Keamanan Sistem

Masalah keamanan merupakan salah satu aspek penting dari sebuah sistem informasi. Sayang sekali masalah keamanan ini sering kali kurang mendapat perhatian dari para pemilik & pengelola sistem informasi. Seringkali masalah keamanan berada di urutan kedua, bahkan di urutan terakhir dalam daftar hal-hal yang dianggap penting. Apabila menggangu performansi dari sistem, seringkali keamanan dikurangi / ditiadakan.

Informasi saat ini sudah menjadi sebuah komoditi yang sangat penting. Bahkan ada yang mengatakan bahwa sudah berada di sebuah *"information-based society*". Kemampuan untuk mengakses dan menyediakan informasi secara cepat dan akurat menjadi sangat esensial bagi sebuah organisasi, baik yang berupa organisasi komersial (perusahaan), perguruan tinggi, lembaga pemerintahan, maupun individual (pribadi). Hal ini dimungkinkan dengan perkembangan pesat di bidang teknologi komputer & telekomunikasi.

Sangat pentingnya nilai sebuah informasi menyebabkan seringkali informasi diinginkan hanya boleh diakses oleh orang-orang tertentu. Jatuhnya informasi ke tangan pihak lain (misalnya pihak lawan bisnis) dapat menimbulkan kerugian bagi pemilik informasi. Untuk itu keamanan dari sistem informasi yang digunakan harus terjamin dalam batas yang dapat diterima.

Jaringan komputer, seperti LAN 1 & Internet, memungkinkan untuk menyediakan informasi secara cepat. Ini salah satu alasan perusahaan / organisasi mulai berbondong-bondong membuat LAN untuk sistem informasinya dan menghubungkan LAN tersebut ke Internet. Terhubungnya LAN / komputer ke Internet membuka potensi adanya lubang keamanan *(security hole*) yang tadinya bisa ditutupi dengan mekanisme keamanan secara fisik. Ini sesuai dengan pendapat bahwa kemudahan (kenyamanan) mengakses informasi berbanding terbalik dengan tingkat keamanan sistem informasi itu sendiri. Semakin tinggi tingkat keamanan, semakin sulit (tidak nyaman) untuk mengakses informasi.

Keamanan informasi adalah bagaimana dapat mencegah penipuan *(cheating*) atau paling tidak mendeteksi adanya penipuan di sebuah sistem yang berbasis informasi, dimana informasinya sendiri tidak memiliki arti fisik.

Keamanan dan management perusahaan

Seringkali sulit untuk membujuk management perusahaan / pemilik sistem informasi untuk melakukan investasi di bidang keamanan. Penyaranan menggunakan *"Risk Management Model*" untuk menghadapi ancaman *(managing threats*). Ada tiga komponen yang memberikan kontribusi kepada Risk, yaitu *Asset*, *Vulnerabilities*, dan *Threats*.

Tabel Kontribusi terhadap Risk

Nama komponen	Contoh & keterangan
Assets (aset)	• hardware • software • dokumentasi • data • komunikasi • linkungan • manusia
Threats (ancaman)	• pemakai *(users*) • teroris • kecelakaan *(accidents*) • crackers • penjahat kriminal • nasib *(acts of God*) • intel luar negeri *(foreign intelligence*)
Vulnerabilities (kelemahan)	• software bugs • hardware bugs • radiasi (layar, transmisi) • tapping, crosstalk • *unauthorized users* • cetakan / printout • keteledoran *(oversight*) • cracker via telepon • storage media

Untuk menanggulangi resiko *(Risk*) dilakukan *countermeasures*" yang dapat berupa :

- usaha untuk mengurangi *Threat*
- usaha untuk mengurangi *Vulnerability*
- usaha untuk mengurangi impak *(impact*)
- mendeteksi kejadian yang tidak bersahabat *(hostile event*)
- kembali *(recover*) dari kejadian

Meningkatnya Kejahatan Komputer

Jumlah kejahatan komputer *(computer crime*), terutama yang berhubungan dengan sistem informasi akan terus meningkat dikarenakan beberapa hal, antara lain :

- Aplikasi bisnis yang menggunakan (berbasis) teknologi informasi dan jaringan komputer semakin meningkat. Sebagai contoh saat ini mulai bermunculan aplikasi bisnis seperti *online*

banking, *electronic commerce (e-commerce*), *Electronic Data Interchange* (EDI), dsb.

