BAB II
LANDASAN TEORI

2.1. Jaringan Komputer

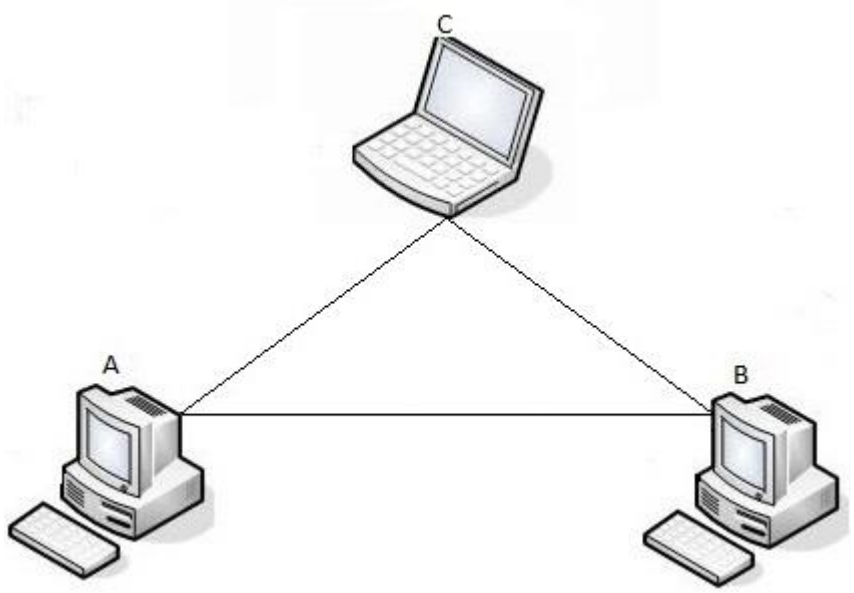
Jaringan komputer adalah sekelompok komputer otonom yang saling berhubungan antara satu dengan yang lainnya menggunakan protokol komunikasi melalui media komunikasi sehingga dapat saling berbagi informasi, program-program, dan menggunakan bersama perangkat keras [9].

2.1.1. Tipe Jaringan

Tipe jaringan komputer dapat dibedakan menjadi dua tipe yaitu:

1. Jaringan *peer to peer*
2. Jaringan *client-server*

2.1.1.1. Jaringan Peer to Peer



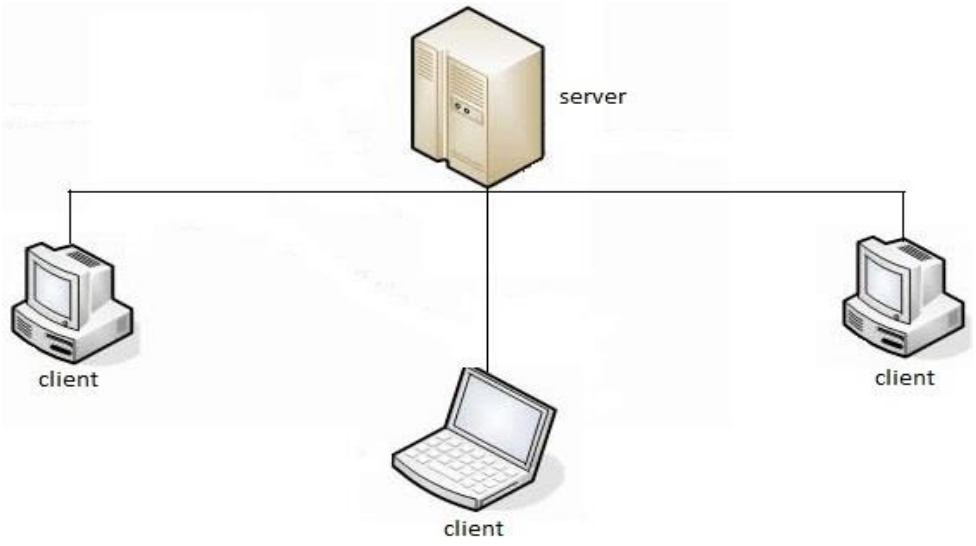
Gambar 2.1 Jaringan *Peer to Peer*

Jaringan *peer to peer* adalah sebuah jaringan komputer dimana setiap komputer disusun dan dihubungkan satu sama lain tanpa adanya kontrol terpusat yang berperan sebagai *server* murni. Pada jaringan *peer to peer* setiap komputer yang terhubung pada jaringan dapat berkomunikasi dengan komputer-komputer lain secara langsung tanpa melalui komputer perantara.

Komputer yang terhubung dalam jaringan *peer to peer* pada prinsipnya mampu untuk bekerja sendiri sebagai sebuah komputer. Untuk membangun jaringan seperti ini, dibutuhkan komputer-komputer yang memiliki kemampuan yang setara karena keamanan dalam jaringan tersebut diatur dan dikontrol oleh masing-masing komputer dalam jaringan tersebut.

Peer to peer juga dikenal dengan istilah *workgroups*. Jumlah pengguna dalam sebuah jaringan *workgroups* biasanya kurang dari 10 pengguna, masing-masing pengguna komputer dapat saling berbagi dan pada umumnya tidak begitu diperlukan pengaturan keamanan data di antara anggota *workgroup* tersebut.

2.1.1.2. Jaringan Client-Server



Gambar 2.2 Jaringan *Client-Server*

Pada jaringan *client-server* terdapat sebuah komputer yang berfungsi sebagai *server*, sedangkan komputer-komputer yang lain berfungsi sebagai *client*. Sesuai namanya maka komputer *server* berfungsi dan bertugas melayani seluruh komputer yang terdapat dalam jaringan tersebut.

Komputer-komputer ini disebut juga dengan *workstation*, yaitu komputer dimana pengguna jaringan dapat mengakses dan memanfaatkan pelayanan yang diberikan komputer *server*. Dalam sebuah jaringan komputer biasanya *workstation* menggunakan komputer yang memiliki kemampuan lebih rendah dibandingkan komputer server, meskipun tidak selalu demikian.

2.1.2. Arsitektur Jaringan Komputer

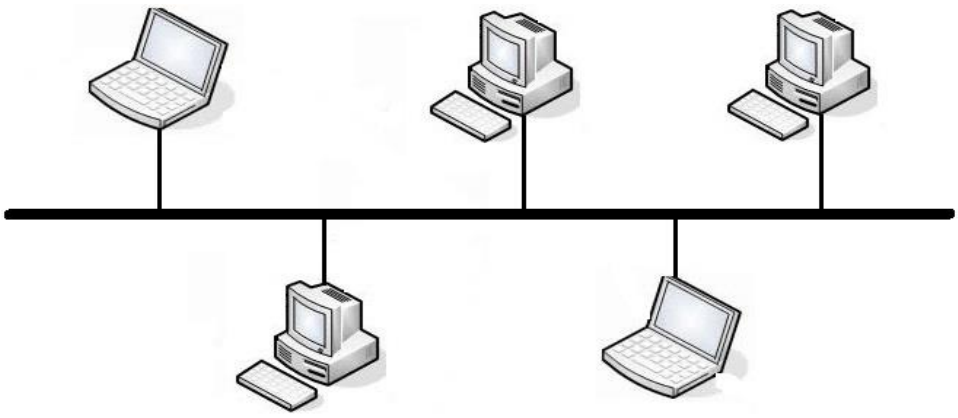
Arsitektur sebuah jaringan komputer dibedakan menjadi arsitektur fisik dan arsitektur *logic*. Arsitektur fisik berkaitan dengan susunan fisik sebuah jaringan komputer, biasa juga disebut dengan topologi jaringan. Sedangkan arsitektur *logic* berkaitan dengan logika hubungan masing-masing komputer dalam jaringan.

Bentuk-bentuk arsitektur jaringan komputer secara fisik adalah sebagai berikut:

1. Topologi *Bus.*
2. Topologi *Ring.*
3. Topologi *Star.*

2.1.2.1. Topologi Bus

Topologi *Bus* merupakan jenis topologi yang memiliki metode paling sederhana. Terdiri dari sebuah kabel yang menghubungkan semua komputer yang tergabung dalam sebuah *bus* atau jalur komunikasi data (kabel). Komputer pada sebuah jaringan berkomunikasi dengan komputer lain berdasarkan sistem pengalamatan data dan meletakkan data melalui kabel dalam bentuk sinyal elektronik.



