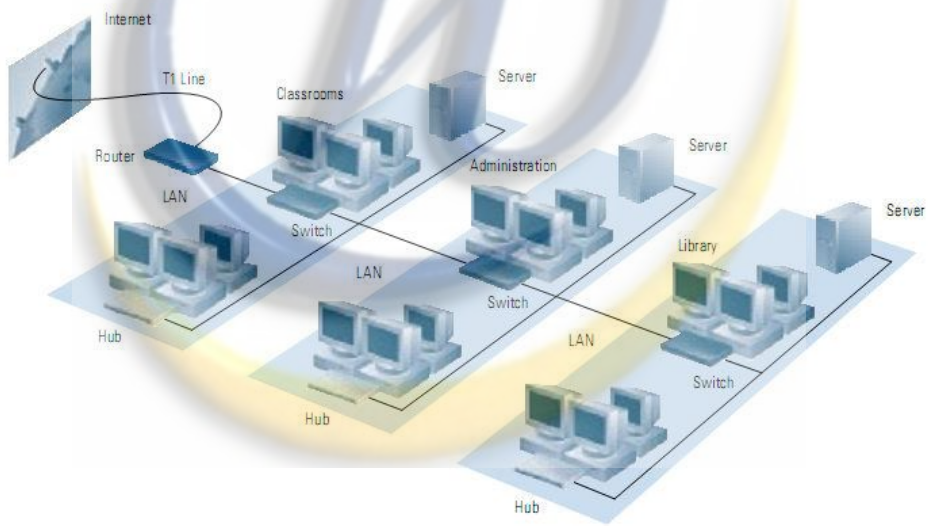
BAB II
LANDASAN TEORI

2.1 Sistem Jaringan Komputer

2.1.1 *Jaringan* Local Area Network *(LAN)*

Local Area Network (LAN) adalah jaringan komputer yang dirancang untuk area geografis terbatas seperti gedung atau kampus. Meskipun LAN dapat digunakan sebagai jaringan terisolasi untuk menghubungkan komputer dalam suatu organisasi untuk tujuan berbagi sumber daya, kebanyakan LAN saat ini juga terkait dengan *wide area network* (WAN) atau Internet. *Pasar LAN telah melihat beberapa teknologi seperti Ethernet, Token Ring, Token Bus*, FDDI, dan ATM LAN. Beberapa teknologi ini bertahan selama beberapa saat, tetapi *Ethernet* sejauh ini merupakan teknologi dominan.[6]



Gambar 2.1 Jaringan LAN (Sumber : www.cisco.com)

Secara garis besar ada beberapa tahapan dalam membangun jaringan LAN, diantaranya[12] ;

- Menentukan teknologi tipe jaringannya (*Ethernet, Fast Ethernet, Token Ring*, FDDI)
- Memilih model perkabelan (Fiber, UTP, Coaxial)
- Menentukan bentuk topologi jaringan (Bus, Ring, dan Star)

- Menentukan teknologi Client/Server atau Peer to Peer
- Memilih Sistem Operasi Server (Windows NT, 2000, XP, atau Linux)

2.1.2 *Pengertian Protocol*

Protocol adalah Satu set formal konvensi yang memungkinkan komunikasi antara dua unit fungsional berkomunikasi. *Protocol* adalah bahasa komputer yang digunakan untuk berbicara satu sama lain. Paling populer adalah TCP/ IP yang digunakan secara resmi di Internet. [12]

1 Model Jaringan 7 *Layer* OSI.

Model OSI terdiri dari 7 *layer*:

1. *Application* : Menyediakan jasa untuk aplikasi pengguna. *Layer* ini bertanggung jawab atas pertukaran informasi antara program komputer, seperti program *e-mail* dan *service* lain yang jalan di jaringan, seperti server printer atau aplikasi komputer lainnya.
2. *Presentation* : Bertanggung jawab bagaimana data dikonversi dan diformat untuk transfer data. Contoh konversi format text ASCII untuk dokumen, gif dan JPG untuk gambar. *Layer* ini membentuk kode konversi, translasi data, enkripsi dan konversi.
3. *Session* : Menentukan bagaimana dua terminal menjaga, memelihara dan mengatur koneksi, bagaimana mereka saling berhubungan satu sama lain. Koneksi di *layer* ini disebut "*session*".
4. *Transport* : Bertanggung jawab membagi data menjadi segmen, menjaga koneksi logika "*end-to-end*" antar terminal dan menyediakan penanganan *error* (*error handling*).

5. *Network* : Bertanggung jawab menentukan alamat jaringan, menentukan rute yang harus diambil selama perjalanan dan menjaga antrian trafik di jaringan. Data pada *layer* ini berbentuk paket.

6. *Data Link* : Menyediakan *link* untuk data, memaketkannya menjadi *frame* yang berhubungan dengan "*hardware*" kemudian diangkut melalui media. Komunikasinya dengan kartu jaringan, mengatur komunikasi *layer physical* antara sistem koneksi dan penanganan *error*.

7. *Physical* : Bertanggung jawab atas proses data menjadi *bit* dan mentransfernya melalui media, seperti kabel dan menjaga koneksi fisik antar sistem.

2.2. *Router* dan *Gateway*

2.2.1. Router

Router adalah perangkat yang akan melewatkan paket IP dari suatu jaringan ke jaringan yang lain, menggunakan metode *addressing* dan *protocol* tertentu untuk melewatkan paket data tersebut.

