BAB II

TINJAUAN PUSTAKA DAN LANDASAN TEORI

2.1 Penelitian Sebelumnya

Dalam penyusunan tugas akhir ini, dilakukan penelitian terhadap sumber-sumber yang ada seperti skripsi dan jurnal yang memiliki hubungan dengan topik yang ditulis oleh penulis. Berikut adalah karya ilmiah yang berhubungan dengan topik tersebut:

Diki Tri Pambudi melakukan penelitian pada tahun 2020 yang berjudul "Membangun *Server RTMP Streaming* di SMP Negeri 1 Balapulang menggunakan *Ubuntu 16.04*". Hasil yang diperoleh adalah Siswa dan siswi SMP Negeri l Balapulang bisa mengakses layanan *RTMP Streaming* melalui aplikasi *VLC*. Kekurangannya yaitu hanya menggunakan protokol *RTMP* dan tidak ada limit akses ke *server RTMP* [4].

Penelitian yang dilakukan oleh Hamid Azwar pada tahun 2020 yang berjudul "Pengiriman Video Secara *Live Streaming* Menggunakan *Dynamic Adaptive Streaming over HTTP (DASH)*". Hasil dari penelitian yang dilakukan menunjukkan bahwa nilai *throughput* pengiriman video mencapai 1,4 Mbps dan nilai *delay* terbesar yang diperoleh adalah 12,5 ms. Persentase permintaan klien untuk kualitas video tertinggi untuk satu klien mencapai 98%, namun ketika digunakan oleh 20 klien, permintaan untuk video kualitas tinggi berkurang menjadi 46%. Hal ini menunjukkan bahwa teknologi *DASH (Dynamic Adaptive Streaming over HTTP)* berhasil diimplementasikan dalam proses *live streaming*. Kekurangannya yaitu masih menggunakan protokol *streaming Dynamic Adaptive Streaming over HTTP (DASH)* [5].

Penelitian yang dilakukan oleh Sri Hartanto dan Minda Mora Purba pada tahun 2020 yang berjudul "Inovasi Pembelajaran Berbasis *Video Streaming* dan Jaringan Telekomunikasi di Perguruan Tinggi". Hasil dari penelitian yang dilakukan adalah mahasiswa dapat mengakses *video streaming* mata kuliah perancangan program komputer melalui jaringan *WLAN*. Kekurangannya yaitu menggunakan *VLC* sehingga mahasiswa harus mengkonfigurasi terlebih dahulu di aplikasi *VLC* [6].

Penelitian yang dilakukan oleh Fika Septiana dan Ady Chandra Nugroho pada tahun 2021 yang berjudul "Rancang Bangun Sistem Perkuliahan Dengan *Video On Demand* Menggunakan Aplikasi *Camtasia*". Hasil dari penelitian yang dilakukan menunjukkan bahwa *video on demand* yang dirancang menggunakan aplikasi *camtasia studio* dapat diakses melalui *web*. Kekurangannya yaitu Menggunakan *camtasia studio* karena harus membeli aplikasinya [7] .

Penelitian yang dilakukan oleh Annabella Medina Aisyah pada tahun 2020 yang berjudul "Rancang Bangun Real-Time Messaging Protocol Untuk Aplikasi Live Streaming Berbasis Cloud Server". Hasil yang diperoleh adalah RTMP pada server streaming dapat bekerja dengan baik dan mampu menampilkan hasil capturing dari aplikasi live streaming berbasis android yang telah dibuat melalui web page monitoring.Kekurangannya yaitu Tidak jelaskan ada tidaknya limit akses ke server RTMP sehingga siapapun bisa jadi streamer atau broadcaster [8].

Tabel 2. 1 Penelitian Sebelumnya

No.	Peneliti	Judul Penelitian	Tujuan Penelitian	Metode Penelitian	Hasil Penelitian	Kekurangan	Perbandingan dengan Penelitian terkait
1	Diki Tri Pambudi (2020)	Membangun *Server RTMP Streaming* di SMP Negeri 1 Balapulang menggunakan *Ubuntu 16.04*	Membuat *desain* dan realisasi dari *server video real time* sebagai sarana pembelajaran bagi siswa dan siswi di SMP Negeri l Balapulang.	Metode dokumentasi adalah salah satu metode pengumpulan data yang digunakan.	Siswa dan siswi SMP Negeri l Balapulang bisa mengakses layanan *RTMP Streaming* melalui aplikasi *VLC*	Hanya menggunakan protokol *RTMP* dan tidak ada limit akses ke *server RTMP*	Pada penelitian yang akan dilakukan menggunakan dua protokol, yaitu *RTMP* dan *HLS* serta terdapat limit akses ter-hadap *server RTMP*

