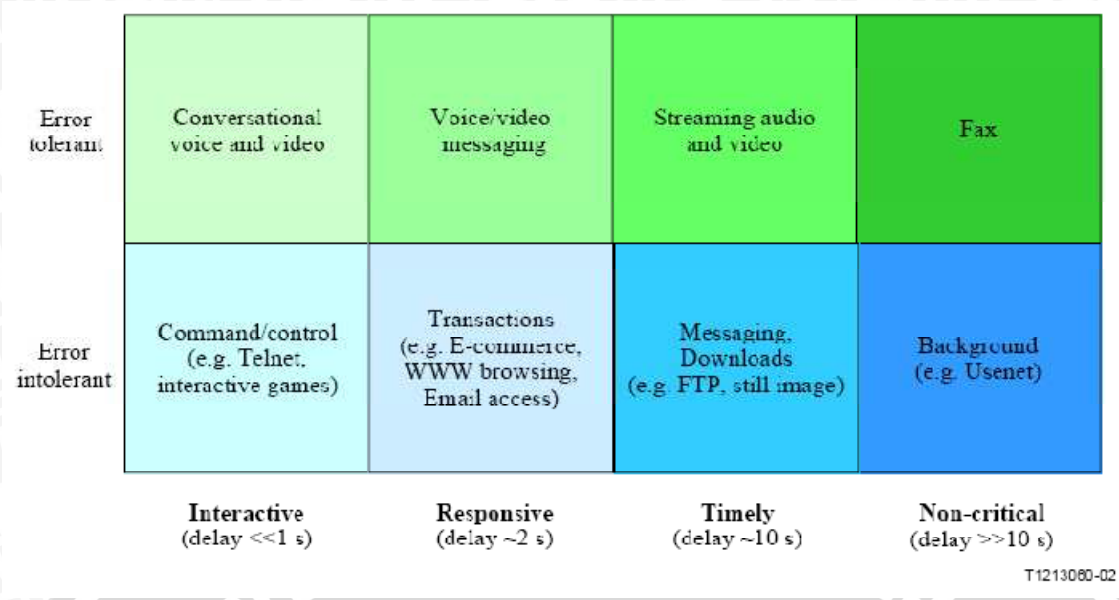
t_D = *delay* dekapsulasi (s)

t_T = *delay* transmisi (s)

t_P = *delay* propagasi (s)

t_W = *delay* antrian (s)

Aplikasi video *streaming* merupakan salah satu teknologi telekomunikasi yang sangat sensitif terhadap *delay*. Sesuai dengan referensi International Telecommunications Union (ITU) G.110, batasan *delay* untuk aplikasi video *streaming* adalah < 10s seperti yang ditunjukkan pada Gambar 2.8.



Gambar 2.8 Batasan *Delay* setiap Aplikasi Multimedia

(**Sumber**: ITU.T G.1010, 2001: 14)

a. *Delay* Proses

Delay proses merupakan waktu yang dibutuhkan untuk memproses paket data dan untuk menentukan kemana data tersebut akan diteruskan. *Delay* proses pada jarngan LTE meliputi *delay* enkapsulasi dan *delay* dekapsulasi.

Delay enkapsulasi adalah waktu yang dibutuhkan untuk menambahkan keseluruhan *header* pada sebuah paket sehingga paket data tersebut dapat tepat sampai ke tujuan. Sedangkan *delay* dekapsulasi adalah waktu yang dibutuhkan untuk melepaskan keseluruhan *header* dari sebuah paket. Besarnya *delay* dekapsulasi dan *delay* enkapsulasi dapat diperoleh dengan menggunakan persamaan di bawah ini (Onno W. Purbo, et al., 2001: 24):

$$t_E = \frac{W_{frame} - L}{C} \times 8 \qquad (2\text{-}12)$$

$$t_D = \frac{W_{frame} - L}{C} \times 8 \qquad (2\text{-}13)$$

dengan:

t_E = *delay* enkapsulasi (s)

t_D = *delay* dekapsulasi (s)

W_{frame} = panjang *frame* (*byte*)

L = panjang paket data di *node* (byte)

C = kecepatan pemrosesan data (bps)

