BAB II

LANDASAN TEORI

2.1 Quality of Service

Quality of Service (QoS) adalah kemampuan sebuah jaringan untuk menyediakan layanan yang lebih baik lagi bagi layanan trafik yang melewatinya. QOS merupakan sebuah sistem arsitektur *end to end* dan bukan merupakan sebuah *feature* yang dimiliki oleh jaringan. *Quality of Service* suatu *network* merujuk ke tingkat kecepatan dan keandalan penyampaian berbagai jenis beban data di dalam suatu komunikasi. *Quality of Service* digunakan untuk mengukur tingkat kualitas koneksi jaringan TCP/IP internet atau intranet (Ningsih, 2004).

Dari definisi diatas dapat disimpulkan QOS (*Quality of Service*) adalah kemampuan suatu jaringan untuk menyediakan layanan yang baik. Oleh karenanya buruk atau baiknya kualitas dan kemampuan suatu jaringan dapat kita ukur melalui unjuk kerja jaringan tersebut. Beberapa parameter yang dijadikan referensi umum untuk dapat mengukur dan melihat unjuk kerja dari suatu jaringan antara lain, *throughput, packet loss*, dan *fairness index.*

2.1.1 Throughput

Throughput adalah kemampuan sebenarnya suatu jaringan dalam melakukan pengiriman data. Biasanya *throughput* selalu dikaitkan dengan *bandwidth* dalam kondisi yang sebenarnya. *Bandwidth* lebih bersifat *fix* sementara *throughput* sifatnya adalah dinamis tergantung trafik yang sedang terjadi (Daryanto, 2010).

$$\textbf{\textit{Throughput}} = \frac{\textit{(jumlah data yang dikirim)}}{\textit{(waktu pengiriman data)}} \qquad (2.1)$$

2.1.2 Packet Loss

Packet loss dapat disebabkan oleh sejumlah faktor, mencakup penurunan signal dalam media jaringan, melebihi batas saturasi jaringan, kesalahan *hardware* jaringan. Beberapa *network transport* protokol seperti TCP menyediakan pengiriman paket yang dapat dipercaya. Dalam hal kerugian paket, penerima akan meminta *retransmission* atau pengiriman secara otomatis (*resends)* walaupun segmen telah tidak diakui. Walaupun TCP dapat memulihkan dari kerugian paket, *retransmitting* paket yang hilang menyebabkan *throughput* yang menyangkut koneksi dapat berkurang. Di dalam varian TCP, jika suatu paket dipancarkan hilang, akan menjadi *resent* bersama dengan tiap-tiap paket yang telah dikirim setelah itu. *Retransmission* ini menyebabkan keseluruhan *throughput* menyangkut koneksi untuk menurun jauh (Zenhadi, 2011).

$$\textbf{\textit{Packet loss}} = \frac{(packet\ transmitted - Packet\ received)}{packet\ transmitted} \text{ X } 100\% \qquad (2.2)$$

2.1.3 Fairness Index

Pengukuran *fairness* digunakan pada jaringan komputer untuk menentukan apakah *users* atau aplikasi telah menerima sumber daya yang adil. Sebuah matrik yang digunakan secara umum untuk menaksir *fairness* adalah Jain's *Fairness* Index (JFI) dengan persamaan (Bhatti, 2008).

$$\textbf{\textit{Fairness Index}} = \frac{(\Sigma\, x_i)^2}{n\Sigma\, x_i^{\,2}} \qquad (2.3)$$

Nilai Xi merupakan nilai *throughput* yang telah diukur, n merupakan jumlah jalur yang ada.

2.2 Load Balance

Load balance adalah teknik untuk mendistribusikan beban trafik pada dua atau lebih jalur koneksi secara seimbang, agar trafik dapat berjalan optimal, memaksimalkan *throughput*, memperkecil waktu tanggap dan menghindari *overload* pada salah satu jalur koneksi (Harry Nugroho, Moch Rezka Utama, 2014).

Selama ini banyak yang beranggapan salah, bahwa dengan menggunakan *load balance*, maka besar *bandwidth* yang diperoleh menjadi dua kali lipat dari *bandwidth* sebelum menggunakan load balance. Hal ini perlu diperjelas dahulu, bahwa *load balance* tidak akan menambah besar *bandwidth* yang diperoleh, tetapi hanya bertugas untuk membagi trafik dari kedua *bandwidth* tersebut agar dapat terpakai secara seimbang. Bahwa dalam menggunakan *load balance* tidaklah seperti rumus matematika 1+1=2, akan tetapi 1+1=1+1.

Dalam sistem *load balance*, proses pembagian bebannya memiliki teknik dan algoritma tersendiri. Pada perangkat *load balance* yang kompleks biasanya disediakan bermacam-macam algoritma untuk pembagian beban ini. Tujuannya adalah untuk menyesuaikan pembagian beban dengan karakteristik dari *server-server* yang ada dibelakangnya.