Router memiliki kemampuan melewatkan paket IP dari satu jaringan ke jaringan lain yang mungkin memiliki banyak jalur diantara keduanya. *Router-router* yang saling terhubung dalam jaringan internet turut serta dalam sebuah algoritma *routing* terdistribusi untuk menentukan jalur terbaik yang dilalui paket IP dari sistem ke sistem lain. Proses *routing* dilakukan secara *hop* by *hop*. IP tidak mengetahui jalur keseluruhan menuju tujuan setiap paket. IP *routing* hanya menyediakan IP *address* dari *router* berikutnya yang menurutnya lebih dekat ke *host* tujuan. [12]

Fungsi :

– Membaca alamat logika / ip *address source* & *destination* untuk menentukan *routing* dari suatu LAN ke LAN lainnya.

– Menyimpan *routing table* untuk menentukan rute terbaik antara LAN ke WAN.

– Perangkat di layer 3 OSI *Layer*.

– Bisa berupa "*box*" atau sebuah OS yang menjalankan sebuah *daemon routing*.

– *Interfaces Ethernet*, Serial, (*Integrated Services Digital Network*) ISDN (*Basic Rate Inteface*)BRI.

Gambar 2.2 *Router*[12]

2.2.2. Gateway

Gateway adalah sebuah perangkat yang digunakan untuk menghubungkan satu jaringan komputer dengan satu atau lebih jaringan komputer yang menggunakan protokol komunikasi yang berbeda sehingga informasi dari satu jaringan komputer dapat diberikan kepada jaringan komputer lain yang protokol berbeda.

Apabila ada suatu komputer yang ingin terkoneksi oleh internet atau terkoneksi dengan komputer lain maka komputer itu harus memasukan ip *gateway* jadi komputer harus melewati gerbang utama pada jaringan internet atau yang lain. [2]

2.3. *Firewall*

Firewall adalah sistem keamanan yang menggunakan *device* atau sistem yang diletakkan di dua jaringan dengan fungsi utama melakukan penyaringan terhadap akses yang akan masuk. Berupa seperangkat *hardware* atau *software*, bisa juga berupa seperangkat aturan dan prosedur yang ditetapkan oleh organisasi. *Firewall* juga dapat disebut sebagai sistem atau perangkat yang mengizinkan lalu lintas jaringan yang dianggapnya aman untuk melaluinya dan mencegah lalu lintas

jaringan yang tidak aman. Umumnya *firewall* diimplementasikan dalam sebuah mesin terdedikasi, yang berjalan pada pintu gerbang (*gateway*) antara jaringan lokal dan jaringan lainnya.

Firewall juga umumnya digunakan untuk mengontrol akses terhadap siapa saja yang memiliki akses terhadap jaringan pribadi dari hak luar. Saat ini, istilah *firewall* menjadi istilah umum yang merujuk pada sistem yang mengatur komunikasi antar dua jaringan yang berbeda.

Firewall atau "tembok penghalang" merupakan sebuah perangkat yang ditujukan untuk melindungi *network* dari "kejahatan dunia luar". Biasanya *firewall* digunakan untuk melindungi LAN dari berbagai serangan atau *intrusions*. Serangan dapat ditujukan kepada *host* tertentu yang dapat menyebabkan data *corrupt* atau *service* menjadi tidak berfungsi. [2]

2.3.1. *Fungsi Firewall*

Firewall berfungsi menjaga keamanan jaringan dari ancaman pihak lain yang tidak berwenang. Mengubah, merusak atau menyebarkan data-data penting perusahaan merupakan contoh ancaman yang harus dicegah. *Firewall* memiliki fungsi ganda yaitu memeriksa paket dan menyaring paket, keduanya merupakan salah satu peran yang paling mendasar dari sebuah *firewall*. Berikut Fungsi-fungsi *firewall* secara umum: [2]

1. Mengontrol dan mengawasi paket data yang mengalir di jaringan. *Firewall* harus dapat mengatur, menyaring dan mengontrol lalu lintas data yang diizin untuk mengakses jaringan *private* yang dilindungi *firewall*. *Firewall* harus dapat melakukan pemeriksaan terhadap paket data yang akan melewati jaringan *private*. Beberapa kriteria yang dilakukan *firewall* apakah memperbolehkan paket data lewati atau tidak, antara lain[2]:
    1. Alamat IP dari komputer sumber;
    2. PortTCP/UDP sumber dari sumber;
    3. Alamat IP dari komputer tujuan;
    4. PortTCP/UDP tujuan data pada komputer tujuan;
    5. Informasi dari headeryang disimpan dalam paket data.
2. Melakukan autentifikasi terhadap akses.

3. Aplikasi *proxy*

   Firewall mampu memeriksa lebih dari sekedar header dari paket data, kemampuan ini menuntut *firewall* untuk mampu mendeteksi *protocol* aplikasi tertentu yang spesifikasi.

4. Mencatat semua kejadian pada jaringan

   Mencatat setiap transaksi kejadian yang terjadi di *firewall*. Ini memungkinkan membantu sebagai pendeteksian dini akan kemungkinan penjebolan jaringan.

Gambar 2.3 Konsep *firewall*.

2.3.2. *Mikrotik Sebagai Firewall*

Firewall beroperasi menggunakan aturan tertentu. Aturan inilah yang menentukan kondisi ekspresi yang memberitahu *router* tentang apa yang harus dilakukan *router* terhadap paket IP yang melewatinya. Setiap aturan disusun atas kondisi dan aksi yang akan dilakukan. Ketika ada paket IP lewat, *firewall* akan mencocokkannya dengan kondisi yang telah dibuat kemudian menentukan aksi apa yang akan dilakukan *router* sesuai dengan kondisi tersebut. [6]

Selain sebagai *gateway*, Mikrotik juga dipadukan dengan kemampuan *firewall* untuk mencegah hal-hal yang menggangu dari pihak lain, mengingat begitu banyaknya aplikasi yang dijalankan oleh pengguna jaringan[2]. Ada

aplikasi yang berjalan normal, tetapi ada juga aplikasi yang bersifat mengganggu kinerja jaringan. Sebagai contoh, paket *broadcast* yang dilakukan oleh *virus* dan paket berlebihan yang sering disebut sebagai *flooding*.