- *Desentralisasi* (*distributed*) server menyebabkan lebih banyak sistem yang harus ditangani. Hal ini membutuhkan lebih banyak operator dan administrator yang handal yang juga kemungkinan harus disebar di seluruh lokasi. Padahal mencari operator dan administrator yang handal adalah sangat sulit.
- Transisi dari *single vendor* ke *multi-vendor* sehingga lebih banyak sistem / perangkat yang harus dimengerti dan masalah *interoperability* antar vendor yang lebih sulit ditangani. Untuk memahami satu jenis perangkat dari satu vendor saja sudah susah apalagi harus menangani berjenis-jenis perangkat.
- Meningkatnya kemampuan pemakai di bidang komputer sehingga mulai banyak pemakai yang mencoba-coba bermain / membongkar sistem yang digunakannya.
- Kesulitan dari penegak hukum untuk mengejar kemajuan dunia komputer dan telekomunikasi yang sangat cepat.
- Semakin kompleksnya sistem yang digunakan, seperti semakin besarnya program *(source code*) yang digunakan sehingga semakin besar probabilitas terjadinya lubang keamanan (yang disebabkan kesalahan pemrograman).
- Semakin banyak perusahaan yang menghubungkan sistem informasinya dengan jaringan komputer yang global seperti Internet. Hal ini membuka akses dari seluruh dunia. Potensi sistem informasi yang dapat dijebol menjadi lebih besar.

Klasifikasi Kejahatan Komputer

Kejahatan komputer dapat digolongkan kepada yang sangat berbahaya sampai ke yang hanya mengesalkan *(annoying*). Berdasarkan lubang keamanan, keamanan dapat diklasifikasikan menjadi empat, yaitu :

1. **Keamanan yang bersifat fisik** *(physical security*) : termasuk akses orang ke gedung, peralatan dan media yang digunakan. Misalnya pernah diketemukan coretan password / manual yang dibuang tanpa dihancurkan, *Wiretapping* / hal-hal yang berhubungan dengan akses ke kabel / komputer yang digunakan, *Denial of service*, yaitu akibat yang ditimbulkan sehingga servis tidak dapat diterima oleh pemakai. *Denial of service* dapat dilakukan misalnya dengan mematikan peralatan / membanjiri saluran komunikasi dengan pesan-pesan (yang dapat berisi apa saja karena yang diutamakan adalah banyaknya jumlah pesan). Beberapa waktu yang lalu ada lubang keamanan dari implementasi protokol TCP/IP yang dikenal dengan istilah *Syn Flood Attack*, dimana sistem *(host*) yang dituju dibanjiri oleh permintaan sehingga menjadi terlalu sibuk dan bahkan dapat berakibat macetnya sistem *(hang*).
2. **Keamanan yang berhubungan dengan orang (personel**) : termasuk identifikasi dan profil resiko dari orang yang mempunyai akses (pekerja). Seringkali kelemahan keamanan sistem informasi bergantung kepada manusia (pemakai dan pengelola). Ada sebuah teknik yang dikenal dengan istilah *"social engineering*" yang sering digunakan oleh kriminal untuk berpura-pura sebagai orang yang berhak mengakses informasi.
3. **Keamanan dari data dan media serta teknik komunikasi** *(communications*). Yang termasuk di dalam kelas ini adalah kelemahan dalam software yang digunakan untuk mengelola data. Seorang kriminal dapat memasang *virus / trojan horse* sehingga dapat mengumpulkan informasi (seperti password) yang semestinya tidak berhak diakses.
4. **Keamanan dalam operasi** : termasuk prosedur yang digunakan untuk mengatur dan mengelola sistem keamanan dan juga termasuk prosedur setelah serangan *(post attack recovery*).

Aspek / servis dari security

Keamanan komputer *(computer security*) melingkupi empat aspek, yaitu *privacy*, *integrity*, *authentication* dan *availability*. Selain keempat hal di atas, masih ada dua aspek lain yang juga sering dibahas dalam kaitannya dengan *electronic commerce*, yaitu *access control* dan *nonrepudiation*.