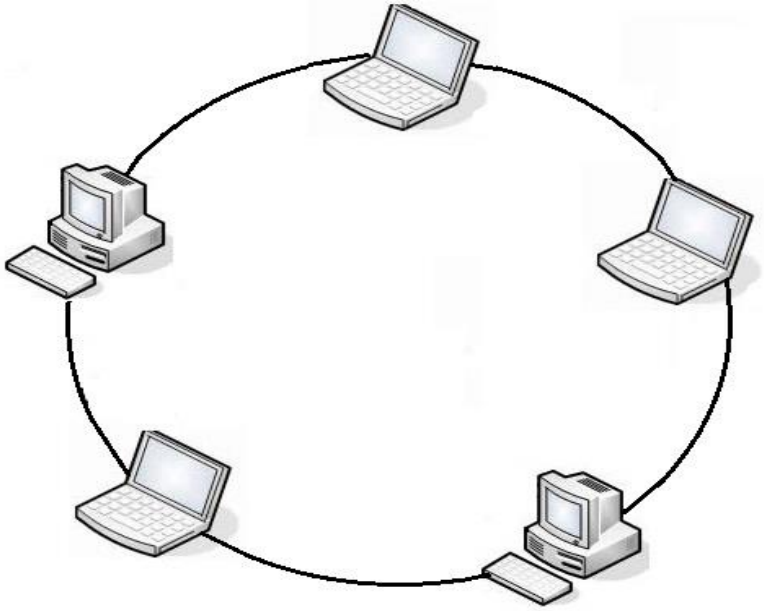
Gambar 2.3 Topologi BUS

Topologi jaringan ini memiliki kelemahan pada tingkat komunikasi data yang cukup padat. Pada kondisi lalu lintas data yang padat, kemungkinan terjadinya tabrakan komunikasi antara beberapa komputer menjadi sangat besar. Hal ini akan berakibat turunnya kecepatan lalu lintas data, yang pada akhirnya

akan menurunkan kinerja jaringan secara keseluruhan. Kelemahan lain dari topologi ini adalah jika terjadi gangguan atau kerusakan pada salah satu lokasi (titik) dalam jaringan maka hal itu akan mempengaruhi jaringan secara keseluruhan.

2.1.2.2. Topologi Ring

Topologi *ring* menghubungkan komputer dengan cara membentuk sebuah lingkaran kabel. Sinyal berjalan mengelilingi lingkaran dengan satu arah dan sinyal tersebut dilewatkan melalui masing-masing komputer. Tidak seperti topologi *bus*, masing-masing komputer bertindak seperti sebuah *repeater* untuk memperkuat sinyal dan mengirimkannya ke komputer lain. Karena sinyal dilewatkan melalui masing-masing komputer, kerusakan dari satu komputer dapat mempengaruhi seluruh jaringan. Namun demikian jaringan ini memiliki kecepatan yang lebih baik bila dibandingkan dengan topologi *bus*.



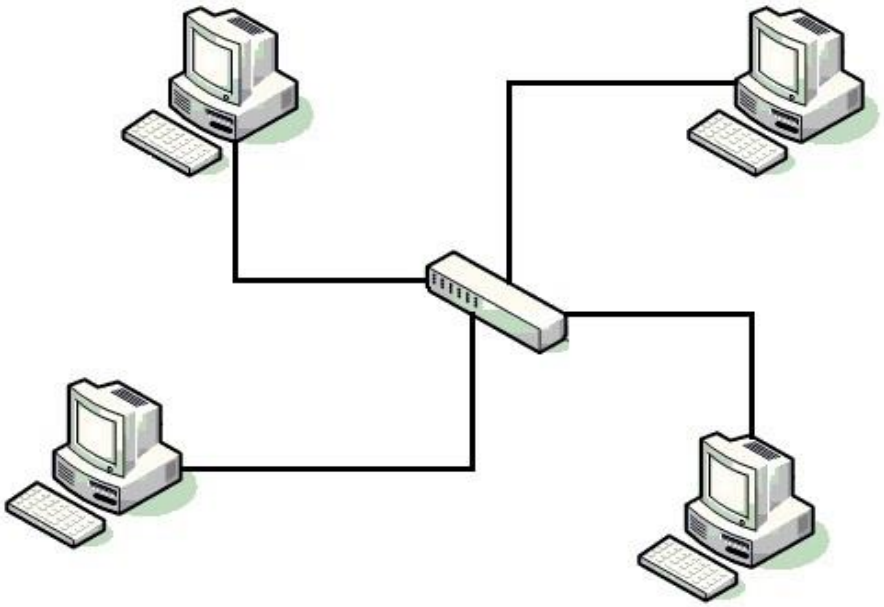
Gambar 2.4 Topologi Ring

2.1.2.3. Topologi Star

Topologi ini masing-masing komputer dalam jaringan dihubungkan ke pusat dengan menggunakan jalur yang berbeda. Komunikasi pada jaringan diatur di pusat jaringan. Dengan digunakannya jalur yang berbeda untuk masing-masing

komputer maka jika terjadi gangguan atau masalah pada salah satu titik dalam jaringan tidak akan mempengaruhi bagian jaringan yang lain. Hal ini juga memungkinkan pengaturan instalasi jaringan yang fleksibel.

Topologi jaringan seperti ini memungkinkan kecepatan komunikasi data yang lebih baik jika dibandingkan topologi yang lain (*bus* dan *ring*). Kelemahan dari topologi ini adalah bahwa kinerja jaringan sangat dipengaruhi oleh kemampuan sentral atau dari jaringan tesebut.



Gambar 2.5 Topologi Star

Sedangkan arsitektur jaringan komputer secara *logic* ada bermacam-macam, bahkan terus dikembangkan bentuk-bentuk jaringan baru. Beberapa bentuk arsitektur jaringan yang telah ada adalah:

1. Arsitektur ArcNet.
2. Arsitektur TokenRing.
3. Arsitektur Ethernet.
4. Arsitektur FDDI.
5. Arsitektur ATM, dan lain-lain

2.2. Internet

Internet, kependekan dari *interconnection-networking*, secara harfiah ialah sistem global dari seluruh jaringan komputer yang saling terhubung menggunakan standar *Internet Protocol Suite* (TCP/IP) untuk melayani miliaran pengguna di seluruh dunia. TCP/IP digunakan sebagai protokol pertukaran paket (*packet switching communication protocol*).

Internet merupakan jaringan komputer yang dibentuk oleh Departemen Pertahanan Amerika Serikat di tahun 1969, melalui proyek ARPA yang disebut ARPANET (*Advanced Research Project Agency Network*), dimana mereka mendemonstrasikan bagaimana dengan *hardware* dan *software* komputer yang berbasis UNIX dapat melakukan komunikasi dalam jarak yang tidak terhingga melalui saluran telepon.

Proyek ARPANET merancang bentuk jaringan, kehandalan, seberapa besar informasi dapat dipindahkan, dan akhirnya semua standar yang mereka tentukan menjadi cikal bakal pembangunan protokol baru yang sekarang dikenal sebagai TCP/IP (*Transmission Control Protocol/Internet Protocol*).

Tujuan awal dibangunnya proyek tersebut adalah untuk keperluan militer. Pada saat itu Departemen Pertahanan Amerika Serikat (*US Department of Defense*) membuat sistem jaringan komputer yang tersebar dengan menghubungkan komputer di daerah-daerah vital untuk mengatasi masalah bila terjadi serangan nuklir dan untuk menghindari terjadinya informasi terpusat, yang apabila terjadi perang dapat mudah dihancurkan.

Pada mulanya ARPANET hanya menghubungkan 4 situs saja yaitu *Stanford Research Institute, University of California, Santa Barbara, University of Utah*, dimana mereka membentuk satu jaringan terpadu di tahun 1969, dan secara umum ARPANET diperkenalkan pada bulan Oktober 1972. Tidak lama kemudian proyek ini berkembang pesat di seluruh daerah, dan semua universitas di negara tersebut ingin bergabung, sehingga membuat ARPANET kesulitan untuk mengaturnya.

Oleh sebab itu ARPANET dipecah manjadi dua, yaitu "MILNET" untuk keperluan militer dan "ARPANET" baru yang lebih kecil untuk keperluan

nonmiliter seperti, universitas-universitas. Gabungan kedua jaringan akhirnya dikenal dengan nama DARPA Internet, yang kemudian disederhanakan menjadi Internet.

2.3. Situs

Sebuah situs web sering pula disingkat *website*, *site* adalah sebutan bagi sekolompok halaman (*domain name*) atau *subdomain* di *World Wide Web* (WWW) di internet. WWW terdiri dari situs web di akses dari sebuah URL yang menjadi "akar" (*root*), yang disebut *homepage* (halaman induk sering diterjemahkan menjadi “beranda”, “halaman muka”), dan biasanya disimpan dalam *server* yang sama. Tidak semua situs web dapat diakses dengan gratis beberapa situs web melakukan pembayaran agar dapat menjadi pelanggan misalnya situs-situs berita, layanan surat elektronik (*e-mail*), dan lain-lain.

Terminologi dari *website* adalah kumpulan dari halaman-halaman situs, yang biasanya terangkum dalam sebuah *domain* atau *subdomain* di dalam *World Wide Web* (WWW) pada internet. Sebuah *web page* adalah sebuah dokumen yang tertulis dalam format HTML (*Hyper Text Markup Language*), yang hampir selalu bisa diakses melalui HTTP, yaitu protokol yang menyampaikan informasi dari *server website-website* tersebut dapat membentuk sebuah jaringan informasi yang sangat besar. Halaman-halaman dari *website* akan bisa diakses melalui URL yang biasa disebut *homepage*. URL ini mengatur halaman-halaman situs untuk menjadi sebuah hirarki, meskipun *hyperlink* yang ada di halaman tersebut mengatur para pembaca dan memberitahukan mereka susunan keseluruhan dan bagian arus informasi ini berjalan. Beberapa *website* membutuhkan subskripsi (data masukan) agar para *user* bisa mengakses situs tersebut.

Penemu *website* adalah Sir Timothy John “Tim” Berners-Lee, sedangkan *website* yang tersambung dengan jaringan, pertamakali muncul pada tahun1991. Maksud dari Tim ketika membuat *website* adalah untuk mempermudah tukar-menukar dan memperbaharui informasi kepada sesama peneliti di tempat dia bekerja. Pada tanggal 30 April 1993, CERN (tempat dimana Tim bekerja) menginformasikan bahwa WWW dapat digunakan gratis oleh semua orang.