Paket dengan ukuran kecil memang tidak mengganggu koneksi jaringan. Namun, jika paket yang kecil tersebut dalam jumlah banyak, hal ini bias menurunkan kinerja jaringan (*down*). Maka disinilah pentingnya memakai *firewall* untuk menghindari jaringan yang bersifat negative.

Pada sistem operasi *Mikrotik*, *firewall* sudah termasuk paket *Mikrotik RouterOS* yang di dalam direktori firewall sendiri terdapat 6 direktori[2]:

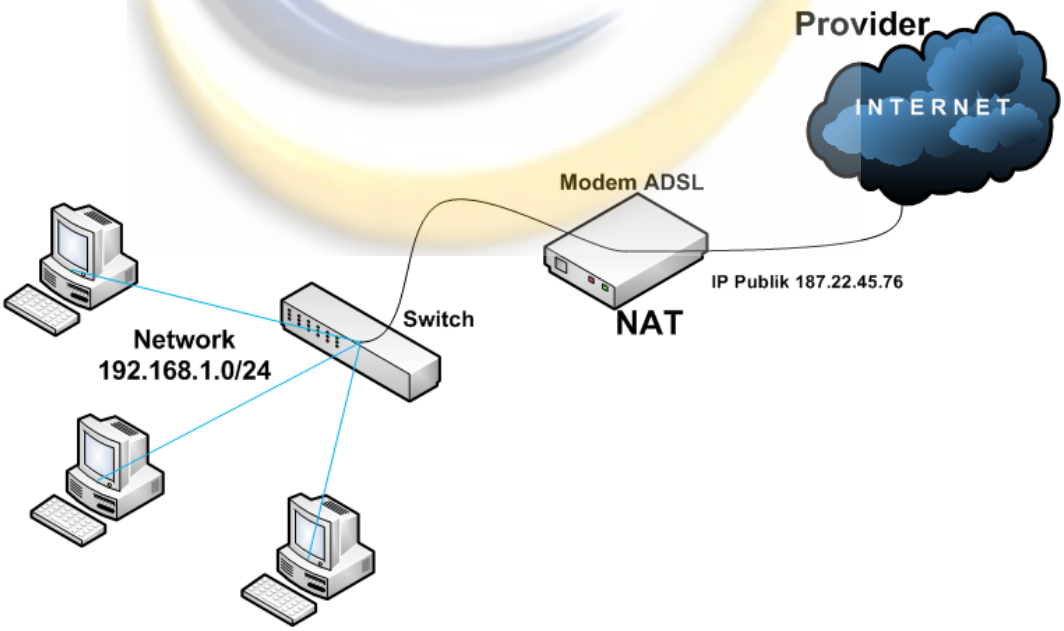
1. *Mangle*, untuk menandai paket dengan suatu tanda khusus sebagai indentitas paket tersebut;
2. NAT, untuk memetakan suatu IP *address* ke IP *address* lain;
3. *Connection*, untuk mengetahui informasi dari suatu koneksi yang aktif, seperti IP *address* asal dan tujuan beserta *port* yang digunakan, jenis protokol yang dipakai;
4. A*ddress-list*, untuk mendefinisikan IP *address* ke dalam group tertentu;
5. *Service port*, untuk mengaktifkan dan mengubah nomer *port* aplikasi;
6. *Filter*, untuk menyaring paket yang masuk atau melewati *router*. *Router* akan meneruskannya jika paket diizinkan lewat dan sebaliknya;
7. *Export*, untuk menyimpan/*backup* semua konfigurasi di dalam direktori *firewall*.

2.4. *Network Address Translator*

Ada dua tipe alamat IP: umum dan pribadi. Alamat umum diberikan kepada kita oleh *Internet Service Provider* (ISP) yang kita pakai untuk berhubungan ke internet. Bagi *host* di dalam organisasi yang tidak memerlukan akses langsung ke internet, alamat IP yang tidak menduplikasi alamat umum yang

sudah diberikan memang dibutuhkan. Untuk memecahkan persoalan alamat ini, para desainer internet mencadangkan suatu bagian dari ruang alamat IP dan menamai ruang ini sebagai ruang alamat pribadi. Suatu alamat IP pada ruang alamat pribadi tidak pernah diberikan sebagai alamat umum. Alamat IP di dalam ruang alamat pribadi dikenal sebagai alamat pribadi. Dengan memakai alamat IP pribadi, kita dapat memberikan proteksi dari para *hacker* jaringan.

Karena alamat IP pada ruang alamat pribadi tidak akan pernah diberikan oleh *Internet Network Information Center* (InterNIC) sebagai alamat umum, maka *route* di dalam internet router untuk alamat pribadi takkan pernah ada. Alamat pribadi tidak dapat dijangkau di dalam internet. Oleh karena itu, saat memakai alamat IP pribadi, kita membutuhkan beberapa tipe *proxy* atau *server* untuk mengonversi sejumlah alamat IP pribadi pada jaringan lokal kita menjadi alamat IP umum yang dapat di *route*. Pilihan lain adalah menerjemahkan alamat pribadi menjadi alamat umum yang valid dengan *network address translator* (NAT) sebelum dikirimkan di internet. Dukungan bagi NAT untuk menerjemahkan alamat umum dan alamat pribadi memungkinkan terjadinya koneksi jaringan-jaringan kantor, rumah atau kantor yang kecil ke internet seperti ditampilkan gambar 2.4 berikut ini. [2]

