BAB 2

LANDASAN TEORI

2.1 Jaringan (*Network*)

Jaringan komputer atau *computer network*, dapat diartikan sebagai dua atau lebih komputer beserta perangkat lain yang dihubungkan agar dapat saling berkomunikasi dan bertukar data serta informasi, sehingga membantu menciptakan efisiensi dalam kerja.

Berdasarkan fungsinya, setiap jaringan komputer ada yang berfungsi sebagai *client* dan ada juga yang berfungsi sebagai *server*. Tetapi ada jaringan yang memiliki komputer yang khusus didedikasikan sebagai *server* sedangkan yang lain sebagai *client*. Ada juga yang tidak memiliki komputer yang khusus berfungsi sebagai *server* saja. Karena itu berdasarkan fungsinya maka ada dua jenis jaringan komputer:

a. *Client-server*

   Yaitu jaringan komputer dengan komputer yang didedikasikan khusus sebagai *server*. Suatu *service* bisa diberikan oleh sebuah komputer atau lebih. Contohnya adalah suatu domain seperti http://www.detik.com yang dilayani oleh banyak komputer *web server*. Atau bisa juga banyak *service* yang diberikan oleh satu komputer. Contohnya adalah *server* jtk.polban.ac.id yang merupakan satu komputer dengan multi *service* yaitu *mail server*, web *server*, *file server*, *database server* dan lainnya.

b. *Peer-to-peer*

   Yaitu jaringan komputer dimana setiap *host* dapat menjadi *server* dan juga menjadi *client* secara bersamaan. Contohnya dalam *file sharing* antar komputer

di Jaringan *Windows Network Neighbourhood* ada 5 komputer (kita beri nama A,B,C,D dan E) yang memberi hak akses terhadap *file* yang dimilikinya. Pada satu saat A mengakses *file share* dari B bernama data_nilai.xls dan juga memberi akses *file* soal_uas.doc kepada C. Saat A mengakses *file* dari B maka A berfungsi sebagai *client* dan saat A memberi akses *file* kepada C maka A berfungsi sebagai *server*. Kedua fungsi itu dilakukan oleh A secara bersamaan maka jaringan seperti ini dinamakan *peer to peer*.

Berdasarkan tipe transmisinya (Tanenbaum 2003, p.15), *network* dapat dibagi menjadi dua bagian besar, yaitu : *broadcast* dan *point-to-point*. Dalam *broadcast network*, komunikasi terjadi dalam sebuah saluran komunikasi yang digunakan secara bersama-sama, dimana data berupa paket yang dikirimkan dari sebuah komputer akan disampaikan ke tiap komputer yang ada dalam jaringan tersebut. Kemudian setiap komputer akan mengecek apakah data tersebut ditujukan untuk dirinya berdasarkan data alamat yang ada dalam paket tersebut. Paket data hanya akan diproses oleh komputer tujuan dan akan dibuang oleh komputer yang bukan tujuan paket tersebut. Sedangkan pada *point-to-point network*, komunikasi data terjadi melalui beberapa koneksi antara sepasang komputer, sehingga untuk mencapai tujuannya sebuah paket mungkin harus melalui beberapa komputer terlebih dahulu. Oleh karena itu, dalam tipe jaringan ini, pemilihan rute yang baik akan menentukan bagus tidaknya koneksi data yang berlangsung.

Klasifikasi jaringan komputer berdasarkan skala geografis :

a. *Local Area Network* (LAN)
b. *Metropolitan Area Network* (MAN)
c. *Wide Area Network* (WAN)

2.1.1 LAN (Local Area Network)

LAN *(Local Area Network)* merupakan suatu jaringan yang memiliki kecepatan tinggi dan mempunyai tingkat kesalahan yang kecil dalam cakupan geografis yang tidak besar. Biasanya LAN menghubungkan beberapa *workstation, printer* dan beberapa *device* yang lain. LAN memberikan beberapa keuntungan kepada penggunanya diantaranya pembagian hak akses *device* dan aplikasinya, pertukaran *file* antara pengguna, dan komunikasi antar pengguna. Suatu LAN dirancang untuk :

a. Beroperasi pada wilayah geografis yang terbatas.
b. Memperbolehkan beberapa *user* untuk mengakses *high-bandwidth* media.
c. Menyediakan koneksi pada *service* lokal secara *full-time*.
d. Menghubungkan *device-device* yang berdekatan secara fisik.

2.1.2 MAN (Metropolitan Area Network)

MAN *(Metropolitan Area Network)* merupakan suatu jaringan dengan jangkauan yang lebih luas dari LAN, biasanya terdiri dari dua atau lebih LAN dalam area geografis yang sama. Penggunaan MAN dapat mencakup perusahaan dengan kantor-kantor cabangnya di satu kota dan dapat berupa jaringan *private* ataupun jaringan *public*.

2.1.3 WAN (Wide Area Network)

WAN *(Wide Area Network)* merupakan suatu jaringan yang mencakup area geografis yang lebih luas, seperti negara atau benua, biasanya merupakan LAN yang saling terhubung yang menyediakan akses ke komputer atau *server* di lokasi lain. WAN menyediakan konektivitas *full-time* dan *part-time*, akses melalui antar muka *serial* dengan kecepatan rendah, kemampuan komunikasi *real-time* kepada *user*, sumber daya *remote* terhubung ke layanan lokal secara *full-time*, layanan *e-mail*, WWW (*World Wide Web*), pemindahan *file*, serta *e-commerce*, dan menghubungkan peralatan yang berjauhan yang dipisahkan dengan jarak yang luas bahkan area global.

2.2 Network Addressing

Setiap komputer pada suatu jaringan harus diberikan *identifier* unik agar dapat dibedakan dengan komputer lainnya. *Identifier* tersebut dapat berupa *physical address* dan *logical address*.

2.2.1 Physical Address

Physical Address atau MAC *address* (*Media Access Control address*) adalah sebuah alamat jaringan yang diimplementasikan pada lapisan *datalink* dalam tujuh lapisan model OSI, yang merepresentasikan sebuah *node* tertentu dalam jaringan. Dalam sebuah jaringan berbasis *Ethernet*, MAC *address* merupakan alamat yang unik yang memiliki panjang 48-*bit* (6 *byte*) yang mengidentifikasikan sebuah komputer, *interface* dalam sebuah *router*, atau *node* lainnya dalam jaringan. MAC *address* juga sering disebut sebagai *Ethernet*

address, physical address, atau *hardware address*.

Gambar 2.1 NIC *(Network Interface Card)*

Penulisan MAC *address* ini dilakukan dalam format heksadesimal.

Tabel 2.1 Format Heksadesimal Penulisan MAC *address*

hh hh hh	hh hh hh
Vendor id	Device id

Contoh dari MAC *address* :

Realtek RTL8139 Family PCI *Fast Ethernet* NIC memiliki MAC *address* 00-A1-B0-A2-45-5A, dimana :

- 00-A1-B0 disini menunjukkan *vendor* ID yakni *Realtek*
- A2-45-5A disini menunjukkan kode unik NIC komputer tertentu.

