ENKRIPSI DAN DEKRIPSI

Kriptografi, secara umum adalah ilmu dan seni untuk menjaga kerahasiaan berita [bruce Schneier - Applied Cryptography]. Selain pengertian tersebut terdapat pula pengertian ilmu yang mempelajari teknik-teknik matematika yang berhubungan dengan aspek keamanan informasi seperti kerahasiaan data, keabsahan data, integritas data, serta autentikasi data [A. Menezes, P. van Oorschot and S. Vanstone - Handbook of Applied Cryptography]. Tidak semua aspek keamanan informasi ditangani oleh kriptografi.
Ada empat tujuan mendasar dari ilmu kriptografi ini yang juga merupakan aspek keamanan informasi yaitu :

1. Kerahasiaan, adalah layanan yang digunakan untuk menjaga isi dari informasi dari siapapun kecuali yang memiliki otoritas atau kunci rahasia untuk membuka/mengupas informasi yang telah disandi.
2. Integritas data, adalah berhubungan dengan penjagaan dari perubahan data secara tidak sah. Untuk menjaga integritas data, sistem harus memiliki kemampuan untuk mendeteksi manipulasi data oleh pihak-pihak yang tidak berhak, antara lain penyisipan, penghapusan, dan pensubsitusian data lain kedalam data yang sebenarnya.
3. Autentikasi, adalah berhubungan dengan identifikasi/pengenalan, baik secara kesatuan sistem maupun informasi itu sendiri. Dua pihak yang saling berkomunikasi harus saling memperkenalkan diri. Informasi yang dikirimkan melalui kanal harus diautentikasi keaslian, isi datanya, waktu pengiriman, dan lain-lain.
4. Non-repudiasi., atau nirpenyangkalan adalah usaha untuk mencegah terjadinya penyangkalan terhadap pengiriman/terciptanya suatu informasi oleh yang mengirimkan/membuat.

Algoritma Sandi

algoritma sandi adalah algoritma yang berfungsi untuk melakukan tujuan kriptografis. Algoritma tersebut harus memiliki kekuatan untuk melakukan (dikemukakan oleh Shannon):
konfusi/pembingungan (confusion), dari teks terang sehingga sulit untuk direkonstruksikan secara langsung tanpa menggunakan algoritma dekripsinya difusi / peleburan (difusion), dari teks terang sehingga karakteristik dari teks terang tersebut hilang.
sehingga dapat digunakan untuk mengamankan informasi. Pada implementasinya sebuah algoritmas sandi harus memperhatikan kualitas layanan/Quality of Service atau QoS dari keseluruhan sistem dimana dia diimplementasikan. Algoritma sandi yang handal adalah algoritma sandi yang kekuatannya terletak pada kunci, bukan pada kerahasiaan algoritma itu sendiri. Teknik dan metode untuk menguji kehandalan algoritma sandi adalah kriptanalisa.

Dasar matematis yang mendasari proses enkripsi dan dekripsi adalah relasi antara dua himpunan yaitu yang berisi elemen teks terang /plaintext dan yang berisi elemen teks sandi/ciphertext. Enkripsi dan dekripsi merupakan fungsi transformasi antara himpunan-himpunan tersebut. Apabila elemen-elemen teks terang dinotasikan dengan P, elemen-elemen teks sandi dinotasikan dengan C, sedang untuk proses enkripsi dinotasikan dengan E, dekripsi dengan notasi D.

Enkripsi : E(P) = C

Dekripsi : D(C) = P atau D(E(P)) = P

Secara umum berdasarkan kesamaan kuncinya, algoritma sandi dibedakan menjadi :

- ✓ kunci-simetris/symetric-key, sering disebut juga algoritma sandi konvensional karena umumnya diterapkan pada algoritma sandi klasik
- ✓ kunci-asimetris/asymetric-key

Berdasarkan arah implementasi dan pembabakan jamannya dibedakan menjadi

- ✓ algoritma sandi klasik classic cryptography
- ✓ algoritma sandi modern modern cryptography

Berdasarkan kerahasiaan kuncinya dibedakan menjadi :

- ✓ algoritma sandi kunci rahasia secret-key
- ✓ algoritma sandi kunci publik publik-key

Pada skema kunci-simetris, digunakan sebuah kunci rahasia yang sama untuk melakukan proses enkripsi dan dekripsinya. Sedangkan pada sistem kunci-asimentris digunakan sepasang kunci yang berbeda, umumnya disebut kunci publik(public key) dan kunci pribadi (private key), digunakan untuk proses enkripsi dan proses dekripsinya. Bila elemen teks terang dienkripsi dengan menggunakan kunci pribadi maka elemen teks sandi yang dihasilkannya hanya bisa didekripsikan dengan menggunakan pasangan kunci pribadinya. Begitu juga sebaliknya, jika kunci pribadi digunakan untuk proses enkripsi maka proses dekripsi harus menggunakan kunci publik pasangannya.

algoritma sandi kunci-simetris

Skema algoritma sandi akan disebut kunci-simetris apabila untuk setiap proses enkripsi maupun dekripsi data secara keseluruhan digunakan kunci yang sama. Skema ini berdasarkan jumlah data per proses dan alur pengolahan data didalamnya dibedakan menjadi dua kelas, yaitu block-cipher dan stream-cipher.

Block-Cipher

Block-cipher adalah skema algoritma sandi yang akan membagi-bagi teks terang yang akan dikirimkan dengan ukuran tertentu (disebut blok) dengan panjang t, dan setiap blok dienkripsi dengan menggunakan kunci yang sama. Pada umumnya, block-cipher memproses teks terang dengan blok yang relatif panjang lebih dari 64 bit, untuk mempersulit penggunaan pola-pola serangan yang ada untuk membongkar

kunci. Untuk menambah kehandalan model algoritma sandi ini, dikembangkan pula beberapa tipe proses enkripsi, yaitu :

- ