

BAB II

TINJAUAN PUSTAKA DAN LANDASAN TEORI

2.1 Tinjauan Pustaka

Yudi Methanoxy, skripsi .(2010): Analisa QOS Radio *Streaming* Pada Local Community Network, aspek yang dibahas dalam skripsi ini adalah dipaparkannya percobaan dan analisa dari implementasi radio *streaming* pada Local Community Network dengan perangkat 802.11n. Dengan metode pengamatan kelayakan yang dilakukan dengan mengacu kepada rekomendasi G.114 ITU-T, dan juga dilakukan penelitian lebih mendalam mengenai maksimal *user* serta parameter pengaturan *encoder* yang paling baik untuk digunakan. Skripsi Yudi Methanoxy sangat membantu penulis dalam memahami metode dan tahapan-tahapan yang harus dipakai dalam pengujian yang dilakukan untuk aplikasi radio *streaming*.

Puput, Eddy, Henry .(2011) dalam jurnal yang berjudul Implementasi Radio Kampus Pada Jaringan Lokal Politeknik Telkom yang membahas tentang implementasi radio *streaming* pada jaringan local serta menjelaskan betapa pentingnya radio *streaming* ada sebagai sarana pengirim informasi yang dapat meningkatkan efisiensi dan efektivitas kerja dengan hasil yang maksimal. Jurnal ini sangat membantu bagi penulis untuk memahami betapa penting radio *streaming* ada pada kampus sebagai media penghantar informasi yang bisa meningkatkan efektivitas dan mobilitas dalam

menerima informasi untuk mahasiswa. Dalam jurnal ini juga dijelaskan metode penelitian yang dilakukan sebagai referensi metode penelitian yang penulis lakukan.

Fitria, Agus .(2009) Dalam SNATI yang berjudul “Internet Radio *Streaming*” membahas tentang internet radio *streaming* yang merupakan sebuah teknologi yang mampu mengompresi atau menyusutkan ukuran file audio dan video agar mudah ditransfer dalam jaringan internet. Jurnal ini membantu penulis dalam memahami pengertian dan metode penelitian internet radio *streaming*.

2.2 Landasan Teori

2.2.1 Konsep Kerja *Streaming*

Streaming adalah sebuah teknologi pengiriman sebuah file melalui jaringan internet secara terus menerus. File yang umum dikirim melalui *streaming* adalah video dan audio, pertama kali *client* meminta sebuah file yang akan di *streaming* ke server, lalu server mulai mengirim file yang diminta dengan memecah nya menjadi paket-paket kecil yang dikirim melalui media internet, lalu di bagian *client* mulai menerima file dan membangun *buffer* yang mana mulai menyimpan informasi sedikit demi sedikit dari keseluruhan file, ketika proses *buffering* masih berjalan dan *buffer* di komputer *client* sudah penuh secara otomatis file dimainkan oleh sistem. Sistem akan membaca informasi dari *buffer* dan tetap melakukan proses menerima file dari server sampai proses *streaming* berakhir.



Gambar 2. 1 Analogi *Streaming*

Untuk menjelaskan hal ini bisa diibaratkan dengan proses keran yang mengalirkan air ke ember dibawahnya. Keran air yang berperan sebagai *server*, air berperan menjadi data, dan ember penerima berperan menjadi *client*. Keran air yang bertindak menjadi *server* mengirimkan air yang bertindak sebagai data yang mengirimkan aliran data ke ember yang dibawah nya secara terus-menerus. Demikian pula selama ada data yang dialirkan maka proses *streaming* akan terus berlanjut, untuk lebih jelasnya bisa dilihat pada gambar 2.1.

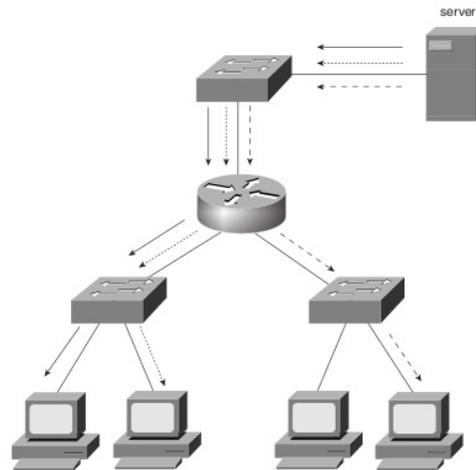
2.2.2 Metode transmisi data

Transmisi adalah proses pengangkutan informasi dari suatu titik ke titik lain yang berada dalam jaringan. Komunikasi dalam jaringan komputer umumnya hanya terjadi antara dua komputer. Dalam perkembangannya pengguna jaringan mulai

merasakan kebutuhan untuk melakukan komunikasi yang melibatkan lebih dari dua pihak secara bersamaan. Beberapa aplikasi dalam jaringan membutuhkan komunikasi banyak titik (*multipoint*), salah satunya adalah *audio broadcast*, *audio broadcast* mengirimkan data *audio* dari satu titik ke banyak titik. Dalam metode nya ada beberapa cara untuk melakukan transmisi data yaitu, *unicast*, *multicast*, dan *broadcast*.