MAC *address* memang harus unik, dan untuk itulah, IEEE *(Institute of Electrical and Electronics Engineers*) mengalokasikan blok-blok dalam MAC *address*. 24 *bit* pertama dari MAC *address* merepresentasikan siapa pembuat

kartu tersebut, dan 24 *bit* sisanya merepresentasikan nomor kartu tersebut. Setiap kelompok 24 *bit* tersebut dapat direpresentasikan dengan menggunakan enam digit bilangan heksadesimal, sehingga menjadikan total 12 *digit* bilangan heksadesimal yang merepresentasikan keseluruhan MAC *address*. Berikut merupakan tabel beberapa pembuat kartu jaringan populer dan nomor identifikasi dalam MAC *address*. Berikut beberapa contoh nama *vendor* yang memiliki MAC *address* yang berbeda :

Tabel 2.2 Nama *Vendor* dengan MAC *Address* yang Berbeda

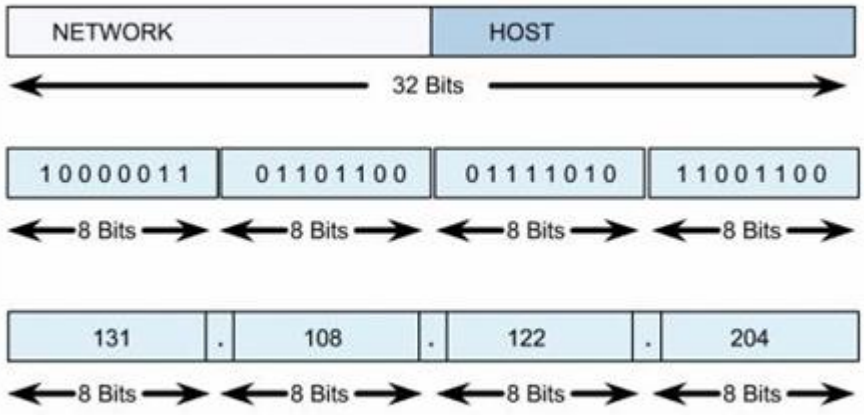
Nama vendor	Alamat MAC
Cisco Systems	00 00 0C
Cabletron Systems	00 00 1D
International Business Machine Corporation	00 04 AC
3Com Corporation	00 20 AF
GVC Corporation	00 C0 A8
Apple Computer	08 00 07
Hewlett-Packard Company	08 00 09

2.2.2 Logical Address

Logical address lebih dikenal sebagai IP *address*, yang berbeda dari MAC *address* dari segi pengalamatannya yang bersifat hierarkis. IP *address* yang umum digunakan adalah IP versi 4 (IPv4), walaupun sekarang sudah

dikembangkan IP versi 6 (IPv6) dan masih ada IPX yang digunakan pada *Novell network.*

IPv4 terdiri dari 32 *bit* yang terbagi dalam 4 segmen yang masing-masing terdiri dari 8 *bit* (1 oktet) dan dinyatakan dalam bentuk biner atau desimal. Walaupun dibagi dalam bilangan bertitik, namun pada dasarnya, komputer membaca IP *address* sebagai deretan bilangan biner sebanyak 32 *bit*. Bilangan bertitik ditambahkan pada bilangan tersebut agar manusia mudah untuk membaca dan menerjemahkan IP *address*.



Gambar 2.2 Susunan bit dalam IPv4

xxxxxxxx . xxxxxxxx . xxxxxxxx . xxxxxxxx → biner

xxx . xxx . xxx . xxx →desimal

Contoh :

11000000.10101000.00000000.00000001 → biner

Diterjemahkan menjadi bilangan desimal bertitik menjadi

192.168.0.1 →desimal

2.3 IP (Internet Protocol)

IP *(Internet Protocol)* adalah protokol lapisan jaringan (*network layer* dalam OSI *Reference Model*) atau protokol lapisan *internetwork* (*internetwork layer* dalam DARPA *Reference Model*) yang digunakan oleh protokol TCP/IP untuk melakukan pengalamatan dan *routing* paket data antar *host-host* di jaringan komputer berbasis TCP/IP. Pengalamatan jaringan *(network addressing*) yang sering digunakan pada IP adalah IPv4 dan MAC *address*.

2.3.1 IPv4

Arsitektur pengalamatan IPv4 yang diatur oleh ARIN *(American Registry for Internet Numbers)* didefinisikan menjadi lima kelas alamat, yaitu:

a. IP Kelas A

Pada kelas A, 8 *bit* pertama mengidentifikasikan *network* dan 24 *bit* sisanya mengidentifikasikan *host*. Kelas ini biasanya digunakan oleh perusahaan yang memiliki jaringan dalam skala yang besar karena memiliki *host* yang paling besar diantara kelas lainnya yaitu maksimum 2^{24}-2 *host*. Alamat IP pada kelas A dimulai dari 1.0.0.0 sampai 126.255.255.255. Salah satu cara untuk mengenali suatu IP adalah IP kelas A atau bukan, selain melihat pada *range* IPnya, dapat melihat pada *bit* pertama pada deretan bilangan binernya. *Bit* pertama pada deretan bilangan biner tersebut haruslah 0, maka itu termasuk dalam kategori IP kelas A.

Tabel 2.3 IP Kelas A

Network	Host	Host	Host
0xxxxxxx . xxxxxxxx . xxxxxxxx . xxxxxxxx			
0-127	0-255	0-255	0-255

b. IP Kelas B

Kelas B memiliki 16 *bit* pertama yang mengidentifikasikan *network* dan 16 *bit* berikutnya mengidentifikasikan *host*. Kelas B memiliki 2 *octet host number* yang memungkinkannya untuk menampung maksimum 2^{16}-2 *host*. Alamat IP kelas B biasanya digunakan untuk jaringan dengan skala menengah. Alamat IP pada kelas B berkisar antara 128.0.0.0 sampai 191.255.255.255. Selain mengenali IP kelas B dari *range* IPnya, IP kelas B dapat dikenali dari dua *bit* pertama pada deretan IP tersebut (dalam biner) yakni 10.

Tabel 2.4 IP Kelas B

Network	Network	Host	Host
10xxxxxx . xxxxxxxx . xxxxxxxx . xxxxxxxx			
128-191	0-255	0-255	0-255

c. IP Kelas C

Pada kelas C, 24 *bit* pertamanya mengidentifikasikan *network* dan 8 *bit* sisanya mengidentifikasikan *host*. Kelas ini memiliki maksimum 2^8-2 *host*. Kelas ini biasanya digunakan untuk jaringan berskala kecil. Alamat pada kelas C dimulai dari 192.0.0.0 sampai 223.255.255.255. IP kelas C dapat dikenali dari tiga *bit* pertama IP (dalam binernya) yakni 110.

Tabel 2.5 IP Kelas C

Network	Network	Network	Host
110xxxxx . xxxxxxxx . xxxxxxxx . xxxxxxxx			
192-223	0-255	0-255	0-255

d. IP Kelas D

IP kelas D mempunyai *byte* pertama 1110xxxx, sehingga rentang alamatnya dimulai dari 224-239. IP kelas D merupakan IP khusus yang tidak dapat dipakai oleh *public* karena satu blok IP kelas ini khusus dipakai untuk keperluan *multicast*. *Multicast* adalah jenis transmisi layaknya *broadcast*, namun dalam skala yang lebih kecil dan dapat ditentukan tujuannya.

Tabel 2.6 IP Kelas D

Host	Host	Host	Host
1110xxxx . xxxxxxxx . xxxxxxxx . xxxxxxxx			
224-239	0-255	0-255	0-255

e. IP Kelas E

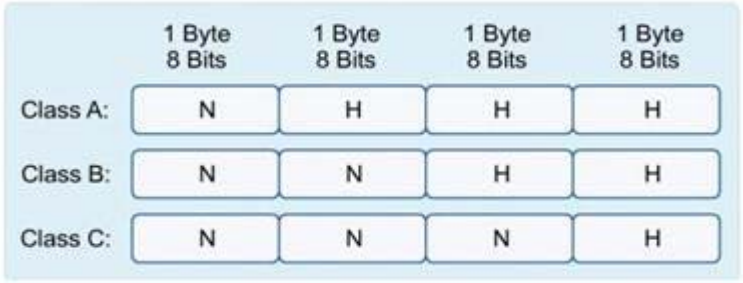
IP kelas E mempunyai *byte* pertama 1111xxxx. IP Kelas E adalah IP yang digunakan hanya untuk keperluan riset dan eksperimen.