Delay enkapsulasi terjadi pada *server*, PDN-GW, S-GW, dan e*Node*B. Besarnya *delay* enkapsulasi didapatkan dengan menggunakan persamaan:

$$t_E = t_{E1} + t_{E2} + t_{E3} + t_{E4} \quad (2\text{-}14)$$

dengan:

t_E = *delay* enkapsulasi total

t_{E1} = *delay* enkapsulasi pada *server*

t_{E2} = *delay* enkapsulasi pada PDN-GW

t_{E3} = *delay* enkapsulasi pada S-GW

t_{E4} = *delay* enkapsulasi pada e*Node*B

Sedangkan *delay* dekapsulasi terjadi di setiap *node* pada jaringan LTE, yaitu PDN-GW, S-GW, dan e*Node*B, dan UE. Besarnya *delay* dekapsulasi dapat dihutung menggunakan persamaan:

$$t_D = t_{D1} + t_{D2} + t_{D3} + t_{D4} \quad (2\text{-}15)$$

dengan:

t_D = *delay* dekapsulasi total

t_{D1} = *delay* dekapsulasi pada PDN-GW

t_{D2} = *delay* dekapsulasi pada S-GW

t_{D3} = *delay* dekapsulasi pada e*Node*B

t_{D4} = *delay* dekapsulasi pada UE

Nilai *delay* proses pada jaringan LTE untuk video *streaming* dapat diperoleh dengan menggunakan persamaan:

$$t_{proc} = t_E + t_D \quad (2\text{-}16)$$

dengan:

t_{proc} = *delay* proses

t_E = *delay* enkapsulasi total

t_D = *delay* dekapsulasi total

- ***Server***

  Pada server terjadi proses enkapsulasi sebelum paket data ditransmisikan ke PDN-GW. Paket data video streaming diberi penambahan header IP. Jumlah total *frame* pada *server* yang dapat dikirimkan ke PDN-GW sesuai dengan persamaan:

$$W_{frame\ server} = P_{VS\text{-}size} + H_{UDP} + H_{IP} + H_{ethernet} + FCS \quad (2\text{-}17)$$

Apabila panjang paket video *streaming* melebihi *Maximum Transfer Unit* (MTU) *Ethernet* sebesar 1500 *byte*, maka paket video *streaming* akan mengalami fragmentasi menjadi beberapa buah frame sesuai dengan persamaan:

$$N_{frame\ Ethernet} = \frac{P_{VS-size}}{MTU_{ethernet}} \quad (2\text{-}18)$$

dengan:

$N_{frame\ Ethernet}$ = jumlah *frame Ethernet*

$MTU_{ethernet}$ = MTU *Ethernet* (1500 byte)

Sehingga jika panjang paket video *streaming* melebihi *Maximum Transfer Unit* (MTU) *Ethernet*, jumlah total *frame* pada *server* yang dapat dikirimkan ke PDN-GW sesuai dengan persamaan:

$$W_{frame\ server} = P_{VS\text{-}size} + [N_{frame\ Ethernet} \times (H_{UDP} + H_{IP} + H_{ethernet} + FCS\ ] \quad (2\text{-}19)$$

Dalam skripsi ini, pada *server* digunakan standar *interface Gigabit Ethernet* dengan kecepatan 1 Gbps. Sehingga *delay* enkapsulasi pada *server* didapatkan dengan persamaaan berikut:

$$t_{E1} = \frac{W_{frame\ server} - P_{VS\text{-}size}}{C_{server}} x8 \quad (2\text{-}20)$$

dengan:

t_{E1} = *delay* enkapsulasi pada *server* (s)

$W_{frame\ server}$ = panjang frame pada *server* (byte)

C_{server} = kecepatan pemrosesan data pada *server* (bps)

- ***Packet Data Network Gateway* (PDN-GW)**

Pada PDN-GW, paket data yang diterima dari *server* mengalami proses dekapsulasi. Kemudian, sebelum paket data aplikasi video *streaming* ditransmisikan ke S-GW, dilakukan proses enkapsulasi terlebih dahulu.

Pada proses dekapsulasi, paket data aplikasi video *streaming* didapatkan sesuai dengan persamaan:

$$W_{VS\ PDN\text{-}GW} = W_{frame\ server} - H_{UDP} - H_{IP} - H_{ethernet} - FCS \quad (2\text{-}21)$$

Dengan mengasumsikan PDN-GW menggunakan standar *Fast Ethernet* dengan kecepatan 100 Mbps, maka besarnya *delay* dekapsulasi pada PDN-GW didapatkan sesuai dengan persamaan:

$$t_{D1} = \frac{W_{frameserver} - W_{VSPDN-GW}}{C_{PDN-GW}} \times 8 \quad (2\text{-}22)$$

dengan:

t_{D1} = *delay* dekapsulasi pada PDN-GW (s)

$W_{VS\ PDN\text{-}GW}$ = panjang paket data video *streaming* di PDN-GW (byte)

$C_{PDN\text{-}GW}$ = kecepatan pemrosesan data di PDN-GW (bps)