Privacy / Confidentiality

Inti utama aspek *privacy / confidentiality* adalah usaha untuk menjaga informasi dari orang yang tidak berhak mengakses. *Privacy* lebih kearah data-data yang sifatnya privat sedangkan *confidentiality* biasanya berhubungan dengan data yang diberikan ke pihak lain untuk keperluan tertentu (misalnya sebagai bagian dari pendaftaran sebuah servis) dan hanya diperbolehkan untuk keperluan tertentu tersebut. Contoh hal yang berhubungan dengan *privacy* adalah e-mail seorang pemakai *(user*) tidak boleh dibaca oleh administrator. Contoh *confidential information* adalah data-data yang sifatnya pribadi (seperti nama, tempat tanggal lahir, *social security number*, agama, status perkawinan, penyakit yang pernah diderita, nomor kartu kredit, dsb) merupakan data-

data yang ingin diproteksi penggunaan dan penyebarannya. Contoh lain dari *confidentiality* adalah daftar pelanggan dari sebuah *Internet Service Provider* (ISP). Serangan terhadap aspek *privacy* misalnya adalah usaha untuk melakukan penyadapan (dengan program *sniffe*r). Usaha-usaha yang dapat dilakukan untuk meningkatkan *privacy* dan *confidentiality* adalah dengan menggunakan teknologi kriptografi.

Integrity

Aspek ini menekankan bahwa informasi tidak boleh diubah tanpa seijin pemilik informasi. Adanya virus, *trojan horse* / pemakai lain yang mengubah informasi tanpa ijin merupakan contoh masalah yang harus dihadapi. Sebuah e-mail dapat saja "ditangkap" *(intercep*t) di tengah jalan, diubah isinya, kemudian diteruskan ke alamat yang dituju. Dengan kata lain, integritas dari informasi sudah tidak terjaga. Penggunaan enkripsi dan digital signature, misalnya, dapat mengatasi masalah ini. Salah satu contoh kasus *trojan horse* adalah distribusi paket program *TCP Wrapper* (yaitu program populer yang dapat digunakan untuk mengatur dan membatasi akses TCP/IP) yang dimodifikasi oleh orang yang tidak bertanggung jawab.

Authentication

Aspek ini berhubungan dengan metoda untuk menyatakan bahwa informasi betul-betul asli / orang yang mengakses / memberikan informasi adalah betul-betul orang yang dimaksud. Masalah pertama, membuktikan keaslian dokumen dapat dilakukan dengan teknologi *watermarking* dan *digital signature*. *Watermarking* juga dapat digunakan untuk menjaga *"intelectual property*", yaitu dengan menandai dokumen / hasil karya dengan "tanda tangan" pembuat. Masalah kedua biasanya berhubungan dengan *access control*, yaitu berkaitan dengan pembatasan orang yang dapat mengakses informasi. Dalam hal ini pengguna harus menunjukkan bukti bahwa memang dia adalah pengguna yang sah, misalnya dengan menggunakan password, biometric (ciri-ciri khas orang), dan sejenisnya. Penggunaan teknologi *smart car*d saat ini kelihatannya dapat meningkatkan keamanan aspek ini.

Availability

Aspek *availability* / ketersediaan berhubungan dengan ketersediaan informasi ketika dibutuhkan. Sistem informasi yang diserang / dijebol dapat menghambat / meniadakan akses ke informasi. Contoh hambatan adalah serangan yang sering disebut dengan *"denial of service attac*k" (DoS *attack*), dimana server dikirimi permintaan (biasanya palsu) yang bertubi-tubi / permintaan yang diluar perkiraan sehingga tidak dapat melayani permintaan lain / bahkan sampai *dow*n, *han*g, *crash*. Contoh lain adalah adanya *mailbom*b, dimana seorang pemakai dikirimi e-mail bertubi-tubi (katakan ribuan e-mail) dengan ukuran yang besar sehingga sang pemakai tidak dapat membuka e-mailnya atau kesulitan mengakses e-mailnya (apalagi jika akses dilakukan melalui saluran telepon).

Access Control

Aspek ini berhubungan dengan cara pengaturan akses kepada informasi. Hal ini biasanya berhubungan dengan masalah *authentication* dan juga *privacy*. *Access control* seringkali dilakukan dengan menggunakan kombinasi *user id*/*password* atau dengan menggunakan mekanisme lain.