Dalam *load balance* di mikrotik, hal-hal yang perlu diperhatikan dalam mengatur *load balance* adalah *static route*, *policy route*, *firewall mangle*, dan *firewall src-nat*. *Static route* dan *policy route*r mengatur *uplink flow*, yaitu kebijakan *routing* yang akan dilalui paket yang telah ditandai, sedangkan *Firewall mangle* adalahpenandaan paket sebelum masuk *routing*.

Ada beberapa metode *load balance*, antara lain PCC dan Nth. Setiap metode

tersebut memiliki kekurangan maupun kelebihan tersendiri, namun lebih dari hal itu, yang paling terpenting dalam menentukan metode *load balance* apa yang akan digunakan adalah harus dimengerti terlebih dahulu karakteristik dari jaringan yang akan diimplementasikan. Berikut ini adalah sedikit pengertian dari masing-masing metode *load balance* dan disertakan pula kekurangan maupun kelebihan.

2.2.1 PCC

PCC merupakan metode yang dikembangkan oleh Mikrotik dan mulai diperkenalkan pada Mikrotik *Router*OS versi 3.24. PCC mengambil bidang yang dipilih dari *header* IP, dan dengan bantuan dari algoritma *hashing* mengubah bidang yang dipilih menjadi 32-bit. Nilai ini kemudian dibagi dengan *denominator* tertentu dan sisanya kemudian dibandingkan dengan *remainder* tertentu, jika sama maka paket akan ditangkap. *Rules* dapat dibuat dengan memilih informasi dari *src-address*, *dst-address*, *src-port*, atau *dst-port* dari bagian *header* IP. *Header* IP memiliki *field* yang berisi beberapa bidang, dua diantaranya adalah alamat IP sumber (*src-address*) paket dan alamat IP tujuan (*dst-address*) dari paket tersebut. Paket TCP dan UDP juga memiliki *header* yang berisi port sumber dan port tujuan (Hafizh, 2011).

PCC Merupakan metode yang mengelompokkan trafik koneksi melalui atau keluar masuk *router* menjadi beberapa kelompok. Pengelompokan ini bisa dibedakan berdasarkan *src-address*, *dst-address*, *src-port*, dan *dst-port*. *Router* akan menyimpan informasi tentang jalur *gateway* yang dilewati data ditiap trafik koneksi, sehingga pada paket-paket selanjutnya yang masih berkaitan dengan paket data sebelumnya akan dilewatkan pada jalur *gateway* yang sama juga. Karena metode PCC melewatkan paket data melalui jalur *gateway* yang sama,

maka metode tersebut mempunyai kekurangan yaitu dapat terjadi *overload* pada salah satu *gateway*.

PCC bekerja dengan cara mengambil beberapa field dari IP header dan TCP atau UDP *header*, kemudian dengan bantuan algoritma *hashing* akan menghasilkan sebuah *output*. *Output* tersebut didapat dengan cara melakukan penjumlahan dari beberapa *field* IP *header*, kemudian dibagi oleh penyebut yang telah ditentukan, dan sisanya jika dibandingan dengan remainder tertentu, jika sama, maka paket akan di*capture*. Kita dapat memilih *source-address*, *destination-address*, *source-port*, dan *destination port* dalam operasi ini (fewi, 2010).

Source-address dan *destination-address* dapat diambil dari IP paket *header* dan *source-port* dan *destination-port* diambil dari TCP atau UDP paket *header*. Salah satu metode *hash* yang dapat digunakan adalah Modulo. Modulo merupakan sebuah operasi bilangan yang menghasilkan sisa pembagian dari suatu bilangan terhadap bilangan lainnya. Misalkan dua bilangan a dan b, a modulo b (disingkat a mod b) adalah bilangan bulat sisa pembagian a oleh b. misalnya, “1 mod 3”. “4 mod 3”, dan “7 mod 3” memiliki hasil 1, karena ketiga bilangan tersebut memiliki sisa 1 jika dibagi oleh 3, sedangkan “9 mod 3” sama dengan 0. Penerapan operasi modulus dalam teori bilangan tergolong aritmatika modulo.

Fungsi *hashing* dipakai karena mempunyai salah satu sifat yang deterministik. Maksudnya adalah jika kita memasukkan input yang bertuliskan “hello” dan menghasilkan output “1”, dan pernyataan itu bersifat mutlak, sehingga jika kita menginputkan “hello” kedua kalinya akan menghasilkan output “1”.

Misalkan *source-address* dari *router client* (*router load balance*) adalah

192.168.1.1 kemudian *destination-address* dari *router* A (256 Kbps) 192.168.3.2 dan *destination-address* dari *router* B (512 Kbps) 192.168.2.2, jika pada saat ada *file video* yang lewat pada *router client*, maka *router client* akan mengecek *ping gateway* pada *router* A dan *router* B (dicari yang terendah). Setelah diketahui bahwa nilai *ping gateway* yang terendah ada pada *router* B (karena *bandwidth*-nya lebih besar dari *router* A), maka *file video* tersebut di-*dekapsulasi* kemudian pada semua *header packet*-nya di *mangle* pada *router* B. Jika dengan algoritma *hashing*, nilai *source-address* + nilai *destination-address* kemudian di modulus jumlah jalur yang ada. Jadi 192+168+1+1+192+168+2+2 mod 2 = 1. Nilai 1 untuk *router* B, nilai 0 untuk *router* A.

2.2.2 Nth

Nth bukanlah sebuah singkatan, melainkan Nth adalah sebuah integer (bilangan ke-N). Nth menggunakan algoritma *round robin* yang menentukan pembagian pemecahan *connection* yang akan di-*mangle* ke rute yang dibuat untuk *load balance*.

Pada dasarnya koneksi yang masuk ke proses di *router* akan menjadi satu arus yang sama, walaupun mereka datang dari *interface* yang berbeda. Maka pada saat menerapkan metode Nth, tentunya akan memberikan batasan ke *router* untuk hanya memproses koneksi dari sumber tertentu saja. Ketika *router* telah membuat semacam antrian baru untuk batasan yang kita berikan diatas, baru proses Nth dimulai.

Didalam Nth terdapat variabel yang harus dimengerti, yaitu

1. *Every*: Angka *every* adalah jumlah kelompok yang ingin dihasilkan. Jadi bila administrator ingin membagi alur koneksi yang ada menjadi 2 kelompok yang

nantinya akan diload balance ke 2 koneksi yang ada, maka angka every = 2.

2. *Packet*: Angka *packet* adalah jumlah koneksi yang akan ditandai atau di-*mangle*. Jika ingin membuat 2 kelompok, tentunya harus membuat 2 *mangle rules*. Pada *rules* tersebut, angka untuk *every* haruslah sama, namun untuk angka *packet* harus berubah. Untuk 2 kelompok, berarti angka *packet* untuk 2 *rules* tersebut adalah 1 dan 2 (Utomo, 2013). Salah satu kekurangan metode Nth ini kemungkinan dapat terjadi terputusnya koneksi yang disebabkan perpindahan *gateway* karena *load balance*.

Misalkan *source-address* dari *router client* (*router load balance*) adalah 192.168.1.1 kemudian *destination-address* dari *router* A (256 Kbps) 192.168.3.2 dan *destination-address* dari *router* B (512 Kbps) 192.168.2.2, jika pada saat *file video* yang lewat pada *router client*, maka *router client* akan men-*dekapsulasi file video*. Kemudian *header packet* yang pertama di-*mangle* pada *router* A (192.168.3.2) dan *header packet* yang kedua di-*mangle* pada *router* B (192.168.2.2). Proses tersebut berlangsung hingga semua *packet* yang melalui *router client* habis.

2.3 Router

Router merupakan perangkat keras jaringan yang memiliki peranan penting dalam mengatur lalu lintas jaringan. *Router* bertugas untuk menangani proses pengiriman data dari jaringan ke jaringan lain. Agar *router* dapat mengetahui bagaimana meneruskan paket ke alamat yang dituju dengan menggunakan jalur terbaik, router menggunakan peta atau *routing table*. Proses routing dilakukan *hop by hop*.

Routing table adalah tabel yang memuat seluruh informasi IP *address* dari

interface router yang lain sehingga *router* yang satu dengan *router* lainnya bisa berkomunikasi. *Routing table* hanya memberikan informasi sedangkan *routing* algoritma yang menganalisa dan mengatur *routing table*.

2.4 Routing

Beberapa jenis *routing* yang sudah diterapkan dan digunakan sebagai berikut :

2.4.1 OSPF

Merupakan protokol *routing link state* dan digunakan untuk menghubungkan *router-router* yang berada dalam satu jaringan sehingga protokol *routing* ini termasuk juga kategori *Interior Gateway Protocol* (IGP). OSPF pertama kali dikembangkan pada tahun 1987 oleh *Internet Engineering Task Force* (IETF) dan yang dipublikasikan pertama adalah OSPFv1. Pada tahun 1991, OSPFv2 mulai dipublikasikan sampai tahun 1998 perkembangan OSPF menjadi OSPFv3 hingga tahun 2008 OSPFv3 ini disempurnakan (Towidjojo, 2012).

2.4.2 Karakteristik OSPF

Protokol *routing* OSPF memiliki beberapa karakteristik sebagai berikut :

- Merupakan protokol *routing link state*, sehingga setiap *router* memiliki gambaran topologi jaringan.
- Menggunakan *Hello Packet* untuk mengetahui keberadaan *neighbore router*.
- *Routing update* hanya dikirimkan bila terjadi perubahan dalam jaringan dan dikirimkan secara *multicast*.
- Dapat bekerja dengan konsep hirarki karena dapat dibagi berdasarkan konsep *area*.