Tabel 2.7 IP Kelas E

Host	Host	Host	Host
1111xxxx . xxxxxxxx . xxxxxxxx . xxxxxxxx			
240-255	0-255	0-255	0-255

Address Class	Number of Networks	Number of Hosts per Network
A	126*	16,777,216
B	16,384	65,535
C	2,097,152	254
D (Multicast)	N/A	N/A

IP Address Class	High-Order Bits	First Octet Address Range	Number of Bits in the Network Address
Class A	0	0 - 127*	8
Class B	10	128 - 191	16
Class C	110	192 - 223	24
Class D	1110	224 - 239	28



Gambar 2.3 Kelas – kelas IP dalam IPv4

2.3.2 Subnet Mask

Subnet mask adalah istilah teknologi informasi yang mengacu kepada angka biner 32 *bit* yang digunakan untuk membedakan *network* ID dengan *host* ID, menunjukkan letak suatu *host*, apakah berada di jaringan lokal atau jaringan luar.

RFC 950 mendefinisikan penggunaan sebuah *subnet mask* yang disebut juga sebagai sebuah *address mask* sebagai sebuah nilai 32-*bit* yang digunakan untuk membedakan *network identifier* dari *host identifier* di dalam sebuah alamat IP. *Bit-bit subnet mask* yang didefinisikan, adalah sebagai berikut :

a. Semua *bit* yang ditujukan agar digunakan oleh *network identifier* diset ke nilai 1.
b. Semua *bit* yang ditujukan agar digunakan oleh *host identifier* diset ke nilai 0.

Setiap *host* di dalam sebuah jaringan yang menggunakan TCP/IP membutuhkan sebuah *subnet mask* meskipun berada di dalam sebuah jaringan dengan satu segmen saja. Entah itu *subnet mask default* (yang digunakan ketika memakai *network identifier* berbasis kelas) ataupun *subnet mask* yang dikustomisasi (yang digunakan ketika membuat sebuah *subnet* atau *supernet*) harus dikonfigurasikan di dalam setiap *node* TCP/IP.

Network address didapat dengan melakukan proses AND antara IP *address* dengan *subnet mask*nya.

Contoh:

- IP *Address* :

  11000000 . 10101000 . 00010000 . 00100000 → 192.168.16.32

- *Subnet mask* :

  11111111 . 11111111 . 11111111 . 00000000 → 255.255.255.0

  ----------AND

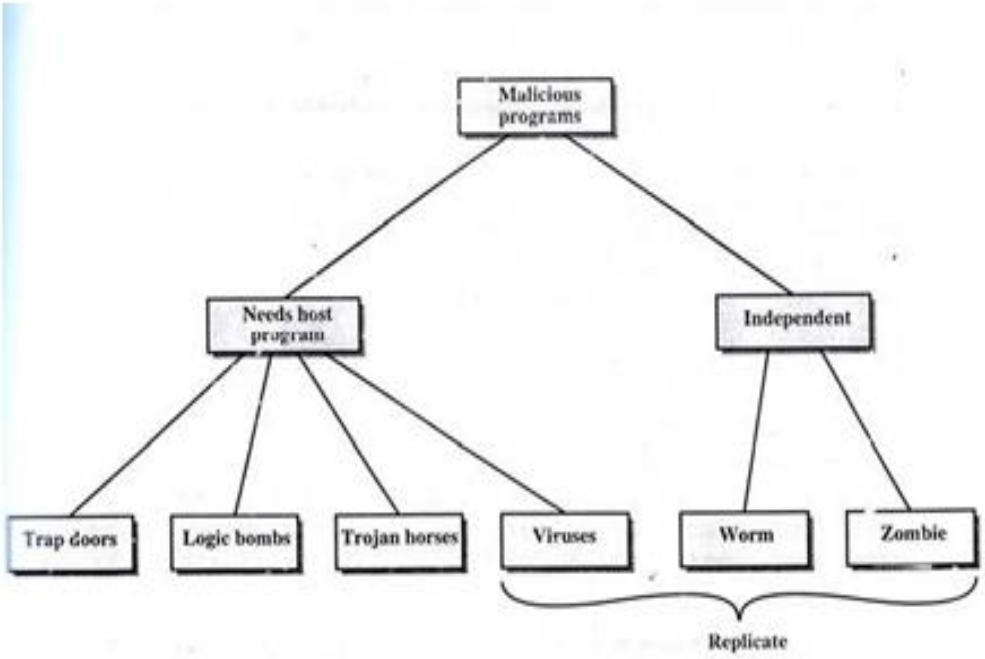
- *Network* ID

  11000000 . 10101000 . 00010000 . 00000000 → 192.168.16.0

Sebagai suatu syarat agar minimal dua *node* di dalam suatu jaringan dapat berkomunikasi satu sama lainnya, dua *node* tersebut harus memiliki IP *address* yang berbeda, namun memiliki *network* yang sama. Dengan demikian dua *node* dalam jaringan tersebut mampu berkomunikasi langsung melalui perantara jaringan biasa (*Hub* atau *Switch*) tanpa harus melewati perantara *router* terlebih dahulu.

2.4 Ancaman Jaringan

2.4.1 Program Berbahaya (Malicious Software)



Gambar 2.4 Taxonomy dari program berbahaya

Gambar di atas memberikan sebuah taksonomi keseluruhan dari ancaman dari keamanan jaringan, atau biasa disebut dengan program-program yang berbahaya. Ancaman-ancaman ini dapat dibagi dalam dua kategori :

1. Ancaman dalam bentuk program yang tidak dapat berjalan tanpa adanya suatu program aplikasi, utilitas atau program sistem (*dependent*).
2. Ancaman dalam bentuk program yang dapat berjalan tanpa adanya suatu program aplikasi, utilitas, atau program sistem serta dapat dijalankan oleh sistem operasi (*independent*).

Program-program ancaman ini juga dapat dibedakan menjadi dua, yaitu program yang dapat berduplikasi dan program yang tidak dapat berduplikasi. Program yang dapat berduplikasi adalah program *independent* yang ketika tereksekusi akan memproduksi satu atau lebih *copy* dari diri mereka sendiri untuk diaktifkan kemudian pada sistem yang sama atau sistem yang berbeda. Sedangkan program yang tidak berduplikasi adalah program yang akan diaktifkan ketika program *host* telah dipanggil untuk menjalankan satu fungsi tertentu.

Meskipun *taxonomy* dari gambar di atas dapat berguna dalam mengorganisasikan informasi yang sedang kita bicarakan, tetapi tidak dapat mencakup semuanya. Dalam kenyataannya, *bom logic* atau *trojan horse* dapat menjadi bagian dari *virus* atau *worm*.

a. Trap door

Trap door adalah suatu titik masuk rahasia ke dalam program yang memperbolehkan seseorang yang tahu keberadaan *trap door* untuk memperoleh akses tanpa melalui prosedur keamanan. *Trap door* telah sering digunakan oleh para *programmer* untuk *debug* dan *test* program. Ini biasanya digunakan ketika *programmer* mengembangkan aplikasi yang mempunyai prosedur autentikasi.