Sebuah *website* bisa berupa hasil kerja dari perorangan atau individu, atau menunjukkan kepemilikan dari sebuah organisasi, perusahaan, dan biasanya *website* itu menunjukkan beberapa topik khusus, atau kepentingan tertentu. Sebuah *website* bisa berisi *hyperlink* yang menghubungkan ke *website* lain.

Website ditulis, atau secara dinamik di konversikan menjadi HTML dan diakses melalui sebuah program *software* yang bisa disebut dengan *web browser*, yang dikenal juga dengan HTTP *Client.* Halaman web dapat dilihat atau diakses melalui jaringan komputer dan internet, perangkatnya bisa saja berupa personal komputer, laptop komputer, *PDA* ataupun *Cellphone*. Sebuah *website* dibuat di dalam sebuah sistem komputer yang dikenal dengan *web server* juga disebut dengan HTTP *Server*, dan pengertian ini juga bisa menunjukkan pada *software* yang dipakai.

Sebuah *website* dinamis adalah *website* yang secara berkala informasi di dalamnya berubah atau *website* ini bisa berhubungan dengan *user* dengan berbagai macam cara atau metode (HTTP *cookies* atau Variabel *Database*, sejarah kunjungan, variabel sesi dan lain-lain) bisa juga dengan cara interaksi langsung menggunakan *form* dan pergerakan *mouse*. Ketika *web server* menerima permintaan dari *user* untuk memberikan halaman tertentu, maka halaman tersebut akan secara otomatis diambil dari media penyimpanan sebagai respon dari permintaan yang diminta oleh *user*.

Sebuah situs dapat menampilkan dialog yang sedang berlangsung di antara dua *user*, memantau perubahan situasi, atau menyediakan informasi yang berkaitan dengan sang *user*. Ada banyak jenis sistem *software* yang dapat dipakai untuk meng-*generate Dynamic Web System* dan Situs Dinamis, beberapa diantaranya adalah *ColdFusion* (CFM), *Active Server Pages* (ASP), *Java Server Pages* (JSP) dan PHP, bahasa program yang mampu untuk meng-*generate Dynamic Web System* dan Situs Dinamis. Situs juga bisa termasuk di dalamnya berisi informasi yang diambil dari satu atau lebih *database* atau bisa juga menggunakan teknologi berbasis XML, contohnya adalah RSS. Isi situs yang statis juga secara periodik di *generate* apabila ada keadaan dimana dia butuh

untuk dikembalikan kepada keadaan semula, maka dia akan di *generate*, hal ini untuk menghindari kinerjanya supaya tetap terjaga.

Seperti yang tertulis di atas, terdapat perbedaan dalam penulisan dari terminologi *website*. Walaupun ¨*Website*¨ sudah secara umum dipakai, namun untuk *Associated Press Stylebook, Reuters, Microsoft, Academia,* dan kamus-kamus yang ada, penulisan yang mereka pakai adalah dengan menggunakan dua kata, yaitu *Web Site*. Hal ini karena *Web* bukanlah terminologi umum, namun ia adalah kependekan dari *World Wide Web.*

2.4. HTTP Respon

1. Kode Respon

   Kode Respon adalah sebuah kode angka yang sesuai dengan respon yang bersangkutan. *Server* merespon dalam salah satu dari empat cara yaitu: *Success, Redirection, Client Error,* atau *Server Error* [10].

Tabel 2.1 Kode-kode Respon Umum

Kode Respon	Keterangan
Success 2xx	
200 OK	*Request* Berhasil dijalankan
Redirection 3xx	
301 *Moved Permanently*	Sumber-sumber *request* diberi URL permanen dan ditempatkan pada *field location*. Kode respon ini berkata, “saya telah dipindahkan, ikuti saya ke tempat yang baru.”

302 *Moved Permanently*	Sumber-sumber request telah diberi URL temporer yang baru dan ditempatkan pada *field* location. Kode respon ini berkata, "saya telah dipindahkan, ikuti saya ke tempat sementaraku yang baru, tetapi jangan bergantung pada saya jika ingin di sini dalam waktu lama."
Client Error 4xx	
400 *Bad Request*	*Request* tidak dimengerti oleh *server*.
401 *Unauthorized*	Sumber-sumber yang diminta memerlukan pembuktian keaslian *user*, biasanya dalam bentuk pembuktian keaslian *Basic* atau yang sejenis.
403 *Forbidden*	*Server* memahami *request* tetapi menolak untuk merespon. Khususnya sewaktu metode GET digunakan untuk menerima respon ini, sedikit atau tak ada informasi lebih jauh lagi. Tetapi, sewaktu metode HEAD dipakai, beberapa *server* akan memberi informasi rinci lebih lengkap mengenai mengapa kondisi seperti ini muncul.
404 *Not Found*	Sumber-sumber yang diminta tak ditemukan.
Server Errors 5xx	
500 *Internal Server Error*	*Server* menemukan kesalahan internal (*internal error*) dalam memproses *request*.

501 *Not Implemented*	*Server* tidak mendukung *request* yang bersangkutan.
502 *Bad Gateway*	*Server* menerima respon cacat dari *server* utama sewaktu ia mengirim *resource* yang diminta. Respon ini khusus untuk *proxy-proxy* HTTP.
503 *Service Unavailable*	*Server* tidak mampu merespon *request* karena *request* sedang meluap.

2. Field Header

   Field Header merupakan informasi tambahan mengenai respon yang berisi *field-field header* yang mengandung beragam tingkatan informasi tambahan.

Tabel 2.2 Definisi *Field Header*

Field Header	Keterangan
Allow	Menampilkan metode-metode yang didukung oleh sumber-sumber yang diminta.
Authorization	Menampilkan daftar mandat-mandat otorisasi untuk pembuktian keaslian HTTP.
Content-Encoding	Menampilkan *encoding* isi tambahan yang sedang dijalankan pada pengembalian data. Dengan informasi ini klien mengetahui secara lebih baik bagaimana menginterpretasikan data yang dikembalikan. Misalnya, *Content-Encoding: x-gzip* berarti isi *gzip* terkompresi.

Content-Length	Menampilkan daftar ukuran body isi dalam angka decimal basis delapan. Misalnya: *Content-Length*: 332.
Content-Type	Menampilkan tipe isi dalam respon. Misalnya *Content-Type*: *text/html* menampilkan *text/html* sebagai tipe isi. *Field* ini membantu klien dalam memahami secara lebih baik bagaimana menampilkan isi dalam *browser*.
Date	Menampilkan daftar tanggal dan jam *server*.
Expires	Menampilkan daftar tanggal dan jam yang menandakan isi telah *out of date*.
From	Menampilkan daftar alamat *e-mail* yang dipakai untuk mengidentifikasi bagian respon isi. *Field* ini jarang digunakan.
Last-Modified	Menampilkan daftar tanggal dan jam yang diyakini *server* sebagai saat terakhir sumber-sumber yang diminta dimodifikasi.
Location	Menampilkan daftar lokasi sumber-sumber yang diminta.
Pragma	Menggambarkan perilaku tambahan untuk request. Misalnya, bila *field header* Pragma dikirim dari *server* dengan perintah "*no-cache*", klien harus me-*load* isi yang dikirim apakah ia sudah meng-*cache* salinannya. Contoh: *Date*: Thu, 22 Apr 2002 01:10:22 GMT.

Referrer	Mengizinkan klien untuk menentukan alamat sumber.
Server	Daftar *software* yang berjalan pada *server*. Dalam banyak hal, informasi ini akurat. Misalnya, *Server*: Microsoft-IIS/5.0. Tetapi, *administrator* yang pintar bisa mengubah informasi ini sesukanya, misalnya "Web Server Saya."
User-Agent	Daftar informasi tambahan mengenai *user agent* (klien) yang meminta informasi. Misalnya, *User-Agent*: Mozilla/5.0 (WinNT).
WWW-Authenticate	Digunakan untuk merespon kode respon 401 *Unauthorized*. *Field* ini menjaga tak masuknya akses ilegal.

3. Data

   Data merupakan isi atau badan (*body*) dari respon. Permintaan *line*, seperti GET / HTTP/1.1 logo.gif atau *Status line*, seperti HTTP/1.1 200 OK. Permintaan / baris status dan *header* semua harus diakhiri dengan <CR> <LF> yaitu, *carriage return* diikuti oleh *line feed.*

   Sistem keamanan komputer digunakan untuk menjamin agar sumber daya tidak digunakan atau dimodifikasi orang yang tidak diotorisasi. Kemanan sistem terbagi menjadi tiga macam, yaitu [11]:

   1. Keamanan eksternal adalah pengamanan yang berkaitan dengan pengamanan fasilitas komputer dari penyusup dan bencana seperti kebakaran /kebanjiran.

2. Keamanan *interface* pemakai, berkaitan dengan identifikasi pemakai sebelum pemakai diijinkan mengakses program dan data yang disimpan.
3. Keamanan internal, berkaitan dengan pengamanan beragam kendali yang dibangun pada perangkat keras dan sistem operasi yang menjamin operasi yang handal dan tak terkorupsi untuk menjaga integritas program dan data.