Besar MSS dapat diperoleh sesuai dengan persamaan berikut:

$$MSS = MTU - H_{GTP} - H_{UDP} - H_{IP} \quad (2\text{-}23)$$

Selanjutnya, paket data yang melebihi MSS akan disegmentasi menggunakan persamaan berikut:

$$N_{datagram} = \frac{W_{VSPDN-GW}}{MSS} \quad (2\text{-}24)$$

Kemudian, paket data aplikasi video *streaming* dienkapsulasi dengan penambahan *header* GTP (GPRS *Tunneling Protocol*), UDP (*User Datagram Protocol*), dan IP (*Internet Protocol*) sesuai dengan persamaan berikut:

$$W_{datagram} = W_{VS\ PDN\text{-}GW} + [N_{datagram} \times (H_{GTP} + H_{UDP} + H_{IP})] \quad (2\text{-}25)$$

Karena panjang datagram IP melebihi *Maximum Transfer Unit* (MTU) *Ethernet* (1500 *byte*), maka datagram IP akan mengalami fragmentasi menjadi beberapa buah frame sesuai dengan persamaan:

$$N_{frameEthernet} = \frac{W_{datagram}}{MTU_{ethernet}} \quad (2\text{-}26)$$

Sehingga, jumlah total *frame* pada PDN-GW yang dapat dikirimkan ke S-GW sesuai dengan persamaan:

$$W_{frame\ PDN\text{-}GW} = W_{datagram} + [N_{frame\ Ethernet} \times (H_{ethernet} + FCS)] \quad (2\text{-}27)$$

Sehingga *delay* enkapsulasi pada PDN-GW didapatkan dengan persamaaan:

$$t_{E2} = \frac{W_{framePDN-GW} - W_{VSPDN-GW}}{C_{PDN-GW}} \times 8 \quad (2\text{-}28)$$

dengan:

t_{E2} = *delay* enkapsulasi pada PDN-GW (s)

$W_{frame\ PDN\text{-}GW}$ = panjang *frame* di PDN-GW (byte)

$C_{PDN\text{-}GW}$ = kecepatan pemrosesan data di PDN-GW (bps)

- ***Serving Gateway* (S-GW)**

Pada S-GW, paket data yang diterima dari PDN-GW mengalami proses dekapsulasi. Kemudian, sebelum paket data aplikasi video *streaming* ditransmisikan ke eNodeB, dilakukan proses enkapsulasi terlebih dahulu.

Pada proses dekapsulasi, paket data aplikasi video *streaming* didapatkan sesuai dengan persamaan:

$$W_{VSS-GW} = W_{framePDN-GW} - H_{GTP} - H_{UDP} - H_{ethernet} - FCS \quad (2\text{-}29)$$

Dengan mengasumsikan S-GW menggunakan standar *Fast Ethernet* dengan kecepatan 100 Mbps, maka besarnya *delay* dekapsulasi pada S-GW didapatkan sesuai dengan persamaan:

$$t_{D2} = \frac{W_{framePDN-GW} - W_{VSS-GW}}{C_{S-GW}} \times 8 \quad (2\text{-}30)$$

dengan:

t_{D2} = *delay* dekapsulasi pada S-GW (s)

$W_{VS\ S\text{-}GW}$ = panjang paket data video *streaming* di S-GW (byte)

$C_{S\text{-}GW}$ = kecepatan pemrosesan data di S-GW (bps)

Besar MSS dapat diperoleh sesuai dengan persamaan berikut:

$$MSS = MTU - H_{GTP} - H_{UDP} - H_{IP} \quad (2\text{-}31)$$

Selanjutnya, paket data yang melebihi MSS akan disegmentasi menggunakan persamaan berikut

$$N_{datagram} = \frac{W_{VSS-GW}}{MSS} \quad (2\text{-}32)$$

Kemudian, paket data aplikasi video *streaming* dienkapsulasi dengan penambahan *header* GTP (GPRS *Tunneling Protocol*), UDP (*User Datagram Protocol*), dan IP (*Internet Protocol*) sesuai dengan persamaan berikut:

$$W_{datagram} = W_{VSS-GW} + [N_{datagram} \times (H_{GTP} + H_{UDP} + H_{IP})] \quad (2\text{-}33)$$

Karena panjang datagram IP melebihi *Maximum Transfer Unit* (MTU) *Ethernet* (1500 *byte*), maka datagram IP akan mengalami fragmentasi menjadi beberapa buah frame sesuai dengan persamaan:

$$N_{frameEthernet} = \frac{W_{datagram}}{MTU_{ethernet}} \quad (2\text{-}34)$$

Sehingga jumlah total *frame* pada S-GW yang dapat dikirimkan ke eNodeB sesuai dengan persamaan:

$$W_{\text{frame S-GW}} = W_{\text{datagram}} + [N_{\textit{frame}\ \text{Ethernet}} \text{ x } (H_{\text{ethernet}} + \text{FCS})] \quad (2\text{-}35)$$