Untuk men*debug* program, *programmer* menginginkan suatu hak khusus atau cara menghindari semua autentikasi. *Trap door* menjadi ancaman ketika digunakan oleh *programmer* jahat untuk memperoleh akses yang tidak diperbolehkan.

b. Logic Bomb

Logic bomb adalah suatu kode yang ditanam dalam suatu program umum yang diatur untuk ‘meledak’ ketika suatu kondisi tertentu dicapai. Contoh kondisi yang bisa dipakai sebagai pemicu adalah kehadiran atau keabsenan suatu *file*, waktu tertentu, atau suatu *user* tertentu yang sedang menjalankan aplikasi.

c. Trojan horse

Trojan horse adalah suatu program yang tampaknya berguna bagi *user*, akan tetapi juga mengandung kode tersembunyi yang apabila diaktifkan akan melakukan suatu fungsi yang berbahaya atau tidak diinginkan. *Trojan horse* dapat digunakan untuk mencapai suatu tujuan tertentu secara tidak langsung dimana tujuan tersebut sebenarnya hanya bisa dicapai oleh *user* yang memiliki autentikasi yang benar. Motivasi umum lainnya dari *Trojan horse* adalah penghancuran data. Program terlihat melakukan fungsi yang berguna, padahal program tersebut secara diam menghapus *file-file user*.

d. Zombie

Zombie adalah program yang secara rahasia mengambil alih komputer yang tersambung dengan *internet* dan lalu menggunakan komputer tersebut untuk melakukan serangan yang sulit dilacak menuju tujuan komputer yang dibuat oleh pembuat *zombie*. *Zombie* digunakan dalam serangan *denial-of-service*, khususnya dalam menargetkan *website*.

e. Virus

Virus adalah sebuah program yang dapat meng'infeksi' program lain dengan memodifikasi mereka, hasil modifikasinya dapat menyebar ke program lain.

Virus dapat melekatkan dirinya pada program lain dan mengeksekusi dirinya sendiri secara rahasia saat program *host* sedang dijalankan. Setelah *virus* dieksekusi, *virus* dapat melakukan beberapa fungsi seperti menghapus *file* dan program.

Pada umumnya, *virus* bekerja pada sistem operasi tertentu dan dalam beberapa kasus bekerja pada *platform hardware* tertentu sehingga *virus* didesain untuk mengambil keuntungan mendetail dari kelemahan sistem tersebut.

f. Worm

Worm adalah program komputer yang menggandakan dirinya sendiri. Program ini menggunakan jaringan komputer untuk mengirimkan atau menggandakan diri ke sistem lain dan biasanya melakukan aksi-aksi yang tidak diinginkan, seperti menghabiskan sumber daya komputer atau mematikan sistem. Tidak seperti *virus*, *worm* tidak perlu menggunakan program *host*.

Program *worm* jaringan menggunakan koneksi jaringan untuk menyebar dari suatu sistem ke sistem lainnya. Sekali aktif dalam suatu sistem, sebuah *worm* jaringan bisa berlaku seperti *virus* komputer, atau dapat mengimplementasikan program *trojan horse*.

2.4.2 Intruder

Satu dari dua ancaman yang paling terkenal dalam hal keamanan jaringan selain virus adalah *intruder*. Secara umum dikenal sebagai *hacker* atau *cracker*. (Stallings 2003, p.566). Berikut ada tiga jenis *intruder*:

a. *Masquerader* adalah individual yang tidak diautorisasi untuk menggunakan komputer yang memasuki kontrol akses sistem untuk mengeksploitasi *account user* yang sah.
b. *Misfeasor* adalah *user* sah yang mengakses data, program, dimana sebenarnya akses ini tidak diperbolehkan, atau *user* yang diperbolehkan untuk akses tertentu tetapi menyalahgunakan izinnya.
c. *Clandestine user* adalah individu yang memperoleh kontrol dari sistem dan menggunakan kontrol ini untuk menghindari *auditing* dan mengakses data atau program yang ada.

Masqueader biasanya dari pihak luar, *misfeasor* biasanya dari pihak dalam, sedangkan *clandestine* user bisa dari pihak dalam maupun pihak luar. Biasanya, *intruder* memerlukan informasi yang diproteksi. Dalam kebanyakan kasus, informasi yang dimaksud dalam bentuk *password*.

2.5 Keamanan Jaringan

Menurut Stallings (2003, p4), arti dari keamanan jaringan adalah melindungi jaringan, tetapi melindungi dalam hal ini adalah masih mempunyai artian luas. Keamanan tidak hanya tentang menjaga orang–orang di dalam jaringan dari dunia luar. Akan tetapi juga menyediakan akses ke dalam jaringan dengan cara yang dikehendaki,

mempersilahkan orang-orang di dalam jaringan itu untuk bekerja sama. Ada beberapa elemen tentang keamanan jaringan yaitu:

a. *Integrity*

   Data yang diterima mestilah sama dengan yang diinginkan.

b. *Reliability*

   Data dapat digunakan secara baik tanpa ada halangan.

c. *Availability*

   Ketersediaan data jika diperlukan.

d. *Security*

   Data yang dikirim maupun yang diterima dilindungi dari akses yang tidak diinginkan.

Berikut adalah program-program yang bisa digunakan untuk lebih mengamankan suatu jaringan, yaitu :

2.5.1 Anti Virus

Anti virus adalah program untuk mendeteksi dan membersihkan komputer dari *virus* (Stallings 2003, p.609). Program *anti virus* ini memindai *file-file* tertentu untuk mendeteksi keberadaan *virus* tertentu. Karena banyak *virus*-*virus* baru yang bermunculan, maka program *anti virus* perlu di-*update* secara teratur untuk dapat mendeteksi dan membersihkan *virus–virus* yang baru.

Ada 4 (empat) generasi *antivirus* yaitu :

a. Generasi pertama : *scanner* sederhana.

   Scanner generasi pertama memerlukan *signature* dari *virus* untuk mengidentifikasi sebuah *virus*.

b. Generasi kedua : *scanner heuristic*.

Scanner generasi kedua tidak memerlukan *signature* yang spesifik, tetapi memakai aturan *heuristic* untuk mencari kemungkinan infeksi *virus*.

c. Generasi ketiga : *trap* aktivitas.

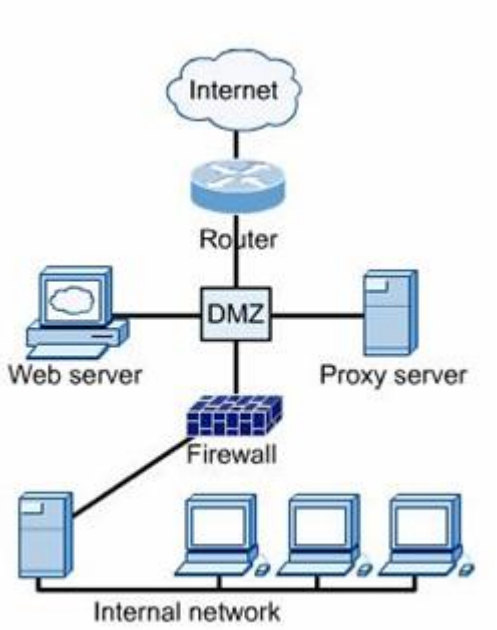
Program generasi ketiga adalah program yang mengidentifikasi sebuah *virus* berdasarkan aksinya daripada struktur dari program yang terinfeksi.

d. Generasi keempat : proteksi berfitur penuh.

Produk generasi keempat merupakan paket yang mengandung variasi teknik *anti virus* yang digunakan secara bersama. Ini termasuk *scanning* dan komponen *trap* aktivitas.

