Rangkuman Sistem Terdistribusi

Bab 7 : Security
”Distributed Systems : Concepts and Design”
Goerge Coulouris, Jean Dollimore, Tim Kindberg

Oleh:

Ninan Kara Gicha Nasution 07/250814/TK/32408

Ega Reti Effendhy 07/252241/TK/32820

UNIVERSITAS GADJAH MADA
YOGYAKARTA

2009

Security

Ninan Kara Gicha N
Ega Reti Effendhy
Jurusan Teknik Elektro UGM

1. PENDAHULUAN

Tindakan Keamanan harus diterapkan didalam sistem komputer. Khususnya untuk system yang memegang masalah keuangan, rahasia atau informasi apapun yang rentan dicuri. Pada gambar dibawah, kami menyajikan evolusi mengenai kebutuhan system keamanan pada system computer dimulai dari tahun 60an dan 70an hingga masa sekarang yang tingkat bahayanya meningkat.

Didalam suatu organisasi biasanya dibuat semacam kebijakan keamanan yang mengatur masalah berbagi sumber daya computer. Kebijakan keamanan ini didukung oleh mekanisme keamanan dari organisasi itu sendiri. Misalnya akses suatu dokumen dibatasi hanya untuk kalangan tertentu saja. Memahami perbedaan mendasar antara kebijakan keamanan dengan mekanisme keamanan sangatlah membantu dalam merancang sebuah system keamanan.

	1965–75	*1975–89*	*1990–99*	*Current*
Platforms	Multi-user timesharing computers	Distributed systems based on local networks	The Internet, wide-area services	The Internet + mobile devices
Shared resources	Memory, files	Local services (e.g. NFS), local networks	Email, web sites, Internet commerce	Distributed objects, mobile code
Security requirements	User identification and authentication	Protection of services	Strong security for commercial transactions	Access control for individual objects, secure mobile code
Security management environment	Single authority, single authorization database (e.g. /etc/ passwd)	Single authority, delegation, repli-cated authorization databases (e.g. NIS)	Many authorities, no network-wide authorities	Per-activity authorities, groups with shared responsibilities

Gambar 1 Evolusi Kebutuhan Keamanan Sistem

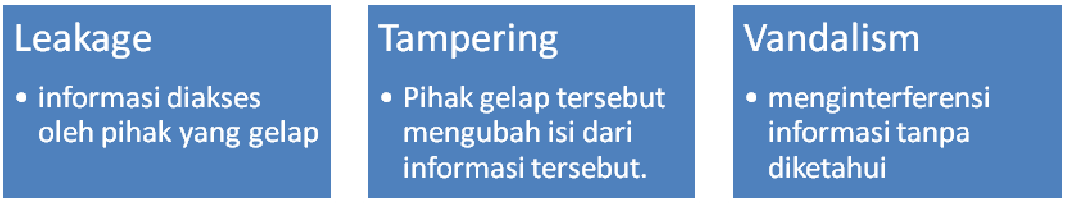
Model Keamanan yang telah dijelaskan pada bab 2 dapat disimpulkan sebagai berikut:

- Proses membungkus resources (seperti bahasa pemrograman level objek dan system lainnya yang berbasis resources) dan membiarkan client mengakses resources tersebut melalui interface mereka. Pengguna (user atau suatu proses) dapat diautorisasi secara eksplisit untuk mengoperasikan resources. Resources harus dilindungi dari akses yang illegal.
- Proses berinteraksi melalui jaringan yang di-share oleh banyak user. Musuh dapat mengakses jaringan. Mereka dapat menyalin atau mencoba membaca pesan apapun yang ditransmisikan melalui jaringan dan mereka pun dapat menambahkan pesan acak, mengubah arah jalur pesan dan sebagainya.

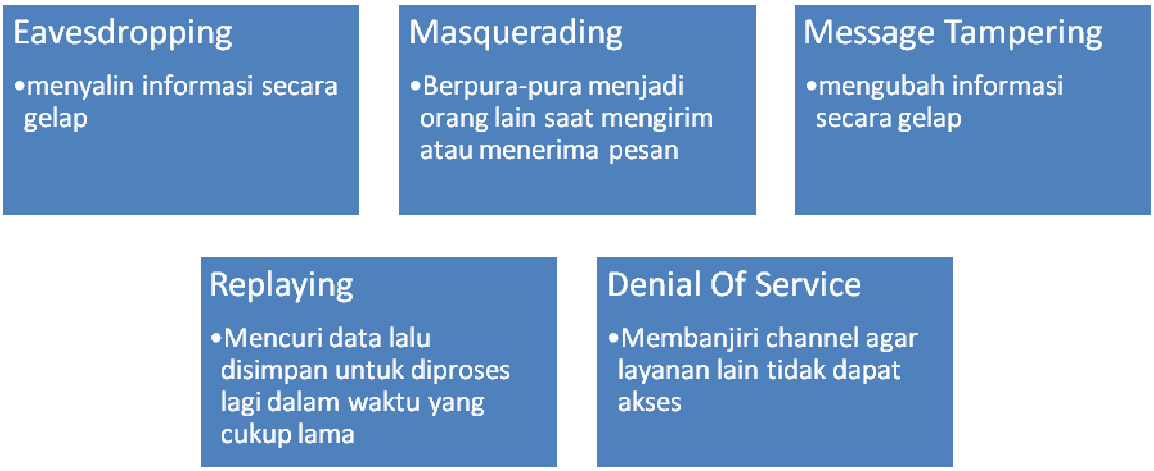
Dalam bab ini akan dibahas lebih jauh mengenai jenis-jenis serangan tersebut beserta teknik untuk menghadapinya.