2.10.1 Unicast

Transmisi *unicast* merupakan transmisi informasi yang menghubungkan antara titik yang satu dengan yang lainnya saja (*point to point*). Setiap komunikasi atau transmisi data yang menuju ke suatu titik atau alamat *unicast* yang lainnya akan memakai satu jalur data, sehingga jika berkomunikasi dengan beberapa titik sekaligus akan menambah jalur data yang dipakai, otomatis jalur jaringan akan menjadi padat oleh lalu-lintas data, apalagi jika yang diakses adalah file multimedia yang umumnya berukuran cukup besar. Hal ini sering kali menyebabkan permasalahan *skalabilitas* pada penerapan metode transmisi *unicast*.

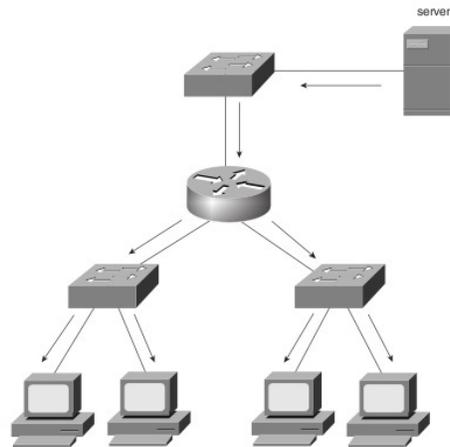


Gambar 2. 2 Transmisi *Unicast*

Bisa dilihat digambar 2.2 setiap *client* dapat meminta data dari *server* dan akan dibuatkan jalur sendiri, jika ada *client* lain yang meminta data walaupun data nya sama tetap akan dibuatkan jalur data yang berbeda.

2.10.2 Broadcast

Transmisi *broadcast* merupakan metode pengiriman data ke semua node yang ada dalam satu jaringan yang sama. *Server* akan mengirimkan data ke semua PC walaupun hanya satu PC yang meminta layanan saja yang akan memproses datanya sedangkan PC yang lain akan mengabaikan data tersebut. Masalah akan besar bila file yang dikirimkan mempunyai ukuran yang cukup besar, maka jalur yang seharusnya dipakai untuk lalu-lintas data lain menjadi terhambat oleh sesuatu yang tidak diinginkan oleh *user* tersebut.



Gambar 2. 3 Transmisi *Broadcast*

Bisa dilihat di gambar 2.3 contoh nya adalah salah satu *client request* sebuah data ke *server* maka *server* akan merespon dengan mengirimkan data tersebut ke semua *client* yang terhubung dalam satu jaringan, walaupun *client* yang lain tidak memintanya.

2.10.3 *Multicast*

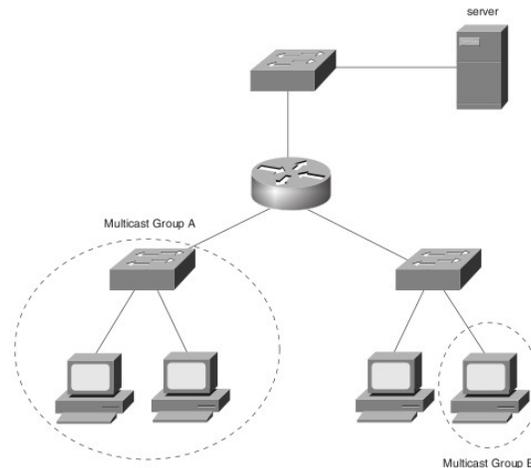
Multicast merupakan mekanisme pengiriman aliran paket data dari satu sumber ke suatu group yang berisi kumpulan *host* penerima. Keuntungan utama dari *IP multicast* adalah kemampuannya untuk penghematan *bandwidth*. Ini dikarenakan sumber *IP multicast* hanya perlu mengirimkan aliran data ke sebuah group yang berisi *host* penerima melalui satu jalur saja. Hal ini mengakibatkan pengiriman data lebih efektif dibanding dengan *broadcast*, dan dapat diterima jauh lebih banyak dibandingkan dengan cara *unicast*.