Non-repudiation

Aspek ini menjaga agar seseorang tidak dapat menyangkal telah melakukan sebuah transaksi. Sebagai contoh, seseorang yang mengirimkan email untuk memesan barang tidak dapat menyangkal bahwa dia telah mengirimkan email tersebut. Aspek ini sangat penting dalam hal *electronic commerce*. Penggunaan *digital signature* dan teknologi kriptografi secara umum dapat menjaga aspek ini. Akan tetapi hal ini masih harus didukung oleh hukum sehingga status dari *digital signature* itu jelas legal.

Serangan Terhadap Keamanan Sistem Informasi

Security attack / serangan terhadap keamanan sistem informasi, dapat dilihat dari sudut peranan komputer / jaringan komputer yang fungsinya adalah sebagai penyedia informasi. Ada beberapa kemungkinan serangan *(attac*k) :

- *Interruptio*n : Perangkat sistem menjadi rusak / tidak tersedia. Serangan ditujukan kepada ketersediaan *(availabilit*y) dari sistem. Contoh serangan adalah "*denial of service attack*".
- *Interceptio*n : Pihak yang tidak berwenang berhasil mengakses aset / informasi. Contoh dari serangan ini adalah penyadapan *(wiretappin*g).
- *Modificatio*n : Pihak yang tidak berwenang tidak saja berhasil mengakses, akan tetapi dapat juga mengubah (*tamper*) aset. Contoh dari serangan ini antara lain adalah mengubah isi dari web site dengan pesan-pesan yang merugikan pemilik web site.
- *Fabricatio*n : Pihak yang tidak berwenang menyisipkan objek palsu ke dalam sistem. Contoh dari serangan jenis ini adalah

memasukkan pesan-pesan palsu seperti e-mail palsu ke dalam jaringan komputer.

Dasar-dasar (*principles*) dan teori-teori yang digunakan untuk pengamanan sistem informasi misalnya berupa kriptografi dan enkripsi (baik dengan menggunakan *private-key* maupun dengan menggunakan *public-key*).

Kriptografi *(cryptograph*y) merupakan ilmu dan seni untuk menjaga pesan agar aman. *"Crypt*o" berarti *"secre*t" (rahasia) dan *"graph*y" berarti *"writin*g". Para pelaku / praktisi kriptografi disebut ***cryptographer*s**. Sebuah algoritma kriptografik *(cryptographic algorith*m) disebut ***cipher***, merupakan persamaan matematik yang digunakan untuk proses enkripsi dan dekripsi. Biasanya kedua persamaan matematik (untuk enkripsi dan dekripsi) tersebut memiliki hubungan matematis yang cukup erat.

Proses yang dilakukan untuk mengamankan sebuah pesan (yang disebut *plaintex*t) menjadi pesan yang tersembunyi (disebut *ciphertex*t) adalah **enkripsi** *(encryptio*n). *Ciphertext* adalah pesan yang sudah tidak dapat dibaca dengan mudah. Terminologi yang lebih tepat digunakan adalah *"encipher"*. Proses sebaliknya, untuk mengubah *ciphertext* menjadi *plaintex*t, disebut **dekripsi** *(decryptio*n). Terminologi yang lebih tepat untuk proses ini adalah *"decipher"*. *Cryptanalysis* adalah seni dan ilmu untuk memecahkan *ciphertext* tanpa bantuan kunci. *Cryptanalyst* adalah pelaku atau praktisi yang menjalankan *cryptanalysi*s.

Enkripsi

Enkripsi digunakan untuk menyandikan data-data / informasi sehingga tidak dapat dibaca oleh orang yang tidak berhak. Dengan enkripsi, data disandikan *(encrypte*d) dengan menggunakan sebuah kunci *(ke*y). Untuk membuka *(decrypt*) data tersebut digunakan juga sebuah kunci yang dapat sama dengan kunci untuk mengenkripsi (untuk kasus *private key cryptograph*y) / dengan kunci yang berbeda (untuk kasus *public key cryptograph*y). Gambar dibawah ini menunjukkan contoh proses enkripsi dan dekripsi dengan dua kunci yang berbeda.