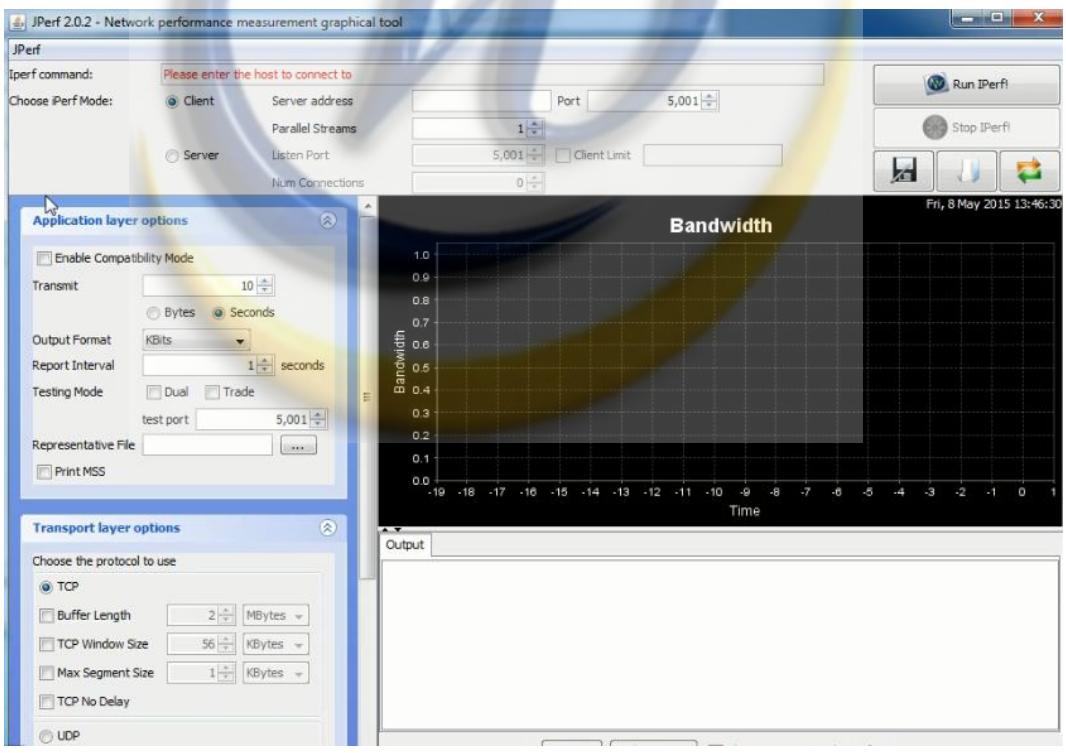


Gambar 2.4 Menghubungkan Jaringan Kecil ke Internet[2].

Sebuah NAT menyembunyikan alamat-alamat IP yang dikelola secara internal dari jaringan-jaringan eksternal dengan menerjemahkan alamat internal pribadi menjadi alamat eksternal umum. Hal ini mengurangi biaya registrasi alamat IP dengan cara membiarkan para pelanggan memakai alamat IP yang tidak terdaftar secara internal melalui suatu terjemahan ke sejumlah kecil alamat IP yang terdaftar secara eksternal. Hal ini juga menyembunyikan struktur jaringan internal, mengurangi resiko penolakan serangan layanan terhadap sistem internal.

2.4. *Jperf*

Jperf adalah versi GUI dari *Iperf*. *Iperf* adalah salah satu *tool* untuk mengukur *troughput bandwidth* dalam sebuah *link network*, agar bisa dilakukan pengukuran diperlukan *Iperf* yang terinstall *point to point*, baik disisi server maupun *client*. Iperf sendiri bisa digunakan untuk mengukur *performance link* dari sisi TCP maupun UDP. [16]

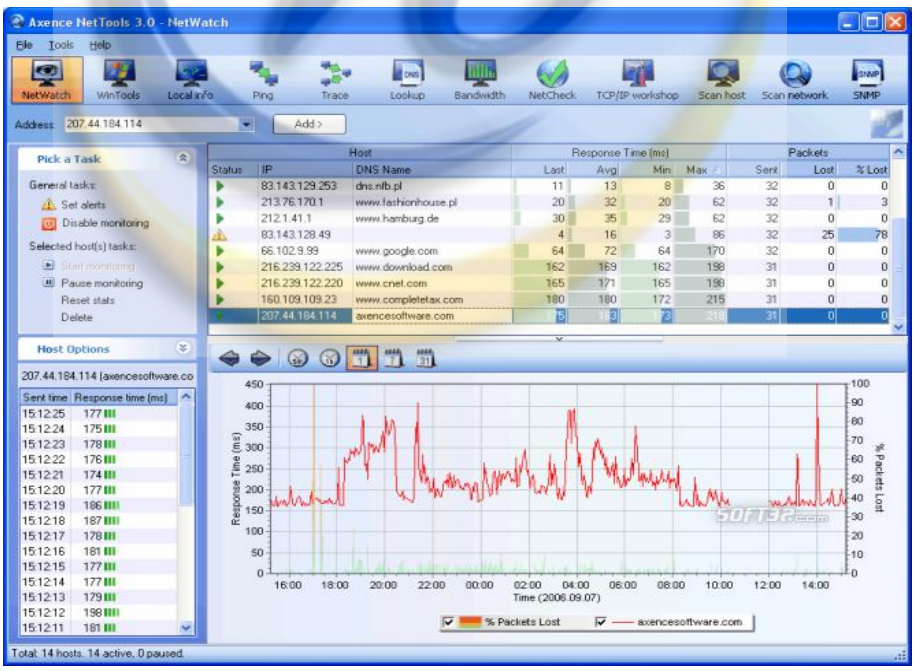


Gambar 2.5 *Jperf*

2.5. *Axence NetTools*

Axence NetTools adalah salah satu *Network analyzer* yang sangat handal. *Tool* ini dipakai unuk mengukur/menganalisa *perfomance network* dan mendiagnosa *problem* yang terjadi pada *network* tersebut. *Axence NetTools* sangat populer karena dilengkapi dengan beberapa *tools* seperti : [17]