Alamat-alamat *multicast* IPv4 didefinisikan dalam ruang alamat kelas D, yakni 224.0.0.0/4, yang berkisar dari 224.0.0.0 hingga 224.255.255.255. Prefiks

alamat 224.0.0.0/24 (dari alamat 224.0.0.0 hingga 224.0.0.255) tidak dapat digunakan karena dicadangkan untuk digunakan oleh lalu lintas *multicast* dalam subnet lokal¹.

Contoh yang menggunakan pendekatan menggunakan metode *multicast* antara lain adalah pembelajaran jarak jauh, pengiriman serentak, informasi saham dan video konferensi. Dan keuntungan menggunakan *IP multicast* adalah:

1. Mengurangi beban disisi pengirim karena hanya mengirimkan satu informasi ke suatu kelompok *multicast*.
2. Memungkinkan untuk mengirimkan suatu informasi serempak kepada banyak penerima.
3. Efisien dalam penggunaan *bandwidth* jaringan.



Gambar 2. 4 Transmisi *Multicast*

Pada gambar 2.4 adalah sebuah contoh gambar pengiriman data transmisi *multicast*. Data dari *server* akan dikirimkan ke salah satu group, bisa ke group A ataupun group

¹ <http://id.wikipedia.org> Diakses tanggal 18 november 2014

B. Jika data hanya dikirimkan ke group A, maka group B tidak akan menerima data dari *server*.

2.2.3 TCP/IP

Inti dari komunikasi adalah sebuah interaksi yang dilakukan oleh dua orang atau lebih yang saling bertukar informasi, tidak jauh berbeda dalam dunia komputer hanya saja object interaktor nya adalah komputer, untuk dapat berkomunikasi pada komputer harus ditambahkan perangkat khusus yang dinamakan *network interface*, bentuk nya bermacam sesuai fungsi dan kegunaan nya.

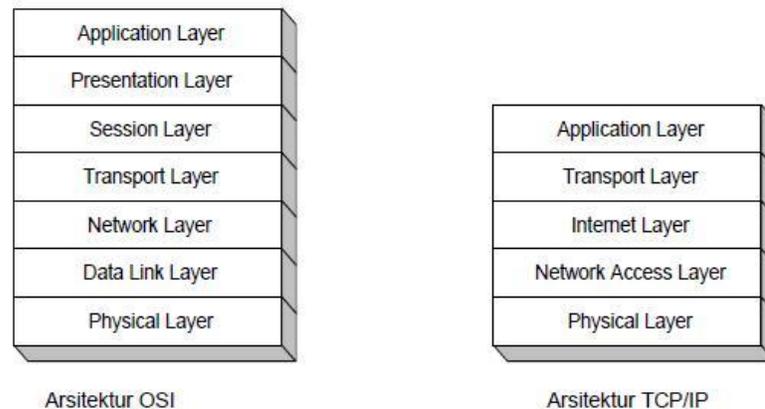
Dalam proses berkomunikasi ada beberapa aspek yang harus diperhatikan seperti jarak dan tujuan. Data haruslah dikirim ketujuan yang tepat, dalam hal jarak pengiriman, data atau informasi yang dikirim tidak akan terlalu susah untuk sampai ke tujuan jika terjadi di jaringan local, berbeda hal nya jika tujuan penerima penerima berada di tempat yang jauh (secara jaringan), kemungkinan data atau informasi yang hilang maupun rusak lebih besar. Tujuan sendiri menjadi faktor yang harus diperhatikan, dalam proses pengiriman data atau informasi harus mencapai tujuan yang tepat, karena bisa saja bukan hanya satu aplikasi yang menunggu kiriman data itu, jadi intinya adalah dalam pengiriman data atau informasi yang terpercaya harus sampai pada komputer yang tepat tanpa kesalahan.

Metode paling mudah untuk memecahkan masalah yang rumit adalah dengan membaginya menjadi bagian-bagian kecil agar lebih mudah dipecahkan. Para ahli jaringan komputer membuat rumusan masalah-masalah diatas menjadi sebuah aturan-

aturan dalam berkomunikasi data. Sekumpulan aturan berkomunikasi data ini disebut *protocol* komunikasi data. *Protocol* adalah sekumpulan aturan standar yang harus diikuti agar bisa terjadi suatu hubungan, komunikasi, dan perpindahan data yang bisa menghubungkan antara dua komputer atau lebih dalam jaringan. TCP/IP adalah sekumpulan *protocol* yang mempunyai tugas masing-masing yang mewujudkan komunikasi antara dua komputer didalam jaringan, Keseluruhan aturan ini harus bekerja sama antara yang satu dengan lainnya. Dengan adanya standarisasi aturan pengiriman data, tugas-tugas setiap *protocol* menjadi jelas dan mudah, semua *protocol* bekerja bersama-sama tanpa harus mengetahui kerja *protocol* yang lain nya.