2.5.2 Firewall

Firewall adalah sebuah sistem atau perangkat yang mengizinkan lalu lintas jaringan yang dianggap aman untuk melaluinya dan mencegah lalu lintas jaringan yang tidak aman. Umumnya, sebuah *firewall* diimplementasikan dalam sebuah mesin terdedikasi, yang berjalan pada *gateway* antara jaringan lokal dan jaringan lainnya. *Firewall* umumnya juga digunakan untuk mengontrol akses terhadap siapa saja yang memiliki akses terhadap jaringan pribadi dari pihak luar. Saat ini, istilah *firewall* menjadi istilah generik yang merujuk pada sistem yang mengatur komunikasi antar dua jaringan yang berbeda. Mengingat saat ini banyak perusahaan yang memiliki akses ke *internet* dan juga tentu saja jaringan korporat di dalamnya, maka perlindungan terhadap aset *digital* perusahaan tersebut dari serangan para *hacker*, pelaku spionase, ataupun pencuri data lainnya, menjadi esensial.



Gambar 2.5 Penempatan Firewall di Dalam Jaringan

Filter paket data oleh *firewall* dilakukan berdasarkan beberapa kriteria, yaitu:

1. IP *Address*

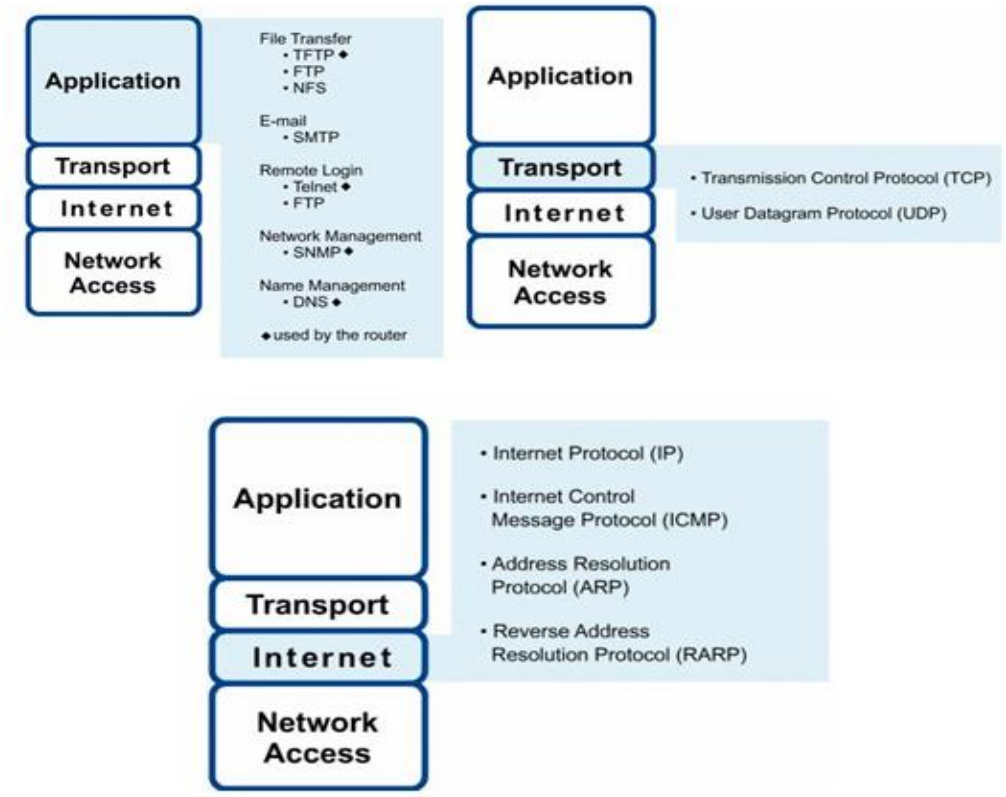
   IP *address* adalah unik. IP *address* terbagi atas 4 oktet, mewakili angka biner 32 *bit* dalam bentuk desimal. Fitur ini menyerupai *Packet Filtering Gateway*.

2. *Domain name*

   Merupakan nama yang dipetakan dari IP *address* yang fungsinya adalah agar IP *address* yang sangat susah diingat akan lebih mudah dituliskan. Dengan filter ini, *firewall* akan dapat memblokir nama *domain* tertentu atau hanya mengijinkan *domain* tertentu yang bisa diakses. Fitur ini menyerupai *Application Level Gateway*.

3. *Protocol*

Protocol merupakan satu set peraturan yang menjadi sebuah standar dalam menggunakan sebuah *service*. *Protocol* pada umumnya adalah teks dan secara sederhana menggambarkan bagaimana *client* dan *server* bisa berkomunikasi. *Protocol* masing-masing berperan di *layer* TCP/IP tertentu seperti digambarkan berikut ini.



Gambar 2.6 Protokol di Tingkat Masing-masing Layer TCP/IP

Beberapa *protocol* yang dapat dikendalikan oleh *firewall* antara lain:

a. IP (*Internet Protocol*) merupakan sistem pengiriman informasi melalui *internet*.

b. TCP *(Transmission Control Protocol*) digunakan untuk memecah dan membangun kembali informasi yang berjalan di dalam *internet*.

c. HTTP *(Hyper Text Transfer Protocol*), digunakan sebagai *protocol* untuk menampilkan halaman *web*.

d. FTP *(File Transfer Protocol*), digunakan untuk *download* dan *upload file* ke *internet*.

e. UDP *(User Datagram Protocol*) digunakan untuk informasi yang tidak memerlukan respons balik seperti *streaming audio* dan video.

f. ICMP *(Internet Control Message Protocol*) digunakan untuk *router* dalam hal pertukaran informasi antara satu *router* dengan yang lainnya.

g. SMTP *(Simple Mail Transfer Protocol*) digunakan untuk mengirim *email* dari *email client* ke *server e-mail* di *internet* sana.

h. SNMP *(Simple Netwokrk Management Protocol*) digunakan untuk mengumpulkan sistem informasi dari *remote* komputer atau *node* tertentu yang SNMP-*supported*.

i. *Telnet*, digunakan untuk mengendalikan komputer di lain tempat dan mengemulasikan terminal.

4. *Port*

Port yang dikenal dengan angka-angka tertentu adalah suatu pintu pada aplikasi tertentu yang berfungsi sebagai jalur tempat keluar masuknya data. Sebagai gambaran, HTTP memiliki *port* 80 dan DNS memiliki *port* 53. *Port* ini tidak tampak secara nyata namun merupakan pintu masuk suatu aplikasi atau layanan tertentu via jaringan. *Firewall* mampu menfilter berdasarkan jenis *port*, misalnya layanan selain 110, 25, dan 80

tidak diperbolehkan, artinya sang pengguna hanya mampu membuka *website*, menerima dan mengirim *email* saja. *Firewall* menggunakan satu atau lebih dari 4 metode yang ada dibawah ini untuk mengatur aliran data yang masuk ataupun yang keluar dari suatu jaringan (Roberta Bragg, 2004. p.231). Metode yang digunakan oleh *firewall* dalam menjalankan fungsinya antara lain:

a) *Packet Filter*

Pada *firewall* tipe ini, suatu paket yang masuk ke dalam jaringan tertentu akan dianalisa dengan beberapa rangkaian *filter*. *Firewall* jenis ini melakukan *filtrasi* terhadap paket-paket yang masuk ke dalam suatu jaringan berdasar sumber paket, tujuan paket, dan atribut dari paket-paket tersebut. Paket-paket yang berhasil di-*filter* tersebut kemudian akan diteruskan ke *requesting system* sedangkan yang tidak berhasil akan dibuang. *Firewall* jenis ini biasanya merupakan bagian dari sebuah *router firewall*.