- Menggunakan *cost* sebagai *metric*, dengan *cost* terendah yang akan menjadi *metric* terbaik.
- Tidak memiliki keterbatasan *hop count*
- Merupakan *routing* protokol *classless*
- Nilai secara *default Administrative Distance* (AD)
- Memiliki fitur *authentication* saat mengirim *routing update* (Towidjojo, 2012).

2.5 NAT (Network Adress Translator)

Network Address Translator atau yang lebih biasa disebut dengan NAT adalah suatu metode untuk menghubungkan lebih dari satu komputer ke jaringan internet dengan menggunakan satu alamat IP. Banyaknya penggunaan metode ini disebabkan karena ketersediaan alamat IP yang terbatas, kebutuhan akan keamanan, dan kemudahan serta fleksibilitas dalam administrasi jaringan (Fathimah, 2009).

Dengan NAT gateway yang dijalankan di salah satu komputer, satu alamat IP tersebut dapat dibagi ke beberapa komputer lain dan mereka bisa melakukan koneksi ke internet secara bersamaan. Berikut ini adalah macam-macam *network address translator*:

2.5.1 Static NAT

Static NAT menggunakan *routing table* yang tetap, atau alokasi translasi alamat IP ditetapkan sesuai dengan alamat asal ke alamat tujuan, sehingga tidak memungkinkan terjadinya pertukaran data dalam suatu alamat IP bila translasi alamat IPnya belum terdaftarkan dalam tabel NAT, Translasi *Static* terjadi ketika sebuah alamat lokal dipetakan ke sebuah alamat global secara *Static*. NAT secara

statis akan melakukan *request* atau pengambilan dan pengiriman paket data sesuai dengan aturan yang telah ditabelkan dalam sebuah NAT.

2.5.2 Dynamic NAT

Dynamic Network Address Translator dimaksudkan untuk suatu keadaan dimana IP *address* terdaftar yang lebih sedikit dari jumlah IP address *un-registered*. *Dynamic* NAT menterjemahkan setiap komputer dengan IP tak terdaftar kepada salah satu IP address terdaftar untuk *connect* ke jaringan. Hal ini agak menyulitkan para penyusup untuk menembus komputer di dalam jaringan karena IP *address* terdaftar yang diasosiasikan ke komputer selalu berubah secara dinamis, tidak seperti pada *Static* NAT yang dipetakan sama. Kekurangan utama dari *Dynamic* NAT ini adalah jika jumlah IP *address* terdaftar sudah terpakai semua, maka untuk komputer yang berusaha *connect* ke jaringan tidak bisa lagi karena IP *address* terdaftar sudah terpakai semua.

2.5.3 Masquerading NAT

Masquerading NAT ini menerjemahkan semua IP *address* tak terdaftar pada jaringan, lalu dipetakan kepada satu IP *address* terdaftar. Agar banyak *client* bisa mengakses jaringan secara bersamaan, *router* NAT menggunakan nomor *port* untuk bisa membedakan antara paket-paket yang dihasilkan oleh atau ditujukan ke komputer-komputer yang berbeda. Solusi *Masquerading* ini memberikan keamanan paling bagus dari jenis-jenis NAT sebelumnya, hal ini dikarenakan asosiasi antara *client* dengan IP tak terdaftar dengan kombinasi IP *address* terdaftar dan nomor *port* di dalam *router* NAT hanya berlangsung saat terjadi satu kesempatan koneksi saja, setelah itu maka akan dilepas.

2.6 Definisi Internet Protocol version 4

IPv4 adalah sebuah jenis pengalamatan jaringan yang digunakan di dalam protokol jaringan TCP/IP yang menggunakan protokol IP versi 4. IP versi ini memiliki keterbatasan yakni hanya mampu mengalamati sebanyak 4 miliar host komputer di seluruh dunia (Setiawan, 2014).

2.6.1 Representasi alamat

Alamat IP versi 4 umumnya diekspresikan dalam notasi desimal bertitik (*dotted-decimal notation*), yang dibagi ke dalam empat buah oktet berukuran 8-bit. Dalam beberapa buku referensi, format bentuknya adalah w.x.y.z. Karena setiap oktet berukuran 8-bit, maka nilainya berkisar antara 0 hingga 255 (meskipun begitu, terdapat beberapa pengecualian nilai). Alamat IP yang dimiliki oleh sebuah *host* dapat dibagi dengan menggunakan subnet mask jaringan ke dalam dua buah bagian, yakni :

- *Host Identifier*/HostID atau *Host address* (alamat host) yang digunakan khusus untuk mengidentifikasikan alamat host (dapat berupa workstation, server atau sistem lainnya yang berbasis teknologi TCP/IP) di dalam jaringan. Nilai *host identifier* tidak boleh bernilai 0 atau 255 dan harus bersifat unik di dalam *network identifier*/segmen jaringan di mana ia berada.
- *Network Identifier*/NetID atau *Network Address* (alamat jaringan) yang digunakan khusus untuk mengidentifikasikan alamat jaringan di mana host berada. Semua sistem di dalam sebuah jaringan fisik yang sama harus memiliki alamat *network identifier* yang sama. *Network identifier* juga harus bersifat unik dalam sebuah *Internetwork* (Setiawan, 2014).

