BAB 2
TINJAUAN PUSTAKA

1.1 Jaringan Komputer

Jaringan komputer adalah sebuah kumpulan perangkat-perangkat komputer dan jaringan yang saling terhubung. Informasi dan data bergerak melalui kabel-kabel sehingga memungkinkan pengguna jaringan komputer dapat saling bertukar informasi, mencetak pada printer yang sama dan bersama-sama menggunakan *hardware*/*software*. Tiap komputer, printer atau perangkat keras yang terhubung dengan jaringan disebut *node*. Sebuah jaringan komputer dapat memiliki dua, puluhan, ribuan atau bahkan jutaan *node*.

1.1.1 Jenis Jaringan Komputer

Ada 3 macam jenis jaringan, yaitu :

1. *Local Area Network* (LAN)

   LAN adalah jaringan yang dibatasi oleh area yang relatif kecil, umumnya dibatasi oleh area lingkungan seperti sebuah perkantoran di sebuah gedung atau sebuah sekolah dan tidak jauh dari sekitar 1 km persegi.

2. *Metropolitan Area Network* (MAN)

   MAN meliputi area yang lebih besar dari LAN, misalnya antar wilayah dalam satu provinsi. Dalam hal ini jaringan menghubungkan beberapa buah jaringan-jaringan kecil ke dalam lingkungan area yang lebih besar, sebagai contoh yaitu jaringan bank dimana beberapa kantor cabang sebuah bank di dalam sebuah kota besar dihubungkan antara satu dengan lainnya.

3. *Wide Area Network* (WAN)

   WAN adalah jaringan yang lingkupnya sudah menggunakan media satelit atau kabel bawah laut, sebagai contoh keseluruhan jaringan bank yang ada di Indonesia atau yang ada di negara-negara lain.

1.1.2 Arsitektur Jaringan Komputer

Arsitektur sebuah jaringan komputer dibedakan menjadi arsitektur fisik dan arsitektur logik. Arsitektur fisik berkaitan dengan susunan fisik sebuah jaringan komputer, bisa juga disebut dengan topologi fisik jaringan yaitu menjelaskan hubungan perkabelan dan lokasi *node* (simpul) atau *workstation*. Sedangkan arsitektur logik berkaitan dengan logika hubungan masing-masing komputer dalam jaringan atau menjelaskan aliran data dari satu *user* ke *user* lainnya dalam jaringan.

Topologi adalah bentuk hubungan dari suatu jaringan (*map of network*). Topologi fisik dari sebuah jaringan adalah merujuk pada konfigurasi kabel, komputer, dan perangkat lainnya. Terdapat tiga topologi fisik dasar yaitu: topologi *bus*,topologi *star*, dan topologi *ring*.

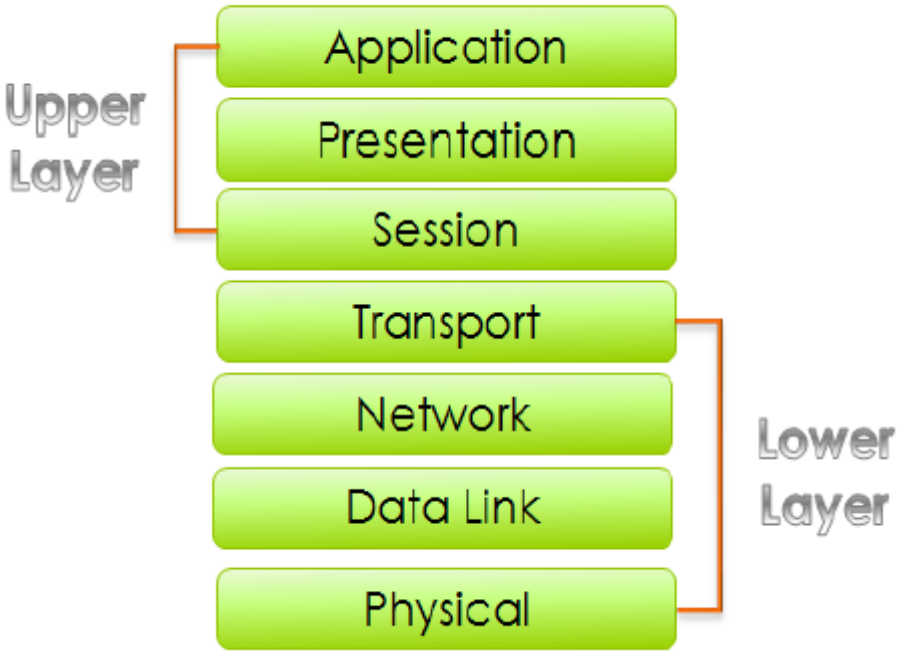
1.2 Protokol Jaringan Komputer

Pada saat *network* baru muncul, kebanyakan komputer hanya dapat berkomunikasi dengan komputer yang dibuat oleh perusahaan yang sama. Sebagai contoh, perusahaan harus menggunakan seluruh solusi dari DECnet atau seluruh solusi dari IBM, tapi tidak bisa kedua – duanya. Pada akhir tahun 1970, *International Organization for Standarization* (ISO) membuat model referensi *Open System Interconnection* (OSI) sebagai solusi untuk mengatasi masalah kompatibilitas ini. Model OSI dimaksudkan untuk membantu para *vendor* (*vendor* di sini adalah perusahaan pembuat perangkat keras atau pembuat perangkat lunak) agar bisa membuat alat – alat dan perangkat lunak yang bisa saling bekerja sama, dalam bentuk protokol – protokol sehingga *network* dengan *vendor – vendor* yang berbeda bisa saling bekerja sama.

1.2.1 Model Referensi OSI

The International Organization for Standarization (ISO) mengembangkan model referensi *Open System Interconnection* (OSI) sebagai pemandu untuk mendefinisikan sekumpulan protokol terbuka. Walaupun minat pada protokol OSI telah berkurang, model referensi OSI tetap merupakan standar yang paling umum

untuk menggambarkan dan membandingkan rangkaian protokol. Model OSI memiliki tujuh *layer* (lapisan).



Gambar 2.1 OSI Layer

a. *Physical Layer* (Lapisan Fisik)

Berfungsi untuk mendefinisikan media transmisi jaringan, metode pensinyalan, sinkronisasi bit, arsitektur jaringan (seperti halnya *Ethernet* atau *Token Ring*), topologi jaringan dan pengabelan. Selain itu, *level* ini juga mendefinisikan bagaimana *Network Interface Card* (NIC) dapat berinteraksi dengan media kabel atau radio[2].

b. *Data Link Layer* (Lapisan Data Link)

Befungsi untuk menentukan bagaimana bit-bit data dikelompokkan menjadi format yang disebut sebagai *frame*. Selain itu, pada *level* ini terjadi koreksi kesalahan, *flow control*, pengalamatan perangkat keras (seperti halnya Media *Access Control Address* (MAC Address)) dan menentukan bagaimana perangkat- perangkat jaringan seperti *hub*, *bridge*, *repeater*, dan *switch layer* 2 beroperasi. Spesifikasi IEEE 802, membagi

level ini menjadi dua level anak, yaitu lapisan *Logical Link Control* (LLC) dan lapisan *Media Access Control* (MAC)[2].

c. *Network Layer* (Lapisan Jaringan)

Network layer berfungsi untuk pengendalian operasi *subnet* dengan meneruskan paket-paket dari satu *node* ke *node* lain dalam jaringan. Masalah desain yang penting adalah bagaimana caranya menentukan *route* pengiriman paket dari sumber ke tujuannya. *Route* dapat didasarkan pada tabel statik yang dihubungkan ke *network*. *Route* juga dapat ditentukan pada saat awal percakapan misalnya *session terminal*. *Route* juga sangat dinamik, dapat berbeda bagi setiap paketnya, dan karena itu, *route* pengiriman sebuah paket tergantung beban jaringan saat itu. Bila pada saat yang sama dalam sebuah *subnet* terdapat terlalu banyak paket, maka ada kemungkinan paket-paket tersebut tiba pada saat yang bersamaan. Hal ini dapat menyebabkan terjadinya *bottleneck* (penyempitan di bagian ujung). Pengendalian kemacetan seperti itu juga merupakan tugas *network layer*[3].

d. *Transport Layer* (Lapisan Transport)

Fungsi dasar *transport layer* adalah menerima data dari *session layer*, memecah data menjadi bagian-bagian yang lebih kecil bila perlu, meneruskan data ke *network layer*, dan menjamin bahwa semua potongan data tersebut bisa tiba di sisi lainnya dengan benar. Selain itu, semua hal tersebut harus dilaksanakan secara efisien, dan bertujuan dapat melindungi *layer-layer* bagian atas dari perubahan teknologi *hardware* yang tidak dapat dihindari.

Dalam keadaan normal, *transport layer* membuat koneksi jaringan yang berbeda bagi setiap koneksi *transport* yang diperlukan oleh *session layer*. Bila koneksi *transport* memerlukan *throughput* yang tinggi, maka *transport layer* dapat membuat koneksi jaringan yang banyak. *Transport layer* membagi-bagi pengiriman data ke sejumlah jaringan untuk meningkatkan *throughput*[3].

e. *Session Layer* (Lapisan Sesi)

Session layer mengijinkan para pengguna untuk menetapkan *session* dengan pengguna lainnya. *Layer* ini membuka, mengatur dan menutup suatu *session* antara aplikasi-aplikasi. Sebuah *session* selain memungkinkan *transport* data biasa, seperti yang dilakukan oleh *transport layer*, juga menyediakan layanan yang istimewa untuk aplikasi-aplikasi tertentu. Sebuah *session* digunakan untuk memungkinkan seseorang pengguna *log* ke *remote timesharing system* atau untuk memindahkan *file* dari satu mesin ke mesin lainnya.

Sebuah layanan *session layer* yang lain adalah untuk melaksanakan pengendalian dialog. *Session* dapat memungkinkan lalu lintas bergerak dalam bentuk dua arah pada suatu saat, atau hanya satu arah saja. Jika pada suatu saat lalu lintas hanya satu arah (analog dengan rel kereta api tunggal), *session layer* membantu untuk menentukan giliran yang berhak menggunakan saluran. Layanan *session* di atas disebut manajemen *token*. Untuk sebagian protokol, adalah penting untuk memastikan bahwa kedua pihak yang bersangkutan tidak melakukan operasi pada saat yang sama. Untuk mengatur aktivitas ini, *session layer* menyediakan *token-token* yang dapat digilirkan. Hanya pihak yang memegang *token* yang diijinkan melakukan operasi kritis[3].

f. *Presentation Layer* (Lapisan Presentasi)

Presentation layer melakukan fungsi-fungsi tertentu yang diminta untuk menjamin penemuan sebuah penyelesaian umum bagi masalah tertentu. Selain memberikan sarana-sarana pelayanan untuk konversi, format dan enkripsi data, *presentation layer* juga bekerja dengan file berformat ASCII, EBCDIC, JPEG, MPEG, TIFF, PICT, MIDI, dan Quick Time.

Presentation layer tidak mengijinkan pengguna untuk menyelesaikan sendiri suatu masalah. Tidak seperti *layer-layer* di bawahnya yang hanya melakukan pemindahan bit dari satu tempat ke tempat lainnya,

presentation layer memperhatikan *syntax* dan semantik informasi yang dikirimkan.

Satu contoh layanan *presentation* adalah encoding data. Kebanyakan pengguna tidak memindahkan *string bit biner* yang *random*. Para pengguna saling bertukar data seperti nama orang, tanggal, jumlah uang, dan tagihan. Item-item tersebut dinyatakan dalam bentuk *string* karakter, bilangan *integer*, bilangan *floating point*, struktur data yang dibentuk dari beberapa item yang lebih sederhana. Terdapat perbedaan antara satu komputer dengan komputer lainnya dalam memberi kode untuk menyatakan *string* karakter contohnya ASCII dan Unicode, *integer* contohnya komplemen satu dan komplemen dua dan lain sebagainya.

Untuk memungkinkan dua buah komputer yang memiliki *presentation* yang berbeda untuk dapat berkomunikasi, struktur data yang akan dipertukarkan dapat dinyatakan dengan cara abstrak, sesuai dengan ”*encoding standard*” yang akan digunakan “pada saluran”. *Presentation layer* mengatur data struktur abstrak ini dan mengkonversi dari *representation* yang digunakan pada sebuah komputer menjadi ”*representation standard*” jaringan, dan sebaliknya[3].

g. *Application Layer* (Lapisan Aplikasi)

Lapisan ini bertugas memberikan sarana pelayanan langsung ke *user*, yang berupa aplikasi-aplikasi dan mengadakan komunikasi dari program ke program. Jika kita mencari suatu *file* dari *file server* untuk digunakan sebagai aplikasi pengolah kata, maka proses ini bekerja melalui *layer* ini. Demikian pula jika kita mengirimkan e-mail, *browse* ke internet, *chatting*, membuka telnet *session*, atau menjalankan FTP, maka semua proses tersebut dilaksanakan di layer ini.

Application layer terdiri dari bermacam-macam protokol. Misalnya terdapat ratusan jenis terminal yang tidak kompatibel di seluruh dunia, kemudian kita memerlukan aplikasi yang diharapkan bekerja pada jaringan dengan bermacam-macam terminal, yang masing-masing memiliki *layout* layar yang berlainan, mempunyai cara urutan penekanan tombol yang

berbeda untuk penyisipan dan penghapusan teks, memindahkan sensor dan sebagainya. Suatu cara untuk mengatasi masalah seperti di atas adalah dengan menentukan terminal *virtual* jaringan abstrak, sehingga editor dan program-program lainnya dapat ditulis agar saling bersesuaian. Untuk menangani setiap jenis terminal, satu bagian *software* harus ditulis untuk memetakan fungsi terminal *virtual* jaringan ke terminal sebenarnya. Fungsi lainnya adalah pemindahan file. Sistem file yang satu dengan lainnya memiliki konvensi penamaan yang berbeda, cara menyatakan baris-baris teks yang berbeda, dan sebagainya. Perpindahan *file* dari sebuah sistem ke sistem lainnya yang berbeda memerlukan penanganan untuk mengatasi adanya ketidak-sesuaian ini. Tugas tersebut juga merupakan pekerjaan *application layer*, seperti pada surat elektronik, *remote job entry*, *directory lookup*, dan berbagai fasilitas bertujuan umum dan fasilitas bertujuan khusus lainnya[3].

1.2.2 Protokol TCP/IP

TCP/IP adalah sekumpulan protokol yang terdapat di dalam jaringan komputer (*network*) yang digunakan untuk berkomunikasi atau bertukar data antar komputer.



Gambar 2.2 TCP/IP Model

TCP/IP merupakan standar protokol pada jaringan internet yang menghubungkan banyak komputer yang berbeda jenis mesin maupun *system* operasinya agar dapat berinteraksi satu sama lain[4].