2.2.3.1 Arsitektur dan *Protocol* Jaringan TCP/IP

Dalam arsitektur jaringan terdapat lapisan-lapisan atau *layer* yang mempunyai tugas yang spesifik. *International Organization for Standardization* (ISO) mengeluarkan suatu standar dalam berkomunikasi dalam jaringan komputer yang dikenal dengan nama *Open Systems Interconnection model* (OSI). *Standard* ini terdiri dari tujuh lapis *protocol*. Dalam TCP/IP hanya terdapat lima lapisan berikut.



Gambar 2. 5 OSI dan TCP/IP

Walaupun berbeda *layer* tetapi arsitektur TCP/IP telah mencukupi semua fungsi dari arsitektur OSI. Fungsi masing masing *layer* arsitektur TCP/IP adalah sebagai berikut.

a. Network Access Layer

Berhubungan dengan pertukaran data antara sistem akhir dan jaringan tempatnya melekat. Lapisan ini menggambarkan bagaimana data dikodekan menjadi sinyal-sinyal dan karakteristik antarmuka tambahan media. Dengan demikian lapisan ini bertanggung jawab menerima dan mengirim data dan dari media fisik. Media fisiknya dapat berupa kabel, serat optik, atau gelombang radio. Karena tugasnya ini, protokol yang ada di layer ini harus mampu menerjemahkan sinyal listrik menjadi data digital yang dapat dimengerti oleh komputer, yang berasal dari peralatan lain yang sejenis.

b. *Internet Layer*

Layer ini setara dengan layer Network dalam OSI, yaitu menyediakan fungsi *routing* sehingga paket dapat dikirim keluar dari segment network lokal ke suatu tujuan yang berada pada suatu network lain. *IP, Internet Protocol*, umumnya digunakan untuk tugas ini. *Internet protocol* bertugas untuk pengalamatan pengiriman data di internet. Setiap komputer yang dikenal sebagai *host* di internet mempunyai sebuah alamat *IP* yang unik untuk yang membedakan dari komputer lain dalam jaringan internet.

c. *Transport Layer*

Lapisan ini bertugas mendefinisikan cara-cara untuk pengiriman *end to end host* secara handal. Lapisan ini menjamin bahwa data yang dikirim bebas dari kesalahan saat pengiriman. Untuk itu lapisan ini memiliki beberapa fungsi penting antara lain.

1. *Flow Control*

Pengiriman data yang telah dipecah-pecah diatur sedemikian rupa agar pengirim tidak sampai mengirimkan data dengan kecepatan yang melebihi kemampuan penerima dalam menerima data.

2. *Error Detection*

Pada saat pengiriman data diberikan sebuah informasi yang bisa digunakan untuk memeriksa apakah data yang dikirimkan bebas dari kesalahan. Jika ditemukan kesalahan maka data tidak akan di

terima dan pihak pengirim akan mengirim kembali data informasi yang salah. Tetapi hal ini memberikan kelemahan karena dapat menimbulkan *delay* dikarenakan proses yang harus diulang.

Pada TCP/IP, *protocol* yang dipergunakan adalah *Transmission Control Protocol* (TCP) atau *User Datagram Protocol* (UDP). TCP dipakai ketika membutuhkan koneksi yang terpercaya karena sifatnya yang *connection oriented* dan memiliki fungsi *flow control* dan *error detection*, TCP dipakai untuk aplikasi-aplikasi yang membutuhkan kehandalan data. Sedangkan UDP dipakai ketika kecepatan adalah yang diutamakan karena sifatnya yang *connectionless* jadi tidak ada pemeriksaan data dan *flow control*, sehingga UDP disebut *unreliable protocol*. Untuk beberapa hal yang menyangkut penyerdehanaan dan efisiensi, beberapa aplikasi memilih UDP sebagai pilihan.