Sehubungan dengan keamanan ini terdapat dua masalah yang penting, yaitu:

1. Kehilangan data, yang dapat disebabkan antara lain oleh:
   a. Bencana, seperti kebakaran, banjir, gempa bumi, perang, maupun kerusuhan.
   b. Kesalahan perangkat keras dan perangkat lunak yang disebabkan oleh tidak berfungsinya pemroses, *disk* yang tidak terbaca, kesalahan telekomunikasi, dan kesalahan program (*bugs*).
   c. Kesalahan manusia, seperti salah dalam memasukkan data, salah memasang *disk*, eksekusi program yang salah, dan kehilangan *disk*.
2. Penyusup (*intruder*) yang terdiri dari:
   a. Penyusup pasif, yaitu membaca data yang tidak diotorisasi.
   b. Penyusup aktif, yaitu mengubah data yang tidak diotorisasi.

Untuk membangun sebuah sistem yang aman dari serangan para *intruder*, adalah penting untuk mengetahui jenis-jenis atau kategori dari *intruder*. Empat contoh kategori:

1. Keingintahuan seorang akan hal-hal pribadi orang lain.

   Banyak orang yang mempunyai PC yang terhubung ke jaringan. Beberapa orang dalam jaringan tersebut suka membaca *e-mail* dan *file* orang lain jika di dalam jaringan tersebut tidak ditempatkan sistem penghalang.
2. Penyusupan oleh orang-orang dalam.

   Pelajar, programmer, operator, dan personil teknis menganggap bahwa mematahkan sistem keamanan komputer lokal merupakan suatu

tantangan. Mereka biasanya sangat ahli dan bersedia mengorbankan banyak waktu untuk melakukan hal tersebut.

3. Keinginan untuk mendapatkan uang.

   Beberapa pemrogram bank mencoba mencuri uang dari bank tempat mereka bekerja dengan mengubah *software* sehingga akan memotong bunga daripada membulatkannya, menyimpan uang kecil untuk mereka sendiri, menarik uang dari akun yang sudah tidak digunakan selama bertahun-tahun, ataupun memeras.

4. E-spionase komersial atau militer.

   E-spionase adalah usaha serius yang diberi dana besar oleh pesaing atau pihak musuh untuk mencuri program, rahasia dagang, ide-ide paten, teknologi, rencana bisnis, dan sebagainya. Seringkali usaha ini melibatkan *wiretapping*, dimana antena diarahkan ke suatu komputer untuk menangkap radiasi elektromagnetis yang memancar dari komputer itu.

Terlihat jelas bahwa jumlah usaha yang berhubungan dengan keamanan dan proteksi tergantung pada siapa musuhnya. Secara garis besar pengamanan sistem komputer mencakup empat hal yang sangat mendasar, yaitu:

1. Pengamanan Fisik

   Pengamanan fisik dapat dilakukan dengan menempatkan sistem komputer pada tempat yang mudah diawasi dan dikontrol untuk mengantisipasi kelalaian/keteledoran dari para *user* yang sering meninggalkan terminal komputer dalam keadaan *logon*. Hal ini memungkinkan pihak lain untuk dapat mengakses beberapa fasilitas sistem komputer bahkan mereka dapat melakukan tindakan perusakan terhadap sistem komputer tersebut.

2. Pengamanan Akses

   Pengamanan akses biasanya menjadi tanggung jawab para administrator sistem. Dalam hal ini seorang *administrator* harus mampu mengontrol dan mendokumentasikan seluruh akses ke sistem komputer dengan baik

sehingga dapat mempercepat dan mempermudah pencarian penyebab masalah dan solusinya bila suatu saat terjadi sesuatu dalam sistem komputer tersebut.

3. Pengamanan Data

   Pengamanan data dilakukan dengan menerapkan sistem tingkatan akses dimana seseorang hanya dapat mengakses data tertentu saja yang menjadi haknya.

4. Pengamanan Jaringan

   Pengamanan jaringan dapat dilakukan dengan menggunakan kriptografi dimana data yang sifatnya *sensitive* dienskripsi terlebih dahulu sebelum ditransmisikan melalui internet.

2.4.1. Ancaman Keamanan

Tujuan dari pengamanan adalah menghindari, mencegah, dan mengatasi ancaman-ancaman terhadap sistem. Kebutuhan akan pengamanan komputer dapat dikategorikan dalam tiga aspek, yaitu:

1. Kerahasiaan, dimana informasi pada sistem komputer itu terjamin kerahasiaannya, hanya dapat diakases oleh pihak-pihak yang diotorisasi, keutuhan serta konsistensi data pada sistem tersebut tetap terjaga.
2. Integritas, dimana sumber daya sistem terjamin hanya dapat dimodifikasi oleh pihak-pihak yang diotorisasi.
3. Ketersediaan, adalah sumber daya sistem komputer terjamin akan tersedia bagi pihak-pihak yang diotorisasi pada saat diperlukan.

2.4.2. Tipe-tipe Ancaman Keamanan

Tipe ancaman terhadap keamanan sistem komputer dapat dimodelkan dengan memandang fungsi sistem komputer sebagai penyedia informasi. Berdasarkan fungsi ini, ancaman terhadap sistem komputer dikategorikan menjadi 4 ancaman, 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.

2. *Modification,*merupakan ancaman terhadap integritas. Orang yang tidak berhak berhasil menyadap lalu-lintas informasi yang sedang dikirim lalu mengubahnya sesuai keinginan orang itu.
3. *Fabrication,* merupakan ancaman terhadap integritas. Orang yang tidak berhak berhasil meniru atau memalsukan suatu informasi sehingga orang yang menerima informasi tersebut menyangka bahwa informasinya berasal dari orang yang dikehendaki oleh si penerima informasi tersebut.

2.4.3. Tipe Ancaman Pada Aplikasi Web

Berikut ini adalah berbagai jenis model serangan yang terjadi pada aplikasi web [12]:

1. *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 contoh lainnya 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.
2. *Cross Site Scripting*

   Cross Site Scripting (XSS) adalah suatu serangan dengan menggunakan 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 serverdalam memvalidasi *input* yang diberikan oleh pengguna. Hal ini memungkinkan halaman yang dihasilkan menyertakan perintah yang sebenarnya tidak diperbolehkan.

3. *Cross Site Request Forgery* (CSRF)

   Cross Site Request Forgery adalah salah satu *vulnerability* pada *Web Application* yang bekerja dengan cara mengeksploitasi suatu task dari sebuah Web dengan memanfaatkan *Autentikasi* yang dimiliki oleh korban. CSRF juga dikenal dengan sebutan "*one link*" *attack*, karena pada implementasinya, sang attacker hanya butuh meng-inject-kan suatu link yang berisi suatu *web task* URL pada halaman tertentu untuk dibuka oleh calon korban, agar ketika si korban membuka halaman tersebut, secara otomatis si korban akan mengeksekusi link URL yang telah di*inject*-kan sang attacker sebelumnya. *Web Application* yang *vulnerable* terhadap CSRF adalah *Web Application* yang memiliki suatu URL *Request* (baik dengan menggunakan method GET maupun POST) yang statis [13].

4. *Web Defacement*

   Serangan dengan tujuan utama merubah tampilan sebuah *website* baik halaman utama maupun halaman lain terkait dengannya diistilahkan sebagai "*Web Defacement*". Hal ini biasa dilakukan oleh para "*attacker*" atau penyerang karena merasa tidak puas atau tidak suka kepada individu, kelompok, atau entitas tertentu sehingga *website* yang terkait dengannya menjadi sasaran utama. Pada dasarnya *deface* dapat dibagi menjadi dua jenis berdasarkan dampak pada halaman situs yang terkena serangan terkait.

   Jenis pertama adalah suatu serangan dimana penyerang mengubah satu halaman penuh tampilan depan. Untuk melakukan hal tersebut biasanya seorang '*defacer*' harus berhubungan secara 'langsung' dengan mesin komputer terkait. Hal ini hanya dapat dilakukan apabila yang bersangkutan sanggup mendapatkan hak akses penuh terhadap mesin, baik itu "*root*

account" atau sebagainya yang memungkinkan *defacer* dapat secara interaktif mengendalikan seluruh direktori terkait. Hal ini umumnya dimungkinkan terjadi dengan memanfaatkan kelemahan pada sejumlah "*services*" yang berjalan di sistem komputer. Jenis kedua adalah suatu serangan dimana penyerang hanya mengubah sebagian atau hanya menambahi halaman yang di-*deface*. Artinya yang bersangkutan men-*deface* suatu situs tidak secara penuh, bisa hanya dengan menampilkan beberapa kata, gambar atau penambahan "*script*" yang mengganggu. Dampaknya biasanya adalah menghasilkan tampilan yang kacau atau mengganggu.

5. *Phising*

   Phishing merupakan sebuah proses "pra-serangan" atau kerap dikatakan sebagai "*soft attack*" dimana sang penyerang berusaha mendapatkan informasi rahasia dari target dengan cara menyamar menjadi pihak yang dapat dipercaya – atau seolah-olah merupakan pihak yang sesungguhnya. Contohnya adalah sebuah *email* yang berisi suatu informasi yang mengatakan bahwa sang pengirim adalah dari Divisi Teknologi Informasi yang sedang melakukan "*upgrading*" sistem, dimana untuk memperlancar tugasnya, sang penerima *email* diminta untuk segera mengirimkan kata kunci "*password*" dari "*user name*" yang dimilikinya. Atau situs sebuah bank palsu yang memiliki tampilan sama persis dengan situs aslinya namun memiliki alamat URL yang miripmirip, sehingga diharapkan sang nasabah akan khilaf dan secara tidak sadar memasukkan kata kunci rahasianya untuk mengakses rekening yang dimaksud.