Enkripsi

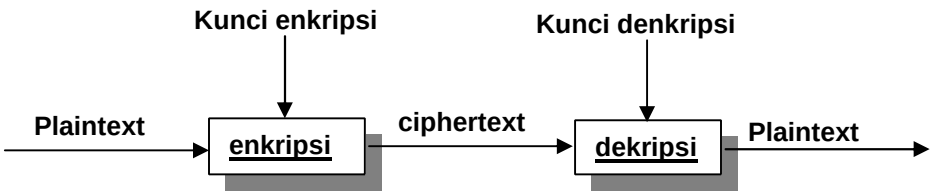
Secara matematis, proses atau fungsi enkripsi *(*E) dapat dituliskan sebagai : *E (M) =C,* dimana : *M* adalah *plaintext (messag*e) dan *C* adalah *ciphertex*t. Proses atau fungsi dekripsi *(*D) dapat dituliskan sebagai : *D (C) = M*

Elemen dari Enkripsi

Ada beberapa elemen dari enkripsi yang akan dijabarkan dalam beberapa paragraf di bawah ini.

Algoritma dari Enkripsi dan Dekripsi. Algoritma dari enkripsi adalah fungsi-fungsi yang digunakan untuk melakukan fungsi enkripsi dan dekripsi. Algoritma yang digunakan menentukan kekuatan dari enkripsi, dan ini biasanya dibuktikan dengan basis matematika.

Kunci yang digunakan dan panjangnya kunci. Kekuatan dari penyandian bergantung kepada kunci yang digunakan. Beberapa algoritma enkripsi memiliki kelemahan pada kunci yang digunakan. Untuk itu, kunci yang lemah tersebut tidak boleh digunakan. Selain itu, panjangnya kunci, yang biasanya dalam ukuran *bi*t, juga menentukan kekuatan dari enkripsi. Kunci yang lebih panjang biasanya lebih aman dari kunci yang pendek. Jadi enkripsi dengan menggunakan kunci 128-bit lebih sukar dipecahkan dengan algoritma enkripsi yang sama tetapi dengan kunci 56-bit. Semakin panjang sebuah kunci, semakin besar keyspace yang harus dijalani untuk mencari kunci dengan cara *brute force attack* atau coba-coba karena keyspace yang harus dilihat merupakan pangkat dari bilangan 2. Jadi kunci 128-bit memiliki keyspace 2 128 , sedangkan kunci 56-bit memiliki keyspace 2 56 . Artinya semakin lama kunci baru bisa ketahuan.



Gambar Diagram proses enkripsi dan dekripsi

Plaintext adalah pesan atau informasi yang dikirimkan. **Ciphertext** adalah informasi yang sudah dienkripsi.

Keamanan sebuah algoritma yang digunakan dalam enkripsi / dekripsi bergantung kepada beberapa aspek. Salah satu aspek yang cukup penting adalah sifat algoritma yang digunakan. Apabila kekuatan dari sebuah algoritma sangat tergantung kepada pengetahuan (tahu / tidaknya) orang terhadap algoritma yang digunakan, maka algoritma tersebut disebut "*restricted algorithm*". Apabila algoritma tersebut bocor / ketahuan oleh orang banyak, maka pesan-pesan dapat terbaca. Tentunya hal ini masih bergantung kepada adanya kriptografer yang baik. Jika tidak ada yang tahu, maka sistem tersebut dapat dianggap aman (meskipun semu). Meskipun kurang aman, metoda pengamanan dengan *restricted algorithm* ini cukup banyak digunakan karena mudah implementasinya dan tidak perlu diuji secara mendalam. Contoh penggunaan metoda ini adalah enkripsi yang menggantikan huruf yang digunakan untuk mengirim pesan dengan huruf lain. Ini disebut dengan "*substitution cipher*".