d. *Application Layer*

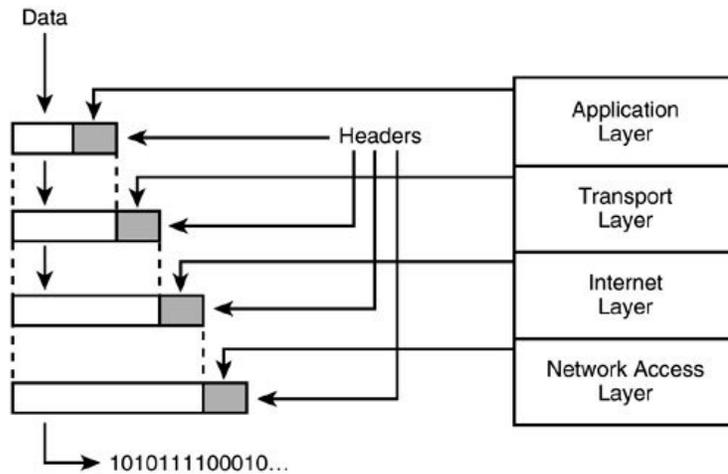
Merupakan lapisan terakhir dalam arsitektur TCP/IP yang berfungsi mendefinisikan aplikasi-aplikasi yang dijalankan pada jaringan. Karena itu, terdapat banyak protokol pada lapisan ini, sesuai dengan banyaknya aplikasi TCP/IP yang dapat dijalankan. Contohnya adalah SMTP (*Simple Mail Transfer Protocol*) untuk pengiriman e-mail, FTP (*File Transfer Protocol*) untuk transfer file, HTTP (*Hyper Text Transfer Protocol*) untuk aplikasi web, NNTP (*Network News Transfer Protocol*) untuk distribusi *news group* dan lain-lain. Setiap aplikasi pada umumnya menggunakan

protokol TCP dan IP, sehingga keseluruhan keluarga protokol ini dinamai dengan TCP/IP.

2.2.4 Proses Pengiriman Data

Setiap lapisan dalam TCP/IP memiliki fungsi-fungsi dalam komunikasi antara dua buah komputer sesuai dengan fungsi *protocol* tersebut. Jika suatu *protocol* menerima data dari lapisan atasnya, maka *protocol* tersebut akan menambahkan sebuah informasi yang relevan sesuai dengan fungsi *protocol* nya yang disebut *header* dengan metode membungkus data tersebut menjadi satu dengan informasi yang ditambahkan dari setiap lapisan. Lalu proses di lanjutkan ke lapisan dibawahnya untuk menambahkan informasi yang dibutuhkan sesuai fungsi dari lapisan yang dilewati.

Sebaliknya jika suatu *protocol* menerima suatu data dari lapisan *protocol* dibawahnya, maka setiap informasi yang dipunya akan dilepas satu-persatu sampai data itu sampai pada komputer penerima. Jadi dalam komunikasi data, data akan tetap hanya saja informasi yang berhubungan untuk pengiriman data tersebut akan berkurang atau bertambah.



Gambar 2. 6 Proses Pengiriman Data

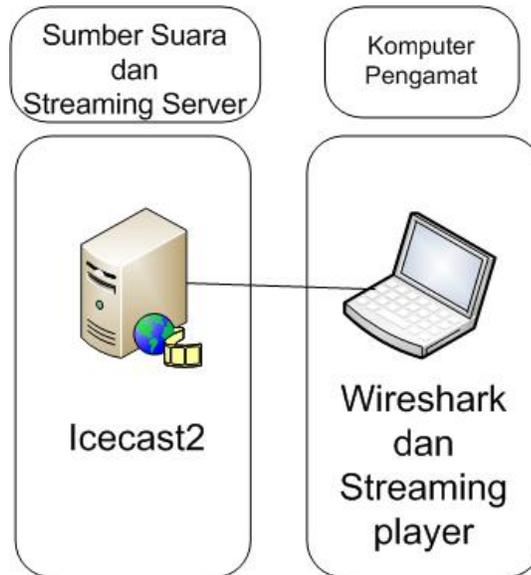
Dari lapisan *transport* data yang telah diberi *header* diteruskan ke lapisan *network*, di lapisan *network* data diberikan penambahan *header* yang berisi alamat tujuan, alamat pengirim dan informasi lain yang dibutuhkan untuk melakukan pengalamatan ke tujuan, kemudian terjadi pengarahannya *routing* data. Pada lapisan ini juga dapat terjadi *segmentasi* data, karena panjang paket yang akan dikirimkan harus disesuaikan dengan kondisi media komunikasi pada jaringan yang akan dilalui. Selanjutnya data menuju *Network Access Layer* dimana data akan diolah menjadi *frame-frame*, *protocol* pada lapisan ini menyiapkan data dalam bentuk yang paling sesuai untuk dikirimkan melalui media komunikasi tertentu. Terakhir data akan sampai pada *Physical Layer* yang akan mengirimkan data dalam bentuk besaran listrik seperti tegangan, arus, gelombang radio, sesuai media yang digunakan.