   Serangan "*phishing*" ini kerap dikategorikan sebagai sebuah usaha "*social engineering*", yaitu memanfaatkan pendekatan sosial dalam usahanya untuk mendapatkan informasi rahasia sebagai alat untuk melakukan penyerangan di kemudian hari. Modus operandi yang paling banyak ditemui saat ini adalah usaha phishing melalui SMS pada telepon genggam, dimana sudah banyak korban yang harus kehilangan uangnya karena diminta untuk melakukan transfer ke rekening tertentu dengan

berbagai alasan yang seolah-olah masuk akal sehingga berhasil menjebak sang korban.

2.4.4. Prinsip Pengamanan Sistem

1. *User Authentification* [14].

   Identifikasi pemakai saat *login* merupakan dasar asumsi sistem proteksi sehingga metode otentifikasi didasarkan pada tiga cara, yaitu sesuatu yang diketahui pemakai, yang dimiliki pemakai, dan mengenai pemakai.

2. Proteksi *Password*

   Password merupakan salah satu otentifikasi yang diketahui pemakai. Teknik pengamanan dengan *password* mempunyai beberapa kelemahan, terutama karena pemakai sering memilih *password* yang mudah diingatnya. Upaya untuk mengamankan proteksi *password* tersebut antara lain:

   a. *Salting*, menambahkan *string* pendek ke *string password* yang diberikan pemakai sehingga mencapai panjang *password* tertentu.
   b. *One-time Password*, *user* harus mengganti *password* secara teratur.
   c. Satu daftar pertanyaan dan jawaban yang panjang, sehingga pada saat *login*, komputer memilih salah satu dari pertanyaan secara acak, menanyakan ke pemakai dan memeriksa jawaban yang diberikan.
   d. Tanggapan-tanggapan / *chalenge respone*, pemakai diberikan kebebasan memilih suatu algoritma.

3. Identifikasi Fisik

   Pendekatan identifikasi fisik ini dilakukan dengan memeriksa apa yang dimiliki pemakai:

   a. Kartu berpita magnetik

      Kartu pengenal dengan selarik pita magnetik umumnya dikombinasikan dengan *password*. *User* akan dapat *login* ke komputer bila memenuhi syarat, yaitu mempunyai kartu dan mengetahui password khusus untuk kartu tersebut.

   b. Sidik fisik

Identifikasi fisik sidik jari atau sidik suara, analisis panjang jari, dan sebagainya.

c. Analisis tanda tangan

Dengan menggunakan pena khusus, pemakai diharuskan untuk membuat tanda tangan. Dalam hal ini yang dibandingkan adalah arah gerakan dan tekanan pena saat *user* membuat tanda tangan.

2.5. SQL Injection

SQL Injection adalah metode ancaman yang mengizinkan *client* untuk mengeksekusi database melalui URL dan mendapatkan akses untuk memperoleh informasi penting. Mekanisme ancaman dilakukan dengan memanfaatkan kesalahan pada kode program yang tidak di*filter*, sehingga menyebabkan terjadinya eksploitasi pada database [4][15].

2.5.1. Penyebab SQL Injection

Kerentanan SQL Injection paling sering terjadi ketika pengembang aplikasi web tidak menjamin bahwa nilai-nilai yang diterima dari *web form*, *cookie*, *parameter input*, dan sebagainya divalidasi atau dikodekan sebelum di *submit* untuk *query SQL* yang akan dieksekusi pada *server database*. Seorang penyerang dapat mengontrol masukan yang dikirimkan ke *query SQL* dan memanipulasi *input*, sehingga data tersebut ditafsirkan sebagai kode bukan sebagai data, dan mungkin dapat mengeksekusi kode pada *database*. Tanpa pemahaman yang baik dari *database* dan kesadaran akan isu-isu keamanan potensial dari kode yang sedang dikembangkan, pengembang aplikasi sering bisa menghasilkan aplikasi tidak aman yang rentan terhadap *SQL Injection*. Berikut ini adalah tipe-tipe kesalahan dalam *SQL Injection* [15]:

a. *Incorrectly Handled Escape Characters.*

Database SQL mengartikan karakter *quote* (') sebagai batas antara kode dan data. Ia diasumsikan bahwa apapun yang mengikuti karakter *quote*, maka itu akan dianggap kode yang kemudian dijalankan. Oleh karena itu, *attacker* dengan sangat mudah untuk mengetahui apakah web tersebut mempunyai celah, dengan menyisipkan karakter *quote* (') pada URL.

b. *Incorrectly Handled Types.*

Karakter *quote* (‘) diinterpretasikan sebagai *string* pemisah dan digunakan sebagai batas antara kode dan data. Ketika berhadapan dengan data *numeric*, maka data tersebut tidak perlu di-enkapsulasi dengan tanda kutip. Apabila data tersebut di-enkapsulasi , maka data *numeric* akan dianggap sebagai *string*.

c. *Incorrectly Handled Query Assembly.*

Terdapat metode *Request* pada *query* sehingga menyebabkan *attacker* dapat memanipulasi permintaan melalui HTTP dengan memisahkan nilai-nilai sebagai parameter inputan.

d. *Incorrectly Handled Errors.*

Masalah ini terjadi ketika pesan kesalahan internal seperti *database* ditampilkan di *browser*. Pesan kesalahan tersebut memberikan informasi bagi *attacker* tentang sistem yang komponen yang terdapat pada sistem.

e. *Incorrectly Handled Multiple Submissions.*

Masalah terjadi pada *web developer* yang cenderung meremehkan pengguna dalam merancang aplikasi. Misalnya, *web developer* berharap bahwa pengguna harus melewati proses 1 untuk mencapai proses 2, kemudian harus melewati proses 2 untuk mencapai proses 3. Namun demikian halnya tidak sedikit pengguna dapat mencapai proses 3 tanpa melewati proses 2.

f. *Insecure Database Configuration.*

Ketika *database* yang baru di *install*, biasanya mengikutsertakan *user* dan *password default*. Apabila *user* dan *password default* tersebut tidak segera diganti, maka sistem dapat diambil alih oleh orang lain.

2.5.2. Pencegahan SQL Injection

Langkah yang dapat ditempuh untuk mengurangi penyusupan ke halaman web dengan *SQL Injection* dengan cara [4][15]:

a. Mem-*filter* dengan tidak membolehkan karakter seperti single *quote, double quote, slash, back slash, semi colon, extended character,* seperti NULL, *carry return, new line, etc,* dalam *string form*:
   − Masukan dari *from users*
   − Parameters di URL
   − Nilai dari *cookie*
b. Untuk nilai *numeric, convert* dulu sebelum melewati statement SQL dengan mengunakan ISNUMERIC untuk meyakinkan itu adalah *integer*.
c. Mengubah "*Startup and run SQL Server*" menggunakan *low privilege user* dalam *SQL Server Security tab*.
d. Ubah *stored procedure* yang tidak terpakai, seperti: *master..Xp_cmdshell, xp_startmail, xp_sendmail, sp_makewebtask.*

2.6. Cross Site Scripting

Cross Site Scripting adalah metode ancaman yang memaksa situs web untuk menampilkan kode berbahaya, yang dijalankan pada *browser* web pengguna. Mekanisme ancaman dilakukan dengan memanfaatkan kesalahan pada kode program yang tidak di-*filter*. Kode tersebut akan dimasukkan ke dalam *webform*, di antaranya *form* buku tamu atau *form* pencarian.[4][16].

2.6.1. Penyebab Cross Site Scripting

Cross Site Scripting terjadi karena tidak adanya penangan nilai masukan pada variabel sehingga masukan tersebut dieksekusi begitu saja sesuai nilai yang ada. Apabila variabel tersebut tidak di filter, maka semua nilai masukan akan dieksekusi [4][16].