Substitution Cipher dengan Caesar Cipher

Salah satu contoh dari "*substitution cipher*" adalah *Caesar Cipher* yang digunakan oleh Julius Caesar. Pada prinsipnya, setiap huruf digantikan dengan huruf yang berada tiga (3) posisi dalam urutan alfabet. Sebagai contoh huruf "a" digantikan dengan huruf "D" dan seterusnya. Transformasi yang digunakan adalah :

plain : a b c d e f g h i j k l m n o p q r s t u v w x y z
cipher : D E F G H I J K L M N O P Q R S T U V W X Y Z A B C

Public-key cryptography lawan symmetric cryptography

Perbedaan prinsip dan penggunaan *public-key cryptography* dan *symmetric cryptography* membutuhkan diskusi tersendiri. Pada *symmetric cryptography*, satu kunci yang sama digunakan untuk melakukan enkripsi dan dekripsi. Pada sistem *public-key cryptography*, enkripsi dan dekripsi menggunakan kunci yang berbeda. Sejak dikembangkannya *public-key cryptography*, selalu timbul pertanyaan mana yang lebih baik. Para pakar kriptografi mengatakan bahwa keduanya tidak dapat dibandingkan karena memecahkan masalah dalam domain yang berbeda. *Symmetric cryptography* merupakan hal yang terbaik untuk mengenkripsi data. Kecepatannya dan keamanan akan *choosen-ciphertext attack* merupakan kelebihannya. Sementara itu *public-key cryptography* dapat melakukan hal-hal lain lebih baik daripada *symmetric cryptography*, misalnya dalam hal *key management*.

Data Encryption Standard (DES)

DES / *Data Encryption Algorithm* (DEA) oleh ANSI dan DEA-1 oleh ISO, merupakan algoritma kriptografi yang paling umum digunakan saat ini. Sejarahnya DES dimulai dari permintaan pemerintah Amerika Serikat untuk memasukkan proposal enskripsi. DES memiliki sejarah dari Lucifer, enkripsi yang dikembangan di IBM kala itu. Horst Feistel merupakan salah satu periset yang mula-mula mengembangkan DES ketika bekerja di IBM Watson Laboratory di Yorktown Heights, New York. DES baru secara resmi digunakan oleh pemerintah Amerika Serikat di tahun 1977. Aplikasi yang menggunakan DES antara lain :

- enkripsi dari password di sistem UNIX
- berbagai aplikasi di bidang perbankan

Memecahkan DES

DES merupakan *block chiper* yang beroperasi dengan menggunakan blok berukuran 64-bit dan kunci berukuran 56-bit. *Brute force attack* dengan mencoba segala kombinasi membutuhkan 2^{56} kombinasi atau sekitar $7x\ 10^{17}$ atau 70 juta milyar kombinasi. DES dengan penggunaan yang biasa *(cookbook mode)* dengan panjang kunci 56 bit saat ini sudah dapat dianggap tidak aman karena sudah berhasil dipecahkan dengan metoda coba-coba *(brute force attack)*. Ada berbagai group yang mencoba memecahkan DES dengan berbagai cara. Salah satu group yang bernama ***distributed.net*** menggunakan teknologi Internet untuk memecahkan problem ini menjadi sub-problem yang kecil (dalam ukuran blok). Pengguna dapat menjalankan sebuah program yang khusus dikembangkan oleh tim ini untuk mengambil beberapa blok, via Internet, kemudian memecahkannya di komputer pribadinya. Program yang disediakan meliputi berbagai operating system seperti Windows, DOS, berbagai variasi Unix, Macintosh. Blok yang sudah diproses dikembalikan ke *distributed.net* via Internet. Dengan cara ini puluhan ribu orang, termasuk penulis, membantu memecahkan DES. Mekanisme ini dapat memecahkan DES dalam waktu 30 hari. Sebuah group lain yang disebut *Electronic Frontier Foundation* (EFF) membuat sebuah komputer yang dilengkapi dengan *Integrated Circuit chip DES cracker*. Dengan mesin seharga US$50.000 ini dapat memecahkan DES 56-bit dalam waktu rata-rata empat sampai lima hari. DES cracker yang mereka kembangkan dapat melakukan eksplorasi keseluruhan dari

56 bit *keyspace* dalam waktu sembilan hari. Dikarenakan 56-bit memiliki 2^{16} (65536) *keyspace* dibandingkan DES dengan 40 bit, maka untuk memecahkan DES 40 bit hanya dibutuhkan waktu sekitar 12 detik. Dikarenakan hukum average, waktu rata-rata untuk memecahkan DES 40 bit adalah 6 detik. Perlu diingat bahwa group seperti EFF merupakan group kecil dengan budget yang terbatas. Dapat dibayangkan sistem yang dimiliki oleh *National Security Agency* (NSA) dari pemerintah Amerika Serikat . Tentunya mereka dapat memecahkan DES dengan lebih cepat.
(cat : Sembilan hari sama dengan 777.600 detik. Jika angka tersebut dibagi dengan 65.536 maka hasilnya adalah sekitar 12 detik.)