Sehingga *delay* enkapsulasi pada S-GW didapatkan dengan persamaaan:

$$t_{E3} = \frac{W_{frameS-GW} - W_{VSS-GW}}{C_{S-GW}} \times 8 \quad (2\text{-}36)$$

dengan:

t_{E3} = *delay* enkapsulasi pada S-GW (s)

$W_{\text{frame S-GW}}$ = panjang *frame* di S-GW (byte)

$C_{S\text{-}GW}$ = kecepatan pemrosesan data di S-GW (bps)

- ***Evolved Node B*** **(eNodeB)**

Pada eNodeB, paket data yang diterima dari S-GW mengalami proses dekapsulasi. Kemudian, sebelum paket data aplikasi video *streaming* ditransmisikan ke UE, dilakukan proses enkapsulasi terlebih dahulu.

Pada proses dekapsulasi, paket data aplikasi video *streaming* didapatkan sesuai dengan persamaan:

$$W_{\text{VS eNodeB}} = W_{\textit{frame}\ \text{S-GW}} - H_{GTP} - H_{UDP} - H_{IP} - H_{\text{ethernet}} - \text{FCS} \quad (2\text{-}37)$$

eNodeB menggunakan *interface* STM-1 (*Synchronous Transport Module*-1) dengan kecepatan 155,52 Mbps. Maka besarnya *delay* dekapsulasi didapatkan dengan menggunakan persamaan:

$$t_{D3} = \frac{W_{frameS-GW} - W_{VSeNodeB}}{C_{eNodeB}} \times 8 \quad (2\text{-}38)$$

dengan:

t_{D3} = *delay* dekapsulasi pada eNodeB (s)

$W_{\text{VS eNodeB}}$ = panjang paket data video *streaming* di eNodeB (byte)

C_{eNodeB} = kecepatan pemrosesan data di eNodeB (bps)

Pada *layer* PDCP (*Packet Data Convergence Protocol*), paket datagram IP video *streaming* dienkapsulasi dengan penambahan *header* sebesar 1 *byte*, yaitu *header* PDU. Banyaknya paket IP pada eNodeB dapat diperoleh sebagai berikut:

$$N_{datagram} = \frac{W_{VSeNodeB}}{MTU} \quad (2\text{-}39)$$

Pada layer PDCP, panjang frame dapat dihitung dengan menggunakan persamaan:

$$W_{frame\ PDCP} = W_{VS\ eNodeB} + H_{PDCP} \quad (2\text{-}40)$$

Selanjutnya pada *layer* RLC (*Radio Link Control*), *frame* PDCP disegmentasi menjadi RLC PDU *fixed size* sebesar 40 *byte.* Sehingga jumlah *frame* RLC sesuai dengan persamaan

$$N_{frameRLC} = \frac{W_{framePDCP}}{40byte} \quad (2\text{-}41)$$

dengan:

$N_{frame\ RLC}$ = jumlah *frame* RLC

$W_{frame\ PDCP}$ = panjang *frame* PDCP (byte)

Setiap *frame* selanjutnya diberi *header* sebesar 2 *byte*, sehingga panjang setiap *frame* RLC sebesar 42 *byte*. Panjang *frame* RLC total yang siap diteruskan ke *layer* MAC (*Medium Access Control*) sesuai dengan persamaan:

$$W_{frame\ RLC\ total} = N_{frame\ RLC} \times W_{frame\ RLC} \quad (2\text{-}42)$$

Pada *layer* MAC, RLC disegmentasi menjadi MAC SDU (*Service Data Unit*) sebesar 42 *byte*. Jumlah *frame* SDU sesuai dengan persamaan:

$$N_{frameMAC} = \frac{W_{frameRLCtotal}}{42byte} \quad (2\text{-}43)$$

dengan:

$N_{frame\ MAC}$ = jumlah *frame* MAC

$W_{frame\ RLC\ total}$ = panjang *frame* RLC total (byte)

Sedangkan panjang *frame* MAC dihitung dengan menggunakan persamaan:

$$W_{frameMAC} = H_{MAC} + 42\ byte \quad (2\text{-}44)$$

Panjang *frame* yang siap ditransmisikan menuju UE merupakan panjang *frame* MAC total sesuai dengan persamaan:

$$W_{frame\ eNodeB} = N_{frame\ MAC} \text{ x } W_{frame\ MAC} \quad (2\text{-}45)$$