No.	Peneliti	Judul Penelitian	Tujuan Penelitian	Metode Penelitian	Hasil Penelitian	Kekurangan	Perbandingan dengan Penelitian terkait
2	Hamid Azwar (2020)	Pengiriman Video Secara *Live Streaming* Menggunakan *Dynamic Adaptive Streaming over HTTP (DASH)*	Melakukan evaluasi pengi-riman video secara *live streaming* pada jaringan *LAN*	Metode *Dynamic Adaptive Streaming over HTTP (DASH)*	Teknologi *DASH (Dynamic Adaptive Streaming over HTTP)* berhasil diimplementasikan dalam proses *live streaming*.	Masih menggunakan protokol *stream-ing Dynamic Adaptive Stream-ing over HTTP (DASH)*	Pada penelitian yang akan dilakukan menggunakan dua protokol, yaitu *RTMP* dan *HLS*

No.	Peneliti	Judul Penelitian	Tujuan Penelitian	Metode Penelitian	Hasil Penelitian	Kekurangan	Perbandingan dengan Penelitian terkait
3	Sri Hartanto dan Minda Mora Purba (2020)	Inovasi Pembelajaran Berbasis *Video Streaming* dan Jaringan Telekomunikasi di Perguruan Tinggi	Mahasiswa dapat mengakses video *streaming* mata kuliah perancangan program kom-puter melalui jaringan *WLAN* kampus ketika mahasiswa tersebut berhalangan hadir	Membangun jaringan WLAN dengan satu server dan dua laptop, Membangun Server Pembelajaran dengan men-setup VLC Media	Mahasiswa dapat mengakses video streaming mata kuliah perancangan program komputer melalui jaringan WLAN	Menggunakan VLC sehingga mahasiswa harus mengkonfigurasi terlebih dahulu di aplikasi VLC	Pada penelitian yang akan dilakukan, user bisa mengakses video live streaming melalui browser

No.	Peneliti	Judul Penelitian	Tujuan Penelitian	Metode Penelitian	Hasil Penelitian	Kekurangan	Perbandingan dengan Penelitian terkait
4	Fika Septiana dan Ady Chandra Nugroho (2021)	Rancang Bangun Sistem Perkuliahan Dengan *Video On Demand* Menggunakan Aplikasi *Camtasia*	Mempermudah mahasiswa mendapatkan materi dan membantu mahasiswa dalam menerima mata kuliah dengan cara yang menarik	*Waterfall*	*Video on demand* yang dirancang menggunakan aplikasi *camtasia studio* dapat diakses melalui *web*	Harus mengeluarkan biaya untuk menggunakan aplikasi *camtasia studio*	Pada penelitian yang akan dilakukan menggunakan *tools open source* yaitu *docker compose*

No.	Peneliti	Judul Penelitian	Tujuan Penelitian	Metode Penelitian	Hasil Penelitian	Kekurangan	Perbandingan dengan Penelitian terkait
5	Annabella Medina Aisyah (2020)	Rancang Bangun *Real-Time Messaging* Protocol Untuk Aplikasi *Live Streaming* Berbasis *Cloud Server*	Membangun *server video live streaming* yang digunakan untuk menyajikan berita terkini secara cepat dan akurat pada PT XYZ	*Video streaming*.	*RTMP* pada *server streaming* dapat bekerja dengan baik dan mampu menampilkan hasil *capturing* dari aplikasi *live streaming* berbasis *android* yang telah dibuat melalui *web page monitoring*.	Tidak jelaskan ada tidaknya limit akses ke *server RTMP* sehingga siapapun bisa jadi *streamer* atau *broadcaster*	Pada penelitian yang akan dilakukan terdapat limit akses terhadap *server RTMP*

2.2 Ringkasan Penelitian Sebelumnya

Pada layanan *video live streaming* yang berdasarkan penelitian sebelumnya masih terdapat kekurangan seperti tidak ada limit akses ke *server RTMP (Real Time Messaging Protocol)* sehingga siapapun bisa menjadi *streamer* atau *broadcaster*. Kemudian ada juga yang masih menggunakan *software* atau aplikasi berbayar seperti *camtasia studio* sehingga harus mengeluarkan biaya untuk membeli aplikasi tersebut. Selanjutnya ada yang hanya menggunakan satu protokol *streaming* saja yaitu *RTMP (Real Time Messaging Protocol)* sehingga pengguna harus mengkonfigurasi menggunakan *software VLC* agar bisa menyaksikan layanan *video live streaming*. Secara garis besar, Penelitian yang penulis ajukan memiliki perbedaan pada penggunaan limit akses terhadap *server RTMP* serta protokol yang dimplementasikan.