1.1 Ancaman terhadap Keamanan dan Tipe Serangan

Ancaman tersebut dibagi menjadi 3 kelompok



Sedangkan metode serangan dapat dikelompokkan menjadi 5 jenis:



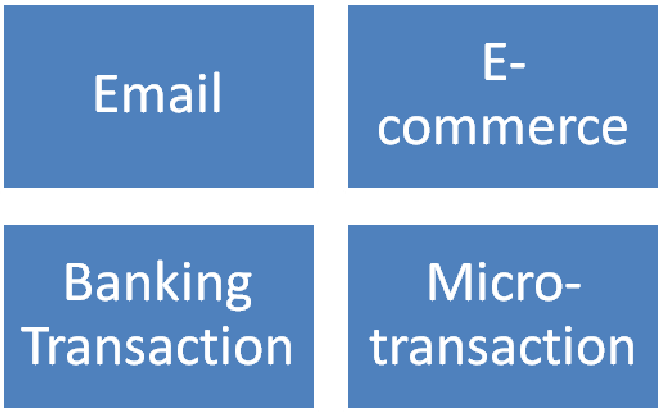
Ancaman untuk mobile code

Java Virtual Machine (JVM) dirancang untuk mobile. JVM memberikan setiap aplikasi sebuah environment khusus untuk mengeksekusi program. Setiap environment memiliki security manager yang menentukan resources yang tersedia untuk aplikasi.
JVM menyediakan 2 cara untuk melindungi environment lokal tersebut, yaitu:

- Kelas yang telah di- download disimpan terpisah dari kelas lokal, berguna untuk mencegah mereka menghapus kelas lokal tersebut.
- Bytecode dicek untuk validasi, berguna mencegah ada kode-kode yang eror

1.2 Mengamankan Transaksi Elektronik

Contoh dari Dokumen Elektronik yang sering dipakai antara lain:

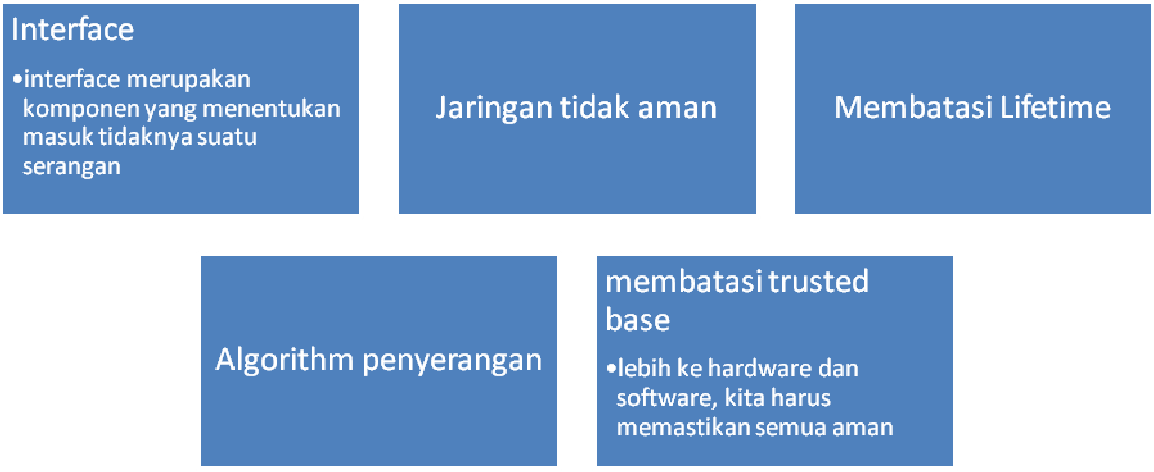


Kebijakan keamanan untuk dokumen-dokumen diatas antara lain:

- Mengautentikasi vendor agar pembeli yakin bahwa dia sedang bertransaksi langsung dengan vendor yang asli
- Menyimpan kartu kredit dan detail lainnya dari pembeli agar tidak jatuh ke tangan yang salah
- Jika bentuk dokumennya bisa di download, maka system harus memastikan bahwa konten tersebut sampai dengan selamat.
- Mengautentikasi identitas ke bank sebelum memberikan dapat mengakses ke data-data pembeli.

1.3 Merancang Sistem Keamanan

Dalam merancang suatu system keamanan yang baik, sebaiknya kita memikirkan kasus terburuknya. Berikut poin-poin yang harus dipikirkan untuk mendapatkan kasus terburuk:



2. OVERVIEW OF SECURITY TECHNIQUES

Penjelasan selanjutnya akan sering menggunakan istilah berikut

K_A	Alice's secret key
K_B	Bob's secret key
K_{AB}	Secret key shared between Alice and Bob
K_{Apriv}	Alice's private key (known only to Alice)
K_{Apub}	Alice's public key (published by Alice for all to read)
$\{M\}_K$	Message M encrypted with key K
$[M]_K$	Message M signed with key K

Gambar 2 Notasi Kriptografi

2.1 Kriptografi

Enkripsi adalah proses mengubah pesan menjadi rahasia. Kriptografi modern telah menyediakan algoritma untuk mengenkripsi dan mendekripsi suatu pesan. Saat ini dikenal 2 macam kunci, kunci public dan kunci privat. Dan dikenal 2 jenis algoritma untuk mengenkripsi:

- *Shared secret keys*, pengirim dan penerima harus saling berbagi kunci
- *Public/private key pairs*, pengirim membungkus pesan dengan kunci public dan penerima harus membukanya dengan kunci privat yang hanya diketahui oleh dirinya.

Untuk lebih jelasnya, akan di jelaskan pada bab-bab selanjutnya.