Sehingga besar *delay* dekapsulasi yang terjadi pada eNodeB didapatkan dengan menggunakan persamaan:

$$t_{E4} = \frac{W_{frameeNodeB} - W_{VSeNodeB}}{C_{eNodeB}} \times 8 \quad (2\text{-}46)$$

dengan:

t_{E4} = *delay* enkapsulasi pada eNodeB (s)

$W_{VS\ eNodeB}$ = panjang paket data video *streaming* di eNodeB (byte)

C_{eNodeB} = kecepatan pemrosesan data di eNodeB (bps)

- ***User Equipment*** **(UE)**

Pada UE, paket data yang diterima dari e*Node*B mengalami proses dekapsulasi. Proses dekapsulasi pada UE sesuai dengan persamaan:

$$W_{VS\ UE} = W_{frame\ eNode\ B} - H_{GTP} - H_{UDP} - H_{IP} - H_{PDCP} - H_{RLC} - H_{MAC} \quad (2\text{-}47)$$

Dengan mengasumsikan *data rate* maksimum sebesar 3,6 Mbps, maka besarnya *delay* dekapsulasi yang terjadi di UE didapatkan dengan persamaan:

$$t_{D4} = \frac{W_{frameeNodeB} - W_{VSUE}}{C_{UE}} \times 8 \quad (2\text{-}48)$$

dengan:

t_{D4} = *delay* dekapsulasi pada UE (s)

$W_{VS\ UE}$ = panjang paket data video *streaming* pada UE (byte)

C_{UE} = kecepatan pemrosesan data pada UE (bps)

b. *Delay* Transmisi

Delay transmisi adalah waktu yang dibutuhkan untuk meletakkan sebuah paket data ke media transmisi. Dipengaruhi ukuran paket data dan kecepatan transmisi. *Delay* transmisi dapat ditentukan dengan persamaan (Mischa Schwartz, 1987 : 132):

$$t_T = \frac{W}{C} \quad (2\text{-}49)$$

Sehingga *delay* transmisi total ditentukan dengan persamaan:

$$t_{T\ tot} = t_{T1} + t_{T2} + t_{T3} + t_{T4} \quad (2\text{-}50)$$

dimana:

t_T tot = *delay* transmisi total (s)

W = panjang *frame* pada *node* (bit)

C = kecepatan transmisi (bps)

t_{T1} = *delay* transmisi pada *server* – PDN-GW (s)

t_{T2} = *delay* transmisi pada PDN-GW – S-GW (s)

t_{T3} = *delay* transmisi pada S-GW – eNodeB (s)

t_{T4} = *delay* transmisi pada eNodeB -UE (s)

c. *Delay* Propagasi

Delay propagasi adalah waktu yang dibutuhkan untuk merambatkan paket data melalui media transmisi dari *server* ke UE. *Delay* propagasi ditentukan dengan persamaan (Forouzan, Behrouz A, 2000 : 215):

$$t_P = \frac{N_{frame}\, x\, R}{v} \qquad (2\text{-}51)$$

Sehingga *delay* propagasi total ditentukan dengan persamaan:

$$t_{p\,tot} = t_{p1} + t_{p2} + t_{p3} + t_{p4} \qquad (2\text{-}52)$$

dimana:

$t_{p\,tot}$ = *delay* propagasi (s)

N_{frame} = jumlah *frame* pada node

R = jarak antar *node* (m)

v = cepat rambat gelombang di cahaya = 3×10^8 (m/s)

t_{p1} = *delay* propagasi pada *server* – PDN-GW (s)

t_{p2} = *delay* propagasi pada PDN-GW – S-GW (s)

t_{p3} = *delay* propagasi pada S-GW - eNodeB (s)

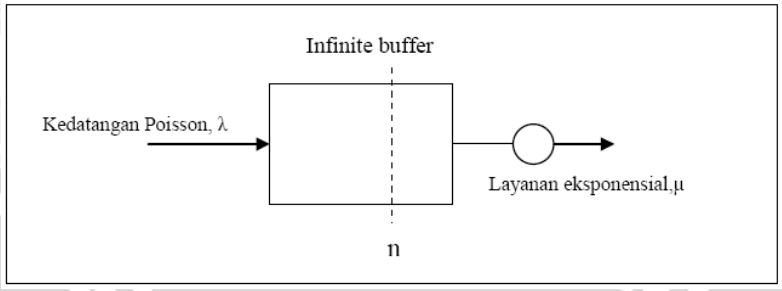
t_{p4} = *delay* propagasi pada eNodeB -UE (s)

d. *Delay Antrian*

Delay antrian adalah waktu dimana paket data berada dalam antrian untuk ditransmisikan. Selama waktu ini, paket data menunggu hingga paket yang lain selesai ditransmisikan. *Delay* antrian dapat dihitung dengan menggunakan model antrian M/M/1. M pertama menunjukkan kedatangan *Poisson*, M kedua berarti distribusi waktu

pelayanan eksponensial, dan 1 menunjukkan 1 menunjukkan jumlah server yang akan melayani pelanggan.