2.6.2. Pencegahan Cross Site Scripting

Langkah yang dapat ditempuh untuk mencegah ancaman *Cross Site Scripting* adalah [4][16]:

a. Teknologi *Static Web Pages*

   Cara terbaik dan efektif untuk menghindari ancaman *Cross Site Scripting* adalah menghindari penggunaan teknologi *dynamic web page*. Halaman

yang statis tentu saja memberikan kontrol yang lebih di sisi server dibandingkan dengan halaman web yang dinamis. Halaman web yang dihasilkan secara statis akan memberikan kelakukan yang lebih pasti dibandingkan halaman web yang dihasilkan secara dinamis. Konsekuensi yang ditanggung adalah penyedia layanan harus merelakan sifat interaktif yang mungkin diinginkan.

b. Metode *POST*

Metode *POST* adalah metode pengirimana data dimana variabel yang dikirimkan tidak disertakan pada link yang digunakan. Metode *POST* menyembunyikan variabel yang dikirimkan dari pengguna. Metode ini menjamin kode tidak dapat diinjeksikan melalui link yang telah didesain oleh penyerang. *Link* merupakan satu satunya cara yang dapat digunakan oleh penyerang untuk mengeksploitasi *Cross Site Scripting*.

c. Pengkodean Karakter Spesial pada *Link*

Untuk me-*non*aktifkan *script* yang diinjeksikan, memerlukan pembuatan aplikasi yang mampu mengkodekan karakter tersebut, sehingga karakter tersebut tidak dapat dimengerti oleh *browser* yang digunakan.

d. Hilangkan kemampuan scripting

Cross Site Scripting disebabkan oleh keberhasilan penyerang menginjeksi kode pada halaman web yang dihasilkan. Jika kode yang diinjeksikan tersebut tidak dapat diinterpretasikan, halaman web dapat ditampilkan dengan aman.

e. *HTTP-Only Cookie*

Metode ini membatasi akses yang dapat dilakukan terhadap *cookie*. Dengan menggunakan *HTTP-only cookie*, browser pengguna masih dapat menerima *cookie* yang dikirimkan oleh penyedia layanan. Namun *cookie* tidak dapat diakses melalui *script* yang dieksekusi pada *browser* pengguna. Jadi *script* yang diinjeksikan kepada *browser* pengguna tidak akan dapat melakukan transfer *cookie* yang ada.

2.7. Unrestricted File Upload

Unrestricted File Upload adalah jenis ancaman yang mengizinkan *user* untuk menyisipkan *file,* biasanya *file* yang di-*upload* telah disediakan oleh *attacker* dan digunakan untuk mengeksekusi kode jarak jauh (*Remote Code Execution*) [5]. Berdasarkan data dari *The Open Web Application Security Project* (OWASP), yang melakukan *survey* mengenai ancaman *file upload* yang sering digunakan oleh *attacker* yaitu *file* dengan ekstensi *".PHP"* [5].

2.7.1. Penyebab Unrestricted File Upload

Sebuah serangan *Unrestricted File Upload* terjadi didasarkan pada kesalahan atau ketidaksengajaan pendeklarasian variabel-variabel. Variabel yang tidak ada penanganan kesalahan dengan baik dapat menjadi penyebab serangan ini. Selain itu, serangan ini dapat terjadi karena web aplikasi menerima dan mengizinkan *attacker* untuk meng-*upload file* tanpa mem-*filter* ekstensi *file* tersebut.

2.7.2. Pencegahan Unrestricted File Upload

Langkah-langkah yang dapat dilakukan untuk mencegah *Unrestricted File Upload* yaitu dengan melakukan dua pendekatan diantaranya:

1. *Blacklist File Extention*

   Blacklist File Extention adalah metode yang menolak masukan yang berbahaya bagi sistem. Pada umumnya metode ini melibatkan masukan yang mengandung karakter, atau pola yang dianggap bisa membahayakan sistem.

2. *Whitelist File Extention*

   Whitelist File Extention adalah metode yang hanya menerima masukan yang tidak berbahaya bagi sistem. Metode ini bisa melibatkan panjang, ukuran, maupun tipe data sebelum diterima oleh sistem.

2.8. Web Application Firewall

Web Application Firewall (WAF) adalah Suatu metode untuk pengamanan pada aplikasi web, yang berupaya mencegah adanya ancaman dari *attacker* [7].

Web Application Firewall dapat bekerja dengan terlebih dahulu melakukan konfigurasi tambahan pada web *server* dan tidak perlu melakukan perubahan pada *script* pembangun aplikasi, sehingga dapat diterapkan pada aplikasi yang sudah berjalan. Seperti *firewall* pada umumnya yaitu melakukan *filter* data masuk dan keluar dan dapat untuk menghentikan *traffic* yang dianggap berbahaya sesuai dengan *rule* yang ditetapkan.

Web Application Firewall memiliki beberapa fungsi, mulai dari monitoring trafik, *secure directory*, pemfilteran *string* dan proteksi terhadap serangan seperti *SQL Injections*, *Cross-Site Scripting*, dan *Unrestricted File Upload*. *Web Application Firewall* membentuk lapisan keamanan yang dapat mendeteksi dan mencegah serangan pada aplikasi web. Adapun tindakan yang dapat dilakukan seperti menghentikan *request* dengan status 403 *forbidden* dan juga dapat melakukan *virtual patching.* Dimana *virtual patching* merupakan suatu *rule* yang diterapkan untuk melakukan *patch* tanpa menyentuh aplikasi guna memblokir *request* yang berbahaya [8].

2.9. Basis Data

Data adalah fakta mengenai objek, orang, dan lain-lain. Sedangkan Informasi adalah hasil analisis dan sintesis terhadap data. Basis data (*database*) adalah kumpulan informasi yang disimpan di dalam komputer secara sistematik sehingga dapat diperiksa menggunakan suatu program komputer untuk memperoleh informasi dari basis data tersebut [17].

2.9.1. Model Data

Model Data adalah kumpulan konsep yang terintegrasi yang menggambarkan data, hubungan antara data dan batasan–batasan data dalam suatu organisasi. Fungsi dari sebuah model data untk merepresentasikan data sehingga data tersebut mudah dipahami. Beberapa manajemen basis data didasarkan pada model data relasional, model data hirarkis, atau model data jaringan.

Untuk menggambarkan data pada tingkat eksternal dan konseptual digunakan model data berbasis objek atau model data berbasis *record*.

1. Model Data Berbasis Objek

Model data berbasis objek menggunakan konsep entitas, atribut dan hubungan antar entitas. Beberapa jenis model data berbasis objek yang umum adalah :

a. Entity-relationship
b. Semantic
c. Functional
d. Object-oriented

2. Model Data Berbasis Record

Pada model data berbasis *record*, basis data terdiri dari sejumlah record dalam bentuk yang tetap yang dapat dibedakan dari bentuknya. Ada 3 macam jenis model data berbasis *record* yaitu :

a. Model data relasional (*relational*)
b. Model data hierarkhi (*hierarchical*)
c. Model data jaringan (*network*)

2.9.1.1. Model Data Hirarkis

Model hierarkis merupakan model data yang dimana data tersebut diatur dengan struktur data *tree*. Struktur ini dapat mewakili informasi menggunakan hubungan *child/parent*: setiap *parent* dapat memiliki banyak *child*, tetapi setiap *child* hanya boleh memiliki satu *parent* (yang dikenal juga dengan hubungan 1-ke-banyak). Seluruh atribut dari record yang ditentukan telah diatur dengan tipe entitas.

2.9.1.2. Model Data Jaringan

Model jaringan distandarisasi pada tahun 1971 oleh *Data Base Task Group* (DBTG). Itulah sebabnya disebut model DBTG. Model ini juga disebut model CODASYL (*Conference on Data System Languages*), karena DBTG adalah bagian dari CODASYL. Model ini menyerupai model hirarkis, dengan perbedaan suatu simpul anak bisa memilki lebih dari satu orang tua. Oleh karena sifatnya demikian, model ini bisa menyatakan hubungan 1:1 (satu arang tua punya satu anak), 1:M (satu orang tua punya banyak anak), maupun N:M (beberapa anak

bisa mempunyai beberapa orangtua). Pada model jaringan, orang tua diseut pemilik dan anak disebut anggota.

2.9.1.3. Model data Relasional

Model relasional adalah model data yang paling banyak digunakan saat ini. Pembahasan pokok pada model ini adalah relasi, yang dimisalkan sebagai himpunan dari record. Deskripsi data dalam istilah model data disebut skema. Pada model relasional, skema untuk relasi ditentukan oleh nama, nama dari tiap *field* (atau atribut atau kolom), dan tipe dari tiap *field*. Basis data relasional memiliki fungsi-fungsi kegunaan sebagai berikut:

1. Mengatur penyimpanan data
2. Mengontrol akses terhadap data
3. Mendukung proses menampilkan dan memanipulasi data

Beberapa istilah yang perlu kita pahami mengenai basis data relasional antara lain:

a. Tabel : Merupakan struktur penyimpanan dasar dari basis data relasional, terdiri dari satu atau lebih kolom (column) dan nol atau lebih baris (row).
b. *Row* (baris) : Baris merupakan kombinasi dari nilai-nilai kolom dalam tabel; sebagai contoh, informasi tentang suatu departemen pada tabel Departmen. Baris seringkali disebut dengan “record”.
c. *Column* (kolom) : Kolom menggambarkan jenis data pada tabel; sebagai contoh, nama departemen dalam tabel Departmen. Kolom di definisikan dengan nama kolom dan tipe data beserta panjang data tertentu.
d. *Field* : *Field* merupakan pertemuan antara baris dan kolom. Sebuah *field* dapat berisi data. Jika pada suatu *field* tidak terdapat data, maka *field* tersebut dikatakan memiliki nilai “*null*”.
e. *Primary key* : kunci utama merupakan kolom atau kumpulan kolom yang secara unik membedakan antara baris yang satu dengan lainnya; sebagai contoh adalah kode departemen. Kolom dengan kategori ini tidak boleh mengandung nilai “*null*”, dan nilainya harus *unique* (berbeda antara baris satu dengan lainnya).

f. *Foreign key* : kunci tamu merupakan kolom atau kumpulan kolom yang mengacu ke *primary key* pada tabel yang sama atau tabel lain. *Foreign key* ini dibuat untuk memaksakan aturan-aturan relasi pada basis data. Nilai data dari *foreign key* harus sesuai dengan nilai data pada kolom dari tabel yang diacunya atau bernilai "*null*".