2.6.2 Jenis Alamat

Alamat IPv4 terbagi menjadi beberapa jenis, yakni sebagai berikut:

- **Alamat Unicast**, merupakan alamat IPv4 yang ditentukan untuk sebuah antarmuka jaringan yang dihubungkan ke sebuah *Internetwork* IP. Alamat *unicast* digunakan dalam komunikasi *point-to-point* atau *one-to-one* .
- **Alamat Broadcast**, merupakan alamat IPv4 yang didesain agar diproses oleh setiap *node* IP dalam segmen jaringan yang sama. Alamat broadcast digunakan dalam komunikasi *one-to-everyone*.
- **Alamat Multicast**, merupakan alamat IPv4 yang didesain agar diproses oleh satu atau beberapa node dalam segmen jaringan yang sama atau berbeda. Alamat multicast digunakan dalam komunikasi *one-to-many* (Setiawan, 2014).

2.6.3 Kelas IPv4

Kelas-kelas alamat jaringan versi 4 :

1. **Kelas A** adalah alamat jaringan berskala besar yang mempunyai nilai oktet pertama 1-126 (desimal) dan oktet pertama 0xxx xxx (biner).
2. **Kelas B** adalah alamat jaringan yang digunakan khusus untuk jaringan berskala menengah sampai besar yang mempunyai nilai oktet pertama 128-191 (desimal) dan oktet pertama 10xx xxx (biner).
3. **Kelas C** adalah alamat jaringan yang digunakan untuk jaringan berskala kecil yang mempunyai nilai oktet pertama 192-223 (desimal) dan oktet pertama 110x xxx (biner).
4. **Kelas D** adalah alamat jaringan yang disediakan khusus hanya untuk IP *multicast* yang mempunyai nilai oktet pertama 224-239 (desimal) dan oktet

pertama 1110 xxx (biner).

5. **Kelas E** adalah alamat jaringan yang bersifat eksperimental atau percoban dan dicadangkan untuk kegunaan di kemudian hari nantinya yang mempunyai nilai oktet pertama 240-255 (desimal) dan oktet petama 1111 xxxx (biner) (Setiawan, 2014).

2.7 TCP/IP (Transmission Control Protocol)

TCP adalah sekumpulan *protocol* yang didesain untuk melakukan fungsi komunikasi pada jaringan komputer. TCP/IP terdiri dari sekumpulan *protocol* komunikasi yang bertanggung jawab atas bagian tertentu dari komunikasi data. Jadi, TCP/IP inilah yang memungkinkan kumpulan komputer untuk berkomunikasi dan bertukar data dalam suatu jaringan. TCP/IP dapat diterapkan dengan mudah disetiap jenis komputer dan *interface* jaringan karena sebagian besar isi kumpulan protokol ini tidak spesifik terhadap satu komputer atau peralatan jaringan tertentu. *Protocol* TCP berfungsi untuk melakukan transmisi data pada *segmen*. Model *protocol* TCP disebut *connection oriented protocol*. Berbeda dengan model *User Datagram Protocol* (UDP) yang disebut *connectionless protocol* (Sugeng, 2010).

Dalam konsep komunikasi data suatu jaringan komputer, ada mekanisme data dari komputer sumber ke komputer yang dituju. Tentunya dalam proses pengiriman yang terjadi tidak semudah yang dipikirkan. Alasan pertama, komputer tujuan berada jauh dari komputer sumber sehingga paket data yang dikirimkan bisa saja hilang atau rusak di tengah jalan. Alasan lainnya, mungkin komputer tujuan sedang mengirim atau menunggu data dari komputer sumber yang lain. Tentunya paket data yang akan dikirimkan diharapkan sampai dengan

tepat tanpa terjadi kerusakan. Untuk mengatur mekanisme komunikasi data tersebut dibutuhkan pengaturan proses pengiriman data yang dikenal sebagai *protocol*. *Protocol* adalah sebuah perangkat lunak yang melekat pada sistem operasi (Sugeng, 2010).