Disiplin antrian yang digunakan adalah FIFO (*First In First Out*). Bentuk model antrian M/M/1 dapat dilihat pada Gambar 2.9.



Gambar 2.9 Model antrian M/M/1

(Sumber: Mischa Schwartz, 1987:31)

Delay antrian terjadi pada setiap *node* pada jaringan LTE, yaitu PDN-GW, S-GW, dan eNodeB. Besarnya *delay* antrian pada *node* ditentukan dengan persamaan (Mischa Schwartz, 1987 : 42):

$$t_w = t_{queue} + t_{serv} \qquad (2\text{-}53)$$

dengan:

t_w = *delay* antrian (s)

t_{queue} = waktu tunggu paket pada *node* (s)

t_{serv} = waktu rata-rata pelayanan *node* (s)

Perhitungan waktu rata-rata pelayanan antrian data di *node* (t_{serv}) dihitung dengan menggunakan persamaan (Mischa Schwartz, 1987 : 23):

$$t_{serv} = \frac{L}{C} \qquad (2\text{-}54)$$

dengan:

μ = kecepatan pelayanan *node* (bps)

L = panjang paket data di *node* (bit)

C = kecepatan transmisi pada *node* (bps)

Sedangkan waktu tunggu paket pada *router* (t_{queue}) dihitung dengan menggunakan persamaan (Mischa Schwartz, 1987 : 42):

$$t_{queue} = \frac{\lambda}{\mu^2(1-\rho)} \quad (2\text{-}55)$$

Untuk performansi sistem antrian, ditunjukkan dalam bentuk ρ (faktor utilisasi), yang nilainya diasumsikan berubah-ubah dengan kenaikan tertentu. Faktor utilisasi bernilai lebih besar dari 0 dan lebih kecil dari 1 dengan kenaikan sebesar 0,1. Hal ini disebabkan karena jika ρ > 1 berarti rata–rata kedatangan melampaui rata–rata pelayanan atau server tidak dapat menjaga rata–rata kedatangan dan panjang antrian yang bertambah tanpa batas. Besarnya faktor utilisasi dapat dihitung dengan menggunakan persamaan:

$$\rho = \frac{\lambda}{\mu} \Rightarrow \lambda = \mu\rho \quad (2\text{-}56)$$

Sehingga dari persamaan 2.51 dan 2.52 *delay* antrian dapat ditentukan dengan persamaan:

$$t_w = \frac{\lambda}{\mu.(\mu-\lambda)} + \frac{1}{\mu} \quad (2\text{-}57)$$

dengan:

t_{queue} = waktu tunggu paket pada *node* (s)

λ = kecepatan kedatangan paket pada *node* (bps)

ρ = faktor utilitas ($0 < \rho < 1$)

μ = kecepatan pelayanan *node* (bps)

Sehingga *delay* antrian total ditentukan dengan persamaan:

$$t_{w\,tot} = t_{w1} + t_{w2} + t_{w3} \quad (2\text{-}58)$$

dengan:

$t_{w\,tot}$ = *delay* antrian total (s)

t_{w1} = *delay* antrian pada PDN-GW (s)

t_{w2} = *delay* antrian pada S-GW (s)

t_{w3} = *delay* antrian pada e*Node*B (s)

2.5.2 Probabilitas *Packet Loss* Video *Streaming* pada Jaringan LTE

Paket *loss* adalah jumlah paket yang hilang dibandingkan dengan paket yang diterima benar. Umumnya perangkat jaringan memiliki *buffer* untuk menampung data yang diterima. Jika terjadi tabrakan yang cukup banyak, *buffer* akan penuh, dan data

baru tidak dapat diterima. Paket yang hilang ini harus ditransmisi ulang, yang akan membutuhkan waktu tambahan. Prosentase *packet loss* maksimum yang diperbolehkan oleh ITU.T G.1010 untuk aplikasi video *streaming* adalah < 1%. Apabila packet loss yang dihasilkan melebihi 1%, kualitas video *streaming* akan buruk.