A. *Netwatch*
B. *Wintools*
C. *Local Info*
D. *Netstat (part of local info)*
E. *Ping*
F. *Trace*
G. *Lookup*
H. *Bandwidth*
I. *Netcheck*
J. TCP/IP *workshop*
K. *Scan host*

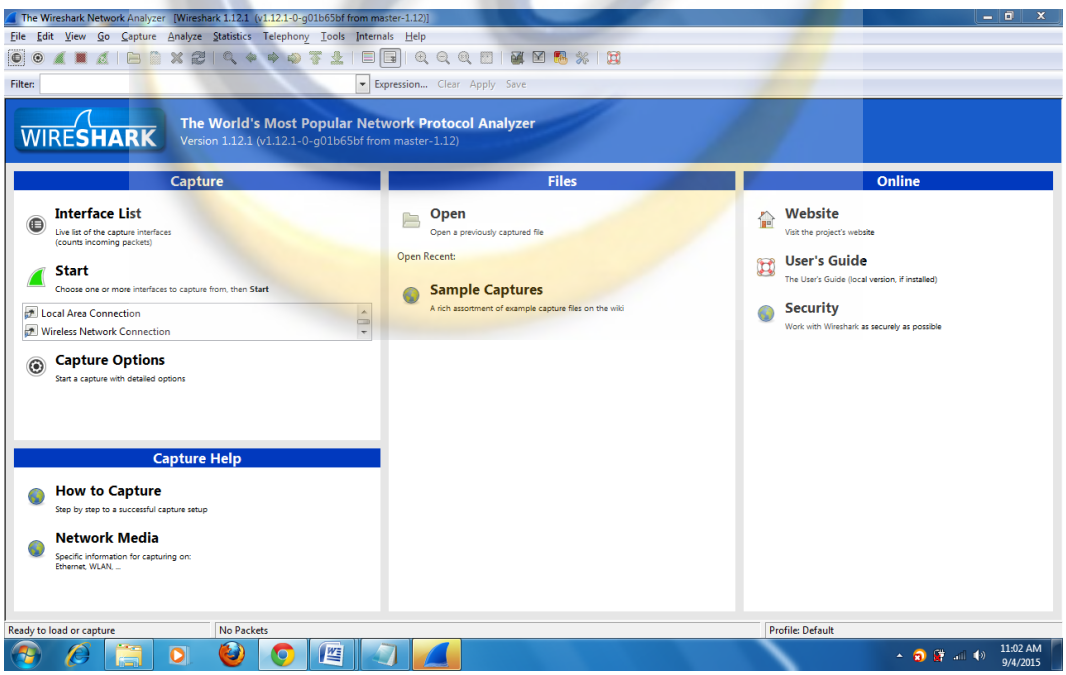


Gambar 2.6 *Axence NetTools*

2.6. *Wireshark*

Wireshark merupakan Network Protocol Analyzer, juga termasuk salat satu network analysis tool atau packet sniffer. Wireshark memungkinkan pengguna mengamati data dari jaringan yang sedang beroperasi atau dari data yang ada di disk, dan langsung melihat dan mensortir data yang tertangkap, mulai dari informasi singkat dan detail bagi masing-masing paket termasuk full header dan porsi data, dapat diperoleh. Wireshark memiliki beberapa fitur termasuk display filter language yang banyak dan kemampuan me-reka ulang sebuah aliran pada sesi TCP.

Paket sniffer sendiri diartikan sebuah tool yang berkemampuan menahan dan melakukan pencatatan terhadap traffic data dalam jaringan. Selama terjadi aliran data dalam jaringgan, packet sniffer dapat menangkap protocol data unit (PDU), melakukan decoding serta analisis terhadap isi paket. Wireshark sebagai salah satu packet sniffer yang diprogram demikian agar mengenali berbagai macam prottokol jaringan. Wireshark juga mampu menampilkan hasil enkapsulasi dan field yang ada di dalam PDU.[18]



Gambar 2.7 *Wireshark*

2.7. *Load Balancing*

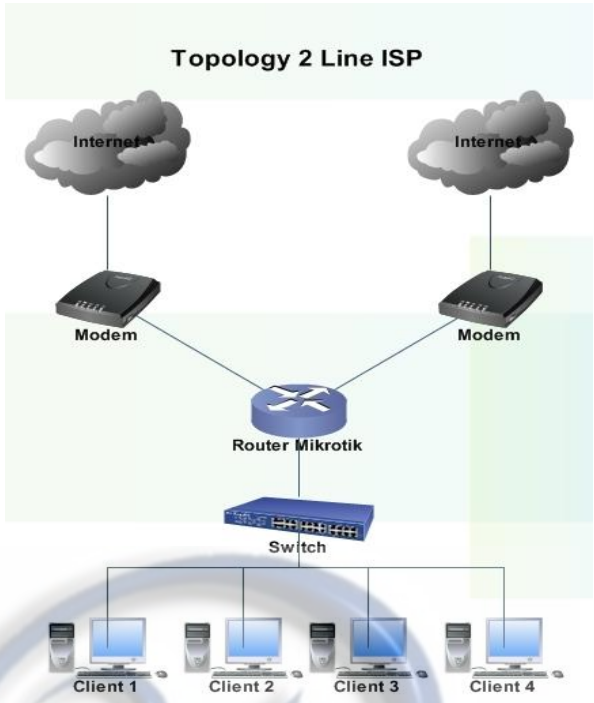
Load Balancing adalah teknik untuk mendistribusikan beban *traffic* pada dua atau lebih jalur koneksi secara seimbang, agar *traffic* dapat berjalan optimal, memaksimalkan *throughput*, memperkecil waktu tanggap dan menghindari *overload* pada salah satu jalur koneksi.