2.9.2. Struktur File Database

Struktur *file* merupakan suatu kumpulan dari data-data yang saling terkait dan berhubungan satu dengan yang lain:

1. Data adalah satu satuan informasi yang akan diolah, dimana sebelum diolah dikumpulkan di dalam suatu *file database*. Pengumpulan data dilakukan secara sistematis menurut struktur *file database* tersebut.
2. *Record* adalah data yang isinya merupakan satu kesatuan seperti Nama, Alamat, Nomor Telepon. Setiap keterangan yang mencakup Nama, Alamat dan Nomor Telepon dinamakan satu *record*, dan setiap *record* diberi nomor urut yang disebut nomor *record* (*Record Number*). Ukuran suatu *file database* ditentukan oleh jumlah *record* yang tersimpan di dalamnya.
3. *Field* adalah sub bagian dari *Record*. Dari contoh isi *record* diatas maka terdiri dari 3 *field*, yaitu *field* Nama, *field* Alamat dan *field* Nomor Telepon.

2.9.3. Database Management System (DBMS)

Menurut Date, Sistem Basis Data adalah sistem terkomputerisasi yang tujuan utamanya adalah memelihara informasi dan membuat informasi tersebut tersedia saat dibutuhkan. Manajemen Sistem Basis Data (*Database Management System* DBMS) adalah perangkat lunak yang didesain untuk membantu dalam hal pemeliharaan dan utilitas kumpulan data dalam jumlah besar. DBMS dapat menjadi alternatif penggunaan secara khusus untuk aplikasi, semisal penyimpanan data dalam *field* dan menulis kode aplikasi yang spesifik untuk pengaturannya [18].

2.9.3.1. Komponen DBMS

DBMS umumnya mengandung komponen-komponen berikut:

1. Hardware, meliputi:
   a. PC sampai dengan jaringan komputer.
   b. Tempat penyimpanan secondary (manegtic disk), I/O device (i.e : disk drives), device Controller, I/O Channels, dan lainnya.
   c. Hardware processor dan main memory, digunakan untuk mendukung saat eksekusi system software database.
2. Software

   DBMS, *operating system, network software* (jika diperlukan) dan program aplikasi pendukung lainnya.
3. Data, meliputi:
   a. Data pada sebuah *system database* baik itu *single-user system* maupun *multi-user system* harus terintegrasi dan dapat digunakan bersama (*Integrated and Shared*).
   b. Digunakan oleh organisasi dan deskripsi dari data disebut *schema Procedures.*
4. Procedures

   Instruksi dan aturan yang harus disertakan dalam mendesain dan menggunakan *database* dan DBMS.
5. People, meliputi:
   a. DA (*Data Administrator*), seseorang yang berwenang untuk membuat keputusan stategis dan kebijakan mengenai data yang ada.
   b. DBA (*DataBase Administrator*), menyediakan dukungan teknis untuk implementasi keputusan tersebut, dan bertanggungjawab atas keseluruhan kontrol system pada *level* teknis.
   c. *Database Designer* (*Logical and Physical*)
   d. *Application Programmers*, bertanggungjawab untuk membuat aplikasi *database* dengan menggunakan bahasa pemrograman yang ada, seperti : C++, Java, dan lainnya.

e. *End Users*, Siapapun yang berinteraksi dengan sistem secara *online* melalui *workstation*/terminal.

2.9.3.2. Kekurangan DBMS

Berikut ini adalah kekurangan DBMS:

1. Rumit (*Complexity*)

   Karena penetapan fungsi dari DBMS yang baik, menyebabkan DBMS menjadi *software* yang cukup rumit. Seluruh *user* harus mengetahui fungsi-fungsi yang ada dengan baik, sehingga dapat memperoleh manfaatnya.
2. Ukuran (*Size*)

   Kerumitan dan banyaknya fungsi yang ada menyebabkan DBMS memerlukan banyak *software* pendukung yang mengakibatkan penambahan tempat penyimpanan dan *memory*.
3. Biaya DBMS (*Cost of* DBMS)
4. Biaya Tambahan Perangkat Keras (*Additional hardware costs*)
5. Biaya Konversi (*Cost of conversion*)
6. *Performance*

   Pada dasarnya DBMS dibuat untuk menyediakan banyak aplikasi, akibatnya mungkin beberapa aplikasi akan berjalan tidak seperti biasanya.
7. *Higher impact of a failure*

   Karena *system* yang terpusat, jika seluruh user dan aplikasi terakses dari DBMS maka kerusakan pada bagian manapun dari sistem, akan menyebabkan operasi terhenti.

2.9.3.3. Fungsi-fungsi DBMS

Berikut ini adalah Fungsi dari DBMS:

1. *Data definition*

   DBMS harus dapat mengolah pendefinisian data.
2. *Data manipulation*

   DBMS harus dapat menangani permintaan-permintaan dari pemakai untuk mengakses data.

3. *Data security and integrity*

   DBMS harus dapat memeriksa keamanan dan integritas data yang didefinisikan oleh DBA.

4. *Data recovery and concurrency*

   DBMS harus dapat menangani kegagalan pengaksesan database yang disebabkan oleh kesalahan *system*, kerusakan *disk* dan sebagainya. DBMS harus dapat memantau pengaksesan data yang konkuren yaitu bila satu data diakses secara bersama-sama oleh lebih dari satu pemakai pada saat bersamaan.

5. *Data dictionary*

   Tempat penyimpanan informasi yang menggambarkan data dalam *database*. Data *dictionary* disebut juga metadata (data mengenai data) berisi tentang:

   a. Nama-nama *user* yang mempunyai wewenang untuk penggunaan DBMS.
   b. Nama-nama *item* data.
   c. Jenis-jenis dan ukuran *item* data.
   d. Batasan untuk masing-masing *item* data.

6. *Performance*

   DBMS harus dapat menangani unjuk kerja dari semua fungsi se-*efisien* mungkin.

2.10. PHP

PHP merupakan kependekan dari kata *Hypertext Preprocessor*. PHP adalah bahasa skrip yang dapat ditanamkan atau disisipkan ke dalam HTML. Pemrograman PHP sangat cocok dikembangkan dalam lingkungan *web*, karena PHP bisa dilekatkan pada *script* HTML atau sebaliknya. PHP dikhususkan untuk pengembangan web dinamis. Maksudnya, PHP mampu menghasilkan website yang secara terus-menerus hasilnya bisa berubah-ubah sesuai dengan pola yang diberikan. Hal tersebut bergantung pada permintaan *client browser*-nya (bisa menggunakan *browser Opera, Internet Explorer, Mozilla*, dan lain-lain).

Pada umumnya, pembuatan web dinamis berhubungan erat dengan *database* sebagai sumber data yang akan ditampilkan. PHP tergolong juga sebagai bahasa pemrograman berbasis *server* (*server side scripting*). Ini berarti bahwa semua *script* PHP diletakkan di *server* dan diterjemahkan oleh *web server* terlebih dahulu, kemudian hasil terjemahan itu dikirim ke browser *client*. Secara teknologi, bahasa pemrograman PHP memiliki kesamaan antara bahasa ASP (*Active Server Page*), *Cold Fusion*, JSP (*Java Server Page*), ataupun Perl.

2.10.1. Tipe Data

PHP tidak memerlukan pendeklarasian tipe data suatu *variable* secara eksplisit, tetapi lebih ditentukan oleh *runtime program* PHP, tergantung pada konteks bagaimana *variable* tesebut digunakan.

2.10.1.1. Jenis Tipe Data

PHP mendukung delapan jenis tipe data, antara lain :

a. Tipe scalar. Tipe *scalar* tidak bisa dipecah lagi menjadi bagian yang lebih kecil; boleh dikatakan merupakan tipe dasar.
   - Boolean
   - Integer
   - Float (floating point, ‘double’)
   - String

b. *Tipe compound*/Tipe campuran
   - Array
   - Object

c. *Tipe special*/Tipe khusus
   - Resource
   - Null

1) *Tipe Boolean*

   Boolean adalah jenis tipe data yang paling sederhana; banyak digunakan untuk mencari nilai kebenaran. *Boolean* bisa bernilai *TRUE* atau *FALSE* dan keduanya bersifat *case insensitive*.

2) *Integer* adalah sederet angka yang dituliskan sebagai set Z={…,-2,-1,0,1,2,..}. Penulisan bilangan *integer* bisa dalam notasi desimal (10-based), heksadesimal (16-based), atau octal (8-based), termasuk penanda (- atau +). 50 Jika digunakan notasi octal, penulisan harus didahului dengan angka 0 (nol) dan jika digunakan notasi heksadesimal, penulisan harus didahului dengan angka 0x;.

3) Tipe Pecahan / *Floating Point*

   Jenis *floating point* merupakan bilangan pecahan dengan presisi tinggi. Termasuk dalam bilangan *floating point,* antara lain *float, double,* dan *real*. Besar ukuran tipe data *float* bergantung pada *platform* yang digunakan; secara umum adalah 1.8 x 10308 dengan tingkat presisi 14 digit desimal. Hal ini mengacu pada standar format 64 bit IEEE.