b) *Application Gateway* (*Proxy Server*)

Suatu informasi yang berasal dari *internet* dan masuk ke dalam jaringan internal akan diambil dan dianalisa oleh *firewall* tipe ini dan kemudian akan dikirimkan ke *requesting system* ataupun sebaliknya, ditahan dan ditolak pada *proxy*. Berbeda dengan *Packet Filtering Firewall*, *firewall* jenis ini lebih spesifik karena melihat *content* dari paket data yang masuk ataupun keluar. *Proxy* ini memiliki sejumlah aturan tertentu, yang biasanya disebut dengan *Access Control List*, *Rules*, atau biasa disebut sebagai *policy* yang mengharuskan paket dengan isi, ukuran, destinasi,

sumber, tipe tertentu boleh masuk atau di-*reject* pada proxy ini. Sebagai contoh, jenis *firewall* ini mampu memblokir paket yang alamat sumbernya memiliki rangkaian kata-kata *"sex"* ataupun *file* dengan *extension "*.mp3".

c) *Circuit-level Gateway*

Firewall jenis ini bekerja pada bagian TCP (*Transport Control Protocol*) *layer* pada lapisan TCP/IP. *Firewall* jenis ini akan melakukan pengawasan terhadap pelaksanaan hubungan awal TCP atau yang biasa disebut sebagai *tcp-handshaking* untuk menentukan apakah hubungan tersebut diperbolehkan atau tidak. Pada prinsipnya, model *firewall* ini hampir serupa dengan *Application Level Gateway*, namun berbeda pada *layer* dimana ia melakukan *filtrasi* (berada pada *layer* keempat).

d) *Stateful Packet-Inspection Firewalls*

Firewall jenis ini adalah *firewall* dengan metode terbaru dimana metode tersebut tidak memeriksa isi dari setiap paket. Metode ini akan membandingkan paket yang diterima dengan *database* yang ada, yang dipercayai sebagai informasi yang masuk. Apabila perbandingan tersebut masuk ke dalam kriteria yang ada pada *database* maka informasi tersebut akan diizinkan untuk masuk ke dalam jaringan. Jika tidak, informasi tersebut akan dihapus. Pada prinsipnya, *firewall* jenis ini adalah penggabungan antara ketiga jenis *firewall* tersebut di atas. *Firewall* ini menjadi *firewall* yang sangat handal karena penggabungan fitur–fitur yang ada pada *firewall* lainnya. Selain itu, berkat penggabungan fitur–

fitur lainnya, *firewall* jenis ini memiliki tingkat keamanan yang teramat tinggi.

2.6 Linux

Istilah *Linux* atau GNU / *Linux* (GNU) juga digunakan sebagai rujukan kepada keseluruhan *distro Linux* (*Linux distribution*), yang didalamnya selalu disertakan program-program lain yang mendukung sistem operasi ini.

Contoh program-program tersebut adalah *web server*, bahasa pemrograman, *database*, tampilan *desktop* (*desktop environment*) (seperti GNOME dan KDE), dan aplikasi/*software* perkantoran (*office suite*) seperti *OpenOffice.org*, *KOffice*, *Abiword*, *Gnumeric*, dan lainnya. *Distro Linux* telah mengalami pertumbuhan yang pesat dari segi popularitas, sehingga lebih populer dari versi UNIX yang menganut sistem lisensi dan berbayar maupun versi UNIX bebas lain yang pada awalnya menandingi dominasi *Microsoft Windows* dalam beberapa sisi.

Linux mendukung banyak perangkat keras komputer, dan telah digunakan di dalam berbagai peralatan dari komputer pribadi, *Super Computer* dan *Embedded System* (seperti telepon seluler dan perekam video pribadi *Tivo*).

Pada mulanya, *Linux* dibuat, dikembangkan dan digunakan oleh peminatnya saja. Kini *Linux* telah mendapat dukungan dari perusahaan besar seperti IBM, *Hewlett-Packard* dan perusahaan besar lainnya. Para pengamat teknologi informatika beranggapan kesuksesan ini dikarenakan *Linux* tidak bergantung kepada *vendor* (*vendor-independence*), biaya operasional yang rendah, dan kompatibilitas yang tinggi dibandingkan versi UNIX berbayar, serta faktor keamanan dan kestabilannya dibandingkan dengan *Microsoft Windows*. Ciri–ciri ini juga menjadi bukti atas

keunggulan model pengembangan perangkat lunak sumber terbuka *open source software*.

Kernel Linux pada mulanya ditulis sebagai proyek hobi oleh pelajar universitas *Finland Linus Torvalds* yang belajar di Universitas *Helsinki*, untuk membuat *kernel Minix* yang gratis dan dapat diedit. (*Minix* adalah projek pelajaran menyerupai UNIX dibuat untuk mudah digunakan dan bukannya untuk digunakan secara komersial.)

Sejarah sistem operasi *Linux* berkaitan erat dengan proyek GNU, proyek program bebas *freeware* terkenal diketuai oleh Richard Stallman. Proyek GNU diawali pada tahun 1983 untuk membuat sistem operasi seperti *Unix* lengkap *compiler*, utilitas aplikasi, utilitas pembuatan dan seterusnya diciptakan sepenuhnya dengan perangkat lunak bebas. Pada tahun 1991, pada saat versi pertama kerangka *Linux* ditulis, proyek GNU telah menghasilkan hampir semua komponen sistem ini kecuali *kernel*. Torvalds dan pembuat *kernel* seperti *Linux* menyesuaikan *kernel* mereka supaya dapat berfungsi dengan komponen GNU, dan seterusnya mengeluarkan sistem operasi yang cukup berfungsi. Oleh karena itu, *Linux* melengkapi ruang terakhir dalam rancangan GNU. Walaupun kernel *Linux* dilisensikan di bawah GNU *General Public License*, ia tidak sebesar proyek GNU.

Bagi mereka yang hanya biasa menggunakan *Windows* atau *Macintosh*, *Linux* mungkin kelihatan lebih sukar disebabkan perbedaan dalam melakukan berbagai kerja komputer. Dan lagi, pengguna perlu menukar program yang sering digunakan, disebabkan program tersebut tidak didapati dalam *Linux* (atau pilihan yang agak terbatas, misalnya permainan komputer). Faktor lain adalah sifat ragu-ragu pengguna yang merasa susah untuk melepaskan sistem operasi mereka (banyak pengguna masih menggunakan *Windows*). Selain itu, kebanyakan komputer didatangkan dengan

Windows siap pakai (*pre-installed*). Faktor-faktor ini menyebabkan perkembangan *Linux* yang agak lambat.

Walau bagaimana pun, kelebihan *Linux* seperti biaya rendah, sekuritas yang lebih aman, dan tidak bergantung pada *vendor*, telah meningkatkan penggunaan yang luas di kalangan korporasi dan perkantoran.

Proses pemasangan (instalasi) yang sukar seringkali menjadi penghalang bagi pengguna baru, namun proses ini sekarang menjadi lebih mudah akhir-akhir ini. Dengan penerimaan Linux oleh beberapa pabrikan PC (komputer pribadi) terbesar, komputer *built-up* dengan distribusi *Linux* banyak ditemukan. Ada juga distribusi *Linux* yang dapat di-*boot* secara langsung dari CD (*Live* CD) tanpa perlu memasangnya ke dalam *Hard Disk*. Contoh-contoh distribusi *Linux* berbentuk *Live* CD adalah *Knoppix/Gnoppix*, *Kubuntu/Ubuntu* dan *Gentoo*. Bahkan saat ini hampir semua distribusi *Linux* menyediakan *Live* CD bagi produknya. Format ISO bagi CD untuk distribusi *Linux* tersebut biasanya dapat di-*download* dari *Internet*, ditulis ke dalam CD ,dan selanjutnya dapat digunakan sebagai *bootable* CD.