Probabilitas *packet loss* total penerapan aplikasi v*ideo streaming* pada suatu jaringan, ditentukan berdasarkan pada probabilitas *packet loss* pada jaringan LTE dan server. Probabilitas *packet loss* video *streaming* yang berbasis protokol UDP/IP seperti pada persamaan

$$\rho_{tot} = 1 - [(1 - \rho_{network})(1 - \rho_{VS})] \quad (2\text{-}59)$$

dimana:

ρ_{tot} = probabilitas *packet loss* video *streaming*

$\rho_{network}$ = probabilitas *packet loss* pada jaringan LTE

ρ_{VS} = probabilitas *packet loss* pada *server*

Prosentase *packet loss* ditentukan dengan Persamaan 2-56 (Wijaya Hendra A, 2005 : 34) :

$$packet\ loss(\%) = \frac{N_{packet\ loss}}{N_{paket} + N_{packet\ loss}} x100\ \% \quad (2\text{-}60)$$

dengan :

$N_{packet\ loss}$ = jumlah paket yang hilang

N_{paket} = jumlah paket yang diterima dengan benar

2.5.2.1 Probabilitas *Packet Loss* pada Server

Protokol UDP/IP sendiri tidak dapat memberikan jaminan bahwa paket akan dikirimkan semua sesuai dengan permintaan. *Packet loss* yang terjadi pada server di *layer transport* dan *layer network* dapat dihitung dengan persamaan:

$$\rho_{VS} = P_{VS\text{-}size} \times \rho_b \quad (2\text{-}61)$$

dimana:

ρ_{VS} = probabilitas *packet loss* pada server

ρ_b = BER (10^{-8})

$P_{VS\text{-}size}$ = panjang paket data video *streaming* (byte)

2.5.2.2 Probabilitas *Packet Loss* pada Jaringan LTE

Probabilitas packet loss pada jaringan LTE dapat diperoleh dari probabilitas bit error rate (BER) di jaringan tersebut, sesuai dengan persamaan:

$$\rho_{LTE} = P_{VStot} \times \rho_{b\text{-}LTE} \quad (2\text{-}62)$$

dimana:

ρ_{VS} = probabilitas *packet loss* pada jaringan LTE

ρ_b = BER LTE pada saat transmisi (tanpa satuan)

$P_{VS\text{-}size}$ = panjang paket video *streaming* yang ditransmisikan pada jaringan LTE (byte)

BER (*bit error rate*) atau dengan sebutan lain probabilitas *error* bit merupakan nilai ukur kualitas sinyal yang diterima untuk sistem transmisi data digital. BER juga dapat didefinisikan sebagai perbandingan jumlah bit *error* terhadap total bit yang diterima. Nilai BER untuk modulasi 64-QAM pada jaringan LTE dapat dihitung dengan persamaan (Andrea Goldsmith, 2005:167):

$$P_b = Q\left(\sqrt{2\frac{E_b}{N_o}}\right) \quad (2\text{-}63)$$

dengan:

P_b = BER LTE pada saat transmisi (tanpa satuan)

$\frac{Eb}{No}$ = rasio energy bit terhadap kerapatan noise pada saat transmisi (dB)

Nilai Q dapat diperoleh dengan distribusi Gaussian menggunakan persamaan (John G. Proakis, 2000: 40):

$$Q(x) = \frac{1}{2} erfc\left(\frac{x}{\sqrt{2}}\right) \quad (2\text{-}64)$$

$$erfc(x) \approx \frac{1}{\sqrt{\pi x}} . e^{-x^2} \quad (2\text{-}65)$$

dengan:

erfc : error function complementary

Eb/No adalah suatu parameter yang berhubungan dengan SNR yang biasanya digunakan untuk menentukan laju data digital dan sebagai ukuran mutu standar untuk kinerja sistem komunikasi digital. Hubungan SNR dengan Eb/No ditunjukkan dalam persamaan:

$$\frac{Eb}{No}(dB) = \frac{S}{N} - 10 \log \frac{B}{R} \quad (2\text{-}66)$$

dengan :

PG = *processing gain* (dB)

Besarnya pengaruh redaman sinyal terhadap sinyal yang ditransmisikan dapat dinyatakan dengan perbandingan antara sinyal dengan noise (SNR) yang dinyatakan dalam persamaan berikut (E. Glatz, 1999):

$$SNR_{(dB)} = P_r(dBm) - N_o(dBm) \quad (2\text{-}67)$$

dengan :

SNR = *signal to noise ratio* (dB)

P_r = daya yang diterima (dBm)

N = daya *noise* saluran transmisi (dBm)

Sedangkan untuk perhitungan daya noise dinyatakan dalam persamaan berikut (E. Glatz, 1999):

$$N_{(dBm)} = 10 \log k \cdot T + 10 \log B + NF \quad (2\text{-}68)$$

dengan :