2.3 Dasar Teori

2.3.1. *Video Live Streaming*

Video live streaming adalah video yang ditayangkan secara *live* ke pengguna melalui *website* atau aplikasi [9]. Pengguna tidak perlu melakukan proses *down-load* apapun untuk dapat menonton *video live streaming* yang tersedia di *platform* atau *website* seperti *YouTube* atau *Instagram*, sehingga pengguna dapat menonton video langsung tanpa harus menunggu proses *download* selesai. *User* bisa menonton *video live streaming* berdasarkan kebutuhannya dengan mengklik video yang akan ditonton.

2.3.2. *Video on Demand*

Video on demand atau VoD adalah layanan yang memungkinkan pengguna untuk meminta video sesuai keinginan mereka [10]. Pengguna dapat misalnya, mencari film pilihan mereka di aplikasi yang dapat digunakan di *PC* atau *smartphone*, lalu mengunduh atau menyewanya untuk ditonton. Biayanya juga bervariasi, ada yang perlu berlangganan, menggunakan biaya paket data, dan ada juga yang bisa didapatkan melalui pulsa atau cara pembayaran lainnya.

VoD juga dapat dianggap sebagai layanan *streaming* video berlangganan yang memungkinkan pengguna memilih acara yang ingin mereka tonton dari saluran televisi mana pun dan dari negara mana pun. Akibatnya, disebutkan bahwa layanan semacam ini mirip dengan ide kuno menyewa film atau kaset. Hanya saja, ide ini dimodernisasi dari waktu ke waktu untuk memudahkan *user* menonton acara favorit mereka.

2.3.3. *Web Server*

Web Server yaitu *tools* untuk melakukan komunikasi melalui *website* antara pengguna dan *server website* dengan menggunakan protokol *HTTP* atau *HTTPS* [11]. *Web server* memungkinkan pengguna yang terbiasa dengan *browser web* (*Chrome, Firefox,* dan lain-lain) untuk menerima permintaan dari *client* yang di tampilkan dalam bentuk halaman *web* sebagai *response server web* terhadap permintaan tersebut.

Manfaat dari *web server*:

1) Membuat *cache* penyimpanan agar *website* dapat lebih cepat diakses.
2) Memeriksa sistem keamanan berdasarkan permintaan klien atau *browser web* yang dilakukan melalui *HTTP* atau *HTTPS*.
3) Sistem memberikan akses data yang diminta oleh *user* agar dapat memastikan keamanan sistem dan kemudahan operasi.

2.3.4. *Nginx*

Nginx adalah *web server* yang dapat menangani lebih banyak lalu lintas atau *traffic* data dalam sebuah *website* [12]. *Website* bisa dibuat lebih handal dan lebih cepat diakses dengan *Nginx*. *Nginx* yang awalnya digunakan hanya sebagai *server HTTP*, dengan kemajuan teknologi saat ini digunakan untuk berbagai fungsi seperti cache *HTTP* dan bisa digunakan pada *operating system* antara lain: *ubuntu*, *debian*.

2.3.5. *RTMP*

Real Time Messaging Protocol (RTMP) yaitu communication protocol yang digunakan untuk streaming [13]. Dikembangkan oleh *Macromedia* dan kemudian diakuisisi oleh *Adobe*, *RTMP* digunakan untuk *streaming audio*, dan video. *RTMP* memungkinkan komunikasi *real-time*. Contohnya, data yang dapat dikomunikasikan melalui *RTMP* antara lain data yang sudah direkam sebelumnya, video langsung, dan lain-lain.

2.3.6. *Video js*

Video js adalah sebuah *tools* untuk memainkan video di *web* yang dibangun menggunakan *HTML5*. Format yang digunakan seperti *DASH* dan *HLS* dapat ditemukan di *browser* [14]. *Media player* ini memberikan bentuk yang sama di seluruh *browser*, memasukkan video di situs *web*, dan mempunyai tampilan yang tidak beda bagi *user*, tidak peduli *device* atau *browser* yang digunakan. *Video.js* dipercaya sebagai pemutar video *HTML 5* yang termasuk stabil.

2.3.7. *HTTP DAN HTTPS*

HTTP (Hypertext Transfer Protocol) adalah *communication protocol* untuk memproses pengiriman data antara *server* dan komputer *user* [15]. Dokumen, file, gambar, dan video dapat ditransfer menggunakan protokol ini. Namun, jika sebuah situs *web* menggunakan *HTTP*, data yang dikirim dari *browser* ke *server* tidak dienkripsi, sehingga menyebabkan *port* 80 dapat disalahgunakan oleh pihak ketiga.

Kemudian untuk mengenkripsi data tersebut dapat menggunakan protokol komunikasi yang disebut sebagai *HTTPS (Hypertext Transfer Protocol Secure)* yang dianggap lebih aman daripada *HTTP*. Saat *sending* dan *receive* data melewati *port* 443 pada sambungan yang dienkripsi menggunakan *SSL/TLS (Transport Layer Security)*, *HTTPS* menggunakan *TCP (Transmission Control Protocol)*.

2.3.8. *HLS Streaming*

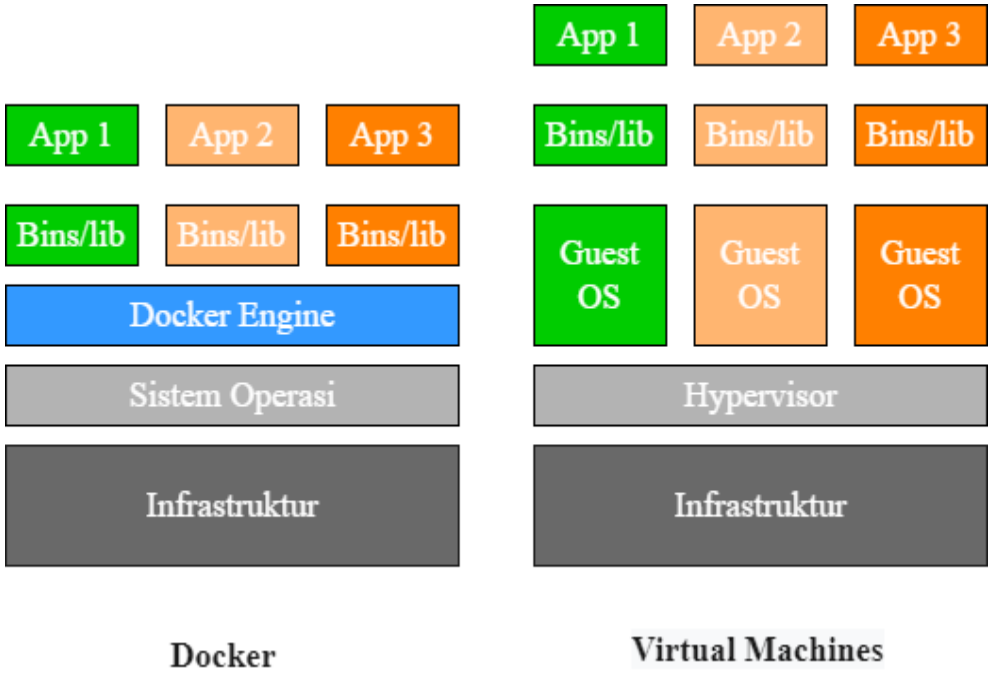
HTTP Live Streaming atau *HLS Streaming* yaitu *protocol video live streaming* yang menggunakan *HTTP* [16]. *HLS Streaming* mentransmisikan data melalui *HTTP* dari *server website* yang sering digunakan untuk ditonton pada *browser* pengguna. *HLS* dibangun untuk kestabilan dan dapat menyesuaikan secara dinamis dengan keadaan jaringan yaitu memaksimalkan penayangan pada jaringan kabel dan nirkabel.

2.3.9. *Docker*

Docker yaitu *software* atau *tools* yang mempermudah pembuatan perangkat lunak dengan menyusun beberapa *services* atau layanan, misalnya *database* dan lain-lain yang dikemas dalam sebuah *container* atau wadah [17]. Saat menggunakan virtualisasi, sebuah sistem operasi harus benar-benar siap karena *server* membutuhkan banyak sumber daya untuk beberapa virtualisasi. Ketika menggunakan *container*, menggunakan *container* dapat menghemat sumber daya karena ukuran file akan lebih kecil dibandingkan dengan metode virtualisasi yang konvensional.

2.3.10. Perbedaan *Docker* dan *Virtual Machines*

VM menggunakan level *virtualisasi hardware* sedangkan *docker* menggunakan level *virtualisasi* sistem operasi, dimana *docker* membagi sistem operasi *kernel* sedangkan *VM* membaginya di *hardware*, hal ini lah yang menyebabkan *docker* terlihat lebih efisien daripada menggunakan *VM* [18]. Untuk lebih jelasnya dapat dilihat pada Gambar 2.1.



Gambar 2. 1 Perbedaan *Docker* dan *Virtual Machines*

Dibawah ini adalah beberapa perbedaan mendasar dari *docker* dan *virtual machines*:

1) Arsitektur: *Virtual machine* mengisolasi lingkungan operasi secara lengkap, dengan memuat sistem operasi lengkap (*OS*), aplikasi, dan *library* di dalam setiap *instance VM*. Sedangkan *Docker* menggunakan konsep *container*, dimana hanya aplikasi dan dependensinya saja yang diisolasi dan dijalankan di atas *host OS* yang sama.
2) Penggunaan Sumber Daya: *Virtual machine* membutuhkan sumber daya yang lebih besar dibandingkan dengan *Docker*. Karena setiap *instance VM* membutuhkan sistem operasi, *memory*, *CPU*, *storage* dan sumber daya lainnya, yang menyebabkan penggunaan sumber daya fisik yang lebih besar. Di sisi lain, *Docker* hanya membutuhkan sumber daya yang lebih sedikit, karena menggunakan *host OS* yang sama untuk menjalankan *container*.
3) Kecepatan: Karena *virtual machine* memuat *OS* lengkap, maka penggunaan sumber daya dan waktu yang dibutuhkan untuk membuat dan menjalankan *instance VM* lebih lama daripada *Docker container*, yang hanya memuat aplikasi dan dependensinya.

2.3.11. *Docker Compose*

Docker compose merupakan alat yang dibuat oleh *docker* yang digunakan untuk membuat satu atau lebih *services* dalam satu *configuration* atau *template* yang berekstensi *.yaml* [19]. Secara sederhana, ini seperti menggabungkan seluruh *Dockerfile* dari masing-masing *services* dalam satu *file yaml (docker-compose file)*, dan dengan satu perintah, *user* dapat menciptakan dan menjalankan seluruh layanan yang sudah dibuat dalam *file yml* tersebut.

2.3.12. *Google Cloud Platform*

Google memiliki produk layanan komputasi awan yang dikenal sebagai *Google Cloud Platform (GCP)*. *GCP* menghadirkan *G-Suite enterprise* versi *Android* dan *Chrome*, serta antarmuka pemrograman aplikasi untuk layanan pemetaan perusahaan dan *machine learning* [20].

Dibawah ini ada beberapa contoh Fitur-Fitur *Google Cloud Platform*:

1) *Google Compute Engine*

   Service ini adalah mesin virtual atau *virtual machine* untuk pengguna yang disediakan oleh layanan (*IaaS*). Selain itu, mesin ini dapat mempercepat proses pengembangan, memiliki penyimpanan *disk* yang kuat, dan performa yang konsisten. *Server* virtual dapat dikonfigurasi dengan berbagai metode, seperti dengan *size* yang telah dibuat sebelumnya atau dengan membuat jenis mesin khusus yang disesuaikan dengan keperluan *user*.

2) *Google Container Engine*

   Fitur selanjutnya yang disediakan oleh *Google Cloud Platform* adalah *Google Container Engine*. *Service* ini adalah untuk *container orchestration* yang *running* di *Google Cloud Platform*, contohnya adalah *Google Kubernetes*, yang memiliki potensi untuk meningkatkan produktivitas pengembang, berfungsi sebagai fondasi untuk Mesin Wadah *Google* itu sendiri. Selain itu, *service* ini dapat meningkatkan fleksibilitas dan efisiensi sumber daya. *Open source* untuk mempersingkat waktu aplikasi ke *client*.