2.2 Penggunaan Kriptografi

Kriptografi digunakan pada aktivitas-aktivitas berikut:

- **Secrecy and integrity**

  Disaat dua pihak akan saling berkomunikasi secara privat, maka suatu proses kriptografi dibutuhkan. Kira-kira skenarionya seperti ini:
  - Alice menggunakan K_{AB} dan setuju atas fungsi enkripsi $E(K_{AB},M)$ untuk mengenkripsi dan mengirimnya ke Bob
  - Bob membaca pesan tersebut dengan mendekripsinya dengan fungsi $D(K_{AB},M)$

- **Authentication**

  Aktivitas ini adalah yang paling sering kita hadapi, skenarionya seperti berikut, scenario ini adalah saat berkomunikasi dengan server:
  - Alice mengirim pesan yang belum terenkripsi ke Sara (server), bertujuan untuk meminta tiket untuk Bob dan memastikan identitas Alice ini ke Sara
  - Sara merespon Alice dengan mengirimkan KA yang berisi tiket. Jadi pesan yang diterima Alice berbentuk {{Ticket} KB,KAB}KA
  - Alice mendekripsi respon tersebut dengan KA
  - Alice mengirim tiket ke Bob
  - Lalu Bob mendekripsi pesan tersebut dengan mendekripsi tiket yang berasal dari Sara terlebih dahulu

- **Digital Signatures**

  Proses ini akan dijelaskan lebih lanjut pada bagian selanjutnya. Digital Signature atau tanda tangan digital ini berfungsi untuk melindungi dokumen elektronik dari bermacam-macam serangan.

2.3 Sertifikat

Sertifikat memiliki fungsi yang sama dengan digital signature. Untuk membuat sertifikat digital ini berguna, ada beberapa hal penting yang dibutuhkan:

- Format standar
- Aggrement dalam pembangunan suatu sertifikat

2.4 Kontrol Akses

Proses mengontrol suatu resources dalam system terdistribusi

2.5 Credential

Credential adalah sekumpulan bukti yang disediakan oleh pusat saat meminta akses ke sebuah resources.

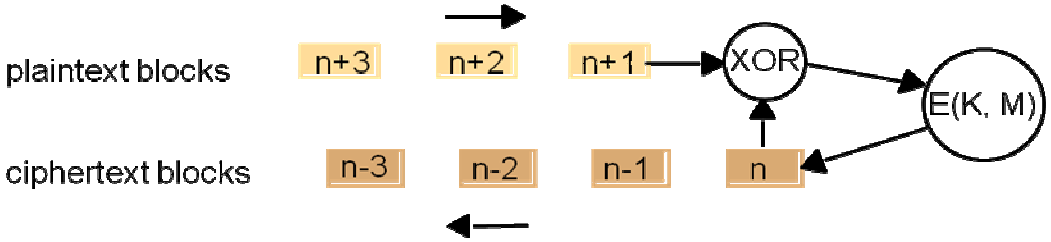
2.6 Firewall

Firewall berfungsi untuk melindungi intranet, menyaring aksi-aksi saat melakukan komunikasi dengan dunia luar. Firewall tidak terlalu efektif saat menghadapi serangan Denial Of Service, karena Firewall tidak sanggup mengurusi data-data yang membanjiri dirinya, karena Firewall termasuk single point defence.

3. CRYPTOGRAPHIC ALGORITHM

Dalam bagian ini akan dijelaskan secara rinci algoritma suatu kriptografi. Namun ada baiknya kita mengenal dasarnya terlebih dahulu:

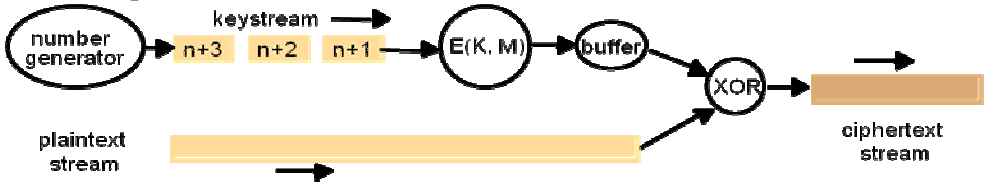
- **Cipher Block Chaining**



Gambar 3 Cipher Block Chaining

Block Cipher ini biasanya berukuran 64 bits. Sebuah pesan dibagi-bagi dalam beberapa blok. Pengirimannya dapat dilihat pada gambar diatas. Dalam Cipher Block Chaining, setiap plaintext block dikombinasikan dengan ciphertext block menggunakan operasi XOR sebelum dienkripsi.

- **Stream Cipher**



Gambar 4 Stream Cipher

Stream cipher ini biasanya digunakan untuk mengenkripsi data stream, seperti percakapan telepon. Stream cipher adalah algoritma enkripsi yang dilakukan secara incremental, merubah plaintext ke ciphertext. Hal ini sangat mudah dilakukan, kuncinya adalah pada keystream generator.

- **TEA (Tiny Encryption function) encryption function**
  Berikut algoritma dari proses enkripsi berdasarkan TEA

```
void encrypt(unsigned long k[], unsigned long text[]) {
    unsigned long y = text[0], z = text[1];                          1
    unsigned long delta = 0x9e3779b9, sum = 0; int n;                2
    for (n= 0; n < 32; n++) {                                        3
        sum += delta;                                                4
        y += ((z << 4) + k[0]) ^ (z+sum) ^ ((z >> 5) + k[1]);        5
        z += ((y << 4) + k[2]) ^ (y+sum) ^ ((y >> 5) + k[3]);        6
    }
    text[0] = y;  text[1] = z;                                       7
}
```

Gambar 5 TEA Encryption Function

3.1 Algoritma Simetris

Schneier [1996] menjelaskan lebih dari 25 algoritma simetris, kita hanya menjelaskan 3 diantaranya:

- **TEA**
  Setelah diatas sudah diperlihatkan algoritma untuk mengenkripsi, sekarang akan diperlihatkan untuk mendekripsinya, berikut:

```
void decrypt(unsigned long k[], unsigned long text[]) {
    unsigned long y = text[0], z = text[1];
    unsigned long delta = 0x9e3779b9, sum = delta << 5;  int n;
    for (n= 0; n < 32; n++) {
        z -= ((y << 4) + k[2]) ^ (y + sum) ^ ((y >> 5) + k[3]);
        y -= ((z << 4) + k[0]) ^ (z + sum) ^ ((z >> 5) + k[1]);
        sum -= delta;
    }
    text[0] = y; text[1] = z;
}
```