Pada bagian penerima alur proses yang terjadi berlawanan dengan proses pengiriman yaitu alurnya bergerak dari bawah keatas. Pada bagian *Physical Layer* data akan diubah bentuknya kedalam data, *protocol* akan memeriksa data tersebut,

jika tidak ditemukan *error* maka *header* yang terpasang akan dilepas dan data akan diteruskan ke lapisan *network*, pada lapisan *network* alamat tujuan akan diperiksa, jika memang data tersebut tujuannya benar maka *header* lapisan *network* akan dilepas dan data akan diteruskan ke layer selanjutnya, tetapi bila salah maka data akan dikirim ke tujuan nya, sesuai dengan informasi yang dimiliki. Lalu jika alamat yang dituju sudah benar, data akan dilanjutkan ke lapisan *transport*, data akan diperiksa kembali dengan informasi yang tersimpan di *header* data. Jika tidak ada kesalahan, paket-paket yang diterima akan disusun kembali sesuai urutan nya pada saat dikirim dan diteruskan pada lapisan aplikasi pada penerima.

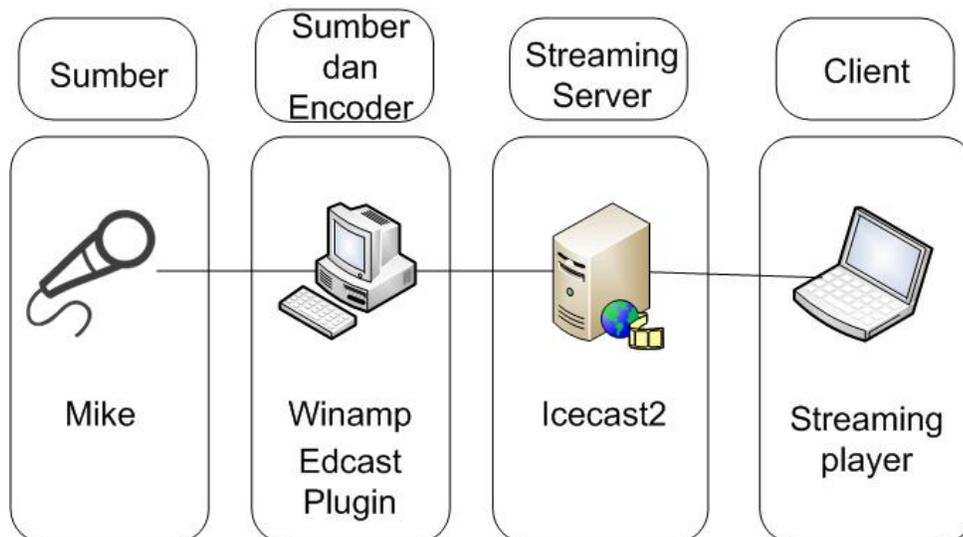
2.2.5 Skema Radio Streaming

Radio *streaming* sendiri dapat dibedakan menjadi dua skema, perbedaannya hanya terdapat pada sumber suara berasal, penggunaannya disesuaikan dengan kebutuhan masing-masing. Hal ini menjadi mungkin dengan adanya fitur Auto DJ di *server streaming*, yaitu memainkan secara otomatis playlist rekaman suara melalui *server streaming*. Pada gambar 2.7 adalah contoh skema radio *streaming* yang pertama, *server streaming* juga dipakai sebagai sumber suara. Pada gambar 2.8 adalah contoh lain dari skema dari radio *streaming*, sumber suara bisa berasal dari sebuah alat input suara seperti mike maupun melalui sebuah file *audio* yang dimainkan melalui komputer penyiar, metode ini biasanya digunakan untuk siaran langsung radio *streaming*.



Gambar 2. 7 Skema 1

Pada gambar 2.7 *server* juga digunakan sebagai *PC* penyiar. Sumber suara berasal dari file MP3 yang dimainkan sesuai urutan *playlist* melalui *server*.



Gambar 2. 8 Skema 2

Pada gambar 2.8 *source audio* bisa berasal dari mike maupun dari komputer sumber.

2.2.6 Parameter Dalam *Audio Streaming*

Dalam *audio streaming* ada beberapa parameter yang harus diperhatikan yaitu *Bit rate* dan *Sample rate*. Kedua parameter ini penting karena dapat mempengaruhi kualitas dari sebuah file *audio streaming*.

1. *Bit Rate*

Bit adalah informasi terkecil yang berbentuk digital, satuan data yang didasari oleh penghitungan bilangan *biner*. Bilangan *biner* adalah bilangan yang digunakan untuk proses data di peralatan elektronik, diwakili oleh serangkain angka 1 dan 0. *Bit rate* adalah suatu ukuran kecepatan *transfer* data yang terkompres dalam suatu waktu. Semakin besar *bit rate*, maka kualitas suara semakin bagus tetapi membutuhkan *bandwidth* yang besar pula.