Secara umum, *load balancing* dapat diartikan sebagai suatu teknik untuk mendistribusikan beban kerja secara merata pada dua atau lebih komputer, *network links*, CPU, *hard drive* atau sumber daya lainnya, untuk mendapatkan pemanfaatan sumber daya yang optimal, memaksimalkan *throughput*, meminimalkan waktu *respond* dan menghindari *overload* [2]. Menggunakan beberapa komponen dengan *load balancing* dapat meningkatkan kehandalan melalui *redudansi*. Layanan *load balancing* biasanya disediakan oleh program khusus atau perangkat keras (seperti *multilayer switch* atau DNS *server*)[2].

Dengan mempunyai banyak link maka optimalisasi utilisasi sumber daya *throughput* atau *respond time* akan semakin baik karena mempunyai lebih dari satu *link* yang bisa saling mem-*backup* pada saat *network down* dan menjadi cepat pada saat *network* normal jika memerlukan *realibilitas* tinggi yang memerlukan 100% koneksi *uptime* dan yang menginginkan koneksi *upstream* yang berbeda dan dibuat saling mem-*backup*.

Dalam jaringan komputer, *load balancing* lebih mengarah kepada pengkombinasian beberapa antarmuka *ethernet* ke dalam satu jalur sehingga dapat di *implementasi* secara bersamaan dengan menghasilkan koneksi yang lebih cepat.

Untuk dapat *implentasi* sistem ini diperlukan suatu perangkat tambahan baik berupa *router Cisco* atau menggunakan solusi *router* dari *Mikrotik* yang lebih ekonomis namun *powerfull*.



Gambar 2.8 *Load Balaancing* dengan Dua *Backbone* Provider (Sumber : www.warungsetting.com).

Dengan konsep yang sederhana, sebuah *load balancing* diletakkan di antara *client* dan *server* seperti terlihat pada Gambar 2.5 akan menampung *traffic* yang datang dan membaginya ke dalam *request-request* individual lalu menentukan *server* mana yang menerima *request* tersebut. Beberapa keuntungan dari penerapan *load balancing* antara lain:

1. *Scalability* : Ketika beban sistem meningkat, kita dapat melakukan perubahan terhadap sistem agar dapat mengatasi beban sesuai dengan kebutuhan.

2. *High Availability* : *Load balancer* secara terus-menerus melakukan pemantauan terhadap *server*. Jika terdapat *server* yang mati, maka *load balancer* akan menghentikan *request* ke *server* tersebut dan mengalikannya ke *server* yang lain.

3. *Manageability* : Mudah ditata meskipun secara fisik sistem sangat besar.

4. *Security* : Untuk semua *traffic* yang melewati *load balancer*, aturan keamanan dapat di *implementasi* dengan mudah. Dengan *private network* digunakan untuk *server*, alamat IP nya tidak akan diakses secara langsung dari luar sistem.

Saat sebuah router mempunyai dua koneksi ke internet (sama atau berbeda ISP-nya), *default gateway* di *router* tetap hanya bisa satu, ditambahpun yang bekerja tetap hanya satu. Jadi misal *router* NAT terhubung ke ISP A melalui *interface* A dan *gateway* A dan ke ISP B melalui *interface* B dan *gateway* B, dan *default gateway* ke ISP A, maka *traffic downlink* hanya akan datang dari ISP A saja. Begitu juga sebaliknya jika dipasang *default gateway* ke ISP B. Penerapan teknik load balancing dapat menyelesaikan permasalahan tersebut dengan menggabungkan *traffic downlink* ISP A dan ISP B sehingga dapat di *imolementasi* secara bersamaan.

Prinsip dari load balance adalah sebagai berikut:[15]

1. Lalu lintas didistribusikan berdasarkan probabilitas.
2. Harus tau seberapa besar tiap *link*, dan didistribusikan sesuai lalu lintas.
3. Berdasarkan kecepatan pada keluaran dan masukan pada *router*, *load balance* dapat diilustrasikan sebagai berikut :

   1 + 1 = 1 + 1

   1 + 1 = ½ + ½ + ½ + ½

   1 + 1 = ¼ + ¼ + ¼ + ¼ + ¼ + ¼ + ¼ + ¼

4. Jika ada dua *gateway*, misal A dan B

   A memiliki *bandwidth* sebesar 1 Mbps dan B memiliki *bandwidth* sebesar 2 Mbps. Maka lalu lintas akan dibagi kedalam 3 aliran, dan mengirim 1 aliran ke A dan 2 aliran ke B.

2.7.1. *Algoritma Load Balancing*

Algoritma pembagian beban digunakan dalam penelitian ini adalah[2] :

1. Algoritma *Round robin*

Round Robin. Algoritma Round Robin merupakan algoritma yang paling sederhana dan banyak digunakan oleh perangkat load balancing. Algoritma ini membagi beban secara bergiliran dan berurutan dari satu server ke server lain sehingga membentuk putaran.

2. *Fastest*

Algoritma yang satu ini melakukan pembagian beban dengan mengutamakan server-server yang memiliki respon yang paling cepat. Server di dalam jaringan yang memiliki respon paling cepat merupakan server yang akan mengambil beban pada saat permintaan masuk.

3. *Ratio*

Ratio (rasio) sebenarnya merupakan sebuah parameter yang diberikan untuk masing-masing server yang akan dimasukkan kedalam sistem load balancing. Dari parameter *Ratio* ini, akan dilakukan pembagian beban terhadap server-server yang diberi rasio. Server dengan rasio terbesar diberi beban besar, begitu juga dengan server dengan rasio kecil akan lebih sedikit diberi beban.