1.2.2.1 Layanan TCP/IP

Berikut ini adalah layanan “tradisional” yang dilakukan TCP/IP :

1. Pengiriman file (*file transfer*). *File Transfer Protocol* (FTP) memungkinkan pengguna komputer yang satu untuk dapat mengirim ataupun menerima file ke komputer jaringan. Karena masalah keamanan data, maka FTP seringkali memerlukan nama pengguna (*username*) dan *password*, meskipun banyak juga FTP yang dapat diakses melalui *anonymous*, alias tidak ber-password.
2. *Remote Login*. *Network Terminal Protocol* (*telnet*) memungkinkan pengguna komputer dapat melakukan login ke dalam suatu komputer di dalam suatu jaringan. Jadi hal ini berarti bahwa pengguna menggunakan komputernya sebagai perpanjangan tangan dari komputer jaringan tersebut.
3. *Computer Mail* digunakan untuk menerapkan sistem *e-mail* (*electronic mail*).
4. *Network File System* (NFS). Pelayanan akses file-file jarak jauh yang memungkinkan klien-klien untuk mengakses file-file pada komputer jaringan jarak jauh walaupun file tersebut disimpan secara lokal.
5. *Remote Execution* Memungkinkan pengguna komputer untuk menjalankan suatu program dari komputer yang berbeda. Biasanya berguna jika pengguna menggunakan komputer yang terbatas, sedangkan ia memerlukan sumber yang banyak dalam suatu sistem komputer. Ada beberapa jenis *remote execution*, ada yang berupa perintah-perintah dasar saja, yaitu yang dapat dijalankan dalam *system* komputer yang sama dan ada pula yang menggunakan "*procedure remote call system*", yang memungkinkan program untuk memanggil *subroutine* yang akan dijalankan di *system* komputer yang berbeda. (sebagai contoh dalam Berkeley UNIX ada perintah "rsh" dan "rexec").

1.2.2.2 Arsitektur TCP/IP

Protokol-protokol TCP/IP memberikan dukungan jaringan untuk menghubungkan seluruh tempat dan *host* serta mengikuti aturan standar tentang bagaimana komputer-komputer berkomunikasi dan bagaimana jaringan-jaringan diinterkoneksikan. Protokol-protokol TCP/IP mengikuti model konsep empat *layer* yang dikenal sebagai model *Department of Defense* (DOD): *Application*, *Transport*, *Internet*, dan *Network Interface*[5].

1. *Layer Application*

   Layer Application terletak di atas pada model TCP/IP konsep empat *layer* dan merupakan tempat program *software* memperoleh akses ke jaringan. *Layer* ini sesuai dengan *layer Session*, *layer Presentation*, dan *layer Application* pada model OSI. Beberapa layanan dan utiliti TCP/IP beroperasi pada *Layer Application*. Layanan dan utiliti tersebut meliputi:

   a. *HyperText Transfer Protocol* (HTTP). HTTP adalah *protocol* yang dipakai untuk mayoritas komunikasi *World Wide Web*. *Windows* 2003 menghadirkan *Internet Explorer* sebagai *client* HTTP dan *Internet Information Services* (IIS) sebagai *server* HTTP.

   b. *File Transfer Protocol* (FTP). FTP adalah suatu layanan Internet yang mentransfer file-file dari satu komputer ke komputer lain. *Internet Explorer* dan FTP utiliti yang termasuk *command-line* bertindak sebagai *client* FTP. IIS menyediakan *server* FTP.

   c. *Simple Mail Transfer Protocol* (SMTP). SMTP merupakan suatu protokol yang dipakai *server mail* untuk mentransfer *e-mail*. IIS dapat mengirimkan pesan-pesan dengan memakai protokol SMTP.

   d. *Telnet*. *Telnet* adalah suatu *protocol* yang menyamai terminal yang dapat dipakai untuk login ke *host* jaringan yang jauh. *Telnet* menawarkan para pemakai suatu kapabilitas dalam mengoperasikan program-program secara jauh dan memudahkan

administrasi yang jauh. *Telnet* secara praktis memang disediakan untuk semua sistem operasi dan mengurangi integrasi dalam lingkungan jaringan yang heterogen. *Windows* 2003 memberikan baik *server* maupun *client Telnet*.

e. *Domain Name System* (DNS). DNS merupakan seperangkat *protocol* dan layanan pada suatu jaringan TCP/IP yang mengizinkan para pemakai jaringan untuk mempergunakan nama-nama hirerarki yang sudah dikenal ketika meletakkan *host* ketimbang harus mengingat dan memakai alamat IP-nya. DNS sangat banyak dipakai di *internet* dan pada kebanyakan perusahaan pribadi dewasa ini.

f. *Simple Network Management Protocol* (SNMP). SNMP memungkinkan untuk mengelola *node* jaringan seperti *server, workstation, router, bridge,* dan *hub* dari *host* sentral. SNMP dapat dipakai untuk mengonfigurasi *device* yang jauh, memantau unjuk kerja jaringan, mendeteksi kesalahan jaringan atau akses yang tidak cocok, dan mengaudit pemakaian jaringan.

2. *Network Application* API

TCP/IP Microsoft menyediakan dua *interface* bagi aplikasi-aplikasi jaringan untuk memakai layanan-layanan stack *protocol* TCP/IP :

a. WinSock. Implementasi *Windows* 2003 sangat banyak memakai *Socket* API (*application programming interface*). *Socket* API adalah mekanisme standar untuk mengakses layanan-layanan datagram dan *session* pada TCP/IP.

b. NetBIOS. Suatu API standar yang dipakai sebagai suatu mekanisme *inter-process communication* (IPC) di lingkungan *Windows*. Meskipun NetBIOS dapat dipakai untuk menyediakan koneksi standar ke protokol-protokol yang mendukung layanan-layanan penyampaian pesan dan penamaan NetBIOS, misalnya TCP/IP dan NetBEUI, namun NetBIOS disertakan di

dalam *Windows* 2003 terutama sekali untuk mendukung aplikasi-aplikasi peninggalan.

3. *Layer Transport*

   Protokol-protokol *transport* menghadirkan sesi komunikasi di antara komputer-komputer dan menentukan tipe layanan *transport* entah berorientasi koneksi (TCP) ataupun berorientasi datagram yang tanpa koneksi (UDP). TCP menyajikan komunikasi berorientasi koneksi yang dapat diandalkan untuk aplikasi-aplikasi yang secara khusus mentransfer sejumlah besar data pada suatu waktu. TCP juga dipakai untuk aplikasi-aplikasi yang memerlukan pengakuan bagi data yang diterima. Namun demikian, UDP menyediakan komunikasi tanpa koneksi dan tidak menjamin mampu mengirimkan paket. Aplikasi-aplikasi yang memakai UDP secara khusus mentransfer sejumlah kecil data pada suatu waktu. Pengiriman data yang dapat diandalkan merupakan tanggung jawab aplikasi. *Layer Transport* pada model DOD sesuai dengan *Layer Transport* pada model OSI.

4. *Layer Internet*

   Protokol-protokol *Internet* meng-*encapsulate* paket-paket menjadi *datagram internet* dan mengoperasikan semua *algoritma routing* yang diperlukan. Fungsi-fungsi *routing* yang dikerjakan *layer internet* memang dibutuhkan untuk memungkinkan *host* meng-*interoperate* dengan jaringan-jaringan lain. *Layer internet* sesuai dengan *layer network* pada model OSI. Lima protokol yang diimplementasikan pada *layer* ini:

   a. *Address Resolution Protocol* (ARP), yang menentukan alamat *hardware* pada *host*.
   b. *Reverse Address Resolution Protocol* (RARP), yang menyediakan resolusi alamat kebalikan pada *host* yang menerima.

c. *Internet Control Message Protocol* (ICMP), yang mengirimkan pesan-pesan kesalahan ke IP ketika terjadi masalah.
d. *Internet Group Management Protocol* (IGMP), yang menginformasikan *router* tentang ketersediaan anggota-anggota pada grup *multicast*.
e. *Internet Protocol* (IP), yang mengalamatkan dan me-*route* paket-paket.

5. *Layer Interface Network*

   Model yang paling mendasar adalah *layer interface network*. Masing-masing tipe *local area network* (LAN), tipe *metropolitan area network* (MAN), tipe WAN, dan tipe *dial-up*, misalnya *Ethernet, Token Ring, Fiber Distributed Data Interface* (FDDI), dan ARCnet, mempunyai persyaratan yang berbeda untuk pengkabelan, pembuatan sinyal, dan penyandian data. *Layer interface network* menetapkan persyaratan yang ekuivalen dengan *layer data link* dan *layer physical* pada model OSI. *Layer interface network* bertanggung jawab untuk mengirimkan dan menerima *frame*, yang merupakan paket-paket informasi yang ditransmisikan pada suatu jaringan sebagai suatu unit tunggal. *Layer interface network* meletakkan *frame* pada jaringan dan mengeluarkan *frame* dari jaringan.

1.3 Lapisan Pada Protokol *Network*

Pada sebuah jaringan komputer terdapat beberapa lapisan protokol yang digunakan sebagai format standar untuk berkomunikasi antar komputer.

1.3.1 IP *Layer*

Pada IP *layer*, paket IP terdiri dari dua bagian yaitu IP *header* dan IP *body*, untuk lebih jelasnya lihat pada Gambar 2.3.



Gambar 2.3 IP Header

Dilihat dari paket *filter*, IP *header* berisi empat informasi, yaitu :

1. IP *Source address* : memiliki panjang 4 byte seperti 192.168.168.3
2. IP *Destination* : memiliki panjang 4 byte seperti 192.168.168.2
3. IP *Protocol Type* : merupakan identitas dari IP *body* apakah paket tersebut berupa paket TCP atau paket UDP, paket ICMP, atau paket lainnya.
4. IP *options field*

1.3.2 TCP *Layer*

Pada TCP *layer* paket dibagi menjadi dua bagian yaitu TCP *header* dan TCP *body*. TCP *header* terdiri dari tiga infomasi yaitu :

1. TCP *source port* : memiliki panjang dua byte yang khusus yang akan diproses pada sistem *client* atau *server* yang dikirim.
2. TCP *destination port* : memiliki panjang dua byte yang akan diproses pada sistem *client* atau *server* yang diterima.
3. TCP *flag field* : berisi *flag* yang digunakan untuk menandai bentuk atau tipe dari paket.

1.3.3 IP (*Internet Protocol*)

Internet Protocol memiliki beberapa lapisan bawah yang berbeda seperti *ethernet*, *token ring*, PPP, dan lain-lain. IP *layer* juga memiliki beberapa lapisan atas yang berbeda seperti TCP, UDP, dan ICMP.

1.3.3.1 IP *Protocol* Lapisan Atas

Internet protocol (IP) merupakan dasar dari semua protokol yang berbeda. Bentuk yang paling umum adalah TCP, UDP dan ICMP.

1.3.3.1.1 *Transmission Control Protocol* (TCP)

Protokol TCP pada umumnya digunakan pada layanan *internet* seperti *telnet*, FTP, SMTP, NNTP, dan HTTP. Protokol TCP adalah protokol yang menghubungkan antara dua *endpoint*. TCP adalah protokol yang dapat memastikan bahwa tujuan dalam keadaan aktif dan data yang diterima tidak rusak.

Protokol TCP akan melakukan pemutusan untuk suatu koneksi yang melanggar dari aturan. Sebagai contoh, jika dalam perjalanan paket TCP hilang atau rusak pada waktu pengiriman maka paket tersebut tidak akan diterima. Jika pengiriman diulangi kembali dan tetap ada data yang hilang atau rusak maka koneksi akan diputus secara otomatis. TCP terbagi menjadi dua bagian yaitu TCP *header* dan TCP *option*.

1. *Header* TCP

   Header TCP mampu menampilkan seluruh mekanisme protokol yang cukup besar dengan panjang maksimum 20 byte. Elemen-elemen dari TCP header dapat dilihat pada Gambar 2.4.



Gambar 2.4 *Header* TCP

Elemen-elemen pada *header* TCP akan dijelaskan pada Tabel 2.1.

Tabel 2.1 Elemen TCP *Header*

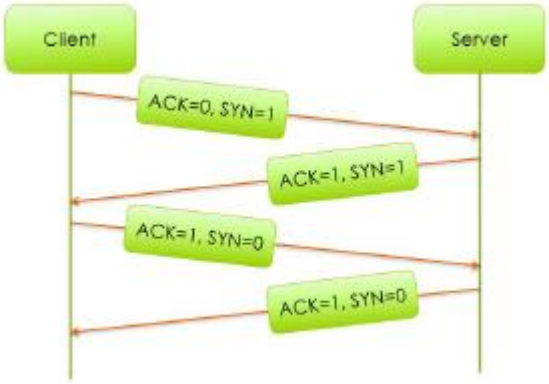
Elemen	Penjelasan
Source Port (16 bit)	*Port* number dari sumber
Destination Port (16 bit)	*Port* number dari tujuan
Sequence Number (32 bit)	Nomor urut data pertama di dalam segmen kecuali bila *flag* SYN diset
Acknowledgment Number (32 bit)	Nomor urut data berikutnya yang merupakan tempat entitas TCP
Data offset (4bit)	Berjumlah 32 bit kata di dalam *header*
Reserved	Bertujuan untuk pengguna-pengguna berikutnya
Flag (6 bit)	Tanda untuk sebuah paket URG,ACK, PSH, RST, SYN, FIN
Window Size	Pengalokasian kredit kontrol alur dalam octet. Memuat sejumlah octet data yang dimulai dengan satu balasan bahwa pengirim ingin menerima data
Checksum (16 bit)	Elemen dari *one complement* dari *sum modulo* $2^{16} - 1$ dari seluruh *word* 16 bit di dalam segmen ditambah dengan *pseudo header*
Urgen Pointer	Menunjuk pada byte terakhir dalam urutan data *urgent*. Hal ini memungkinkan penerima mengetahui berapa banyak data *urgent* yang datang
Data	Berisi data yang dikirimkan

2. TCP *Option*

   TCP *option* berisi beberapa *flag* atau tanda. Beberapa *flag* tersbut diantaranya yaitu :

   1. URG (*Urgent*)
   2. ACK (*Acknowledgment*)
   3. PSH (*Push*)
   4. RST (*Reset*)
   5. SYN (*Synchronize*)
   6. FIN (*Finish*)

URG dan PSH digunakan untuk melakukan identifikasi pada data. PSH menerima *buffer* dan beberapa program dari data, sedangkan URG secara umum menandai data dari pengirim yang disebut dengan *out of band data*. ACK dan SYN digunakan untuk membuat *handshake* pada protokol TCP dengan tiga cara. Pada Gambar 2.5 diperlihatkan bagaimana prokol TCP melakukan proses *handshaking*.



Gambar 2.5 *Handshaking* Pada Protokol TCP

RST dan FIN merupakan cara untuk menutup suatu koneksi. RST berfungsi sebagai tanda menutup koneksi secara tidak normal karena terjadi kesalahan atau *error*, sedangkan FIN berfungsi sebagai tanda untuk menutup koneksi secara normal.

1.3.3.1.2 *User Datagram Protocol* (UDP)

UDP menyediakan layanan koneksi untuk prosedur-prosedur pada level aplikasi. Pada dasarnya UDP merupakan suatu layanan protokol yang kurang bisa diandalkan karena kurang memberikan perlindungan dalam pengiriman dan duplikasi data. *Datagram* merupakan suatu paket *switching*, sebuah paket terpisah-pisah dari paket lain yang membawa informasi yang memadai untuk *routing* dari *Data Terminal Equipment* (DTE) sumber ke DTE tujuan tanpa harus menetapkan koneksi antara DTE dan jaringan[6].



Gambar 2.6 User Datagram Protocol

Pada dasarnya struktur UDP hampir sama dengan TCP, UDP *header* berisi *source* dan *destination port number*, seperti halnya *source* TCP dan *destination port number*, hanya saja pada UDP *header* tidak terdapat *flag*, *sequence number*, atau bit ACK. Berikut ini adalah gambar dari struktur UDP yang diperlihatkan pada Gambar 2.6.