Gambar 6 TEA decryption Function

Berikut ini algoritma TEA yang dipakai :

```
void tea(char mode, FILE *infile, FILE *outfile, unsigned long k[]) {
/* mode is 'e' for encrypt, 'd' for decrypt, k[] is the key. */
    char ch, Text[8]; int i;
    while(!feof(infile)) {
        i = fread(Text, 1, 8, infile);        /* read 8 bytes from infile into Text
*/
        if (i <= 0) break;
        while (i < 8) { Text[i++] = ' ';}     /* pad last block with spaces */
        switch (mode) {
        case 'e':
            encrypt(k, (unsigned long*) Text); break;
        case 'd':
            decrypt(k, (unsigned long*) Text); break;
        }
        fwrite(Text, 1, 8, outfile);          /* write 8 bytes from Text to outfile
*/
    }
}
```

Gambar 7 TEA in Use

Algoritma singkat ini memiliki kelebihan aman dan enkripsi yang cepat.

- **DES**
  DES (Data Encryption Standard) dikembangkan oleh IBM. Standar DES, fungsi enkripsi memetakan 64 bit plaintext input menjadi 64 bit output terenkripsi menggunakan kunci 56 bit.

- **IDEA**
  IDEA (International Data Encryption Algorithm) merupakan pengembangan dari DES. Seperti TEA, IDEA menggunakan kunci 128 bit untuk mengenkripsi 64 bit block.

3.2 Algoritma Asimetris

Jenis-jenis algoritma asimetris:

- **RSA**
  RSA (Rivest, Shamir and Adelman) didesain untuk public key cipher. Berikut algoritma singkatnya
  Langkah-langkah algoritma RSA adalah sebagai berikut:
  - Pilih dua angka prima besar sebagai p dan q
  - Hitung modulus n yang merupakan perkalian kedua bilangan prima p dan q
    $n = pq$
  - Hitung fungsi Euler
    $\varphi(n) = (p-1)(q-1)$
  - Pilih kunci enkripsi e yang merupakan bilangan prima terhadap $\varphi(n) = (p-1)(q-1)$
  - Hitung kunci dekripsi d
    $d \equiv e^{-1} \pmod{\varphi(n)}$
  - Bilangan e dan n disebut kunci public
  - Bilangan d disebut kunci privat
    Untuk mengenkripsi pesan:
  - Pesan adalah m
  - Jadikan cipher text c dengan cara
    $c \equiv m^e \pmod{n}$
    Untuk mendekripsi cipher text
  - Chipher text adalah c

- Jadikan pesan asli m
  $m \equiv cd \pmod{n}$
  Berikut ini adalah contohnya:
  Jika ditentukan $p = 17$ dan $q = 31$, maka
  $n = pq = 17 \times 31 = 527$
  $\varphi(n) = (p-1)(q-1) = 16 \times 30 = 480$
  jika ditentukan $e = 7$, maka
  $d \equiv e{-}1 \pmod{\varphi(n)} \equiv 7{-}1 \pmod{480} \equiv 343$
  kunci deskripsi d dihitung menggunakan algoritma Euclid
  $ed \equiv 7 \times 343 \pmod{480} \equiv 2401 \pmod{480} \equiv 1$
  kunci public (e, n) diperlukan untuk mengenkripsi pesan m
  jika $m = 2$, maka enkripsinya adalah
  $c \equiv me \pmod{n} \equiv 27 \pmod{527} \equiv 128$
  untuk mendekripsi, diperlukan kunci privat d untuk membaca pesan m
  $m \equiv cd \pmod{n} \equiv 128343 \pmod{527} \equiv 2$

- **Elliptic Curve Algorithm**

  Algoritma ini dijelaskan secara rinci pada buku Menezes. Kunci yang diapakai diturunkan dari cabang-cabang matematika yang berbeda-beda. Tidak seperti RSA, tingkat keamanan tidak didasarkan pada sulitnya angka yang dipilih. Kunci yang terpendek justru yang paling aman.

3.3 Hybrid Cryptographic Protocols

Kriptografi kunci public ini sangat tepat untuk e-commerce karena tidak dibutuhkan mekanisme distribusi secret key. Tidak perlu untuk mengautentikasi kunci public, yang dibutuhkan hanya sertifikat digital yang dikirimkan bersama kunci. Penjelasanan lebih lanjut akan diperoleh pada studi kasus pada bab selanjutnya.

4. Digital Signature

- ***Digital signing***

Dokumen atau pesan elektronik M dapat ditandatangani oleh A dengan mengenkripsi kopian M dengan kunci K_A dan menyertakannya pada kopian *plaintext* M dan identitas A. Dokumen yang ditandatangani terdiri dari : M,A, $[M]K_A$

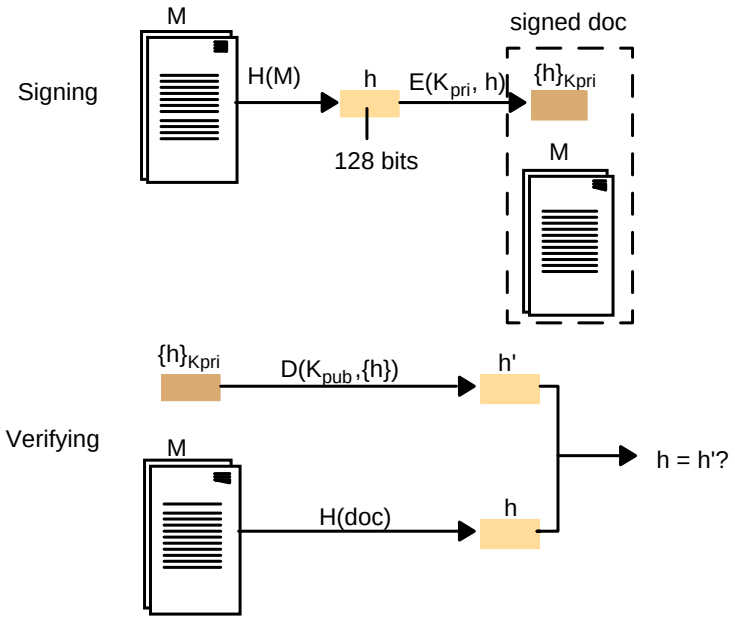
- ***Digest Function***

Digest Function juga disebut dengan *secure hash function* H(M)

4.1 Digital Signature dengan Kunci Publik

Berikut ini adalah metode untuk A menandatangani pesan M dan B memverifikasinya.