2.7 Piranti Lunak Keamanan dan Pemantauan Jaringan

2.7.1 IP-Tables

IP-Tables merupakan suatu kumpulan instruksi-instruksi yang berada di dalam *kernel Linux* yang mengizinkan modul-modul *kernel* untuk melakukan pemanggilan kembali paket-paket yang melalui suatu jaringan lalu lintas data. Didalamnya juga dapat ditambahkan aturan-aturan yang bertujuan untuk melakukan penyaringan terhadap paket-paket data yang masuk.

Fungsi utama *IP-tables* adalah:

a. *Stateless Packet Filtering*
b. *Stateful Packet Filtering*
c. Tiap jenis NAT (*Network Address Translation*) atau NAPT
d. Fleksibel dan Infrastuktur yang luas

2.7.2 Squid

Squid merupakan sebuah *proxy server* yang mempunyai kehandalan yang sangat baik untuk *web client,* FTP, *gopher,* dan objek-objek data HTTP. *Proxy* melakukan fungsi *caching* dalam menangani permintaan terhadap suatu paket data. *Caching internet object* merupakan cara untuk menyimpan objek-objek *internet* yang diminta oleh FTP, HTTP, dan *gopher*. *Caching* dari *Squid* membantu *web browser* untuk menghemat *bandwidth*.

2.7.3 ClamAV

ClamAV merupakan *software anti virus* keluarga UNIX yang banyak digunakan dalam integrasi bersama dengan *mail server* untuk melakukan *scanning* terhadap sistem dan *email* (terutama yang memiliki *attachment* didalamnya). Aplikasi ini menyediakan *daemon* yang fleksibel, pendeteksian per-baris, dan alat untuk *update* otomatis dari *internet*. Proteksinya juga terbilang cukup baik karena *database antivirus*-nya dapat dengan mudah di-*update*.

2.7.4 Spamassasin

Spamassasin adalah *filter mail* untuk mendeteksi *email* yang merupakan *spam*. *Spamassasin* dapat mendeteksi *spam* dengan melihat bagian *header* maupun *content* menggunakan data statistik hasil penyaringan *email*. Keunggulan *Spamassasin* dari yang lain adalah menggunakan banyak teknik pendekatan, modular, dan dapat diperbaharui melalui pengembangan *anti spam* yang ada.

Fitur-fitur yang ditawarkan *Spamassasin*:

a. Tes Judul *email*
b. Tes frase dalam isi *email*
c. Penyaringan dengan metode *Bayes*
d. Alamat *Whitelist* dan *Blacklist* otomatis
e. Kolaborasi dengan *database spam* seperti *Pyzor*

2.7.5 Snort

Snort adalah IDS (*Intrusion Detection System)* yang dapat berguna untuk analisa *real-time traffic* dan *logging* paket data dalam jaringan IP. *Snort* berfungsi mencocokkan protokol yang digunakan oleh paket data dan mendeteksi berbagai serangan jaringan. *Snort* memberikan fitur filtrasi dengan menggunakan aturan-aturan dalam lalu lintas paket. *Snort* juga memiliki fitur peringatan yang *real-time* sehingga tindak lanjut terhadap suatu aksi dapat dilakukan dengan segera.

Fungsi utama *Snort*, antara lain :

a. *Packet Sniffer*
b. *Packet Logger*
c. IDS (*Intrusion Detection System*)

2.8 UTM (Unified Threat Management)

Menjaga keamanan jaringan perusahaan menjadi semakin menantang tiap tahunnya, dan keamanan jaringan telah menjadi salah satu isu paling penting yang dihadapi oleh bisnis sekarang ini. Ancaman yang baru dan terus berubah bermunculan dalam jumlah yang banyak, dan tidak ada organisasi yang kebal terhadap resiko tersebut. (http://www.watchguard.com/docs/whitepaper/wg_UTMBusinessOverview.pdf)

Setiap kali suatu ancaman baru yang lebih canggih muncul, defenisi inti dari "secure network" berubah saat itu juga. Menurut IBM *Internet Security Systems X-Force Research and Development Team*, lebih dari 7.247 titik kelemahan baru keamanan jaringan *Internet* yang ditemukan pada tahun 2006, dan 88,4% dari itu semua dapat dieksploitasi secara jarak jauh.

Saat suatu jaringan disusup oleh *intruders*, serangan *Denial of Service* (DoS), atau *virus*, keseluruhan organisasi menjadi terancam. Hal ini dapat menyebabkan sumber daya operasional perusahaan, data pelanggan, *tools* dan teknologi, dan modal intelektual berada dalam ancaman pencurian, penyalahgunaan, atau perusakan oleh pihak ketiga. Serangan jaringan dapat berupa banyak bentuk, yaitu :

1. *Network Intrusion*

   Pada skenario *intrusion*, seorang *hacker* yang tidak memliki hak akses mencoba untuk memasuki jaringan secara jarak jauh untuk tujuan yang tidak baik.

2. DoS / DDoS *Attacks*

   Pada serangan *DoS*, sistem atau jaringan dibuat menjadi tidak berguna, dengan cara memonopoli sumber daya sistem. *Distributed Denial of Service* (DDoS) melibatkan banyak sistem komputer (kemungkinan ratusan) semuanya mengirimkan traffic ke target tertentu.

3. *Virus* dan *Worms*

   Virus adalah program komputer yang menginfeksi program lain dengan copy dari dirinya sendiri, tetapi ditransfer dari sistem ke sistem lain dengan beberapa mekanisme luar seperti *e-mail*. *Virus* melakukan kerusakan ketika program yang terinfeksi dijalankan. Ini sangat berbeda dengan *worm*, yang merupakan program komputer yang memiliki kapasitas untuk meng-*copy* dirinya sendiri secara berulang-ulang pada sistem komputer yang lain. *Worm* dapat membawa *code* yang membahayakan.

4. *Adware* dan *Spyware*

   Adware adalah aplikasi *software* yang meng-*install* dirinya sendiri, biasanya tanpa izin *user*, dan menampilkan *banner-banner* iklan sembari program itu berjalan. Mereka muncul dalam bentuk *pop-up window* atau *bar* yang muncul di layar komputer. Aplikasi ini juga dapat mengubah *properties* dari *browser* seperti *home page*. *Spyware* mirip dengan

adware tetapi biasanya tidak menampakkan diri dengan *pop-up* atau cara lain. *Spyware* menggunakan kode untuk mendapatkan informasi pribadi dari *user* dan mengirimkannya ke pihak ketiga tanpa sepengetahuan *user*.

5. *Rootkits*

   Rootkit menggabungkan diri pada sistem operasi dan menghalangi perintah-perintah yang digunakan untuk menjalankan fungsi dasar, seperti mengakses *file* pada *hard drive*. *Rootkit* bersembunyi antara sistem operasi dan program yang bergantung pada sistem operasi tersebut. *Rootkit* mengontrol apa yang dapat dilihat dan dilakukan oleh program tersebut.

6. *DNS Poisoning – Domain Name System* (DNS) *server* dimanipulasi untuk mengalihkan tujuan *traffic* untuk menuju ke situs *web* yang berbahaya.

Suatu jaringan dapat juga menjadi lemah setiap kali bisnis berkembang dan berubah. Di saat jaringan menjadi semakin kompleks dan diharapkan untuk mendukung tujuan bisnis, suatu *firewall* yang *simple* tidak memiliki kapasitas untuk menyediakan keamanan yang dibutuhkan pada jaringan tersebut. Di sinilah solusi UTM (*Unified Threat Management*) dapat menjadi jawaban yang tepat.