Seperti yang telah di jelaskan sebelumnya, *bit* adalah informasi terkecil yang berbentuk digital, besaran *bit* sangat berpengaruh terhadap kualitas dan ukuran dari sebuah file *audio*. Untuk memudahkan dapat mengambil contoh dalam penyimpanan gambar yang bilangan *bit* nya berbeda, ini dapat dilakukan karena dalam dunia digital file *audio* dan gambar disimpan secara digital pada komputer yang diwakili oleh bilangan *biner*.

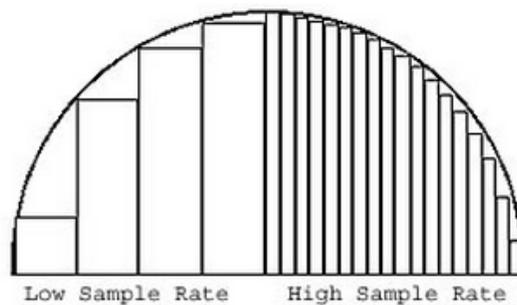


Gambar 2. 9 Perbedaan kualitas gambar

Bisa dilihat pada gambar 2.9 perbedaan hasil kualitas gambar yang disimpan dalam *bit* yang berbeda, gambar A pada gambar 2.9 tersimpan dalam 24 *bit* terlihat lebih jelas dan detail, sedangkan untuk gambar B pada gambar 2.9 yang tersimpan dalam 16 *bit* terasa kurang jelas dan kurang detail. Sekarang terjemahkan kedalam dunia *audio digital*. *Audio* yang direkam pada *bit* yang rendah akan terasa kurang jelas dan kasar. Dan merekam *audio* pada 24 *bit* akan menghasilkan sebuah kualitas suara yang lebih tinggi sehingga menghasilkan suara yang lebih jernih dan jelas. Dengan menyimpan *audio* pada *bit* yang lebih tinggi maka akan menghasilkan file suara dengan ukuran yang lebih besar. Dalam dunia *streaming*, mengalirkan sebuah suara dengan *bit rate* yang besar akan menghasilkan kualitas suara yang bagus tetapi berakibat pada kebutuhan *bandwidth* yang besar.

2. *Sample Rate*

Sample rate adalah jumlah “tangkapan” atau suara yang tertangkap dari audio yang direkam setiap detiknya. *Stream audio* secara terus menerus dikodekan secara digital seperti kamera film yang menangkap objek yang bergerak. Sebagai ilustrasi adalah sebuah kamera film yang dapat menangkap gerakan object yang bergerak sebanyak 30 gerakan per detik, dan object bergerak itu bergerak 60 gerakan perdetik, maka hanya terekam 30 gerakan perdetik yang menghasilkan kualitas gambar yang kurang bagus dan kurang detail. Dengan meningkatkan *frame rate* nya menjadi lebih tinggi misal 60 gerakan perdetik maka akan dihasilkan sebuah gerakan asli yang berkualitas seperti object yang direkam. Begitupula dengan suara, semakin tinggi *sample rate* akan mewakili sumber bunyi secara tepat dan akurat.



Gambar 2. 10 Contoh gambar perbandingan *sample rate*

Pada gambar 2.10 adalah ilustrasi perbedaan *sample rate* yang tinggi dengan yang rendah yang di presentasikan dengan blok-blok dan suara sumber yang dipresentasikan dengan kurva. Dapat dilihat pada gambar 2.10 perbedaan kualitas suara yang dihasilkan dari perbandingan *sample rate* yang digunakan. Dalam penggunaan *sample rate* yang rendah akan menghasilkan kualitas suara yang tidak jernih dan kurang detail dikarenakan banyak sekali *gap* informasi yang tidak terpenuhi, sedangkan jika menggunakan *sample rate* yang tinggi hanya sedikit sekali *gap* celah yang ada, sehingga menghasilkan suara yang detail dan jernih hampir sama dengan suara sumbernya.