- A membuat sepasang kunci K_{pub} dan K_{priv} dan menyebarkan kunci public Kpub
- A membuat digest dari pesan M, H(M) menggunakan fungsi hash H dan mengenkripnya menggunakan kunci privat K_{priv} untuk mendapatkan tandatangan $S = \{H(M)\}\ K_{priv}$
- A mengirim pesan yang sudah ditandatangani $[M]K = M,S$ ke B
- B mendekrip S menggunakan K_{pub} dan membuat *digest* dari M, H(M). Jika cocok, tandatangan valid.

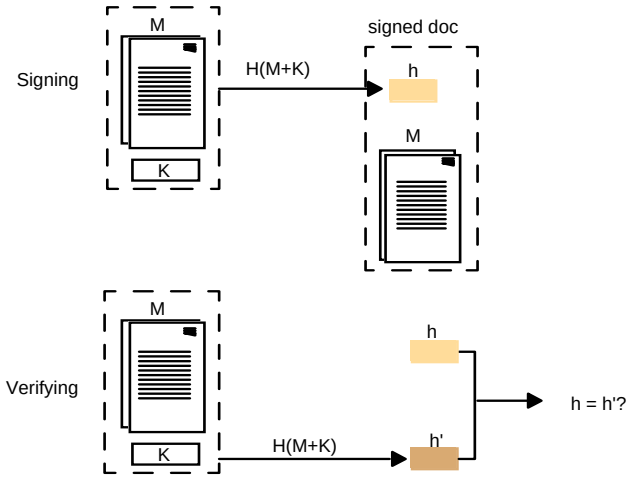


Gambar 8 Digital Signature dengan Kunci Publik

4.2 Digital Signature Dengan Kunci Privat-MAC

Pada proses ini, pesan yang dikirimkan tidak di enkripsi karenanya dikirimkan melalui kanal yang aman (*secure channel*). Karena kanal aman menyediakan komunikasi yang aman antara sepasang proses, kunci rahasia yang dibagi pakai dapat dilakukan dngan menggunakan metode *hybrid* (yang telah dijelaskan sebelumnya) untuk menghasilkan *low-cost signature* yang disebut *MAC (Message Autentication Codes)*. Prosesnya adalah sebagai berikut.

- A membuat kunci random atau acak K untuk tandatangan dan mendistribusikannya menngunakan kanal yang aman kepada orang yang butuh untuk memverifikasi pesan dari A
- Untuk setiap dokumen M yang akan ditandatangani, A menggabungkan M dan K lalu membuat digestnya h=H(M+K) dan mengirimkan dokumenyang sudah ditandatangani [M]K = M, h ke orang yang ingin memverifikasinya (h adalah MAC)
- Penerima, B menggabungkan kunci rahasia K dengan dokumen yang diterima M dan membentuk digest h' = H(M+K). Jika h = h' maka tandatangan valid.



Gambar 9 Low-cost Signature dengan Pertukaran Kunci Privat

4.3 Secure Digest Function

*MD5.*Menggunakan empat putaran, masing-masing melakukan satu dari empat fungsi nonlinier untuk masing-masing dari enam belas 32-bit segmen dari 512-bit blok dari sebuah text. Hasilnya adalah 128-bit digest.

SHA. Algoritma ini menghasilkan 160-bit digest. Algoritmanya berdasar pada MD4 dengan tambahan beberapa operasi. Tidak lebih cepat dari MD5 tetapi lebih aman dari serangan brute force dan birthday style.

4.4 Standar Sertifikat Dan Autoritas Sertifikat

X.509 .Sertifikat ini mengumpulkan kunci public dan entity bernama menjadi *subject.* Pengikatnya adalah tanda tangan yang dikeluarkan oleh issuer. Sertifikat tersebut memiliki periode validitas. Dibawah ini adalah format sertifikat X.509. Format ini ada dalam protocol SSL untuk mengautentikasi kunci public klien. Perusahaan atau organisasni ternama akan bertindak sebagai *certificate authorities* yang akan menyimpan dan menjamin data serfitikat dan pemiliknya jika ada yang melakukan verifikasi.

Subject	Distinguished Name, Public Key
Issuer	Distinguished Name, Signature
Period of validity	Not Before Date, Not After Date
Administrative information	Version, Serial Number
Extended Information	