2.8 Lapisan Network

Lapisan *network* bertanggung jawab mengirim dan menerima data dari media fisik. Media fisik ini berupa kabel, serat optik atau gelombang radio. Karena tugasnya, *protocol* pada layer ini harus mampu menerjemahkan sinyal listrik menjadi data digital yang dimengerti oleh komputer yang berasal dari peralatan lain yang sejenis. Pada lapisan *network*, internet bertanggung jawab dalam proses pengiriman paket ke alamat yang tepat. Pada layer ini terdapat tiga macam *protocol*, yaitu IP, ARP, dan ICMP. IP (*Internet Protocol*) berfungsi untuk menyampaikan paket data ke alamat yang tepat. ARP (*Address Resulotion Protocol*) merupakan *protocol* yang digunakan untuk menemukan alamat *hardware* dari *host* / komputer yang terletak pada *network* yang sama. Sedangkan ICMP (*Internet Control Massage Protocol*) merupakan *protocol* yang digunakan untuk mengirimkan pesan dan melaporkan kegagalan pengiriman data (Sugeng, 2010).

2.9 Lapisan Transport

Layer *transport* berisi protokol yang bertanggung jawab untuk berkomunikasi antara dua *host*. Pada lapisan transport ini menggunakan *Acknowledgement positif* dan *Acknowledgement negative* pada aliran datanya. *Acknowledgement positif* akan memberitahukan pesan apabila data yang ditransfer telah sampai sedangkan *Acknowledgement negative* jika paket yang ditransfer

tidak sampai ke tujuan maka akan terjadi pengiriman ulang. *Protocol* TCP dan UDP termasuk menggunakan lapisan *transport* (Sugeng, 2010).

2.10 Prinsip Kerja TCP/IP

Pada saat melakukan tugasnya, protokol TCP memiliki beberapa prinsip kerja. Prinsip kerja sebuah protokol ini akan menjadi referensi bagi pembuat program atau admin jaringan untuk memilih protokol apa yang nanti akan digunakan untuk bisa melakukan trasnmisi data (Sugeng, 2010).

2.10.1 Connection Oriented

Sebelum data dapat ditransmisikan antara dua *host*, dua proses yang berjalan pada lapisan aplikasi harus melakukan negosiasi untuk membuat sesi koneksi terlebih dahulu. Proses pembuatan koneksi TCP disebut juga dengan "*Three-way Handshake*". Tujuan metode ini adalah agar dapat melakukan sinkronisasi terhadap nomor urut dan nomor *acknowledgement* yang dikirimkan oleh kedua pihak dan saling bertukar ukuran TCP Window.

Client : SYN ->*Server* : *Client* akan mengirimkan SYN ke *Server*

Server: SYN-ACK ->*Client* : *Server* merespon SYN *Client* dengan mengirimkan SYN-ACK ke *Client*

Client : ACK ->*Server* : Setelah menerima SYN-ACK dari *Server*, *Client* mengirim ACK ke *Server*.

Setelah melewati handshake tadi, baru kemudian koneksi terbentuk (established). Bisa dikatakan *device* yang menggunakan protokol TCP ini akan melakukan kesepakatan terlebih dahulu sebelum transmisi data terjadi.

TCP menggunakan proses jabat tangan yang sama untuk mengakhiri koneksi yang dibuat. Hal ini menjamin dua *host* yang sedang terkoneksi tersebut telah menyelesaikan proses transmisi data dan semua data yang ditransmisikan telah diterima dengan baik. Koneksi TCP ditutup dengan menggunakan proses terminasi koneksi FIN (TCP *connection termination*) (Sugeng, 2010).

2.10.2 Reliable Transmission

Data yang dikirimkan ke sebuah koneksi TCP akan diurutkan dengan sebuah nomor urut yang unik disetiap bit data dengan tujuan agar data dapat disusun kembali setelah diterima. Pada saat transmisi, bisa jadi data dipecah / difragmentasi, hilang, atau tiba di *device* tujuan tidak lagi urut. Pada saat data diterima, paket data yang duplikat akan diabaikan dan paket yang datang tidak sesuai dengan urutannya akan diurutkan agar dapat disusun kembali (Sugeng, 2010).

2.10.3 Error Detection

Jika terjadi *error*, misalnya ada paket data yang hilang pada saat proses transmisi, bisa dilakukan pengiriman ulang data yang hilang. Untuk menjamin integritas setiap segmen TCP, TCP mengimplementasikan penghitungan TCP Checksum (Sugeng, 2010).

2.10.4 Flow Control

Mendeteksi supaya satu *host* tidak mengirimkan data ke *host* lainnya terlalu cepat. *Flow Control* akan menjadi sangat penting ketika bekerja di lingkungan dimana *device* satu dengan *device* yang lain memiliki kecepatan komunikasi jaringan yang beragam. Sebagai contoh, ketika PC mengirimkan data ke *smart phone*. kemampuan PC dengan *smart phone* tentu berbeda. *Smart phone*

lebih lambat dalam memproses data yang diterima daripada PC, maka TCP akan mengatur aliran data agar *smart phone* tidak kewalahan (Sugeng, 2010).

2.10.5 Segment Size Control

Mendeteksi besaran MSS (*Maximum Segment Size*) yang bisa dikirimkan supaya tidak terjadi IP *fragmentation*. MSS adalah infomasi ukuran data terbesar yang dapat ditransmisikan oleh TCP dalam bentuk segment tunggal. Informasi MMS ini dalam format *Bytes*. Untuk performa terbaik, MSS bisa ditetapkan dengan ukuran yang cukup kecil untuk menghindari fragmentasi IP. Fragmentasi IP dapat menyebabkan hilangnya paket dan retransmisi yang berlebihan (Sugeng, 2010).

2.10.6 Congestion Control

Prinsip kerja TCP yang terakhir yang cukup penting adalah *Congestion Control*. TCP menggunakan beberapa mekanisme untuk mencegah terjadinya *congestion* pada *network*. mekanisme yang dilakukan salah satunya adalah mengatur aliran data yang masuk ke dalam jaringan (Sugeng, 2010).

2.11 Mikrotik

Mikrotik adalah sistem operasi dan perangkat lunak yang dapat digunakan untuk menjadikan komputer menjadi router *network* yang handal, mencakup berbagai fitur yang dibuat untuk ip *network*, cocok digunakan oleh ISP dan *provider hotspot*. Untuk instalasi Mikrotik tidak dibutuhkan piranti lunak tambahan atau komponen tambahan lain. Mikrotik didesain untuk mudah digunakan dan sangat baik digunakan untuk keperluan administrasi jaringan komputer seperti merancang dan membangun sebuah sistem jaringan komputer skala kecil hingga yang kompleks (Sugeng, 2010).

2.11.1 Mikrotik RB941-2nd (haP Lite)

Router ini adalah salah satu varian *Routerboard* seri 900 yang memungkinkan digunakan di segala kondisi. Dengan fitur routerOS yang cukup banyak router ini bisa dipasang di kantor dan di rumah.

Mikrotik menggunakan standart power yang baru di varian ini yaitu MicroUSB 5V, hal ini memungkinkan pemasangan *Routerboard* ini menggunakan *charger hand phone / Smart phone* atau bahkan menggunakan *powerbank* yang banyak beredar di pasaran.

Sumber : mikrotik.co.id

Gambar 2.1 Mikrotik RB941-2nd (haP Lite)

2.11.2 Fitur Mikrotik

Beberapa fitur yang diberikan oleh Mikrotik yaitu :

1 Address List : Pengelompokan IP Address berdasarkan nama

2 Asynchronous : Mendukung serial *PPP dial-in*/*dial-out*, dengan otentikasi CHAP,PAP, MSCHAPv1 dan MSCHAPv2, Radius, *dial ondemand*, modem pool hingga 128 *port*s.

3	Bonding	:	Mendukung dalam pengkombinasian beberapa antarmuka *ethernet* ke dalam 1 pipa pada koneksi cepat.
4	Bridge	:	Mendukung fungsi *bridge spinning tree, multiple bridge interface, bridging firewalling*.
5	Data *Rate Management*	:	QoS berbasis HTB dengan penggunaan burst, PCQ, RED,SFQ, FIFO *queue*, CIR, MIR, *limit* antar *peer to peer*.
6	DHCP	:	Mendukung DHCP tiap antarmuka; *DHCP Relay; DHCP Client, multiple network DHCP; static and dynamic DHCP leases*.
7	*Firewall* dan NAT	:	Mendukung penyaringan koneksi *peer to peer*, *source* NAT dan *destination* NAT. Mampu mem*filter* berdasarkan MAC, IP *address, range port*, protokol IP, pemilihan opsi protokol seperti ICMP,TCP *Flags* dan MSS.
8	*Hotspot*	:	*Hotspot gateway* dengan otentikasi RADIUS. Mendukung *limit data rate*, SSL,HTTPS.
9	IPSec	:	Protokol AH dan ESP untuk IPSec; MODP *Diffie-Hellmann groups* 1, 2, 5; MD5 dan *algoritma SHA1 hashing; algoritma enkirpsi* menggunakan DES, 3DES, AES-128, AES-192, AES-256; *Perfect Forwarding Secresy* (PFS) MODP *groups* 1, 2,5