IDC (*International Data Corporation*), yang merupakan sebuah perusahaan analisis dan penelitian pasar yang mengkhususkan dalam Teknologi Informasi dan Telekomunikasi, mendefinisikan aplikasi keamanan UTM (*Unified Threat Management*) sebagai suatu produk yang menggabungkan dan mengintegrasikan berbagai fitur keamanan menjadi suatu *platform hardware* tunggal. Kualifikasi didalam kategori ini meliputi kemampuan *firewall* jaringan, *network* IDP (*Intrusion Detection and*

Prevention), dan *anti virus gateway*. Dalam kenyataan, perusahaan besar menawarkan layanan kontrol keamanan bervariasi berdasarkan kebutuhan.

Intensi awal dari UTM adalah konsolidasi dari beberapa teknologi keamanan jaringan menjadi satu sistem. Awalnya, solusi UTM adalah *firewall* yang mengintegrasikan *Intrusion Detection/Prevention Systems* (IDS/IPS). Saat ini, banyak solusi UTM juga memasukkan *Anti-X* (*Spam* dan *malware*) dan fungsi VPN. UTM terus berkembang dengan fungsionalitas yang meluas. (http://www.cisco.com/en/US/solutions /collateral/ns340/ns517/ns224/ns376/white_paper_Defining_the_U_in_UTM.pdf)

Teknologi UTM yang potensial meliputi: *Identity management, Data Leak Prevention* (DLP), VOIP *security gateways, Unified Communications and Collaboration* (UCC) *security gateway*, dan *network access control* (NAC). Beberapa solusi UTM bahkan menambahkan fitur *routing* dan *switching*.

2.9 Web Server

Web Server adalah program aplikasi *server* yang menangani *request* halaman *web* kepada *client* melalui jaringan komputer. *Web Server* menyimpan halaman-halaman *web, script*, program, *file* pendukung, dan menyajikannya dengan HTTP. HTTP (*HyperText Transfer Protocol*) adalah protokol *layer* aplikasi *web* yang dipakai *web server*. HTTP menggunakan *port* 80 untuk berkomunikasi. HTTP diimplementasikan pada dua program, yaitu program *client* dan program *server*.

2.9.1 Apache

Apache Web Server adalah salah satu aplikasi *web server* yang sering digunakan untuk menangani halaman *web* yang berbasis HTML, PHP, XML, dan sebagainya. *Apache* merupakan *web server* dengan lisensi *open source* yang berarti penggunaannya dapat dipakai dengan luas dan pengembangannya dapat dilakukan oleh semua orang. *Apache* pertama kali dikembangkan oleh NCSA (*National Center for Supercomputing Applications*), *University of Illinois*.

Apache menawarkan beberapa karakteristik berikut :

a. Handal dan flesibel serta memenuhi standar HTTP / 1.1.
b. Dapat dikonfigurasi dan diperluas dengan modul *third party*.
c. Dapat diubah dengan menggunaka *Apache* API (*Application Program Interface*).
d. Pada *web server Apache* disediakan *source code* yang bersifat *open source.*
e. Dapat dijalankan pada sistem operasi yang berbeda (*multiplatform*).

2.10 Perl dan PHP

Perl (*Practical Extraction and Report Language*) adalah sebuah bahasa pemrograman yang dikembangkan oleh Larry Wall. Bahasa *perl* diimplementasikan dalam sebuah *interpreter* yang bernama *perl* yang tersedia untuk berbagai macam *operating system*, mulai dari UNIX, MS-DOS, sampai ke *Macintosh*.

Perl sangat bermanfaat dan mudah digunakan untuk memproses data, baik yang berbentuk teks maupun yang berbentuk *binary*. Ini disebabkan *perl* memiliki fasilitas

pattern matching dan *regular expression* yang sangat ampuh. Inilah yang menyebabkan *perl* sering digunakan untuk mengimplementasikan CGI-*script* di *World Wide Web*.

PHP merupakan bahasa pemrograman *web* yang bersifat *server-side* dimana semua operasi (pengambilan data, pencatatan data, penghapusan data, dan sebagainya) dilakukan di *server* sehingga keamanan *database* terjamin. PHP diperkenalkan pada tahun 1995 oleh Rasmus Lerdof, yang pada awalnya berupa *script perl* sederhana yang digunakan untuk menghitung pengunjung yang datang ke situs Rasmus. Versi pertama diperkenalkan dengan nama *Personal Home Page Tools* yang kemudian sampai saat ini telah berkembang menjadi versi 5 yang menggunakan *Zend Engine*.

2.11 Database

Menurut Connolly dan Begg (2005, p15), *database* adalah kumpulan-kumpulan data yang saling berhubungan secara logis dan deskripsi dari data-data tersebut, yang dirancang untuk memenuhi kebutuhan informasi dari perusahaan. Menurut Ramakrishnan dan Gehrke (2003, p4), *database* adalah sebuah kumpulan data yang mendeskripsikan aktivitas-aktivitas dari satu atau lebih organisasi yang berhubungan.

Dari kedua pengertian diatas, dapat disimpulkan bahwa *database* adalah kumpulan data yang saling berhubungan dan menggambarkan aktivitas atau proses bisnis dalam suatu perusahaan.

Software atau aplikasi yang bertugas untuk mengatur, menyimpan, memodifikasi data disebut dengan *software database engine* dan lebih resminya disebut dengan DBMS (*DataBase Management System*). Ada banyak sekali aplikasi DBMS ini mulai yang berjalan di PC (*Personal Computer*) sampai ke komputer skala *mainframe*. Contoh-contoh dari aplikasi *database engine* misalnya seperti:

a. SQL *Server*, dibuat oleh *Microsoft*.

b. MS *Access*, dibuat oleh *Microsoft*.

c. *Oracle Database*, dibuat oleh *Oracle*.

d. MySQL, dibuat oleh MySQL AB.

e. *Firebird*, dibuat oleh komunitas *open source* berdasarkan dari kode *Interbase*.

f. *PostgreSQL*, dibuat oleh komunitas *open source*.

g. DB2, dibuat oleh IBM.

2.11.1 MySQL

MySQL adalah sebuah RDBMS (*Relational DataBase Management System*) yang bersifat *open source*. Dengan konsep RDBMS, MySQL tidak menyimpan data kedalam sebuah area yang besar namun ke dalam tabel-tabel *database*. MySQL mengimplementasikan konsep *client-server* yang terdiri dari *daemon mysqld* dan beragam jenis aplikasi *client* dan *library*.

MySQL adalah *open source database server* yang menawarkan fitur yang tidak kalah baiknya dengan *database server* lain serperti SQL *Server 2000*. MySQL pertama kali dikeluarkan tahun 1998 yang penggunaannya langsung mendapatkan tempat istimewa di dunia pengembangan. *MySQL* juga bersifat *multiplatform* dan dapat digunakan dalam UNIX, *MacOS*, dan *Microsoft Windows*.

Awalnya MySQL dijalankan pada sistem operasi *Unix* dan *Linux*. Tapi, bagi para penggemar *Windows* pun sekarang sudah tersedia MySQL versi *Windows*nya. Mereka yang menggunakan *Linux* (*RedHat*, *Mandrake*, dsb),

biasanya MySQL sudah ter-*install* secara *default*. Bila belum bisa di-*install* maka dengan cukup mudah dapat meng-*install*-nya menggunakan RPM (*Redhat Package Manager*).