4. *Least connection*

Algoritma Least connection akan melakukan pembagian beban berdasarkan banyaknya koneksi yang sedang dilayani oleh sebuah server. Server dengan pelayanan koneksi yang paling sedikit akan diberikan beban yang berikutnya akan masuk.

5. Algoritma *Hasing*

Algoritma *hashing* merupakan algoritma yang menggunakan informasi dari IP *Address* yang berasal dari permintaan *client* pada saat

mengirimkan *request* ke suatu *server*. Informasi IP *Address* yang digunakan tergantung dari penerapan sistem tertentu untuk WAN *Link Load Balancing*, informasi IP *Address* yang digunakan adalah informasi dari IP *Address* tujuan paket. Semua *request* untuk tujuan IP *Address* tertentu akan dikirim ke *gateway* yang sama.

2. Fungsi *Hashing.*

Hash adalah fungsi yang diberi input dan menghasilkan *output* yang *deterministic*. Itu berarti bahwa ketika sebuah fungsi *hash* mengenkripsi suatu *input* yang berbunyi "halo" lalu akan menghasilkan *output* "1". Dengan sifat dari *hashing* yang *deterministic* maka dapat ditetapkan bahwa jika fungsi *hash* tersebut mengenkripsi *input* berupa "halo" pada saat kedua kalinya atau seterusnya, maka sudah dipastikan akan menghasilkan *output* yang sama yaitu "1".

2.7.2. *Sistem Load Balancing*

Sistem *load balancing* sebenarnya dapat dibuat dengan banyak cara. Pembuatannya tidak terikat oleh sebuah *operating system* saja atau hanya dapat dibuat oleh sebuah perangkat saja. Namun secara garis besar cara pembuatan sistem *load balancing* terbagi menjadi tiga kategori besar, yaitu *load balancing* dengan menggunakan DNS *round robin*, *Integrated load balancing* dan *Dedicated load balancing*[2]. Ketiga jenis ini memiliki cara kerja yang unik dan berbeda satu sama lain, tetapi tetap menuju suatu hasil akhir yang sama yaitu menciptakan sebuah sistem yang lebih menjamin kelangsungan hidup jaringan dibelakangnya dan membuatnya lebih skalabel[2].

2.8. Metode *load balancing*

2.7.1. Nth

Nth *load balance* merupakan suatu teknik *load balance* yang membentuk suatu deret tertentu (Nth), yang nantinya akan digunakan sebagai suatu sistem antrian di dalam *mangle rule* yang dibentuk. Nth

diimplementasikan dalam suatu deret yang terdiri dari *every*, *packet* dan *counter* yang akan direalisasikan dalam suatu deret *interger*. Pada metode *load balance* ini, paket data yang masuk akan ditandai sebagai suatu variabel ‘n’ dalam tipe data *integer*. [6]

Koneksi *load balance* menggunakan *multi gateway* ini disebut dengan metode *round robin* karena beban terbagi secara berurutan dan bergiliran dari *gateway* yang satu ke *gateway* yang lain oleh karena itu *gateway* yang digunakan selalu bergantian dan tidak tetap (*random*). [6]

2.7.2. *Per Connection Classifier (*PCC*)*

Per Connection Classifier merupakan metode yang menspesifikasikan suatu paket menuju gateway suatu koneksi tertentu. PCC mengelompokkan traffic koneksi yang keluar masuk router menjadi beberapa kelompok. Pengelompokan ini bisa dibedakan berdasarkan src-address, dst-address, src-port dan dst-port. Mikrotik akan mengingat-ingat jalur gateway yang telah dilewati di awal traffic koneksi. Sehingga pada paket-paket data selanjutnya yang masih berkaitan akan dilewatkan pada jalur gateway yang sama dengan paket data sebelumnya yang sudah dikirim. Kelebihan metode ini mampu menspesifikasikan gateway untuk tiap paket data yang masih berhubungan dengan data yang sebelumnya sudah dilewatkan pada salah satu gateway. Kekurangannya beresiko terjadi overload pada salah satu gateway yang disebabkan oleh pengaksesan situs yang sama.[7]

2.7.2. *Static route*

Static route dengan Address list adalah metode load balancing yang mengelompokkan suatu range IP address untuk dapat di atur untuk melewati salah satu gateway dengan menggunakan static routing. Metode ini sering di gunakan pada warnet yang membedakan PC untuk browsing dengan PC untuk Game Online. Mikrotik akan menentukan jalur gateway yang di pakai dengan membedakan src-address pada paket data. [7]

2.7.2. *Equal Cost Multi Path (ECMP)*

Equal Cost Multi Path adalah pemilihan jalur keluar secara bergantian pada gateway. Contohnya jika ada dua gateway, dia akan melewati kedua gateway tersebut dengan beban yang sama (Equal Cost) pada masing-masing gateway. [7]

2.9. Definisi QoS (Quality Of Service)

QoS adalah kemampuan suatu jaringan untuk menyediakan layanan yang baik dengan menyediakan kapasitas jaringan, mengatasi *jitter* dan *delay* (waktu tunda). QoS dirancang untuk membantu pengguna menjadi lebih produktif dengan memastikan bahwa pengguna mendapatkan kinerja yang handal dari aplikasi aplikasi berbasis jaringan. QoS mengacu pada kemampuan jaringan untuk menyediakan layanan yang lebih baik pada trafik jaringan tertentu melalui teknologi yang berbeda-beda. QoS merupakan suatu tantangan yang besar dalam jaringan berbasis IP dan internet secara Keseluruhan[8].

Teknologi QoS adalah teknologi yang memungkinkan administrator jaringan untuk dapat menangani berbagai efek akibat terjadinya konjesti pada lalu lintas aliran paket dari berbagai layanan. Penanganan QoS dilakukan dengan memanfaatkan sumber daya jaringan secara optimal, dibandingkan dengan menambah kapasitas fisik jaringan tersebut.