2.2.7 Arsitektur Radio Streaming

2.2.7.1 Streaming media server

Streaming media server adalah aplikasi yang berfungsi sebagai *repeater* untuk mengirimkan *streaming* audio dari komputer penyiar kepada pendengar melalui internet. *Streaming media server* dipasang di *server* yang biasanya menggunakan linux sebagai sistem operasinya. *Streaming media server* banyak macamnya, yang

paling populer contohnya adalah SHOUTCAST dan ICECAST. Fungsi dari keduanya sama yaitu aplikasi yang bertugas untuk melewatkan file *audio* yang dikirim dari komputer penyiar kepada pendengar melalui internet. Perbedaan mencolok diantara keduanya adalah penggunaan *port* dan *mount point*. Dengan ICECAST, dapat menyiarkan siaran yang berbeda melalui *port* yang sama dengan *mount point* yang berbeda.

2.2.7.2 Broadcast Tool

Selain disisi *server*, komponen yang harus ada adalah *broadcast tool*. Modul ini harus terpasang di komputer penyiar. *Broadcast tool* berfungsi sebagai alat untuk mengubah atau mengkonversi masukan (input) suara yang berbentuk analog ke format *audio* digital agar dapat ditangkap oleh *streaming media server* untuk kemudian disiarkan ke seluruh dunia melalui internet. Sama seperti *streaming media server*, *broadcast tool* juga banyak macam-macamnya, untuk komputer dengan sistem operasi windows dapat menggunakan edcast atau DSP plug-in. Sedangkan untuk komputer dengan sistem operasi Linux ataupun Mac dapat memasang modul BUTT (*Broadcast Using This Tool*).

2.2.7.3 Pendengar

Ada beberapa cara yang dapat dilakukan untuk mendengarkan siaran radio *streaming*, yang pertama adalah menggunakan sebuah browser, yang kedua dengan

menggunakan audio player seperti winamp, dengan cara memasukkan alamat, *port* dan *mount-point* sesuai dengan siaran yang ingin didengarkan.

2.2.8 QoS (*Quality of Service*)

Quality of Service (QoS) adalah kemampuan menyediakan jaminan dan performansi layanan pada suatu jaringan. QoS sebagai bentuk suatu ukuran atas tingkatan layanan yang disampaikan ke *client*. Layanan multimedia *streaming* terutama *audio streaming* merupakan salah satu jenis aplikasi internet yang sering diakses oleh user. *Audio streaming* adalah layanan multimedia yang membutuhkan layanan yang handal, karena membutuhkan *bandwidth* yang tinggi serta *delay* yang kecil agar menghasilkan layanan yang dapat dinikmati secara interaktif dan juga inti dari proses *streaming* adalah pengiriman file multimedia harus tiba ditujuan dengan tepat tanpa ada gangguan . Metode pengujian QoS dengan parameter *delay*, *throughput* dan *packet loss* digunakan sebagai metode yang dipakai dalam penelitian penulis kali ini.

Pengujian QoS biasanya didasarkan beberapa parameter, yaitu *delay*, *packet loss* dan *throughput*.

a. Delay

Delay adalah waktu tunggu yang dibutuhkan untuk seluruh pesan sampai ditujuannya. *Delay* dapat dipengaruhi oleh jarak, media fisik, kongesti atau juga waktu proses yang lama. Menurut rekomendasi ITU

(*International Telecommunication Union*), besar nya *delay* dapat diklasifikasikan sebagai berikut.

Kategori Delay	Besar Delay
Sangat Bagus	< 150 ms
Bagus	150 ms s/d 300 ms
Sedang	300 ms s/d 450 ms
Buruk	> 450 ms

b. *Packet Loss*

Merupakan gambaran suatu kondisi yang menunjukkan jumlah total paket hilang yang dapat terjadi karena *collision* ataupun *congestion* dalam jaringan. Nilai *packet loss* sesuai dengan rekomendasi ITU (*International Telecommunication Union*) sebagai berikut.

Kategori Degredasi	Packet Loss
Sangat Bagus	0 %
Bagus	3 %
Jelek	15 %
Sangat Jelek	25 %

c. *Throughput*

Throughput adalah suatu ukuran yang merepresentasikan suatu ukuran kecepatan *bandwidth* yang diukur dalam rentang waktu tertentu. *Throughput* biasanya dalam satuan *kbps*, contohnya adalah ketika seorang

user yang mempunyai *bandwidth* 32 *kbps* dan ingin mengunduh sebuah file berukuran 64 *kb* dengan estimasi waktu unduh 2 detik, ternyata kenyataannya memerlukan waktu 8 detik. Jadi jika ukuran file yang diunduh adalah 64 *kb* yang memerlukan waktu 8 detik maka *bandwidth* yang sebenarnya atau *throughput* dapat dihitung dengan rumus ukuran file yang diunduh dibagi estimasi waktu keseluruhan waktu unduh sebuah file adalah

$$64 \text{ kb} / 8 \text{ detik} = 8 \text{ kbps.}$$