1.3.3.1.3 *Internet Control Message Protocol* (ICMP)

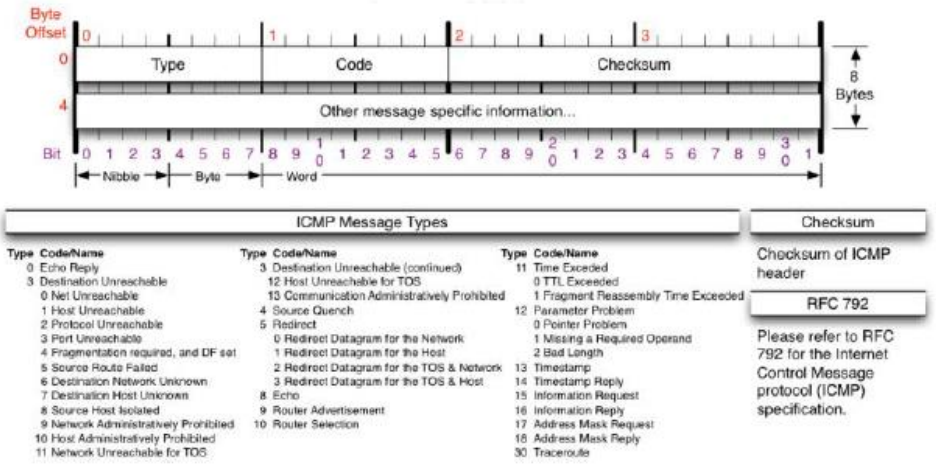
Internet Control Message Protocol (ICMP) adalah protokol pada TCP/IP yang bertugas mengirimkan pesan-pesan kesalahan dan kondisi lain yang memerlukan perhatian khusus. Hal ini dapat dilakukan dengan mengevaluasi pesan yang dihasilkan oleh ICMP. Jenis pesan pada ICMP ada dua yaitu ICMP *error message* dan ICMP *query message*. Dengan memanfaatkan kedua pesan ini seorang *hacker* dimungkinkan untuk mengumpulkan informasi sebanyak-banyaknya sebelum melakukan penyerangan seperti ICMP *sweep*, *traceroute* dan OS *fingerprinting* dan menyiapkan strategi yang diperlukan untuk melakukan penyerangan seperti ICMP *route redirect*, ICMP *informational message*, ICMP *router discovery message* dan ICMP *flood*[6]. Contoh-contoh dari paket ICMP dijelaskan pada Tabel 2.2.

Tabel 2.2 Paket ICMP

Paket ICMP	Penjelasan
Echo request	Apa yang dikirim *host* ketika *user* menjalankan perintah *ping*
Echo response	Apa yang direspon *host* terhadap *echo request*
Time exceeded	Apa yang dikembalikan *router* ketika menentukan bahwa suatu paket terjadi *looping*
Destination unreachable	Apa yang *router* kembalikan ketika

	tujuan dari suatu paket tidak bisa dicapai oleh beberapa alasan, misalkan suatu sistem sedang *down*
Redirect	Respon apa yang *router* kirim ke *host* suatu paket telah dikirim ke *router* yang lain

ICMP berada pada *level* yang sama pada *internet protocol* yang berada pada arsitektur TCP/IP. Pesan yang telah disusun dan kemudian ditransfer dalam lintasan *internet protocol* dan membungkus pesan tersebut dengan sebuah *header* IP dan kemudian mentransmisikan datagram yang dihasilkan dengan cara ditransfer biasa, dikarenakan pesan dari ICMP ditransmisikan di dalam *datagram* IP sehingga pengirim tidak dijamin keamanannya. Oleh karena itu penggunaan ICMP tidak bisa diandalkan. Pada Gambar 2.7 diperlihatkan format ICMP *message*.



Sumber:http://www.siongboon.com/projects/2006-03-06_serial_communication/

Gambar 2.7 ICMP *Message*

1.4 Perangkat Jaringan Komputer

Dalam jaringan komputer terdapat beberapa perangkat keras yang dibutuhkan agar dapat menghubungkan komputer satu dengan komputer lainnya. Perangkat-perangkat keras itu diantaranya : kabel UTP, *router*, *switch*, *switch manageable*, *network interface card* (NIC).

1.4.1 Kabel *Unshielded Twisted Pair* (UTP)

Kabel UTP[7] digunakan sebagai penghubung antar perangkat keras satu ke perangkat keras yang lain agar perangkat tersebut dapat berhubungan. Kabel ini memiliki delapan jalur yang dapat digunakan. Kabel UTP memiliki beberapa jenis kategori, seperti yang dijelaskan pada Tabel 2.3.

Tabel 2.3 Kategori Kabel UTP

Kategori	Keterangan
Cat 1	Digunakan diperangkat telephone pada jalur ISDN (*Integrated Service Digital Network*), juga untuk menghubungkan modem dengan line telephone
Cat 2	Transmisi suara hingga 4 Mbps, digunakan pada topologi Token Ring
Cat 3	Transmisi data hingga 10 Mbps, digunakan pada topologi Token Ring atau 10BaseT
Cat 4	Transmisi data hingga 16 Mbps, digunakan pada topologi Token Ring
Cat 5e	Transmisi data hingga 100 Mbps, digunakan pada topologi Star dan Tree. Teknologi yang disupport ethernet 16 Mbps,Fast Ethernet 100 Mbps, Token Ring 16 Mbps
Cat 6	Transmisi data hingga 2,5 Gbps, Jarak hingga 200 meter
Cat 7	Transmisi data hingga 10 Gbps

Ada dua jenis pemasangan kabel UTP yang umum digunakan yaitu *straight* dan *cross-over*. *Straight* biasanya digunakan untuk menghubungkan perangkat-perangkat keras yang berbeda, contoh komputer dan *switch*. Kabel *straight* ini dipasang sejajar antar pin pada konektor RJ45. *Cross-over* biasanaya digunakan untuk menghubungkan perangkat-perangkat keras yang sama, contohnya komputer dengan komputer. Kabel *Cross* ini dipasang dengan menghubungkan pin 1 ke pin 3, pin 2 ke pin 6, pin 3 ke pin 1, pin 6 ke pin 2 dan pin-pin yang lain tetap dihubungkan sesuai dengan nomor pin pada RJ45.

1.4.2 Router

Router adalah perangkat internetworking yang meneruskan paket antara jaringan dengan memproses informasi *routing* yang termasuk dalam paket atau datagram (informasi protokol Internet dari lapisan 3). Informasi *routing* sering diolah dalam hubungannya dengan tabel *routing* (atau *forwarding table*). Router menggunakan tabel *routing* untuk menentukan di mana untuk meneruskan paket. (Sebuah tujuan dalam tabel *routing* dapat mencakup "nol" antarmuka, juga dikenal sebagai "lubang hitam" antarmuka karena data bisa masuk ke dalamnya, namun, tidak ada proses lebih lanjut dilakukan untuk mengatakan data.) [8]

1.4.3 Switch

Switch adalah perangkat jaringan komputer yang digunakan untuk menghubungkan beberapa *network segment*[8]. Istilah *switch* juga merujuk pada *multi-port network bridge* (Jembatan jaringan *multi-port*) yang memproses dan mengirimkan data pada layer kedua pada OSI. *Switch* yang juga memproses data pada layer ketiga OSI disebut juga *multilayer switch*. Fungsi dari *switch* adalah membuat jalur distribusi data tersendiri tanpa mengganggu jalannya distribusi data lain yang sedang berjalan. Sehingga data bisa berjalan *full duplex* (kirim terima secara bersamaan) dan mempunyai bandwidth tersendiri. Hal ini berbeda dengan *switch* yang menjalankan data dalam *half duplex* (kirim-terima secara bergantian) dan membagi semua *bandwidth* ke semua jalur, yang bisa menyebabkan terjadinya tabrakan data. Misal ada dua komputer A dan B, dan dua lagi C dan D. Keduanya dihubungkan dengan *switch* yang sama. Maka komputer A dan B dapat saling berkomunikasi tanpa mengganggu C dan D, begitu pula sebaliknya. Dari jenis konfigurasi datanya *switch* dapat dibedakan menjadi :

1. *Unmanaged switch*

   Switch jenis ini tidak dapat dikonfigurasi dan bersifat *plug and play*. *Switch* jenis ini bisa ditemukan di rumah atau di perusahaan atau jaringan kecil. [8]



Gambar 2.8 Unmanaged switch

2. *Managed switch*

   Mempunyai fitur untuk memodifikasi operasi dari *switch*. Biasanya cara untuk mengaturnya lewat CLI (*Command Line Interface*) yang diakses lewat *console*, telnet atau *secure shell*. Biasanya juga ditanamkan SNMP (*Simple Network Management Protocol*) yang diatur melalui *web browser*. Jenis ini terbagi menjadi dua jenis :

   1. *Smart Switch* (*Intelligent Switch*)

      Adalah *managed switch* dengan fitur manajemen yang terbatas. Pengaturannya melalui *web browser*, pengaturan tersebut antara lain terdiri dari VLAN, *port-bandwith* dan *duplex*.[8]

   2. *Enterprise Managed Switch* (*Full Managed Switch*)

      Adalah managed *switch* dengan fitur manajemen yang lengkap. Pengaturan dilakukan melalui *web browser* atau CLI, dengan pengaturan tambahan sepertu *backup* konfigurasi *switch*.[8]



Gambar 2.9 Managed switch

Dibanding *Smart Switch, switch* ini memiliki pengaturan lebih lengkap. Contoh perangkat keras ini yaitu Cisco Catalyst 2960. Pada jenis *switch* ini mendukung untuk aplikasi SPAN (*Switched Port Analyzer*), SPAN ini digunakan untuk mengirimkan salinan paket data pada *switch port* (atau seluruh VLAN) ke *switch port*

lain yang terhubung pada suatu *network monitoring* seperti IDS. *Administrator* menggunakan fitur ini untuk menganalisa data pada jaringan. Proses ini membantu *administrator* unutk *memonitoring* performa *network* dan akan membantu memperingatkan ketika terjadi suatu masalah.

1.4.4 Network Interface Card (NIC)

Network Interface Card adalah perangkat keras pada jaringan komputer yang digunakan sebagai *interface* komputer untuk berhubungan dengan jaringan computer[8]. Pada NIC terdapat *port ethernet* yang digunakan untuk memasukan konektor RJ45.



Gambar 2.10 Network Interface Card

NIC menentukan kecepatan data komunikasi yang dapat diproses oleh komputer.

1.5 Keamanan Jaringan Komputer

Keamanan jaringan secara umum adalah komputer yang terhubung ke jaringan, mempunyai ancaman keamanan yang lebih besar dibandingkan komputer yang tidak terhubung ke jaringan. Namun keamanan komputer biasanya bertentangan dengan *network access*, dimana bila *network access* semakin mudah, maka *network security* semakin rawan, dan bila *network* semakin baik maka *network access* semakin tidak nyaman. Suatu *network* dirancang sebagai

komunikasi data dengan tujuan meningkatkan akses ke sistem komputer, sementara *security* dirancang untuk mengendalikan akses. Sebagai analogi, keamanan di rumah dilakukan dengan memberi kunci di pintu rumah, tidak dengan cara memblokir jalan di depan rumah. Hal seperti ini juga diterapkan pada *network security*. Keamanan dijaga untuk setiap *host-host* tertentu, tidak langsung pada *network*-nya.

Langkah awal dalam mengembangkan rencana *network security* yang efektif adalah dengan mengenali ancaman yang mungkin datang. Dalam RFC 1244, *Site security Handbook*, dibedakan tiga tipe ancaman, yaitu :

1. Akses tidak sah oleh orang yang tidak memiliki wewenang.
2. Kesalahan informasi, segala masalah yang dapat menyebabkan diberikannya informasi yang penting atau sensitif kepada orang yang salah, yang seharusnya tidak boleh mendapatkan informasi tersebut.
3. Penolakan terhadap *service*, segala masalah mengenai *security* yang menyebabkan sistem mengganggu pekerjaan-pekerjaan yang produktif.

1.5.1 Tujuan Keamanan Jaringan Komputer

Pada dasarnya tujuan dari keamanan komputer yaitu :

1. *Confidentiality*, merupakan usaha untuk menjaga informasi dari orang yang tidak berhak mengakses. *Confidentiality* biasanya berhubungan dengan informasi yang diberikan ke pihak lain.[9]
2. *Integrity*, Keaslian pesan yang dikirim melalui sebuah jaringan dan dapat dipastikan bahwa informasi yang dikirim tidak dimodifikasi oleh orang yang tidak berhak dalam perjalanan informasi tersebut.[9]
3. *Availability*, aspek *availability* atau ketersediaan berhubungan dengan ketersediaan informasi ketika dibutuhkan.[9]

Tabel 2.4 CIA

	Definitions	***Tools***
Confidentiality	a. *Concealment of info and resource* b. *Hide existence of info and resource*	a. *Encryption* b. *Access control*
Integrity	a. *Trustworthiness off info and resource* (*Authentication*) b. *Correctness of data* (*data integrity*)	a. *Preventions* b. *Detection*
Availability	*Ability to use info and resources*	a. *System design* b. *Statstical models of use*

1.5.2 Tipe Ancaman Keamanan Sistem Komputer

Fungsi sistem komputer dapat digunakan sebagai dasar untuk menentukan model tipe ancaman dari suatu sistem komputer. Berdasarkan fungsi sistem komputer sebagai penyedia informasi, ancaman terhadap sistem komputer dikategorikan menjadi empat, yaitu :

1. *Interruption*, merupakan suatu ancaman terhadap *availability*, informasi atau data yang ada dalam sistem komputer dirusak, dihapus, sehingga jika dibutuhkan maka sudah tidak ada lagi[10].
2. *Interception*, merupakan ancaman terhadap kerahasiaan (*secrecy*). Informasi yang ada di dalam sistem disadap oleh orang yang tidak berhak[10].
3. *Modification*, merupakan ancaman terhadap integritas. Orang yang tidak berhak berhasil menyadap lalu-lintas informasi yang sedang dikirim lalu mengubahnya sesuai keinginan orang itu[10].
4. *Fabrication*, merupakan ancaman terhadap integritas. Orang yang tidak berhak berhasil meniru atau memalsukan suatu informasi sehingga orang yang menerima informasi tersebut menyangka informasi tersebut berasal dari orang yang dikehendaki oleh si penerima informasi tersebut[10].

1.5.3 Bentuk Ancaman Jaringan Komputer

Berikut beberapa ancaman yang umum ditemui pada jaringan nirkabel :

1. MAC *Spoofing*

   Penyerang berusaha mendapatkan koneksi ke dalam jaringan dengan mengambil alamat NIC dari suatu perangkat komputer pada jaringan tersebut[11].

2. ARP *Spoofing*

   Penyerang menangkap penyebaran paket ARP dari *access point* dan kemudian mengirimkan balasan ARP fiktif sehingga informasi perangkat dari penyerang akan terpetakan ke dalam tabel ARP untuk kemudian mendapatkan hak akses kedalam jaringan[11].

3. *Man in the Middle Attack*

   Metode serangan ini biasanya didahului dengan ARP *spoofing* kemudian penyerang menempatkan perangkat yang dimilikinya sebagai sebuah komputer fiktif yang akan terlihat resmi dari sisi *access point*[11].