Gambar 10 Format Sertifikat X.509

5. Studi Kasus:

5.1 Needham-Schroeder

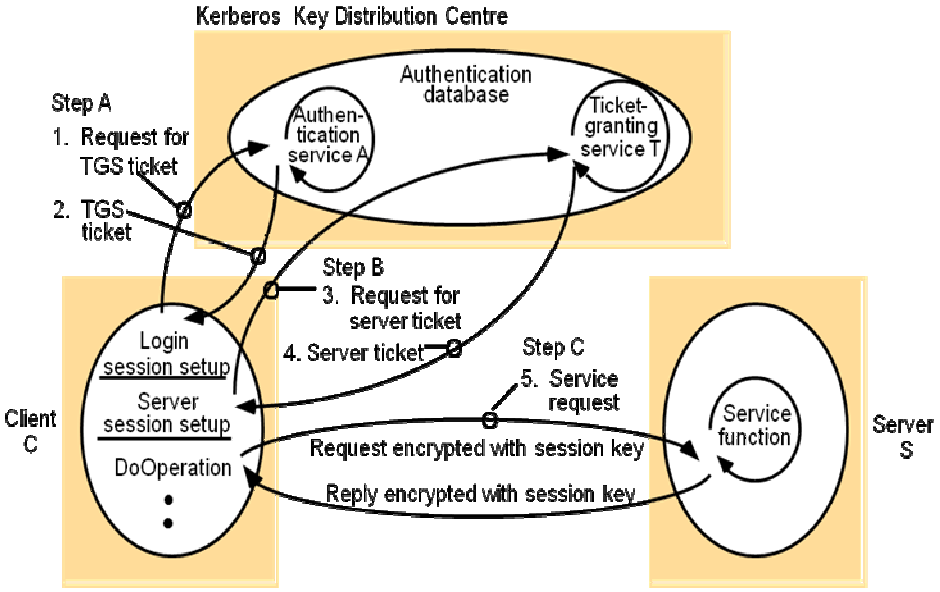
Protokol ini berdasar pada pembangkitan dan pentransmisian tiket oleh server autentikasi. Tiket adalah pesan terenkripsi berisi kunci rahasia yang digunakan untuk komunikasi antara A dan B. Gambar berikut menceritakan alur transmisi pesan.

Header	*Message*	*Notes*
1. A → S:	A, B, N_A	A requests S to supply a key for communication with B.
2. S → A:	$\{N_A, B, K_{AB}, \{K_{AB}, A\}_{K_B}\}_{K_A}$	S returns a message encrypted in A's secret key, containing a newly generated key K_{AB}, and a 'ticket' encrypted in B's secret key. The nonce N_A demonstrates that the message was sent in response to the preceding one. A believes that S sent the message because only S knows A's secret key.
3. A → B:	$\{K_{AB}, A\}_{K_B}$	A sends the 'ticket' to B.
4. B → A:	$\{N_B\}_{K_{AB}}$	B decrypts the ticket and uses the new key K_{AB} to encrypt another nonce N_B.
5. A → B:	$\{N_B - 1\}_{K_{AB}}$	A demonstrates to B that it was the sender of the previous message by returning an agreed transformation of N_B.

Gambar 11 Protokol Autentikasi Kunci Privat Needham-Schroeder

S adalah server autentikasi. N_A dan N_B adalah *nonces* yaitu nilai integer yang ditambahkan pada pesan untuk menandai kesegarannya (baru). Jika protocol ini dapat dijalankan dengan sempurna, komunikasi antara A dan B dilakukan dengan mengenkripsi pesan dengan K_{AB} yang hanya diketahui oleh mereka dan S. Kelemahan protocol ini adalah tidak dapat meyakinkan bahwa pesan 3, segar(baru) karena tanpa *nonces*. Kelemahan ini diatasi Kerberos

5.2 Kerberos



Gambar 12 Arsitektur Kerberos

Gambar di atas memperlihatkan arsitektur Kerberos. Kerberos mempunyai tiga macam objek keamanan :

- Tiket. Diberikan pada klien oleh *TGS* untuk presentasi ke server tertentu dan memverifikasi bahwa pengirim telah diautentikasi Kerberos. Tiket berisi waktu kadaluwarsa dan kunci session yang digunakan klien dan server.

- Autentikasi. Tanda yang dibuat klien dan dikirim ke server untuk membuktikan identitas klien yang berisi nama klien da pewaktuan yang dienkripsi dengan kunsi session.
- Kunci session. Kunci rahasia yang dibangkitkan acak oleh Kerberos diberikan ke klien untuk komunikasi dengan server tertentu.
  Tiket Kerberos memiliki periode yang tetap untuk waktu awal *t1* dan waktu berakhir *t2*. Tiket untuk klien C untuk mengakses server S adalah sebagai berikut.

  {C, S, t1, t2, KCS}KS atau {Tiket (C, S)}KS

  Berikut ini urutan penyampaian pesannya.

A.	*Obtain Kerberos session key and TGS ticket, once per login session*	
Header	*Message*	*Notes*
1. C → A: Request for TGS ticket	C, T, n	Client C requests the Kerberos authentication server A to supply a ticket for communication with the ticket-granting service T.
2. A → C: TGS session key and ticket	$\{K_{CT}, n\}_{K_C}$, $\{ticket(C,T)\}_{K_T}$ containing C, T, t_1, t_2, K_{CT}	A returns a message containing a ticket encrypted in its secret key and a session key for C to use with T. The inclusion of the nonce n encrypted in K_C shows that the message comes from the recipient of message 1, who must know K_C.

Notation:

A	Name of Kerberos authentication service.	n	A nonce.
T	Name of Kerberos ticket-granting service.	t	A timestamp.
C	Name of client.	t_1	Starting time for validity of ticket.
		t_2	Ending time for validity of ticket.

B.	*Obtain ticket for a server S, once per client-server session*	
3. C → T: Request ticket for service S	$\{auth(C)\}_{K_{CT}}$, $\{ticket(C,T)\}_{K_T}$, S, n	C requests the ticket-granting server T to supply a ticket for communication with another server S.
4. T → C: Service ticket	$\{K_{CS}, n\}_{K_{CT}}$, $\{ticket(C,S)\}_{K_S}$	T checks the ticket. If it is valid T generates a new random session key K_{CS} and returns it with a ticket for S (encrypted in the server's secret key K_S).

C.	*Issue a server request with a ticket*	
5. C → S: Service request	$\{auth(C)\}_{K_{CS}}$, $\{ticket(C,S)\}_{K_S}$, *request*, n	C sends the ticket to S with a newly generated authenticator for C and a request. The request would be encrypted in K_{CS} if secrecy of the data is required.

D.	*Authenticate server (optional)*	
6. S → C: Server authentication	$\{n\}_{K_{CS}}$	(Optional): S sends the nonce to C, encrypted in K_{CS}.

Gambar 13 Penyampaian Pesan pada Kerberos

Aplikasi Kerberos.