Hash function - integrity checking
Salah satu cara untuk menguji integritas sebuah data adalah dengan memberikan "*checksum*" / tanda bahwa data tersebut tidak berubah. Cara yang paling mudah dilakukan adalah dengan menjumlahkan karakter-karakter / data-data yang ada sehingga apabila terjadi perubahan, hasil penjumlahan menjadi berbeda. Cara ini tentunya mudah dipecahkan dengan menggunakan kombinasi data yang berbeda akan tetapi menghasilkan hasil penjumlahan yang sama.

Pada sistem digital biasanya ada beberapa mekanisme pengujian integritas seperti antara lain:
- parity checking
- checksum
- hash function

Hash function merupakan fungsi yang bersifat satu arah dimana jika kita masukkan data, maka dia akan menghasilkan sebuah "*checksum*" / "*fingerprint*" dari data tersebut. Ada beberapa *hash function* yang umum digunakan, antara lain:
- MD5
- SHA

Meski sebuah sistem informasi sudah dirancang memiliki perangkat pengamanan, dalam operasi masalah keamanan harus selalu dimonitor. Hal ini disebabkan oleh beberapa hal, antara lain:
- Ditemukannya lubang keamanan *(security hol*e) yang baru. Perangkat lunak dan perangkat keras biasanya sangat kompleks sehingga tidak mungkin untuk diuji seratus persen. Kadang-kadang ada lubang keamanan yang ditimbulkan oleh kecerobohan implementasi.
- Kesalahan konfigurasi. Kadang-kadang karena lalai / alpa, konfigurasi sebuah sistem kurang benar sehingga menimbulkan lubang keamanan. Misalnya *mode (permission* atau kepemilikan) dari berkas yang menyimpan password (/etc/passwd di sistem UNIX) secara tidak sengaja diubah sehingga dapat diubah / ditulis oleh orang-orang yang tidak berhak.
- Penambahan perangkat baru (hardware dan/atau software) yang menyebabkan menurunnya tingkat security / berubahnya metoda untuk mengoperasikan sistem. Operator dan administrator harus belajar lagi. Dalam masa belajar ini banyak hal yang jauh dari sempurna, misalnya server atau software masih menggunakan konfigurasi awal dari vendor (dengan password yang sama).

Sumber lubang keamanan
Lubang keamanan *(security hol*e) dapat terjadi karena beberapa hal, yaitu : salah disain *(design fla*w), salah implementasi, salah konfigurasi dan salah penggunaan.

Salah Disain
Lubang keamanan yang ditimbulkan oleh salah disain umumnya jarang terjadi. Akan tetapi apabila terjadi sangat sulit untuk diperbaiki. Akibat disain yang salah, maka biarpun dia diimplementasikan dengan baik, kelemahan dari sistem akan tetap ada. Contoh sistem yang lemah disainnya adalah algoritma enkripsi Caesar cipher, dimana karakter digeser 3 huruf. Meskipun diimplementasikan dengan programming yang sangat teliti, siapapun yang mengetahui algoritmanya dapat memecahkan enkripsi tersebut. Contoh lain lubang keamanan yang dapat dikategorikan kedalam kesalahan disain adalah disain urutan nomor *(sequence numberin*g) dari paket TCP/IP. Kesalahan ini dapat dieksploitasi sehingga timbul masalah yang dikenal dengan nama *"IP spoofin*g", yaitu sebuah host memalsukan diri seolah-olah menjadi host lain dengan membuat paket palsu setelah mengamati urutan paket dari host yang hendak diserang. Bahkan dengan mengamati cara mengurutkan nomor packet bisa dikenali sistem yang digunakan. Mekanisme ini digunakan oleh program *nmap* dan *queso* untuk mendeteksi *operating system* (OS) dari sebuah sistem, yang disebut *fingerprintin*g.