4. *Denial of Service*

   Metode serangan dengan mengirimkan paket data dalam jumlah yang sangat besar terhadap jaringan yang menjadi targetnya secara terus - menerus. Hal ini dapat mengganggu lalu - lintas data bahkan kerusakan sistem jaringan[11].

1.6 Jenis Serangan Jaringan Komputer

Jenis dan teknik serangan yang mengganggu jaringan komputer beraneka jenis, diantaranya adalah :

1.6.1 *Port Scanning*

Merupakan suatu proses untuk mencari *port* yang terbuka pada suatu jaringan komputer. Dari hasil *scanning* akan didapat letak kelemahan sistem tersebut. Pada dasarnya sistem *port scanning* mudah untuk dideteksi, tetapi

penyerang akan menggunakan beberapa cara metode untuk menyembunyikan serangan[12].

1.6.2 *Teardrop*

Teknik ini menggunakan tekni penyerangan dengan mengeksploitasi proses *disassembly-reassembly* paket data. Dalam jaringan *internet* sering kali data harus dipotong menjadi paket yang lebih kecil untuk menjamin reliabilitas dan proses *multiple* akses jaringan. Pada proses pemotongan data paket yang normal, setiap potongan diberi informasi *offset* data yang berbunyi "Potongan byte ini merupakan potongan 600 byte dari total 800 byte paket yang dikirimkan". Program memanipulasi potongan data sehingga terjadi *overlapping* antara paket yang diterima di bagian penerima setelah potongan-potongan paket disusun kembali[12].

1.6.3 IP *spoofing*

Teknik ini bekerja dengan mengganti alamat IP pengguna yang lain yang bukan penyerang sebenarnya. Hal ini terjadi karena salah rancang (*design flaw*) bagian urutan nomor (*sequence number*) dari paket TCP/IP. Dalam beberapa kasus, penyerang menggunakan satu alamat IP sumber yang spesifik pada semua paket IP yang keluar untuk membuat semua pengembalian paket IP dan pesan ICMP ke pemilik alamat tersebut[12].

1.6.4 ICMP *flood*

Penyerang melakukan eksploitasi dengan tujuan untuk membuat target *host* menjadi terganggu, yang disebabkan oleh pengiriman sejumlah paket yang besar ke arah target *host*. Eksploitasi sistem ini dilakukan dengan mengirimkan suatu perintah "ping" dengan tujuan *broadcast* atau *multicast* dimana pengirim dibuat seolah-olah adalah target *host*. Semua balasan dikembalikan ke target *host*. Hal inilah yang menyebabkan *host* target menjadi terganggu dan menurunkan kinerja jaringan bahkan dapat menyebabkan *denial of service*[12].

1.6.5 UDP *flood*

UDP *flood* mengaitkan dua sistem tanpa disadari. Dengan cara *spoofing*, *User Datagram Protocol* (UDP) *flood attack* akan menempel pada servis UDP di salah satu mesin yang untuk keperluan "percobaan" akan mengirimkan sekelompok karakter ke mesin lain, yang diprogram untuk meng- *echo* setiap kiriman karakter yang diterima melalui servis. Karena paket UDP tersebut di-*spoofing* diantara ke dua mesin tersebut maka yang terjadi adalah "banjir" tanpa henti paket kiriman karakter yang tidak berguna diantara diantara kedua mesin tersebut. Untuk menanggulangi UDP *flood*, *disable* semua servis UDP di semua mesin di jaringan, atau dengan menyaring semua servis UDP yang masuk pada *firewall*[12].

1.6.6 *Packet interception*

Gangguan jenis ini dilakukan dengan membaca paket di saat paket tersebut sedang mengalami *packet sniffing*. *Packet interception* merupakan cara penyerang untuk mendapatkan informasi yang ada di dalam paket tersebut. Hal ini dapat dicegah dengan mengenkripsi terlebih dahulu, sehingga penyerang akan mengalami kesulitan untuk membuka paket tersebut[12].

1.6.7 Smurf *attack*

Gangguan jenis ini biasanya dilakukan dengan menggunakan IP *spoofing*, yaitu mengubah nomor IP dari datangnya *request*. Dengan menggunakan IP *spoofing*, respons dari ping tadi dialamatkan ke komputer yang IP-nya di-*spoof*. Akibatnya, komputer tersebut akan menerima banyak paket. Hal ini dapat mengakibatkan pemborosan penggunaan *bandwidth* jaringan yang menghubungkan komputer tersebut[12].

1.6.8 DOS (*Denial Of Service*)

DOS merupakan salah satu serangan yang banyak ditemui dalam dunia networking saat ini. Kita tidak pernah akan tahu kapan kita akan mendapat serangan ini. Serangan DOS dapat terjadi kapan saja pada jaringan, yang dapat

kita lakukan adalah menyusun dan menerapkan metode penangan untuknya. Serangan ini dapat ditujukan kepada siapa saja, bahkan ke personal. Namun yang paling sering terkena dampaknya adalah server- server besar seperti yahoo, google, serta perbankan yang secara langsung memberikan pelayanannya melalui jaringan. Ada beberapa contoh misal pada anggota parpol, mungkin bila anda seorang yang berkecimpung di dunia politik atau pengurus parpol, anda akan berpikir serangan ini ditujukan untuk menjatuhkan citra politik anda dan parpol. Bila anda seorang pengusaha serangan ini bisa saja ditujukan oleh saingan bisnis anda. Bila kita memutuskan untuk memberikan pelayanan melalui jalur networking maka kita juga harus siap menanggung resikonya, salah satunya adalah serangan DOS (*Denial Of Service*).

Faktor penyebab yang paling banyak ditemukan di dunia nyata sebenarnya karena keisengan belaka atau dengan kata lain ”*its just for fun*”. Pembuktian jati diri di kalangan *underground* juga bukanlah alasan yang salah, sebagian komunitas berpendapat bila seseorang telah dapat melumpuhkan sebuah sistem dengan cara yang bermacam – macam maka ia baru bisa diterima dalam komunitas tersebut. ”*welcome to Underground World*”. Seorang Elite yang beretika tentunya tidak diharapkan untuk melakukan serangan *Denial Of Service* ini. Serangan ini biasanya bertujuan untuk mematikan pelayanan dari komputer atau jaringan yang diserang. Dampaknya akan sangat besar bagi perusahaan atau instansi yang menyediakan jasa terutama bagi perbankan. Korban yang terkena serangan ini tidak dapat memberikan pelayanan yang seharusnya. Serangan Dos ini dapat menghambat bahkan mematikan pelayanan pada sebuah sistem, sehinggan pengguna yang absah tidak dapat menerima atau mendapatkan pelayanan yang seharusnya. Bayangkan saja bila sebuah perusahaan bank tidak dapat memberikan pelayanan kepada nasabahnya, maka akan sangat fatal akibatnya bagi kelangsungan perusahaan tersebut. Atau sebuah *provider internet* yang tidak dapat memberikan *bandwith*-nya kepada klien, maka akan berdampak tidak hanya kepada *provider* tersebut yang tentunya harus membayar jaminan koneksi sesuai yang mereka terapkan namun juga kepada penyewa koneksi. Bila

salah satu penyewanya adalah sebuah warnet atau bahkan banking, tentunya akan berdampak luas.

Serangan *Denial Of Service* ini pada dasarnya sulit untuk dideteksi, kecuali bila penyerang telah melakukan beberapa kali percobaan dengan alamat ip yang sama. Tentu akan sangat mudah untuk membloknya. DOS cukup sulit untuk diatasi karena serangan ini pada dasarnya juga berkaitan dengan pelayanan yang diberikan, sebuah sistem dengan tingkat keamanan yang tinggi biasanya memberikan kenyamanan yang rendah bagi penggunanya. Bayangkan bila *server* yahoo dijadikan perantara untuk menyerang, tentu akan sangat membingungkan bagi *administrator* disebuah penyedia jasa *internet*. Sang *administrator* tidak bisa semerta-merta memblok alamat ip dari yahoo karena yang akan terkena dampaknya adalah pengguna jasa *internet* tersebut. Maka perlu dicari solusi yang tepat oleh *provider internet* tersebut.

Saat seseorang ingin melakukan serangan, ada banyak sekali cara yang dapat dilakukan. Klasifikasi dari serangan DOS ini, diantaranya *syn attack* dan UDP *flood*. Serangan DOS biasa saat ini sudah tidak terlalu besar lagi dampaknya, karena para ahli jaringan telah banyak menemukan metode pengamanan terbaru dan menambal patch-patch pada sebuah sistem yang dapat memungkinkan seseorang dengan mudahnya melakukan DOS.

Serangan DOS bukan merupakan hal yang sulit untuk dilakukan, namun bila penyerang ingin memberikan dampak yang besar, maka dibutuhkan pengetahuan dan sumber daya yang memadai untuk melakukan hal tersebut. Pengamatan tentang target juga merupakan hal-hal yang tidak boleh dilupakan. Mengatasi DOS secara kasar dapat dilakukan dengan cara mematikan layanan yang diserang atau mungkin mematikan server kita, namun akan berakibat kepada user absah yang ingin mengakses *server* kita (”*destination unreachable*”).

1.6.9 SQL *Injection*

Pada dasarnya SQL *Injection* merupakan cara mengeksploitasi celah keamanan yang muncul pada *level* atau "layer" *database* dan aplikasinya. Celah keamanan tersebut ditunjukkan pada saat penyerang memasukkan nilai "*string*" dan karakter-karakter yang ada dalam instruksi SQL, perintah tersebut hanya diketahui oleh sejumlah kecil individu yang berusaha untuk mengeksploitasinya. Karena tipe data yang dimasukkan tidak sama dengan yang seharusnya, maka terjadi sebuah aktivitas "liar" yang tidak terduga yang dapat mengakibatkan individu yang tidak berhak dapat masuk ke dalam sistem yang telah terproteksi. Dikatakan sebagai sebuah "injeksi" karena aktivitas penyerangan dilakukan dengan cara memasukkan string khusus untuk melewati filter logika hak akses pada *website* atau sistem komputer yang dimaksud[10].

1.6.10 *Cross Site Scripting*

Cross Site Scripting (XSS) adalah suatu serangan dengan mekanisme *injection* pada aplikasi *web* dengan memanfaatkan metode HTTP GET atau HTTP POST. *Cross Site Scripting* biasa digunakan oleh pihak-pihak yang berniat tidak baik dalam upaya mengacaukan konten website dengan memasukkan naskah program sebagai bagian dari teks masukan melalui formulir yang tersedia. XSS memanfaatkan lubang kelemahan keamanan yang terjadi pada penggunaan teknologi *dynamic page*. Serangan jenis ini dapat diakibatkan oleh kelemahan yang terjadi akibat ketidakmampuan *server* dalam memvalidasi input yang diberikan oleh pengguna. Hal ini memungkinkan halaman yang dihasilkan menyertakan perintah yang sebenarnya tidak diperbolehkan[10].

1.7 IDS (*Intrusion Detection System*)

Intrusion Detection System (IDS) dapat didefinisikan sebagai *tool*, metode, sumber daya yang memberikan bantuan untuk melakukan identifikasi dan memberikan laporan terhadap aktifitas jaringan komputer. Ada beberapa alasan menggunakan IDS, diantaranya adalah :

1. Mencegah ancaman keamanan yang terus meningkat, karena banyak ditemukan kegiatan illegal yang diperbuat oleh orang-orang yang tidak bertanggung jawab.
2. Mendeteksi serangan dan pelanggaran keamanan sistem jaringan yang tidak bias dicegah oleh sistem yang umum dipakai, seperti *firewall*.
3. Mendeteksi serangan awal (biasanya *network probe* dan aktivitas *doorknob rattling*), Langkah awal dari suatu serangan pada umumnya adalah melakukan penyelidikan atau menguji sistem jaringan yang akan menjadi target, untuk mendapatkan titik-titik dimana mereka akan bisa masuk.
4. Mengamankan file yang keluar dari jaringan.
5. Sebagai pengendali *security design* dan administrator terutama bagi perusahaan yang besar.
6. Menyediakan informasi terhadap gangguan secara langsung, meningkatkan diagnosis, *recovery* dan mengkoreksi faktor-faktor penyebab serangan.

1.7.1 Jenis IDS

Pada dasarnya IDS terbagi menjadi dua jenis, yaitu :

1. *Host Based*

   IDS *host based* bekerja pada *host* yang akan dilindungi. IDS jenis ini dapat melakukan berbagai macam tugas untuk mendeteksi serangan yang dilakukan pada *host* tersebut. Keunggulan IDS *Host based* adalah pada tugas-tugas yang berhubungan dengan keamanan *file*. Misalnya ada atau tidaknya *file* yang telah diubah atau adakah usaha untuk mendapatkan akses ke *file-file* yang *sensitive*.

2. *Network Based*

   IDS *Network based* biasanya berupa mesin yang khusus dipergunakan untuk melakukan *monitoring* seluruh segmen dari jaringan. IDS ini akan mengumpulkan paket-paket data yang terdapat pada jaringan dan

kemudian menganalisanya serta menentukan apakah paket itu berupa suatu paket yang normal atau suatu serangan.

1.7.2 Metode Pada IDS

Ada dua metode yang digunakan pada *intrusion detetion system*, diantaranya :

1. *Signature Based*

   IDS yang berbasis pada *signature* akan melakukan pengawasan terhadap paket-paket dalam jaringan dan melakukan pembandingan terhadap paket-paket tersebut dengan basis data *signature* yang dimiliki oleh sistem IDS ini atau atribut yang dimiliki oleh percobaan serangan yang pernah diketahui. Cara ini hampir sama dengan cara kerja aplikasi antivirus dalam melakukan deteksi terhadap *malware*. Intinya adalah akan terjadi keterlambatan antara terdeteksinya sebuah serangan di internet dengan *signature* yang digunakan untuk melakukan deteksi yang diimplementasikan didalam basis data IDS yang digunakan. Jadi bisa saja basis data *signature* yang digunakan dalam sistem IDS ini tidak mampu mendeteksi adanya sebuah percobaan serangan terhadap jaringan karena informasi jenis serangan ini tidak terdapat dalam basis data *signature* sistem IDS ini. Selama waktu keterlambatan tersebut sistem IDS tidak dapat mendeteksi adanya jenis serangan baru[13].

2. *Anomaly Based*

   IDS jenis ini akan mengawasi *traffic* dalam jaringan dan melakukan perbandingan *traffic* yang terjadi dengan rata-rata *traffic* yang ada (stabil). Sistem akan melakukan identifikasi apa yang dimaksud dengan jaringan "normal" dalam jaringan tersebut, berapa banyak *bandwidth* yang biasanya digunakan di jaringan tersebut, protokol apa yang digunakan, *port-port* dan alat-alat apa saja yang biasanya saling berhubungan satu sama lain didalam jaringan tersebut, dan memberi peringatan kepada administrator ketika dideteksi ada yang tidak normal, atau secara signifikan berbeda dari kebiasaan yang ada[13].

1.8 IPS (*Intrusion Prevention System*)

Intrusion Prevention Systems (IPS), juga dikenal sebagai *Intrusion Detection and Prevention Systems* (IDPS), adalah peralatan keamanan jaringan yang memonitor jaringan dan / atau sistem kegiatan untuk aktivitas berbahaya. Fungsi utama IPS adalah untuk mengidentifikasi aktivitas berbahaya, log informasi tentang kegiatan ini, mencoba untuk memblokir / menghentikannya, dan melaporkannya[14].