10	ISDN	:	Mendukung ISDN *dial-in/dial-out*. Dengan otentikasi PAP, CHAP, MSCHAPv1 dan MSCHAPv2, Radius. Mendukung 128K *bundle*, Cisco HDLC, x751, x75ui, x75bui *line* protokol.
11	M3P	:	Mikrotik *Protokol Paket Packer* untuk *wireless links* dan *ethernet*.
12	MNDP	:	*Mikrotik Discovery Neighbour Protokol*, juga mendukung *Cisco Discovery Protokol* (CDP).
13	*Monitoring / Accounting*	:	Laporan *Trafic* IP, *log*, *statistik graph* yang dapat diakses melalui HTTP.
14	NTP	:	*Network* Time Protokol untuk *server* dan *clients*; sinkronisasi menggunakan *system* GPS.
15	*Poin to Point Tunneling Protocol*	:	PPTP, PPPoE dan L2TP *Access Consentrator;* protokol otentikasi menggunakan PAP, CHAP, MSCHAPv1, MSCHAPv2; otentikasi dan laporan Radius; enkripsi 28MPPE; kompresi untuk PPoE; *limit data rate*.
16	Proxy	:	*Cache* untuk FTP dan HTTP *proxyserver*, HTTPS *proxy*; *transparent proxy* untuk DNS dan HTTP; mendukung protokol SOCKS; mendukung *parent proxy*; static DNS.
17	*Routing*	:	*Routing* statik dan dinamik; RIP v1/v2, OSPF v2, BGP v4.

18	SDSL	:	Mendukung *Single Line* DSL; mode pemutusan jalur koneksi dan jaringan.
19	*Simple Tunnel*	:	Tunnel IPIP dan EoIP (*Ethernet over* IP).
20	SNMP	:	*Simple Network Monitoring Protocol mode akses read-only*.
21	Synchronous	:	V.35, V.24, E1/T1, X21, DS3 (T3) *media types*; *sync*- PPP, Cisco HDLC; *Frame Relay line protokol*; ANSI-617d (ANDI atau *annex* D) dan Q933a (CCITT atau *annex* A); *Frame Relay* jenis LMI.
22	*Tool*	:	*Ping, Traceroute; bandwidthtest; ping flood; telnet*; SSH; *packet sniffer*; Dinamik DNS update.
23	UPnP	:	Mendukung antarmuka *Universal Plug and Play*
24	VLAN	:	Mendukung *Virtual* LAN IEEE 802.1q untuk jaringan *ethernet* dan *wireless*; *multiple* VLAN; VLAN *bridging*.
25	VoIP	:	Mendukung aplikasi *voice over* IP.
26	WinBox	:	Aplikasi mode GUI untuk me*remote* dan mengkonfigurasi MikroTik RouterOS serta VRRP yang mendukung *Virtual Router Redudant Protocol* (Sinaga, 2013).

2.12 Layanan

Sebuah sistem yang terdiri atas komputer-komputer yang didesain untuk dapat berbagi sumber daya, berkomunikasi, dan dapat mengakses informasi.

Bertujuan agar setiap bagian dari jaringan komputer dapat meminta dan memberikan layanan. Ada beberapa layanan untuk media pengiriman seperti FTP (*File Transfer Protocol*). FTP tersebut memiliki 2 jenis, yaitu FTP server dan FTP client. Beberapa ringkasanmengenai FTP server dan FTP client, sebagai berikut :

2.12.1 FTP Server

File Transfer Protocol (FTP) *Server* merupakan perangkat lunak yang bertanggung jawab untuk menerima permintaan protokol FTP dari *Client*. FTP ini berfungsi untuk men*download* atau meng*upload file* antar komputer (Ozan, 2012).

2.12.2 FTP Client

FTP *Client* merupakan aplikasi untuk mengelola dan men*transferfile* antar *Client* dan *Server*. Pada umumnya digunakan untuk men*download file* ke *Server*. Ada beberapa aplikasi FTP diantaranya Filezila, FireFTP, dan masih banyak lagi (Ozan, 2012).

2.13 Network Monitoring

Monitoring jaringan dibutuhkan untuk melakukan pengawasan pada jaringan yang dilakukan, agar jaringan tersebut selalu terkontrol dan apabila terputus dapat diketahui langsung oleh *user*. Pada tugas akhir ini *software* yang digunakan untuk monitoring jaringan yaitu Wireshark.

2.13.1 Wireshark

Wireshark merupakan salah satu *tool monitoring* jaringan yang berfungsi untuk mengawasi lalu lintas pada jaringan komputer dan dapat menganalisa keseluruhan jaringan komputer (Cahyaningtyas, 2013). Logo wireshark dapat dilihat pada Gambar 2.2

Sumber: http://www.wireshark.org

Gambar 2.2 Wireshark

Wireshark dapat melihat dan meyimpan informasi mengenai paket keluar dan masuk dalam jaringan yang terkirim dan diterima.

2.13.2 Tujuan dan Manfaat Wireshark

Manfaat dari software Wireshark, sebagai berikut :

- Menangkap informasi yang dikirim dan diterima
- Mengetahui aktivitas dalam jaringan komputer
- Mengetahui dan menganalisa kinerja jaringan komputer
- Mengamati keamanan jaringan computer (Cahyaningtyas, 2013).