3) *Google Cloud Storage*

   Service ini adalah penyimpanan data di *cloud*, sehingga *user* tidak perlu tempat penyimpanan fisik yang besar pada *device* seperti *smartphone* atau *computer*, namun *user* tetap dapat mengakses data di *cloud* kapan saja dan dari mana saja.. *Cloud Datastore* pada *storage non-relasional NoSQL*, *Cloud SQL* pada *storage relasional* penuh *MySQL*, dan *database Cloud Bigtable Google* adalah semua opsi penyimpanan database yang ditawarkan oleh *Google*.

2.3.13. *Streamlabs*

Streamlabs adalah aplikasi untuk *video live streaming* bagi para *content creator*. Aplikasi ini tersedia dalam versi yang berbayar dan gratis. *Streaming* layar atau siarkan kamera pengguna ke *platform* media sosial seperti *Twitch, YouTube, dan Facebook* saat bermain *game seluler* juga bisa dilakukan dengan aplikasi ini. Pengguna bisa melalukan percakapan dengan banyak orang dari seluruh dunia sambil menonton permainan favorit mereka dan melakukan *streaming* langsung petualangan harian mereka secara *real time* dengan fitur yang sama yang ada di perangkat seluler seperti *Streamlabs Desktop*. Selain itu, ini kompatibel dengan *widget Streamlabs* seperti *Eventist, Alert Box, dan Chat Box* [21].

2.3.14. *Wireshark*

Wireshark adalah *software* gratis yang dapat digunakan untuk menangkap dan memindai lalu lintas data di jaringan internet [22]. *Wireshark* mendukung berbagai format *file* untuk menangkap data dan melacak paket, termasuk *.pcapng* dan *.libpcap*. Selain itu, *Wireshark* juga dilengkapi dengan alat dekripsi yang dapat menampilkan paket yang dienkripsi dari berbagai *protocol* jaringan internet, seperti *WEP* dan *WPA/WPA2* yang digunakan saat ini.

2.3.15. *TIPHON*

Standar *TIPHON* (*Telecommunications and Internet Protokol Harmonization over Network*) untuk parameter *QoS* dikembangkan oleh *ETSI* (*European Telecommunications Standart Institute*). Tujuan *ETSI* yaitu membantu mengembangkan standar telekomunikasi dan aspek multimedia lainnya antara *user network* [23]. *ETSI (Eeuropean Telecommunications Standart Institude)* adalah sebuah organisasi di Eropa yang bertanggung jawab untuk pembentukan standar telekomunikasi teknis yang didirikan pada tahun 1988.

2.3.15. Quality of Service (QoS)

Quality of Service (QoS) yaitu cara untuk menguji kualitas layanan suatu jaringan [24]. Seorang *network administrator* dapat mengutamakan *traffic* tertentu menggunakan *QoS*. Menampilkan *Quality of Service* yang tidak sama berdasarkan keperluan layanan jaringan adalah tujuan dari *QoS*. Parameter *Quality of Service* terdiri dari *Throughput*, *Packet Loss*, dan *Delay*. Berdasarkan penelitian yang dilakukan oleh [25] menghitung rumus dari ke-3 parameter tersebut adalah pada Penomoran Rumus (2.1), Penomoran Rumus (2.2), dan Penomoran Rumus (2.3):

1) *Throughput*

Throughput adalah *speed* (rate) *transfer* data yang aktual (nyata) dan dihitung dalam satuan bps (bit per second).

$$\textit{Throughput} = \frac{\text{Jumlah paket yang diterima (bytes)}}{\text{waktu pengiriman (second)}} \text{ X 8 (kbps)} \quad (2.1)$$

2) *Packet Loss*

Packet Loss yaitu parameter yang menjelaskan suatu keadaan di mana terdapat jumlah paket yang *loss* dalam jaringan.

$$\textit{Packet loss} = \frac{\text{Jumlah paket yang dikirm – paket yang diterima}}{\text{jumlah paket yang dikirim}} \text{ X 100\%} \quad (2.2)$$

3) *Delay (Latency)*

Delay (Latency) adalah proses pengiriman dan penerimaan data yang tertunda dalam jaringan.

$$\text{Rata – Rata } \textit{Delay} = \frac{\text{Total } \textit{Delay}}{\text{Total Paket yang diterima}} \quad (2.3)$$