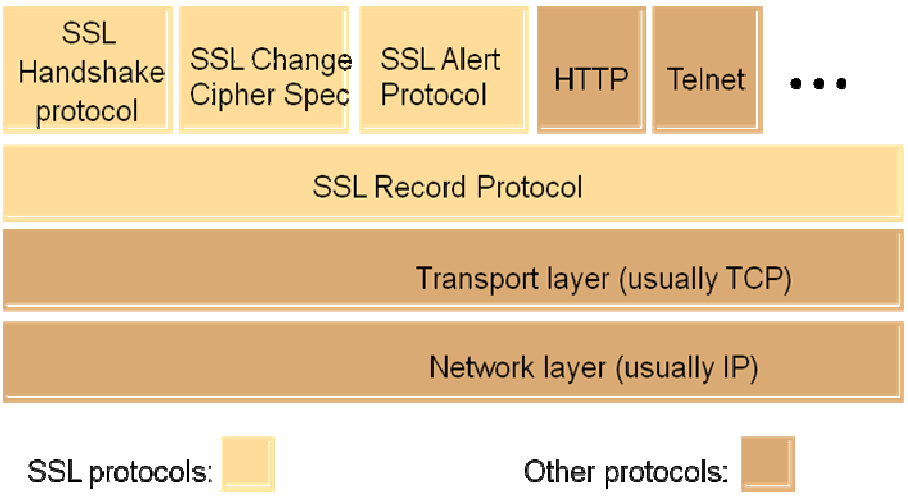
Login dengan Kerberos. Ketika pengguna mendaftar ke workstation. Program login mengirim namanya ke servis autentikasi Kerberos. Jika user dikenali, maka akan dijawab dengan kunci session dan nonce yang dienkripsi dengan password pengguna dan tiket untuk TGS. Program login kemuian mendekrip kunci session dan nonce dengan password pengguna.Jika password benar, pengguna dapat memulai sesi login di workstation.

Mengakses server dengan Kerberos. Kapanpun program yang berjalan di workstation membutuhkan akses ke servis yang baru, akan meminta tiket servis ke TGS.

Implementasi Kerberos. Kerberos diimplementasikan sebagai server yang menjalankan mesin yang aman menggunakan enkripsi DES. Servis Kerberos bersifat scalable-dunia dibagi menjadi domain autoritas autentikasi yang terpisah, disebut realm, masing-masing memiliki server Kerberos sendiri. Kebanyakan pengguna hanya terdaftar di satu realm tetapi Ticket Granting Server Kerberos terdaftar di semua realm. Pengguna dapat mengautentikasi diri sendiri pada server di realm lain melalui TGS mereka.

Kritik untuk Kerberos. Protocol versi 5 membiarkan nonces di autentikator diimplementasikan sebagai timestamp atau urutan angka. Padahal, nonces harus unik dan server memegang daftar nonces yang diterima dari klien untuk memastikan klen membalas. Selain itu keamanan Kerberos juga bergantung pada waktu hidup sesi yang terbatas.

5.3 SSL

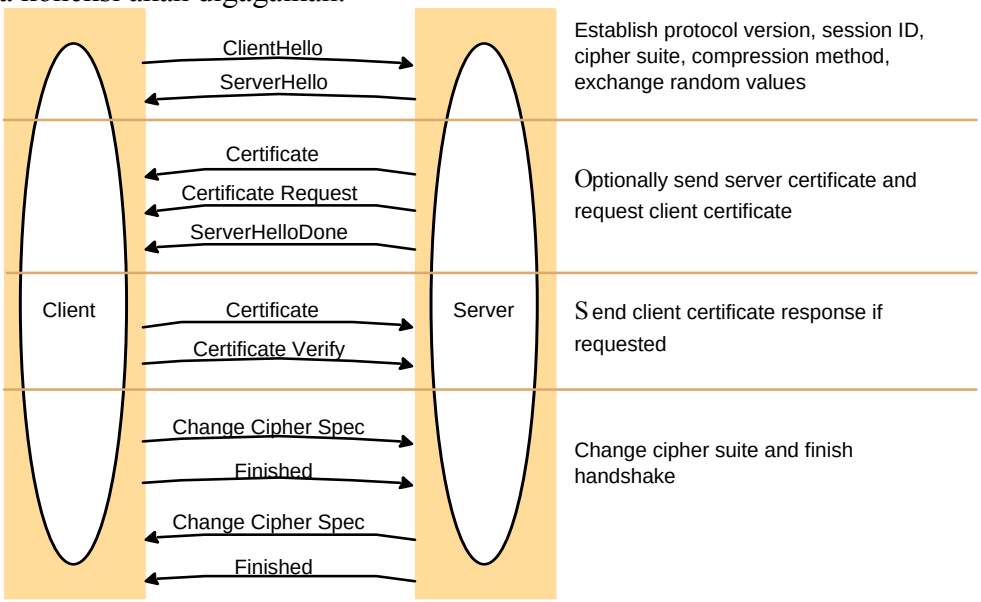


Gambar 14 Stack Protokol SSL

SSL terdiri dari dua layer (seperti pada gambar di atas) *SSL Record Protocol*, yang mengimplementasikan kanal aman, enkripsi dan autentikasi pesan yang

ditransmisikan melalui protocol yang *connection-oriented*; dan layer *handshake* yang berisi protocol handshake SSL dan dua protocol yang berhubungan yang menjaga sesi SSL antara klien dan server. SSL menyimpan protocol pada layer session-level, digunakan untuk mentransmisikan data secara transparan antara sepasang proses untuk menjamin integritas dan autentitasnya. Demikian tadi pengadaan kanal aman sebagai properti dari SSl. Walaupun begitu setiap direksi bebas menentukan apakah akan menggunakan dekripsi dan autentikasi atau tidak.