Implementasi kurang baik
Lubang keamanan yang disebabkan oleh kesalahan implementasi sering terjadi. Banyak program yang diimplementasikan secara terburu-buru sehingga kurang cermat dalam pengkodean. Akibatnya cek atau testing yang harus dilakukan menjadi tidak dilakukan. Sebagai contoh, seringkali batas ("*boun*d") dari sebuah *"arra*y" tidak dicek sehingga terjadi yang disebut *out-of-bound*

array / *buffer overflow* yang dapat dieksploitasi (misalnya overwrite ke variable berikutnya). Lubang keamanan yang terjadi karena masalah ini sudah sangat banyak dan yang mengherankan terus terjadi, seolah-olah para programmer tidak belajar dari pengalaman. Contoh lain sumber lubang keamanan yang disebabkan oleh kurang baiknya implementasi adalah kealpaan memfilter karakter-karakter yang aneh-aneh yang dimasukkan sebagai input dari sebuah program (misalnya input dari *CGI-script* 2) sehingga sang program dapat mengakses berkas / informasi yang semestinya tidak boleh diakses.

Salah konfigurasi

Meskipun program sudah diimplementasikan dengan baik, masih dapat terjadi lubang keamanan karena salah konfigurasi. Contoh masalah yang disebabkan oleh salah konfigurasi adalah berkas yang semestinya tidak dapat diubah oleh pemakai secara tidak sengaja menjadi *"writeabl*e". Apabila berkas tersebut merupakan berkas yang penting, seperti berkas yang digunakan untuk menyimpan password, maka efeknya menjadi lubang keamanan. Kadangkala sebuah komputer dijual dengan konfigurasi yang sangat lemah. Ada masanya workstation Unix di perguruan tinggi didistribusikan dengan berkas /etc/aliases (berguna untuk mengarahkan e-mail), /etc/utmp (berguna untuk mencatat siapa saja yang sedang menggunakan sistem) yang dapat diubah oleh siapa saja. Contoh lain dari salah konfigurasi adalah adanya program yang secara tidak sengaja diset menjadi *"setuid roo*t" sehingga ketika dijalankan pemakai memiliki akses seperti *super user (roo*t) yang dapat melakukan apa saja.

Salah menggunakan program atau sistem

Salah penggunaan program dapat juga mengakibatkan terjadinya lubang keamanan. Kesalahan menggunakan program yang dijalankan dengan menggunakan account root (super user) dapat berakibat fatal. Sering terjadi cerita horor dari sistem administrator baru yang teledor dalam menjalankan perintah "rm -rf" di sistem UNIX (yang menghapus berkas atau direktori beserta sub direktori di dalamnya). Akibatnya seluruh berkas di sistem menjadi hilang mengakibatkan *Denial of Service* (DoS). Apabila sistem yang digunakan ini digunakan bersama-sama, maka akibatnya dapat lebih fatal lagi. Untuk itu perlu berhati-hati dalam menjalan program, terutama apabila dilakukan dengan menggunakan account administrator seperti *root* tersebut. Kesalahan yang sama juga sering terjadi di sistem yang berbasis MS-DOS. Karena sudah mengantuk, misalnya, ingin melihat daftar berkas di sebuah direktori dengan memberikan perintah "dir *.*" ternyata salah memberikan perintah menjadi "del *.*" (yang juga menghapus seluruh file di direktori tersebut).

Penguji keamanan sistem

Dikarenakan banyaknya hal yang harus dimonitor, administrator dari sistem informasi membutuhkan *"automated tool*s", perangkat pembantu otomatis, yang dapat membantu menguji atau meng-evaluasi keamanan sistem yang dikelola. Untuk sistem yang berbasis UNIX ada beberapa tools yang dapat digunakan, antara lain:

- *Cops*
- *Tripwire*
- *Satan/Saint*
- *SBScan*: localhost security scanner

Untuk sistem yang berbasis Windows NT ada juga program semacam, misalnya program *Ballista* yang dapat diperoleh dari :
<http://www.secnet.com>
Selain program-program (tools) yang terpadu *(integrate*d) seperti yang terdapat pada daftar di atas, ada banyak program yang dibuat oleh hackers untuk melakukan "coba-coba". Program-program seperti ini, yang cepat sekali bermunculuan, biasanya dapat diperoleh (download) dari Internet melalui tempat-tempat yang berhubungan dengan keamanan, seperti misalnya "*Rootshel*l".