4) *Tipe String*

   String boleh dikatakan sebagai serangkaian karakter. Besarnya karakter sama dengan *byte*. Ada tiga cara penulisan *string* yaitu :

   a. *Single quoted* / petik tunggal

      Penulisan *string* dengan *single quoted* harus diawali dan diakhiri petik tunggal (‘…’). Cara ini sedikit membingungkan jika ada karakterkarakter khusus (*escaped character*) yang disertakan. Misalnya, jika ada karakter petik tunggal yang akan disisipkan, karakter tersebut harus didahului dengan backslash (\).

   b. *Double quoted* / petik ganda

      Penulisan *string* dengan *double quoted* harus diawali dan diakhiri tanda petik ganda (“…”). Pada dasarnya, cara penulisan *double quoted* hampir 51 sama dengan *single quoted,* tetapi *double quoted* lebih fleksibel dan memiliki lebih banyak *escaped character* yang bisa disisipkan.

5) *Tipe Array*

Array adalah sekumpulan data yang disimpan dalam suatu variable dengan nama yang sama. Untuk membedakan antara data satu dan data yang lain, digunakan *index* atau *keys*. Setiap data dalam *array* disebut *element*. *Array* dalam PHP sedikit berbeda dengan *array* pada bahasa pemrograman yang lain. Pada umumnya, setiap elemen *array* harus memiliki kesamaan tipe, tetapi hal tersebut tidak berlaku bagi PHP. Di dalam PHP, diperbolehkan adanya perbedaan jenis tipe data di setiap elemen *array*, karena perlu diingat bahwa PHP tidak memerlukan pendeklarasian variabel, termasuk variabel *array*. Dengan kata lain, setiap elemen *array* bisa memiliki berbagai macam jenis tipe data yang tidak sejenis dalam satu nama variabel *array*. *Array* dapat diciptakan dengan pernyataan *array()* yang disertai dengan "*key=>value*" lebih dari satu, di setiap "*key=>value*" harus dipisahkan dengan tanda koma (",").

6) *Tipe Null*

Nilai *null* mewakili suatu variabel yang tidak memiliki nilai apa pun. Tipe *null* dikenali PHP sejak versi 4 ke atas. Beberapa kemungkinan jika variabel *null* terjadi:

a. Jika variabel diberi nilai dengan konstanta NULL;
b. Variabel belum diberi nilai atau belum di-set sama sekali;
c. Jika perintah *unset()* dikenakan pada suatu variabel.

2.11. 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 merupakan open source penghubung sistem manajemen database. Ini didasarkan pada struktur bahasa query (SQL) yang akan digunakan untuk menambah, menghapus, dan mengubah informasi di database. Standar perintah SQL, seperti *ADD*, *DROP*, *INSERT, UPDATE* dan dapat digunakan dengan MySQL.

2.11.1.Tipe Data

Berikut ini adalah tipe data dalam MySQL.

Tabel 2.3 Tipe Data

Tipe Data	Keterangan
INT(M) [UNSIGNED]	Angka -2147483648 s/d 2147483647
FLOAT(M,D)	Angka pecahan
DATE	Tanggal Format : YYYY-MM-DD
DATETIME	Tanggal dan Waktu Format : YYYY-MM-DD HH:MM:SS
CHAR(M)	String dengan panjang tetap sesuai dengan yang ditentukan. Panjangnya 1-255 karakter
VARCHAR(M)	String dengan panjang yang berubah-ubah sesuai dengan yang disimpan saat itu. Panjangnya 1 – 255 karakter
BLOB	Teks dengan panjang maksimum 65535 karakter
LONGBLOB	Teks dengan panjang maksimum 4294967295 karakter

2.12. Web Server

Server web dapat merujuk baik pada perangkat keras ataupun perangkat lunak yang menyediakan layanan akses kepada pengguna melalui protokol komunikasi *HTTP* atau *HTTPS* atas berkas-berkas yang terdapat pada suatu situs web dalam layanan ke pengguna dengan menggunakan aplikasi tertentu seperti *browser web* [19].

Fungsi utama sebuah server web adalah untuk mentransfer berkas atas permintaan pengguna melalui protokol komunikasi yang telah ditentukan. Disebabkan sebuah halaman web dapat terdiri atas berkas teks, gambar, *video*, dan lainnya pemanfaatan server web berfungsi pula untuk mentransfer seluruh aspek

pemberkasan dalam sebuah halaman web yang terkait termasuk di dalamnya teks, gambar, *video*, atau lainnya.

Macam-macam *Web Server* diantaranya [19]:

1. *Web Server Apache*
2. *Web Server Litespeed*
3. *Web Server Nginx*
4. *Web Server IIS*

2.12.1. Web Server Apache

Web Server Apache merupakan *web server* yang paling banyak dipergunakan di Internet. Program ini pertama kali didesain untuk sistem operasi lingkungan UNIX. Namun demikian, pada beberapa versi berikutnya *Apache* mengeluarkan programnya yang dapat dijalankan di Windows NT. *Apache* mempunyai program pendukung yang cukup banyak. Hal ini memberikan layanan yang cukup lengkap bagi penggunanya. Beberapa dukungan *Apache*:

1. Kontrol Akses
2. CGI (*Common Gateway Interface*)
3. PHP (*Personal Home Page/PHP Hypertext Processor*)
4. SSI (*Server Side Includes*)

2.12.2. Web Server Litespeed

Web Server Litespeed merupakan teknologi baru yang diciptakan untuk menggantikan *Web Server Apache* di masa mendatang. *Litespeed* menurut hasil riset mempunyai kelebihan yaitu 50% lebih cepat dalam memproses PHP dan lebih cepat 6x dibandingkan *Web Server Apache.*

2.12.3. Web Server Nginx

Nginx dengan cepat memberikan konten statis dengan penggunaan efisien sumber daya sistem. Hal ini dapat menyebarkan dinamis *HTTP* konten di jaringan menggunakan *FastCGI handler* untuk *script*, dan dapat berfungsi sebagai perangkat lunak yang sangat mampu penyeimbang beban. Nginx menggunakan *asynchronous-event* pendekatan untuk menangani permintaan yang diprediksi memberikan kinerja yang lebih bawah beban, kontras dengan *Apache HTTP*

server model yang menggunakan berulir atau proses yang berorientasi pada pendekatan-permintaan penanganan.

2.12.4. Web Server IIS

Internet Information Service (IIS) adalah komponen yang dapat digunakan untuk mengelola web, *File Transfer Protocol* (FTP), *Ghoper*, dan *NNTP*. Komponen IIS terdapat pada sistem operasi Windows NT ,Windows 2000, Windows XP, 7 dan Windows Server 2003. Pada Windows 98, agar dapat mengelola web diperlukan komponen *Personal Web Server* (PWS) walaupun tidak terdapat fasilitas untuk FTP. PWS juga merupakan bagian dari IIS. Komponen PWS terdapat pada CD master Windows 98 dan terletak pada direktori \add-ons\pws. Instalasi PWS dapat dilakukan dari <Drive CD ROOM>:\ADD-ONS|PWS|SETUP.

2.13. Virtual Private Server

VPS (*virtual private server*) yaitu sebuah komputer atau *server*, dimana *server* ini di sewa untuk dipakai trading. VPS dapat diakses dimanapaun melalui jaringan internet. Layaknya sebuah PC/Laptop VPS bisa di-*install software* atau untuk penyimpan data. Secara global VPS sering digunakan untuk *Cloud Computing*, *Software Bot*, Menjalankan *Software robot forex* (untuk trading) dan lain-lain [20].

Fungsi VPS sesuai uraian di atas sudah jelas yaitu sebagai pengganti dan pembantu sebuah perangkat seperti komputer yang bisa dijalankan selama 24 jam tanpa henti, yang bisa digunakan untuk meyimpan data penting ataupun menjalankan *software* yang *realtime* (Selama masih berlangganan).

Spesifikasi sebuah VPS itu sendiri berbagai macam, baik dari segi *Harddisk*, *memory*, jenis *processor*, pilihan sistem operasinya (*Windows/Linux*/Lainnya), dan sebagainya. VPS sudah terhubung dengan internet selama 24 jam dengan kecepatan tinggi agar setiap *user* bisa dengan mudah mengaksesnya. VPS biasanya diakses melalui komputer pribadai menggunakan *software Remote Desktop Connection* (RDC) yang biasanya sudah tersedia di operasi sistem *WINDOWS*.

2.14. OWASP DirBuster

DirBuster adalah aplikasi java *multi thereaded* yang dirancang untuk *brute force* direktori dan *file name* pada *server* web/aplikasi [21]. Saat ini sudah banyak *tool* sejenis, namun seringkali hanya mengandalkan direktori dan *file* yang terbatas.

DirBuster hadir dengan 9 daftar yang berbeda, ini membuat DirBuster sangat efektif dalam menemukan *file-file* tersembunyi dan direktori. Dan jika itu tidak cukup DirBuster juga memiliki opsi untuk melakukan kekerasan murni, yang meninggalkan direktori tersembunyi dan *file* tersembunyi.

Prinsip-prinsip dasar bagaimana DirBuster kerjanya adalah dengan menggunakan "*Fail Case*". Ini berarti bahwa DirBuster akan mencoba untuk menentukan apa sebenarnya "*Fail Case*" adalah sebelum pengujian. DirBuster kemudian akan menentukan sesuatu (*file* atau direktori) untuk hadir jika tes dilaksanakan mengembalikan hasil yang berbeda dari "*Fail Kasus*".