IPS dianggap sebagai tambahan dari IDS karena IDS dan IPS memantau lalu lintas jaringan dan / atau sistem kegiatan untuk aktivitas berbahaya. Perbedaan utama adalah, tidak seperti IDS, IPS ditempatkan *in-line* dan dapat aktif mencegah / blok intrusi yang terdeteksi[15][16]. Lebih khusus, IPS dapat mengambil tindakan seperti mengirim alarm, melakukan pemblokiran atas paket berbahaya, melakukan konfigurasi ulang atas koneksi dan / atau memblokir lalu lintas dari alamat IP[17]. Sebuah IPS dapat juga melakukan pengecekan *Cyclic Redundancy Check* (CRC) atas kesalahan-kesalahan, *unfragment packet streams*, mencegah masalah TCP *sequencing*, dan membersihkan *transport* dan *network layer* yang tidak diinginkan pada pilihan lapisan jaringan[15][18].

1.8.1 Klasifikasi IPS

Intrusion Prevention Systems (IPS) dapat diklasifikasi menjadi empat tipe yang berbeda, antara lain[14][19] :

1. *Network-based intrusion prevention system* (NIPS)

   Memonitor seluruh jaringan untuk lalu lintas yang mencurigakan dengan menganalisis aktivitas protocol.

2. *Wireless intrusion prevention systems* (WIPS)

   Memonitor jaringan nirkabel untuk lalu lintas yang mencurigakan dengan menganalisis protokol jaringan nirkabel.

3. *Network behavior analysis* (NBA)

   Memeriksa lalu lintas jaringan untuk mengidentifikasi ancaman-ancaman yang menghasilkan arus lalu lintas yang tidak biasa, seperti

serangan *distributed denial of service* (DDoS), bentuk-bentuk tertentu dari malware dan kebijakan pelanggaran.

4. *Host-based intrusion prevention system* (HIPS)

   Paket perangkat lunak yang diinstal dengan tujuan memonitor sebuah host untuk aktivitas yang mencurigakan dengan menganalisis peristiwa yang terjadi dalam host.

1.8.2 Metode Pada IPS

Sebagian besar *Intrusion Prevention Systems* (IPS) menggunakan salah satu dari tiga metode deteksi, diantaranya[16][20] :

1. *Signature-Based Detection*

   Signature berbasis IDS memonitor paket dalam Jaringan dan membandingkan dengan pola serangan yang telah ditentukan sebelumnya pra-konfigurasi dan dikenal sebagai *signatures*.

2. *Statistical anomaly-based detection*

   Sebuah *Statistical anomaly-based* IDS menentukan aktivitas jaringan yang normal seperti macam apa bandwidth umumnya digunakan, jenis protokol apa yang digunakan, jenis port apa dan perangkat umumnya menghubungkan bagi satu sama lain dan *alert* administrator atau pengguna ketika lalu lintas terdeteksi anomali (tidak normal).

3. *Stateful Protocol Analysis Detection*

   Metode ini mengidentifikasi penyimpangan dari protokol dengan membandingkan peristiwa yang diamati dengan "profil yang telah ditetapkan definisi yang berlaku umum atas aktivitas normal”[16].

1.9 SSH (*Secure Shell*)

Pada tahun 1995, Tatu Ylönen, seorang peneliti di Helsinki University of Technology, Finlandia, merancang versi pertama protokol (sekarang disebut SSH-1) karena didorong oleh peristiwa serangan pembongkaran sandi di jaringan universitas. Tujuan dari pembuatan SSH adalah untuk menggantikan fungsi rlogin, TELNET, dan rsh protokol, yang tidak memberikan

otentikasi kuat atau menjamin kerahasiaan. Ylönen merilis SSH sebagai *freeware* pada bulan Juli 1995, dan tool tersebut berkembang dengan cepat untuk mendapatkan popularitas. Menjelang akhir 1995, basis pengguna SSH telah tumbuh hingga 20.000 pengguna di lima puluh negara.

Secure Shell (SSH) adalah sebuah protokol jaringan kriptografi untuk komunikasi data yang aman, login antarmuka baris perintah, perintah eksekusi jarak jauh, dan layanan jaringan lainnya antara dua jaringan komputer. Ini terkoneksi, melalui saluran aman atau melalui jaringan tidak aman, server dan klien menjalankan server SSH dan SSH program klien secara masing-masing[21]. Protokol spesifikasi membedakan antara dua versi utama yang disebut sebagai SSH-1 dan SSH-2.

Aplikasi yang paling terkenal dari protokol ini adalah untuk akses ke akun shell pada sistem operasi mirip Unix, tetapi juga dapat digunakan dengan cara yang sama untuk akun pada Windows. Ia dirancang sebagai pengganti Telnet dan protokol remote shell lainnya yang tidak aman seperti rsh Berkeley dan protokol rexec, yang mengirim informasi, terutama kata sandi, dalam bentuk teks, membuat mereka rentan terhadap intersepsi dan penyingkapan menggunakan penganalisa paket[22] Enkripsi yang digunakan oleh SSH dimaksudkan untuk memberikan kerahasiaan dan integritas data melalui jaringan yang tidak aman, seperti Internet.

SSH menggunakan kriptografi kunci publik untuk mengotentikasi komputer *remote* dan biarkan komputer *remote* untuk mengotentikasi pengguna, jika perlu[21] Ada beberapa cara untuk menggunakan SSH; salah satunya adalah dengan menggunakan secara otomatis *public-privat key* pasangan untuk dengan sederhana mengenkripsi koneksi jaringan, dan kemudian menggunakan otentikasi password untuk login.

Penggunaan yang lain dengan menghasilkan secara manual pasangan *public-privat key* untuk melakukan otentikasi, yang memungkinkan pengguna atau program untuk login tanpa harus menentukan password. Dalam skenario ini, siapa pun dapat menghasilkan pasangan yang valid dari kunci yang berbeda (publik dan privat). Kunci publik ditempatkan pada semua komputer yang harus

memungkinkan akses ke pemilik *private key* yang valid (pemilik menjaga rahasia kunci privat). Sementara otentikasi didasarkan pada kunci privat, kunci itu sendiri tidak pernah ditransfer melalui jaringan selama otentikasi. SSH hanya memverifikasi apakah orang yang sama yang menawarkan kunci publik juga memiliki kunci pribadi yang valid. Dalam semua versi SSH adalah penting untuk memverifikasi kunci publik yang tidak diketahui, yaitu mengaitkan kunci publik dengan identitas, sebelum menerima mereka dengan valid. Menerima serangan kunci publik tanpa validasi akan mengotorisasi penyerang yang tidak sah sebagai pengguna yang valid.

1.9.1 Penggunaan SSH

SSH adalah sebuah protokol yang dapat digunakan untuk berbagai aplikasi. Beberapa aplikasi di bawah ini mungkin membutuhkan fitur-fitur yang hanya tersedia atau yang kompatibel dengan klien atau server SSH yang spesifik. Sebagai contoh, menggunakan protokol SSH untuk mengimplementasikan VPN adalah dimungkinkan, tapi sekarang hanya dapat dengan implementasi server dan klien OpenSSH. 123

1. Dalam kombinasi dengan SFTP, sebagai alternatif yang aman untuk FTP transfer file.
2. Dalam kombinasi dengan rsync untuk mem-backup, menyalin dan me-mirror file secara efisien dan aman.
3. Untuk *port forwarding* atau *tunneling port* (jangan disamakan dengan VPN yang rute paket antara jaringan yang berbeda atau menyambung dua wilayah *broadcast* menjadi satu).
4. Untuk digunakan sebagai VPN yang terenkripsi penuh. Klien yang mendukung fitur ini.
5. Untuk meneruskan X11 melalui beberapa host.
6. Untuk *browsing web* melalui koneksi *proxy* yang dienkripsi dengan klien SSH yang mendukung protokol SOCKS.
7. Untuk mengamankan *mounting* direktori di *server remote* sebagai sebuah sistem file di komputer lokal dengan menggunakan SSHFS

8. Untuk mengotomasi *remote monitoring* dan pengelolaan *server* melalui satu atau lebih dari mekanisme seperti yang dibahas di atas.

1.9.2 Arsitektur SSH

SSH-2 protokol memiliki arsitektur internal (didefinisikan dalam RFC 4.251) pada lapisan terpisah dengan baik. Yaitu:

1. Lapisan transportasi (RFC 4253)

   Lapisan ini menangani pertukaran kunci awal dan server otentikasi dan set up enkripsi, kompresi dan integritas verifikasi. Lapisan ini memperlihatkan ke lapisan atas sebuah antarmuka untuk mengirim dan menerima paket teks terang hingga masing-masing 32.768 byte (atau lebih yang diperbolehkan oleh implementasi). Lapisan transportasi juga mengatur ulang pertukaran kunci, biasanya setelah 1 GB data yang ditransfer atau setelah 1 jam telah berlalu, tergantung mana yang lebih cepat.

2. Lapisan otentikasi pengguna (RFC 4252)

   Lapisan ini menangani otentikasi klien dan menyediakan sejumlah metode otentikasi. Otentikasi *client-driven*: ketika seseorang diminta untuk memasukkan password, mungkin diminta oleh klien SSH, bukan *server*-nya. *Server* hanya menanggapi permintaan otentikasi klien. Metode otentikasi pengguna yang sering digunakan meliputi:

   1. Password

      sebuah metode untuk otentikasi password secara langsung, termasuk fasilitas yang memungkinkan sandi untuk diubah. Metode ini tidak diimplementasikan pada semua program.

   2. Kunci Publik

      sebuah metode untuk otentikasi berbasis kunci publik, biasanya mendukung setidaknya pasangan kunci DSA atau RSA, pada implementasi lain juga mendukung sertifikat X.509.

3. keyboard-*interactive* (RFC 4256)

   sebuah metode serbaguna di mana *server* akan mengirimkan satu atau lebih *prompt* untuk memasukkan informasi sehingga klien menampilkannya dan mengirimkan kembali tanggapan oleh pengguna. Digunakan untuk menyediakan otentikasi password sekali-waktu seperti S/Key atau SecurID. Digunakan oleh beberapa konfigurasi OpenSSH dimana PAM bertindak sebagai penyedia otentikasi host yang mendasar agar secara efektif dapat menyediakan otentikasi password, namun kadang-kadang menyebabkan kegagalan untuk login dengan klien yang hanya mendukung metode otentikasi password biasa.

4. metode otentikasi GSSAPI

   Sebuah skema *extensible* untuk melakukan otentikasi SSH menggunakan mekanisme eksternal seperti Kerberos 5 atau NTLM, menyediakan satu kemampuan sign on untuk sesi SSH. Metode ini biasanya digunakan pada implementasikan SSH komersial untuk digunakan dalam organisasi, meskipun OpenSSH memang memiliki implementasi kerja GSSAPI.

3. Lapisan koneksi

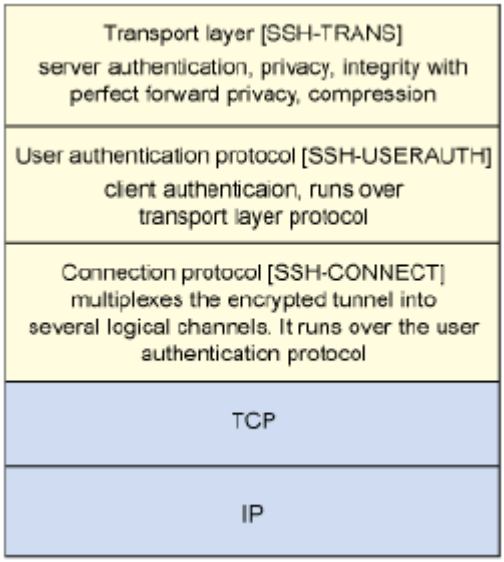
   Lapisan ini mendefinisikan konsep kanal, kanal permintaan dan permintaan global menggunakan layanan yang disediakan SSH. Sebuah koneksi SSH dapat melayani beberapa kanal secara bersamaan, masing-masing mentransfer data dalam dua arah. Permintaan kanal tersebut digunakan untuk menyambungkan saluran data spesifik secara *out-of-band*, seperti perubahan ukuran jendela terminal atau *exit code* dari sebuah proses *server-side*. Klien SSH meminta sebuah *port server-side* untuk diteruskan

menggunakan sebuah permintaan global. Jenis saluran standar yang tersedia adalah:

1. shell untuk terminal, SFTP dan *request exec* (termasuk transfer SCP)
2. *direct*-tcpip untuk koneksi klien-ke-*server* yang diteruskan
3. *forwarded*-tcpip for *server-to-client forwarded connections* *forwarded*-tcpip untuk koneksi *server*-ke-klien yang diteruskan

4. SSHFP DNS record (RFC 4255) menyediakan sidik jari kunci publik untuk membantu memverifikasi keaslian host.

Fungsi lapisan transportasi sendiri sebanding dengan TLS; lapisan otentikasi pengguna sangat *extensible* dengan metode otentikasi khusus; dan lapisan sambungan menyediakan kemampuan untuk membuat banyak sesi sekunder ke dalam satu koneksi SSH, sebuah fitur yang sebanding dengan BIP dan tidak tersedia di TLS.



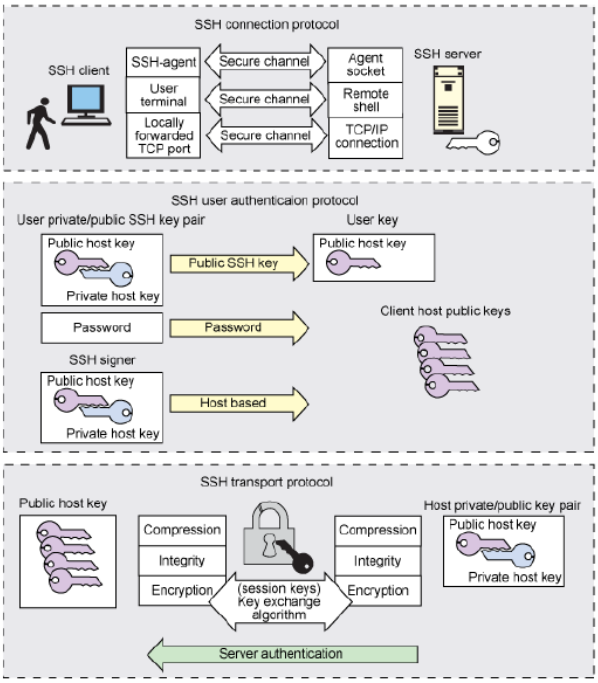
Gambar 2.11 SSH Protocol Logical Layers

CSI layer name	SSH elements	
Application	File transfers, z-modem uploads and downloads, remote SSH sessions, terminal display, and control such as vt100	7
Presentation	Encryption, data translation, and data compression	6
Session	Session creation and destruction, X11, port forwarding, checksums, synchronization, tokenizing	5
Transport		4
Network		3
Data link		2
Physical		1

Gambar 2.12 SSH within the Seven-layer OSI Model

1.9.3 Pasangan kunci privat dan publik untuk SSH

Gambar 2.13 dibawah ini menjelaskan proses transaksi kunci privat dan publik pada protokol jaringan SSH.