QoS bertujuan untuk menyediakan kualitas layanan yang berbeda-beda untuk beragam kebutuhan akan layanan di dalam jaringan IP, sebagai contoh untuk menyediakan *bandwidth* yang khusus, menurunkan hilangnya paket-paket, menurunkan waktu tunda dan variasi waktu tunda di dalam proses transmisinya. QoS menawarkan kemampuan untuk mendefinisikan atribut-atribut layanan yang disediakan, baik secara kualitatif maupun kuantitatif. QoS memiliki fungsi-fungsi sebagai berikut[9] :

1. Pengkelasan paket untuk menyediakan pelayanan yang berbeda-beda untuk kelas paket yang berbeda-beda,
2. Penanganan *congestion* (kongesti) untuk memenuhi dan menangani kebutuhan layanan yang berbeda-beda,

3. Pengendalian lalu lintas paket untuk membatasi dan mengendalikan pengiriman paket-paket data,

4. Pensinyalan untuk mengendalikan fungsifungsi perangkat yang mendukung komunikasi di dalam jaringan IP.

Pada jaringan berbasis packet switched, kualitas layanan dipengaruhi oleh berbagai faktor, yang dapat dibagi menjadi faktor manusia dan faktor teknis. Faktor-faktor manusia meliputi: stabilitas layanan, ketersediaan layanan, waktu tunda, dan informasi pengguna. Faktor-faktor teknis meliputi: *realibility*, *scalability*, *effectiveness*, *maintainability*, *Grade of Service* (GOS), dan lain lain. Terdapat banyak hal bisa terjadi pada paket ketika mereka melakukan perjalanan dari asal ke tujuan, yang mengakibatkan masalah-masalah berikut dilihat dari sudut pandang pengirim dan penerima,atau yang sering disebut sebagai parameterparameter QoS[8].

2.9.1. *Parameter QoS*

2.9.1.1. Throughput

Throughput merupakan *rate* (kecepatan) transfer data efektif, yang diukur dalam *bit per second* (bps). Throughput merupakan jumlah total kedatangan paket yang sukses yang diamati pada *destination* selama interval waktu tertentu dibagi oleh durasi interval waktu tersebut. [10]

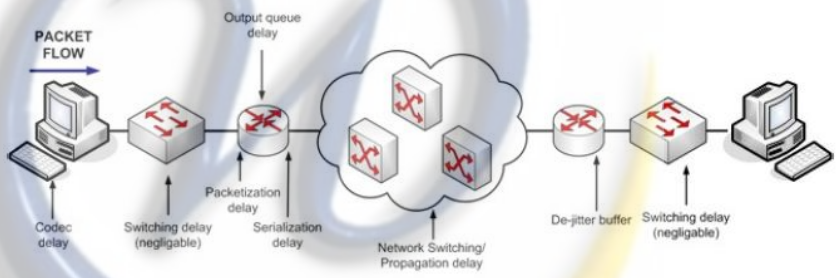
2.9.1.2. Delay

Waktu tunda (*delay*) merupakan akumulasi berbagai waktu tunda dari ujung ke ujung pada jaringan Internet. Waktu tunda mempengaruhi kualitas layanan (QoS) karena waktu tunda menyebabkan suatu paket lebih lama mencapai tujuan. ITU-T G.114 merekomendasikan waktu tunda tidak lebih besar dari 150 ms untuk berbagai aplikasi, dengan batas 400 ms untuk komunikasi suara yang masih dapat diterima. Rekomendasi tersebut ditunjukkan di Tabel 1. sebagai berikut: [10]

Tabel 1.1 Pengelompokan waktu tunda berdasarkan ITU G.114

Waktu Tunda (ms)	Kualitas
0 - 150	Baik
150 - 400	Cukup, masih dapat diterima
> 400	Buruk

Waktu tunda *end-to-end* seperti ditunjukkan Gambar 1 terdiri atas waktu tunda pengkodean (*codec delay*), waktu tunda paketisasi *(packetization delay*), waktu tunda serialisasi (*serialization delay*), waktu tunda propagasi (*propagation delay*), dan waktu tunda akibat *jitter buffer* (*dejitter buffer delay*).



Gambar 2.9 Waktu tunda (*delay*) pada jaringan[10]

2.9.1.3. Jitter

Variasi waktu tunda (jitter) merupakan perbedaan selang waktu kedatangan antar paket di terminal tujuan. Variasi waktu tunda dapat disebabkan oleh terjadinya kongesti, kurangnya kapasitas jaringan, variasi ukuran paket, serta ketidakurutan paket. [10]

Tabel 1.2 Standar *jitter* berdasarkan ITU G.114

Variasi waktu tunda (ms)	Kualitas
0 – 20	Baik
20 – 50	Dapat diterima
> 50	Tidak dapat diterima

2.9.1.4. Packet Loss

Packet Loss didefinisikan sebagai kegagalan transmisi paket mencapai tujuannya. Paket hilang dapat disebabkan oleh pembuangan paket di jaringan (*network loss*) atau pembuangan paket di *gateway*/terminal sampai kedatangan terakhir (*late loss*). *Network loss* secara normal disebabkan kemacetan (*router buffer overflow*), perubahan rute secara seketika, kegagalan *link*, dan *lossy link* seperti saluran nirkabel. Kemacetan atau kongesti pada jaringan merupakan penyebab utama dari paket hilang. Tabel 3 menunjukkan rekomendasi nilai paket hilang yang mempengaruhi kualitas layanan (QoS). [10]

Tabel 1.3 Rekomendasi nilai paket hilang berdasarkan ITU G.114

Paket Hilang (%)	Kualitas
0 – 1	Baik
1 – 5	Dapat diterima
> 10	Tidak dapat diterima