N = daya *noise* saluran transmisi (dBm)

k = konstanta Boltzman (1,38 x 10^{-23} J/K)

T = suhu *absolute* (300º K)

NF = *noise figure* (11,2 dB)

B = *bandwidth* (Hz)

Daya yang diterima oleh penerima sangat dipengaruhi oleh propagasi sinyal dari pemancar ke penerima. Sehingga daya yang diterima dapat dinyatakan dalam persamaan berikut (Robert G. Winch, 1998: 184) :

$$P_r(dBm) = P_t - FSL - L_t - L_r + G_r + G_t \quad (2\text{-}69)$$

Sedangkan nilai FSL (*Free Space Loss*) dapat diperoleh menggunakan persamaan berikut (Andrea Goldsmith, 2005: 49)

$$FSL = -20 \log \left(\frac{\lambda}{4\pi d} \right) \quad (2\text{-}70)$$

dengan:

P_r : daya terima *receiver* (dBm)

P_t : daya pancar *transmitter* (dBm)

FSL : *free space loss* (dB)

L_t : *transmitter losses* (*cable loss*) (dB)

L_r : *receiver losses* (*body loss*) (dB)

G_r : *gain receiver* (dBi)

G_t : *gain transmitter* (dBi)

λ : Panjang gelombang (m)

f : Frekuensi kerja sistem (Hz)

d : jarak antara pemancar dan penerima (m)

c : kecepatan cahaya ($3x10^8$ m/s)

2.5.3 *Throughput*

Throughput adalah kecepatan maksimum jaringan saat tidak ada data yang hilang pada pentransmisiannya atau banyaknya data yang diterima dengan benar oleh *user*. *Throughput* yang mungkin didapat dengan memperhatikan probabilitas paket diterima dalam keadaan salah dapat dihitung dengan persamaan (Mischa Schwartz, 1987:129)

$$T = \frac{1}{t_v} = \frac{(1-\rho_{tot})}{t_l[1+(\alpha-1)\rho_{tot}]} \qquad (2\text{-}71)$$

dimana:

T = *throughput* (bps)

t_v = waktu rata-rata transmisi untuk mengirimkan paket yang benar (s)

t_1 = waktu transmisi sebuah paket data atau *frame* (s)

ρ_{tot} = probabilitas frame yang salah

α = konstanta

Sedangkan parameter α dapat dihitung dengan persamaan (Mischa Schwartz, 1987:129):

$$t_T = t_l + t_{out} \qquad (2\text{-}72)$$

dan:

$$\alpha = \frac{t_T}{t_l} = 1 + \frac{t_{out}}{t_l} \qquad (2\text{-}73)$$

dengan:

t_T = waktu total yang dibutuhkan untuk mentransmisikan sebuah paket (s)

t_l = waktu yang dibutuhkan untuk mentransmisikan sebuah paket (s)

t_{out} = waktu yang dibutuhkan untuk menerima *acknowledge* atau interval waktu antara pengiriman sebuah paket dengan pengiriman paket selanjutnya/ *fixed timed out interval* (s)

Waktu transmisi *frame* (t_l) ditentukan dengan persamaan (Mischa Schwartz, 1987:132):

$$t_l = \frac{\left(PL_{frame} + H_{frame}\right) x 8}{C_{trans}} \quad (2\text{-}74)$$

dengan:

PL_{frame} = *payload frame* (byte)

H_{frame} = *header frame* (byte)

C_{trans} = kecepatan transmisi node (bps)

Sedangkan waktu yang dibutuhkan untuk menerima *acknowledge* dapat dihitung dengan menggunakan persamaan (Mischa Schwartz, 1987:129):

$$t_{out} = 2t_p + 2t_I + t_{pro} \quad (2\text{-}75)$$

dengan:

t_{out} = waktu yang dibutuhkan untuk menerima *acknowledge*/*fixed timed out interval* (s)

t_p = *delay* propagasi untuk satu *frame* (s)

t_{pro} = *delay* proses total untuk satu *frame* (s)

Nilai dari t_{pro} dan t_P dapat dihitung sesuai dengan persamaan:

$$t_{pro} = \frac{t_{proc}}{N_{frame}} + \frac{t_{wtotal}}{N_{frame}} \quad (2\text{-}76)$$

$$t_P = \frac{t_{ptotal}}{N_{frame}} \quad (2\text{-}77)$$

dengan:

t_{proc} = *delay* proses (s)

$t_{W\ total}$ = *delay* antrian total (s)

t_{ptotal} = *delay* propagasi total (s)

N_{frame} = jumlah frame