Algoritma enkripsi dan autentikasi yang dapat dinegosiasikan. Tidak semua software pada klien sama dengan pada server. SSL memiliki algoritma yang dapat dinegosiasikan sehingga jika salah satu pihak tidak menggunakan algoritma tersebut maka koneksi akan digagalkan.

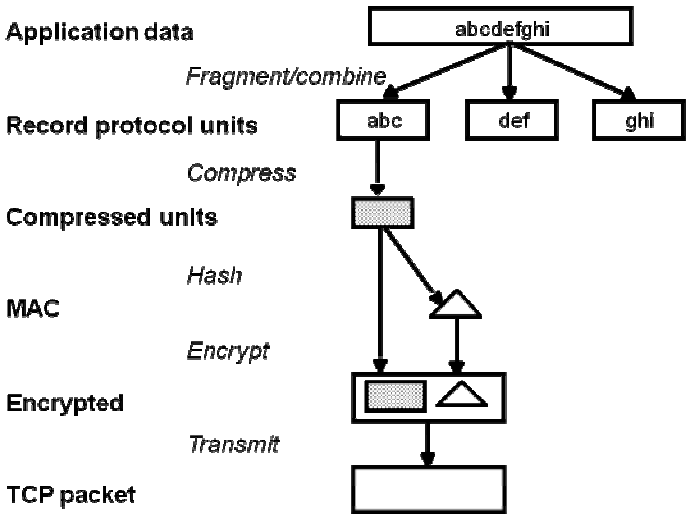


Gambar 15 Protokol handshake SSL

Protocol *handshake* pada SSL diilustrasikan pada gambar di atas. koneksi dimulai dengan pertukaran persetujuan dan parameter yang diperlukan untuk enkrisi dan autentikasi. Proses ini menggunakan kunci public untuk memverifikasi sertifikat. SSL menyediakan beberapa fungsi kriptografi yang disebut sebagai *chipper suite* yang kemudia dipilih oleh klien setelah ditawarkan server. Kemudian salah satu pihak mengirim *pre-master secret* ke pihak lain untuk membangkitkan dua kunci session untuk mengenkripsi pesan. Setelah itu session pun dimulai.

Component	*Description*	*Example*
Key exchange method	the method to be used for exchange of a session key	RSA with public-key certificates
Cipher for data transfer	the block or stream cipher to be used for data	IDEA
Message digest function	for creating message authentication codes (MACs)	SHA

Gambar 16 Konfigurasi Handshake SSL



Gambar 17 Protokol Record SSL

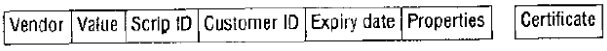
Seperti gambar di atas. sebelum ditransmisikan, pesan terlebih dahulu dibagi menjadi blok yang kemudian dikompres. Lalu blok dienkripsi sekaligus dengan MAC (tandatangan digital) –nya. Blok siap dikirim.

5.4 Milicent

Milicent adalah suatu protocol yang digunakan untuk transaksi elektronik. Protokol ini menggunakan *skrip* yaitu bentuk dari uang digital yang valid hanya untuk vendor tertentu. Scrip mempunyai beberapa vitur berikut:

- Memiliki nilai hanya bagi vendor spesifik
- Digunakan atau dihabiskan hanya sekali
- Tamper-resistant
- Hanya dapat digunakan oleh pemilih yang berhak
- Dapat diproduksi dan divalidasi secara efisien

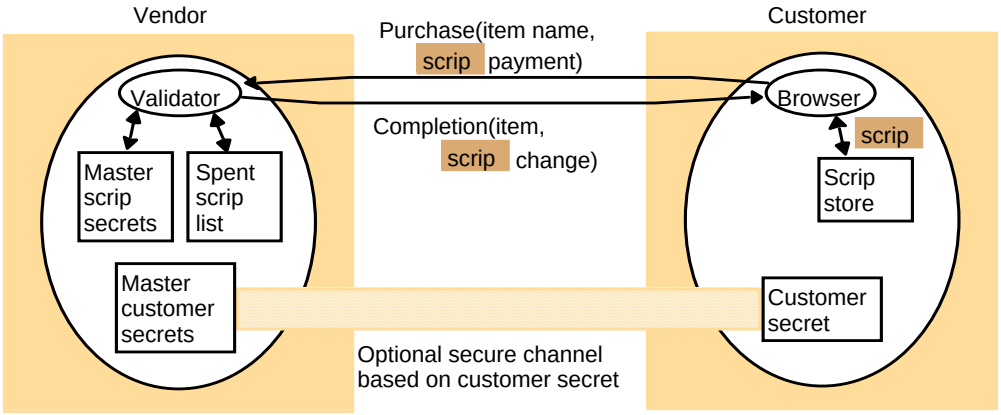
Skrip dibangkitkan dan didistribusikan oleh *Broker* yaitu server yang menukar skrip dengan uang cash. Kostumer dapat membeli skrip dari beberapa vendor pada satu broker. Skrip direpresentasikan dengan format berikut :



Gambar 18 Format Skrip

Scenario. Alice ingin membeli produk Venetia. Ia membeli skrip yang cocok pada Bob (broker) berdasar saran Venetia. Alice memesan produk pada Venetia dengan menyertakan skrip. Venetia memeriksa skrip apakah benar atas nama Alice. Jika nilai skrip amasih bersisa, Venetia akan mengirim skrip baru senilai sisanya kepada Alice.

Implementasi. Sebelum membangkitkan skrip, vendor membangkitkan *64 bit master skrip secret* yang kemudian digunakan untuk membuat sertifikat yang merupakan isi dari tanda tangan digital pada skrip.



Gambar 1 Arsitektur Millicent

6. Kesimpulan

Ancaman pada system Keamanan saat ini sangat marak terjadi. Untuk melindungi system tersebut maka diciptakanlah system kriptografi yang terdiri atas kunci public dan kunci privat. Dan yang paling penting dalam merancang suatu system keamanan adalah dengan memikirkan kasus-kasus terburuk yang dapat terjadi pada suatu system.