Gambar 2.13 Diagram transaksi pasangan kunci privat-publik SSH, sebagaimana didefinisikan dalam model arsitektur SSH

Untuk membantu memvalidasi identitas, SSH memiliki kapasitas / kemampuan untuk manajemen kunci dan agen terkait. Ketika dikonfigurasi dengan otentikasi kunci publik, kunci membuktikan identitas pengguna ke remote SSH host. Identitas berbasis SSH terdiri dari dua bagian: sebuah kunci publik dan sebuah kunci privat. Kunci privat SSH adalah identitas pengguna untuk koneksi SSH *outbound* dan harus dijaga kerahasiaannya. Ketika pengguna memulai sesi SSH atau SCP ke host atau server jauh, hal ini dikatakan SSH client. Melalui algoritma matematika, sebuah kunci privat seperti kartu identitas elektronik; kunci publik seperti gembok atau gerbang mekanisme untuk kartu identitas pengguna. Kunci privat mengatakan, "Ini benar-benar adalah Fred Smythe"; kunci publik mengatakan, "Ya, Anda memang nyata Fred Smythe, Anda sekarang telah dikonfirmasi: Silahkan masuk.".

Kunci publik pengguna mewakili siapa pengguna yang akan memungkinkan akses masuk ke melalui pintu atau kunci. Kunci publik tidak perlu dirahasiakan; mereka tidak dapat digunakan untuk kompromi sistem atau untuk akses yang tidak beralasan ke dalam sistem. Pada sistem Linux atau UNIX, pasangan kunci privat dan public ini disimpan dalam file teks ASCII; pada sistem Windows, beberapa program menyimpan pasangan kunci sebagai file text, beberapa di registry Windows.

Beberapa identifikasi menggunakan beberapa kunci privat dapat dibuat dengan konfigurasi SSH Protokol 2. Mari kita lihat bagaimana untuk menghasilkan, menyiapkan, dan mengkonfigurasi pasangan kunci privat dan public SSH pada host Linux, dapat dilihat pada gambar 2.13[23].

1.10 *Short Message Service* (SMS)

SMS adalah sebuah layanan yang banyak diaplikasikan pada sistem komunikasi tanpa kabel yang merupakan salah satu fasilitas dari GMS yang dikembangkan dan distandarisasi oleh suatu badan yang bernama *European Telecomunication Standards Institute* (ETSI) sebagai bagian dari pengembangan GSM fase 2, yang terdapat pada dokumentasi GSM 03.40 dan GSM 03.38. Fitur SMS ini memungkinkan perangkat Stasiun Seluler Digital (*Digital Cellular*

Terminal, seperti ponsel) untuk dapat mengirim pesan dalam bentuk *alphanumeric* dan menerima pesan-pesan teks dengan panjang sampai dengan 160 karakter melalui jaringan GSM. (ETSI, 1996)

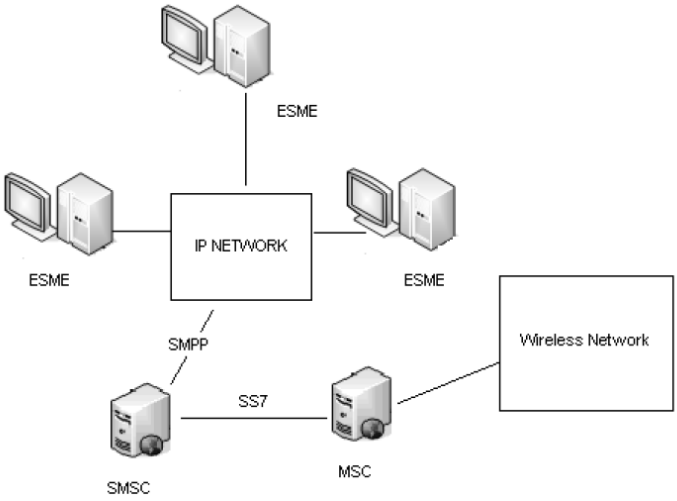
Isu SMS pertama kali muncul dibelahan Eropa pada sekitar tahun 1991 bersama sebuah teknologi komunikasi *wireless* yang saat ini cukup banyak pengunanya, yaitu *Global System for Mobile Comunication* (GSM). Dipercaya bahwa pesan pertama yang dikirimkan menggunakan SMS dilakukan pada bulan Desember 1992, dikirimkan dari sebuah *Personal Computer* (PC) ke telepon *mobile* (bergerak) dalam jaringan GSM milik Vodafone Inggris. Perkembangannya kemudian merambah ke benua Amerika, dipelopori oleh beberapa operator komunikasi bergerak berbasis digital seperti BellSouth Mobility, PrimeCo, Nextel, dan beberapa operator lain.

Layanan SMS merupakan layanan yang bersifat *nonreal time* dimana sebuah *short message* dapat di-*submit* ke suatu tujuan, tidak peduli apakah tujuan tersebut aktif atau tidak. Karakteristik utama SMS adalah SMS merupakan sebuah sistem pengiriman data dalam paket yang bersifat *out-of-band* dengan *bandwidth* kecil, dengan karakteristik ini pengiriman suatu *burst data* yang pendek dapat dilakukan dengan efisiensi yang sangat tinggi.

Dalam SMS mekanisme utama yang dilakukan dalam sistem adalah sistem melakukan pengiriman *short message* dari satu terminal pelanggan ke terminal yang lain. Hal ini dapat dilakukan berkat adanya sebuah Entitas dalam sistem SMS yang bernama *Short Message Service Center* (SMSC), disebut juga *Message Center* (MC). SMSC merupakan sebuah perangkat yang melakukan tugas *store and forward* trafik *short message*. Di dalamnya termasuk penentuan atau pencarian rute tujuan akhir dari *short message*. Sebuah SMSC biasanya didesain untuk dapat menangani *short message* dari berbagai sumber seperti *Voice mail System* (VMC), *Web-based message, Email Integration, External Short Messaging Entities* (ESME), dan lain-lain. Dalam interkoneksi dengan entitas dalam jaringan komunikasi *wireless* seperti *Home location Register* (HLR) dan *mobile Switching Center* (MSC), SMSC biasanya selalu menggunakan *Signal Transfer Point* (STP).

1.10.1 Arsitektur Jaringan SMS

Layanan SMS dibangun dari berbagai entitas yang saling terkait dan mempunyai fungsi dan tugas masing-masing. Tidak ada satu pun dalam sistem SMS yang dapat bekerja secara parsial. Entitas dalam jaringan SMS ini disebut juga elemen jaringan SMS. Secara umum yang dintegrasikan dengan jaringan *wireless*, adalah sebagai berikut :



Gambar 2.14 Arsitektur Dasar Jaringan SMS

1.10.2 Elemen Dasar Jaringan SMS

1. *Short Messaging Entities* (SME), suatu piranti yang dapat menerima dan mengirim pesan pendek.
2. *Short Message Service Center* (SMSC), kombinasi perangkat lunak dan perangkat keras yang brtanggung jawab memperkuat, menyimpan dan meneruskan pesan pendek antar SME dan piranti bergerak (*mobile phone).*
3. *Signaling System 7* (SS7), protokol *signalling* yang umum digunakan dalam jeringan telepon seluler.
4. *Base Station System* (BSS), berfungsi mengendalikan satu atau lebih BTS dan bertanggung jawab dalam pemberian sumber data transmisi sinyal radio elektromagnetis antara MSC dan *mobile phone*.
5. *Home Location Register* (HLR), basis data yang digunakan untuk penyimpanan permanen, pengelolaan dan profil layanan.

6. *Visitor Location Register* (VLR), basis data yang berisi informasi temporal mengenai pelanggan yang berasal dari suatu HLR yang roaming ke HLR lainnya.

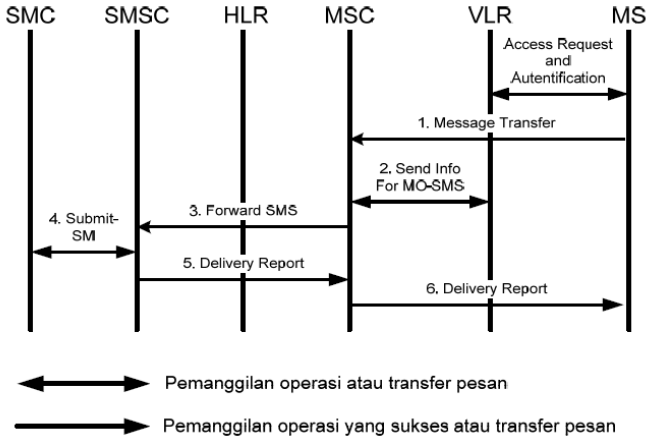
1.10.3 Elemen Layanan

SMS terdiri dari beberapa elemen layanan yang relevan terhadap penerimaan dan pengiriman pesan pendek yaitu :

1. *Message Expiration*, SMSC akan menyimpan dan mencoba mengirimkan kembali pesan yang mengalami kegagalan sampai pengiriman tersebut berhasil.
2. *Priority*, Untuk memberi tanda pesan-pesan yang penting dan membedakannya dari pesan biasa.

Sistem SMS memiliki dua layanan dasar *point-to-point* bagi pelanggan yaitu:

a. *Mobile-Oriented* (MO) *Short Message*, dikirimkan dari *mobile phone* yang MO-*Capable* ke SMSC dan dapat ditujukan ke *mobile phone* lainnya. Pada layanan ini selalu ada laporan yang dikirimkan ke *mobile phone*, baik yang mengkonfirmasikan pengiriman pesan pendek ke SMSC ataupun mengkonfirmasikan kegagalan pengiriman dan mengidentifikasikan penyebabnya.

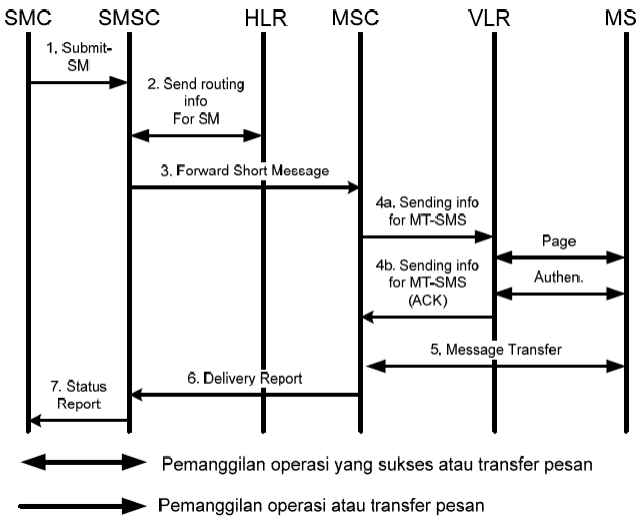


Gambar 2.15 Skenario pengiriman MO-SM

Keterangan :

1. MS mengirimkan SM ke MSC
2. MSC menginterogasi VLR utuk membuktikan bahwa pengiriman pesan tersebut tidak melanggar permintaan layanan pembatasan yang telah ditetapkan.
3. MSC mengirimkan pesan pendek ke SMSC dengan menggunakan operasi *Forward Short Message.*
4. SMSC mengirimkan pesan pendek ke SME
5. SMSC memberitahu MSC mengenai keberhasilan operasi *Forward Short Message.*
6. MSC mengembalikan hasil dari operasi MO-SM ke MS.

b. *Mobile Terminated* (MT) *Short Message*, dikirimkan dari SMSC ke *mobile phone* dan dapat sampai ke SMSC dari *mobile phone* lainnnya melalui MO-SM, pada layanan ini juga terdapat laporan yang diberikan kepada SMSC yang isinya bisa berupa konfirmasi pengiriman pesn pendek ke *mobile phone* maupun informasi kegagalan pengiriman pesan.



Gambar 2.16 Skenario pengiriman MT-SM

Keterangan :

1. Pesan pendek dikirimkan dari SME ke SMSC.
2. Setelah menyelesaikan pengolahan internalnya, SMSC menginterogasi HLR dan menerima informasi *routing* untuk pelanggan *mobile*.

3. SMSC mengirimkan pesan pendek ke MSC dengan menggunakan operasi *Forward Short Message.*
4. MSC mengambil informasi pelanggan dari VLR. Operasi ini dapat melibatkan prosedur autentifikasi.
5. MSC mengirimkan pesan pendek ke MS.
6. MSC mengembalikan hasil dari operasi *Forward Short Message* ke SMSC.
7. Jika diminta oleh SME, SMSC akan mengembalikan laporan status yang mengindikaikan pengiriman pesan pendek.

1.11 *Global System for Mobile Communications* (GSM)

Komunikasi sekarang ini tidak mengenal jarak dan waktu. Seseorang dapat melakukan komunikasi dari tempat yang berbeda tampa mengalami kesulitan. Ini semua karena adanya salah satu teknologi yang popular GMS (*Global System for Mobile Communication*) yaitu standardisasi untuk komunikasi mobile.

Alokasi spektrum frekuensi untuk GSM awalnya dilakukan pada tahun 1979. Spektrum ini terdiri atas dua buah *sub-band* masing-masing sebesar 25MHz, antara 890MHz -915MHz dan 935MHz - 960MHz. Sebuah *sub-band* dialokasikan untuk frekuensi *uplink* dan *sub-band* yang lain sebagai frekuensi *downlink.*

Kedua *sub-band* tersebut dibagi lagi menjadi kanal-kanal, sebuah kanal pada satu *sub-band* memiliki pasangan dengan sebuah kanal pada *sub-band* yang lain. Tiap *sub-band* dibagi menjadi 124 kanal, yang kemudian masing-masing diberi nomor yang dikenal sebagai *Absolute Radio Frequency Channel Number* (ARFCN). Jadi sebuah *handphone* yang dialokasikan pada sebuah ARFCN akan beroperasi pada satu frekuensi untuk mengirim dan satu frekuensi untuk menerima sinyal.

1.12 AT-*Command*

AT-*Command* atau *ATtention Command* yaitu perintah AT (Hayes AT*Command)* yang digunakan untuk berkomunikasi dengan terminal (modem) melalui gerbang serial pada komputer. Dengan penggunaan perintah AT, dapat

diketahui atau dibaca kondisi dari terminal, seperti mengetahui kondisi sinyal, kondisi baterai, mengirim pesan, membaca pesan, menambah item pada daftar telepon, dan sebagainya. Beberapa jenis ponsel memiliki *extended* AT *Command* yang bisa digunakan untuk mengambil informasi jenis, model hp, nomor *Internasional Mobile Station Equipment Identity* (IMEI), SIM *Subscriber Identification Number* (IMSI), status batere, kekuatan sinyal, nama operator, lokasi dan cell ID.

Tabel 2.5 Perintah AT-*Command*

AT-*Command*	Singkatan	Fungsi
AT+C	*Attention + Command*.....	Awal perintah hayes untuk....
AT+CGMI	Get Manufactured Identification	Menampilkan ID pabrik pembuat
AT+CGMM	Get Mobile Model	Menampilkan ID model *Handphone*
AT+CGSN	Get Serial Number	Menampilkan serial nomor IMEI
AT+COPS	Operator Selection	Memilih operator jaringan
AT+CLCK	Lock	Merubah penguncian telepon
AT+CPWD	Change Password	Merubah *password*
AT+CLIP	List Incoming Phone	Menampilkan nomor telepon masuk
AT+CPIN	Pin	Memasukan PIN
AT+CBC	Battery Charge	Mengisi batere
AT+CSQ	Signal Quality	Menampilkan kualitas sinyal
AT+CPBW	Phone Book Write	Memasukan data pada buku telepon
AT+CRSM	Restricted SIM Access Mode	Memblokir akses SIM
AT+CAOC	Advice Of Charge	Menampilkan biaya panggilan
AT+CCLK	Clock Mode	Pengaturan Jam
AT+CALM	Alert Sound Mode	Pengaturan jenis nada dering
AT+CRSL	Ringer Sound Level	Pengaturan volume nada dering
AT+CMUT	Mute Control	Pengaturan tanpa suara
AT+CLVL	Mute Control	Pengaturan volume pengeras suara
AT+CVIB	Vibration Mode	Pengaturan nada getar
AT+CPBR	Phone Book Read	Menampilkan daftar buku telepon
AT+CLCC	List Curent Call	Menampilkan penelepon terakhir
AT+CMGR	Read message	Membaca pesan berdasarkan

		index pesan
AT+CMGL	List message	Membaca pesan berdasarkan kategori
AT+CMGS	Send message	Mengirim pesan

Pada tabel 2.5 yaitu menjelaskan tentang perintah-perintah AT-*Command* yang digunakan untuk berkomunikasi dengan terminal (Modem) melalui gerbang serial pada komputer.

1.13 Regex

Regex adalah sebuah *pattern matching*, yaitu notasi yang digunakan untuk mencocokkan *teks* dan data, dan memanipulasinya. Biasanya lebih sering digunakan untuk *string*.

Contoh: validasi input *email*/*password*, *searching*, *replace*. Regex memiliki 2 fungsi utama yakni mencari dan mengganti, mencari suatu pola tertentu dalam *text* lalu menggantinya menjadi pola yang lain. Dengan menggunakan Regex, kita dapat dengan mudah menemukan pola karakter spesifisik dari sejumlah *teks*. Kita juga dapat dengan mudah mengganti, memodifikasi atau menghapus suatu *substring* dengan cepat dan akurat sesuai dengan Regex yang telah kita buat. Keuntungan menggunakan regex adalah tidak ada keterbatasan untuk ukuran data *numeric*.

Regular Expression digunakan oleh banyak *teks editor*, *utilities*, dan bahasa pemrograman untuk pencarian dan memanipulasi teks berdasarkan pola. Misalnya, Perl, Ruby, PHP,VB dan Tcl memiliki *engine Regular Expression* yang kuat dibangun pada *syntax* mereka.

Sering kali orang beranggapan bahwa regex susah dan membingungkan. Namun sebenarnya regex sangatlah membantu dalam menemukan pola-pola kalimat. Sehingga percobaan terhadap semua kemungkinan pola kalimat tidak perlu dilakukan. Sebuah alasan yang sangat bagus untuk menggunakan regex adalah karena regex sangatlah *powerfull*. Pada level rendah regex dapat mencari sebuah penggalan kata. Pada level tinggi regex mampu melakukan kontrol terhadap data. Baik mencari, menghapus dan merubah.

1.13.1 Prinsip Kerja Regex

Engine RegEx terdiri dari 2 jenis *Text-directed engine* dan *regex-directed engine* atau ada juga yang mengatakan DFA (*Deterministic Finite Automaton*) dan NFA (*Nondeterministic Finite Automaton*) *engines*. Namun jenis mesin *engine* yang lebih banyak diminati adalah *regex-directed engine*, disamping itu *feature*-nya lebih hebat dari *text-directed engine*.

Misalnya saja kita mencocokan kata "regex" dengan kata "belajar regex sekarang", pertama-tama regex akan memulai mencocokan karakter pertama yaitu huruf "R" dengan "B", karena tidak cocok maka akan dilanjutkan ke tempat selanjutnya yaitu "R" dengan "E", karena masih belum cocok juga proses ini akan terus berlanjut, hingga akhirnya menemukan hurup "R" yaitu di posisi ke-7, hasil ini akan disimpan di memori bahwa telah valid pada posisi ke-7, kemudian akan meneruskan langkah selanjutnya mencocokan hurup "E" dengan " " dan ternyata tidak cocok, engine akan mulai melakukan pencocokan kembali. Karena huruf "E" tidak ditemukan maka kali ini *engine* akan kembali menggunakan huruf "R" dan dicocokan pada posisi ke-9 yakni huruf "R". Karena valid, *engine* menggunakan huruf berikutnya yakni huruf "E" dan dicocokan dengan posisi ke-10 yakni huruf "E". Proses ini akan terus diulang hingga mencapai posisi akhir suatu himpunan *string*. Nah jika kata "regex" ditemukan oleh kata "belajar regex sekarang" maka *engine* akan melaporkan bahwa *Regular Expression* telah valid, meskipun masih ada karakter yang belum di validasi (kata "sekarang"). Itu adalah cara kerja NFA yang lama, berbeda dengan mesin NFA yang sekarang, sudah lebih baik, yaitu meskipun string yang dicari telah *match* tetap akan diteruskan sampai benar-benar valid ditemukan. Setelah itu baru dibuat laporan sukses. Berbeda dengan *engine* dari NFA, engine DFA memilik cara kerja yakni membandingkan semua karakter secara serempak, hal ini membuat *total memory* yang dibutuhkan lebih besar dari NFA, sehingga teknologi NFA lebih diminati.

Beberapa sistem yang telah menggunakan teknologi NFA ini diantaranya adalah Perl, Java, .NET languages, PHP, Python, Ruby dan masih banyak lagi.

1.13.2 Dasar Teori Regex

Salah satu fungsi yang dapat digunakan untuk operasi regex adalah: ereg(string_pola, string_sumber); Fungsi ini akan menghasilkan *TRUE*, jika ternyata pola cocok dengan sumber dan *FALSE* jika keduanya tidak cocok. Pola dari fungsi di atas adalah rumus dari data *text* yang diinginkan. Cara membuat rumus dengan menggunakan karakter *meta*. singkatnya *meta* itu adalah karakter yang memiliki arti tertentu. contoh pada perintah DOS adalah DIR *.*, karakter * tidak diartikan sebagai karakter bintang oleh komputer melainkan mempunyai arti khusus.

Ada beberapa karakater *meta* yang perlu di ketahui, yaitu : Karakter *meta* | (Garis Vertikal). Karakter | artinya menjadikan karakter *meta text* sebagai pilihan pada pola untuk di cocokan dengan sumber. Contoh: ereg(Uus | Lukman |Nanang, Lukman) Artinya akan *TRUE* karena Lukman adalah salah satu pilihan dari pola. namun jika sumbernya ternyata adalah Hasan, hasilnya akan *FALSE* alias tidak cocok, karena hasan tidak ada pada salah satu pilihan tersebut. (dan), karakter kurung buka dan kurung tutup karakter kurung buka dan kurung tutup berguna untuk mengelompokan beberapa kata atau data text agar dianggap menjadi suatu kesatuan. Contoh: ereg((Uus Musailini)|(Lukmanul Hakim)|Nanang, Lukmanul); Hasil penyataan regex di atas adalah *FALSE* alias tidak cocok , karena Lukmanul Hakim adalah satu kesatuan, sehingga tidak sama Lukmanul saja.

Tabel 2.6 Tabel Pola Umum Pada Regex

Pola	Penjelasan
[ ]	Ekspresi kurung. cocok dengan satu karakter yang berada dalam kurung, misal: *pattern* "a[bcd]i" cocok dengan *string* "abi", "aci", dan "adi". penggunaan *range* huruf dalam kurung diperbolehkan, misal : pattern "[a-z]" cocok dengan salah satu karakter diantara string "a" sampai "z". *pattern* [0-9] cocok dengan salah satu angka. jika ingin mencari karakter "-" juga, karakter tersebut harus diletakkan di depan atau di belakang kelompok, misal: "[abc-]".

[^]	Cocok dengan sebuah karakter yang tidak ada dalam kurung, berlawanan dengan yang diatas. misal: *pattern* "[^abc]" cocok dengan satu karakter apa saja kecuali "a", "b", "c".
?	Cocok dengan nol atau satu karakter sebelumnya. misal: pattern "*died*?" cocok dengan *string* "die" dan "died".
+	Cocok dengan satu atau lebih karakter sebelumnya. misal: "yu+k" cocok dengan "yuk", "yuuk", "yuuuk", dan seterusnya.
*	Cocok dengan nol atau lebih karakter sebelumnya. misal: *pattern* "hu*p" cocok dengan *string* "hp", "hup", "huup" dan seterusnya.
{x}	Cocok dengan karakter sebelumnya sejumlah x karakter. misal: *pattern* "[0-9]{3}" cocok dengan bilangan berapa saja yang berukuran 3 digit.
{x,y}	Cocok dengan karakter sebelumnya sejumlah x hingga y karakter. misal: *pattern* "[a-z]{3,5}" cocok dengan semua susunan huruf kecil yang terdiri dari 3 sampai 5 huruf.
!	Jika diletakkan di depan *pattern*, maka berarti "bukan". misal *pattern* "!a.u" cocok dengan string apa saja kecuali string "alu", "anu", "abu", "asu", "aiu", dan seterusnya .
^	Jika diletakkan di depan *pattern*, akan cocok dengan awal sebuah *string*.
( )	*Grouping*. digunakan untuk mengelompokkan karakter - karakter menjadi *single* unit. *string* yang cocok dalam *pattern* yang berada dalam tanda

	kurung dapat digunakan pada operasi berikutnya. semacam *variable*.
\	*Escape Character*, mengembalikan fungsi meta *character* menjadi karakter biasa. pada beberapa *system* dapat berarti sebaliknya, yaitu *meta character* menggunakan *escape character* didepannya .
$	Jika diletakkan di belakang *pattern*, akan cocok dengan akhir sebuah string.

1.14 UML

Pada Oktober 1994 Dr. James Rumbaugh yang mengembangkan Object Modelling Technique (OMT) bergabung dengan perusahaan Rational Software. Sebelumnya juga bergabung Grady Booch yang mengembangkan Object Modelling Design (OOD). Duet mereka pada Oktober 1995 menghasilkan Unified Method versi 0.8, yang menjadi cikal bakal dari UML (*Unified Modelling language*) sebagai bahasa pemodelan standar untuk aplikasi object oriented[24].

Pada tahun 2002 lahir UML versi 2.0 dengan penambahan dan penggantian diagram menjadi 13 buah diagram. Diagram-diagram ini terbagi menjadi 3 kategori :

a) *Structural diagrams*

Menggambarkan elemen dari spesifikasi yang mengabaikan waktu. Terdiri dari : *Class Diagram, Object Diagram, Component Diagram, Deployment Diagram, Composite Structure Diagram dan Package Diagram.*

b) *Behavior diagram*

Menggambarkan ciri-ciri *behavior*/*method*/*function* dari sebuah sistem atau *business process*. Terdiri dari : *Use Case Diagram, Activity Diagram dan State Machine Diagram*.

c) *Interaction diagram*

Bagian dari *behavior* diagram yang menggambarkan *object interactions*. Terdiri dari : *Communication Diagram*, *Interaction Overview Diagram*, *Sequence Diagram dan Timing Diagram*.

Karena UML sangat fleksibel, ada juga cara melihat diagram UML berdasar kategori berikut :

a) *Static Diagram*

Menunjukkan segi static dari sistem. Kategori ini sama dengan *structural diagram*.

b) *Dynamic Diagram*

Menunjukkan bagaimana sistem berkembang setiap waktu. Meliputi *state-machine diagram* dan *timing diagram*.

c) *Functional Diagram*

Menunjukkan detail dari perilaku (*behavior*) dan algoritma bagaimana sistem memenuhi perilaku yang diinginkannya. Kategori ini termasuk *use case*, *interaction* dan *activity diagram*.

1.14.1 Analisa dan perancangan berorientasi obyek

Analisa dan desain berorientasi obyek berarti merumuskan dan menyelesaikan masalah serta menghasilkan suatu hipotesa dan diagnosa (solusi), memodelkannya dengan pendekatan/paradigma obyek (obyek adalah riil punya atribut/data dan perilaku).

Dalam melakukan analisa dan perancangan sistem berorientasi obyek menggunakan UML (*Unified Modelling Language*) untuk memodelkannya. Sedangkan alat (*tool*) *visual modelling* yang digunakan untuk menggambarkan model analisa dan perancangan adalah Star UML.

1.14.2 Analisa Berorientasikan Obyek

Object oriented analysis adalah metode analisis yang memeriksa *requirements* (syarat atau keperluan yang harus dipenuhi suatu sistem) (Suhendar dan Hariman, 2002:11).

Dalam tahap ini kegiatan-kegiatan yang dilakukan dalam menganalisa sistem sebagai berikut :

a) Menganalisa sistem yang ada dan mempelajari apa yang dikerjakan oleh sistem yang ada.
b) Menspesifikasikan sistem yaitu spesifikasi masukan yang digunakan database yang ada, proses yang dilakukan dan keluaran yang dihasilkan.

Tujuan dari analisa berorientasi obyek yaitu untuk menentukan kebutuhan pemakai secara akurat.

Pendekatan-pendekatan yang dipakai dalam analisa berorientasi obyek antara lain :

a) Pendekatan *top down*, yaitu memecahkan masalah ke dalam bagian-bagian terkecil atau per level sehingga mudah untuk diselesaikan.
b) Pendekatan modul, yaitu membagi sistem ke dalam modul-modul yang dapat beroperasi tanpa ketergantungan.
c) Penggunaan alat-alat bantu dalam bentuk grafik dan teks sehingga mudah untuk dimengerti serta dikoreksi apabila terjadi perubahan.

Pendekatan dalam analisa berorientasi obyek dilengkapi dengan alat-alat dan teknik-teknik yang dibutuhkan dalam pengembangan sistem, sehingga hasil akhir dari sistem yang dikembangkan akan didapatkan sistem yang terdefinisi dengan baik dan jelas.

1.14.3 *Use Case Diagram*

Use case diagram menggambarkan kebutuhan sistem dari sudut pandang *user*. Digunakan untuk menggambarkan hubungan antara internal sistem dan eksternal sistem atau hubungan antara *use case* dan aktor.

1.14.4 *Actor*

Actor adalah sesuatu (entitas) yang berhubungan dengan sistem dan berpartisipasi dalam *use case*. *Actor* menggambarkan orang, sistem atau entitas

eksternal yang secara khusus membangkitkan sistem dengan input atau masukan kejadian-kejadian, atau menerima sesuatu dari sistem. *Actor* dilukiskan dengan peran yang mereka mainkan dalam *use case*, seperti *Operator*, Konsumen dan lain-lain.



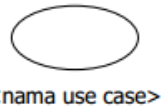
Gambar 2.17 Bentuk *Actor* dalam UML

Dalam *use case* diagram terdapat satu aktor pemulai atau *initiator actor* yang membangkitkan rangsangan awal terhadap sistem, dan mungkin sejumlah aktor lain yang berpartisipasi atau *participating actor*. Akan sangat berguna untuk mengetahui siapa *actor* pemulai tersebut.

1.14.5 *Use Case*

Use case yang dibuat berdasar keperluan aktor merupakan gambaran dari "apa" yang dikerjakan oleh sistem, bukan "bagaimana" sistem mengerjakannya. *Use case* diberi nama yang menyatakan apa hal yang dicapai dari interaksinya dengan aktor.

Dalam UML use case dinotasikan dengan gambar :

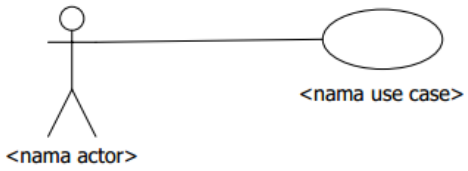


Gambar 2.18 Bentuk *Use Case* dalam UML

1.14.6 *Relationship*

Relasi (*relationship*) digambarkan sebagai bentuk garis antara dua simbol dalam *use case diagram*. Relasi antara *actor* dan *use case* disebut juga dengan asosiasi (*association*). Asosiasi ini digunakan untuk menggambarkan bagaimana hubungan antara keduanya.

Relasi-relasi yang terjadi pada *use case diagram* bisa antara *actor* dengan *use case* atau *use case* dengan *use case*.



Gambar 2.19 Bentuk *Relationship* dalam UML

Relasi antara *use case* dengan *use case* :

a. *Include*

   Pemanggilan *use case* oleh *use case* lain atau untuk menggambarkan suatu *use case* termasuk di dalam *use case* lain (diharuskan). Contohnya adalah pemanggilan sebuah fungsi program. Digambarkan dengan garis lurus berpanah dengan tulisan <<include>>.

b. *Extend*

   Digunakan ketika hendak menggambarkan variasi pada kondisi perilaku normal dan menggunakan lebih banyak kontrol *form* dan mendeklarasikan *ekstension* pada *use case* utama. Atau dengan kata lain adalah perluasan dari *use case* lain jika syarat atau kondisi terpenuhi. Digambarkan dengan garis berpanah dengan tulisan <<extend>>.

c. *Generalization/Inheritance*

   Dibuat ketika ada sebuah kejadian yang lain sendiri atau perlakuan khusus dan merupakan pola berhubungan *base-parent use case*. Digambarkan dengan garis berpanah tertutup dari *base use case* ke *parent use case*.

1.14.7 *Activity Diagram*

Diagram aktivitas menggambarkan proses bisnis dan urutan aktivitas-aktivitas yang mendukung penggambaran tindakan sistem baik yang

bersifat kondisional maupun paralel. Tindakan kondisional dilukiskan dengan cabang (*branch*) dan penyatuan (*merge*).

Sebuah branch memiliki sebuah *transition* masuk atau yang disebut dengan *incoming transition* dan beberapa *transition* keluar atau yang disebut dengan *outgoing transition* dari branch yang berupa keputusan-keputusan. Hanya satu dari *outgoing transition* yang dapat diambil, maka keputusan-keputusan tersebut harus bersifat *mutually exclusive*. [*else*] digunakan sebagai keterangan singkat yang menunjukkan bahwa *transition* "*else*" tersebut harus digunakan jika semua keputusan yang ada pada *branch* salah.

Sebuah *merge* memiliki banyak input *transition* dan sebuah *output*. *Merge* menandakan akhir dari suatu kondisi yang diawali dengan sebuah *branch*. Selain *branch* dan *merge*, di dalam diagram aktivitas terdapat pula *fork* dan *join*. *Fork* memiliki satu *incoming transition* dan beberapa *outgoing transition*. Sedangkan pada *join, outgoing transition* diambil atau digunakan hanya ketika semua *state* pada *incoming transition* telah menyelesaikan aktivitasnya.

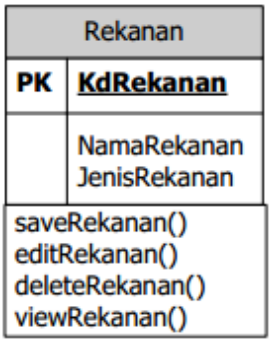
1.14.8 *Sequence Diagram*

Diagram yang menggambarkan bagaimana obyek berinteraksi dengan obyek lainnya melalui pesan (*message*) yang disampaikan, disusun dalam urutan kejadian atau waktu dan secara khusus berasosiasi dengan *use case*.

1.14.9 *Class Diagram*

Class diagram merupakan bagian yang paling penting dalm analisa dan perancangan berorientasi obyek. Dalam UML diagram kelas digunakan untuk memodelkan *static structure* dari sistem informasi.

Kelas merupakan himpunan dari obyek yang sejenis yang mempunyai atribut (*attribute*) dan perilaku (*behaviors/method*) yang sama. Atribut adalah sebuah nilai data karakteristik yang dimiliki oleh obyek sebuah kelas sedangkan *method* adalah perilaku atau operasi yang dikenakan oleh suatu kelas. Pada gambar kelas terdapat tiga bagiannya.



Gambar 2.20 Bentuk Class dalam UML

Diagram kelas menggambarkan struktur obyek sistem, dimana diperlihatkan hubungan amtar mereka. Diagram kelas merupakan fondasi untuk *component diagram* dan *deployment diagram*.

Secara garis besar terdapat 3 jenis *class*. Ketiga jenis *class* tersebut dikelompokkan berdasarkan fungsi dan karakternya masing-masing, yaitu :

a. *Entity Class Diagram*

Merupakan paket utama dari sistem yang berisi kumpulan kelas berupa entitas-entitas yang membentuk sistem dan menjadi landasan untuk menyusun basis data pada model data konseptual.



Gambar 2.21 Bentuk *Entity Class* dalam UML

b. *Control Class Diagram*

Berisi kumpulan kelas yang menjadi kontrol program termasuk koneksi dengan basis data dan merupakan kelas perantara atau penghubung antara *entity class* dengan kelas antar muka pemakai (*interface*).



Gambar 2.22 Bentuk *Control Class* dalam UML

c. *Boundary Class Diagram*

Berisi kumpulan kelas yang menjadi *interface* antara pemakai (*user*) dengan sistem, seperti tampilan *form* untuk pencetakan.



Gambar 2.23 Bentuk *Boundary Class* dalam UML

1.14.10 *Package Diagram*

Package (paket) adalah mekanisme pengelompokan yang digunakan untuk menandakan pengelompokan elemen-elemen model. Sebuah *package* dapat mengandung beberapa paket lain di dalamnya. *Package* digunakan untuk memudahkan pengorganisasian elemen-elemen model.

1.15 DBMS

Sistem manajemen basis data (Bahasa Inggris: *database management system*, DBMS), atau kadang disingkat SMBD, adalah suatu sistem atau perangkat lunak yang dirancang untuk mengelola suatu basis data dan menjalankan operasi terhadap data yang diminta banyak pengguna[25]. Contoh tipikal SMBD adalah akuntansi, sumber daya manusia, dan sistem pendukung pelanggan, SMBD telah berkembang menjadi bagian standar di bagian pendukung (*back office*) suatu perusahaan. Contoh SMBD adalah Oracle, SQL server 2000/2003, MS Access, MySQL dan sebagainya. DBMS merupakan perangkat lunak yang dirancang untuk dapat melakukan utilisasi dan mengelola koleksi data dalam jumlah yang besar. DBMS juga dirancang untuk dapat melakukan manipulasi data secara lebih mudah. Sebelum adanya DBMS, data pada umumnya disimpan dalam bentuk *flat file*, yaitu *file teks* yang ada pada sistem operasi. Sampai sekarangpun masih ada aplikasi yang menimpan data dalam bentuk *flat* secara langsung.

Menyimpan data dalam bentuk *flat file* mempunyai kelebihan dan kekurangan. Penyimpanan dalam bentuk ini akan mempunyai manfaat yang optimal jika ukuran filenya relatif kecil, seperti file password pada sistem operasi

Unix dan Unix-*like*. File password pada umumnya hanya digunakan untuk menyimpan nama yang jumlahnya tidak lebih dari 1000 orang.

Selain dalam bentuk *flat file*, penyimpanan data juga dapat dilakukan dengan menggunakan program bantu seperti spreadsheet. Penggunaan perangkat lunak ini memperbaiki beberapa kelemahan dari *flat file*, seperti bertambahnya kecepatan dalam pengolahan data. Namun metode ini masih memiliki banyak kelemahan, diantaranya adalah masalah manajemen dan keamanan data yang masih kurang. Penyimpanan data dalam bentuk DBMS mempunyai banyak manfaat dan kelebihan dibandingkan dengan penyimpanan dalam bentuk flat file atau *spreadsheet*, diantaranya :

1. Performa yang dapat dengan penyimpanan dalam bentuk DBMS cukup besar, sangat jauh berbeda dengan *performance* data yang disimpan dalam bentuk *flat file*. Disamping memiliki unjuk kerja yang lebih baik, juga akan didapatkan efisiensi penggunaan media penyimpanan dan memori.
2. Integritas data lebih terjamin dengan penggunaan DBMS. Masalah redudansi sering terjadi dalam *flat file*. Redudansi adalah kejadian berulangnya data atau kumpulan data yang sama dalam sebuah database yang mengakibatkan pemborosan media penyimpanan.
3. *Independensi*. Perubahan struktur database dimungkinkan terjadi tanpa harus mengubah aplikasi yang mengaksesnya sehingga pembuatan antarmuka ke dalam data akan lebih mudah dengan penggunaan DBMS.
4. Sentralisasi. Data yang terpusat akan mempermudah pengelolaan database. kemudahan di dalam melakukan bagi pakai dengan DBMS dan juga kekonsistenan data yang diakses secara bersama-sama akan lebiih terjamin dari pada data disimpan dalam bentuk file atau worksheet yang tersebar.
5. Keamanan. DBMS memiliki sistem keamanan yang lebih fleksibel daripada pengamanan pada file sistem operasi. Keamanan dalam

DBMS akan memberikan keluwesan dalam pemberian hak akses kepada pengguna.

1.16 C# .NET

Sekarang ini Microsoft telah dapat merealisasikan visi "Sistem operasi Windows dalam setiap PC dan PC dalam setiap desktop". Salah satu tool untuk mengembangkan aplikasi .NET adalah Microsoft C# .NET. C# .NET bersama dengan Visual C++ .NET, Visual VB .NET, Visual J++ .NET dan Visual J# .NET merupakan bagian dari Microsoft Visual Studio .NET.[26]

C# .NET adalah bahasa pemrograman untuk membuat aplikasi berbasis Windows, aplikasi form Web ASP .NET, layanan Web XML dan aplikasi *mobile* seperti komputer Palm dan Pocket PC. C# .NET dibangun di atas fondasi .NET Framework (lingkungan kerja .NET).

Setiap generasi baru dari perangkat lunak bahasa pemrograman datang karena adanya keterbatasan dari generasi sebelumnya. Teknologi *device*, *hardware*, *network* dan internet baru yang muncul menyebabkan bahasa pemrograman yang ada tidak lagi menjadi alat yang ideal untuk mengembangkan perangkat lunak yang dapat bekerja dengan teknologi baru tersebut.

Sekarang untuk pertama kalinya, *platform* pengembang perangkat lunak yang lengkap, Microsoft .NET telah didesain dari dasar dengan internet sebagai fokus utamanya (walaupun tidak secara eksklusif hanya untuk pengembang internet saja). Banyak inovasi baru yang berada dalam *platform* ini akan mengatasi keterbatasan dari *tool-tool* dan teknologi lama. Kelebihan C# .NET 2010 terletak pada tampilannya yang lebih canggih dibandingkan dengan edisi bahasa C dan C++.

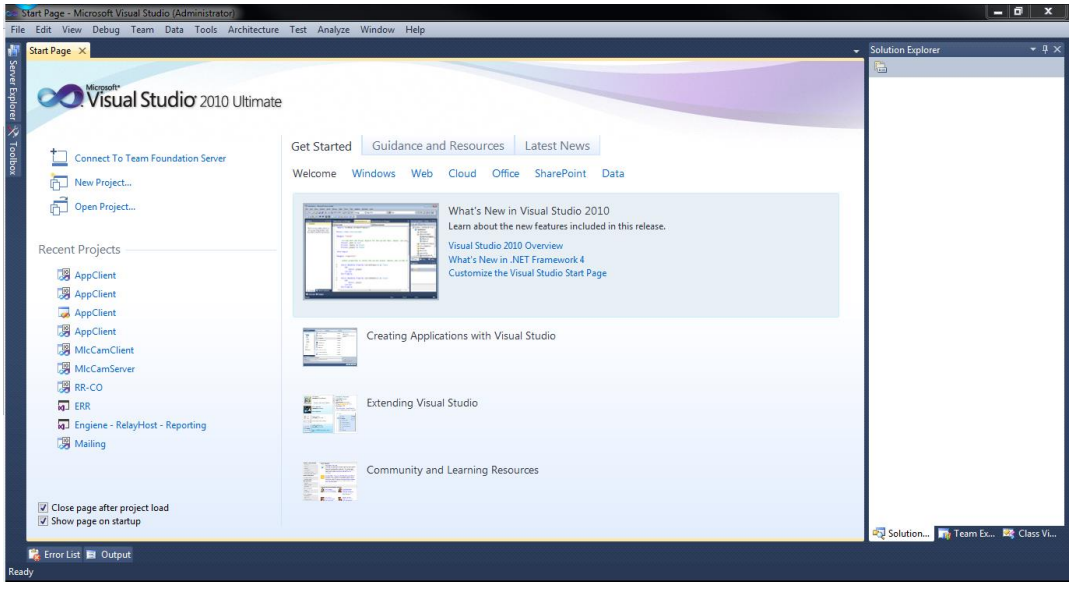
Selain memiliki kelebihan, C# .NET 2010 memiliki kekurangan. Kekurangan C# .NET 2010 yang terlihat jelas adalah beratnya aplikasi ini apabila dijalankan pada komputer yang memiliki spesifikasi sederhana.

Syarat minimal yang harus dipenuhi oleh komputer agar C# .NET 2010 dapat diinstal adalah :

[Sumber : Microsoft Visual Studio 2010 *Documentation*]

Tabel 2.7 Spesifikasi Minimal Visual Basic.NET 2010

Processor	*Computer that has a 1.6GHz or faster processor.*
Operating System	1. Windows XP (x86) *with Service Pack* 3 - *all editions except Starter Edition* 2. Windows Vista (x86 & x64) *with Service Pack* 2 - *all editions except Starter Edition* 3. Windows 7 (x86 & x64) 4. Windows Server 2003 (x86 & x64) *with Service Pack* 2 5. Windows Server 2003 R2 (x86 & x64) 6. Windows Server 2008 (x86 & x64) *with Service Pack* 2 7. Windows Server 2008 R2 (x64)
RAM	1 GB (32 Bit) *or* 2 GB (64 Bit) RAM (*Add* 512 MB *if running in a virtual machine*)
Hard Disk	1. 3GB *of available hard disk space* 2. 5400 RPM *hard disk drive*
CD or DVD *Drive*	*Not Required*
Display	*DirectX* 9 *capable video card running at* 1024 x 768 *or higher-resolution display*



Gambar 2.24 Tampilan utama Visual Studio 2010

1.17 MySQL

MySQL adalah sebuah perangkat lunak sistem manajemen basis data SQL (bahasa Inggris: *database management system*) atau DBMS yang *multithread*, *multi-user*, dengan sekitar 6 juta instalasi di seluruh dunia. MySQL B membuat MySQL tersedia sebagai perangkat lunak gratis dibawah lisensi GNU *General Public License* (GPL), tetapi mereka juga menjual dibawah lisensi komersial untuk kasus-kasus dimana penggunaannya tidak sesuai dengan penggunaan GPL. Tidak sama dengan proyek-proyek seperti Apache, dimana perangkat lunak dikembangkan oleh komunitas umum, dan hak cipta untuk kode sumber dimiliki oleh penulisnya masing-masing, MySQL dimiliki dan disponsori oleh sebuah perusahaan komersial Swedia MySQL AB, dimana memegang hak cipta hampir atas semua kode sumbernya. Kedua orang Swedia dan satu orang Finlandia yang mendirikan MySQL AB adalah: David Axmark, Allan Larsson, dan Michael "Monty" Widenius.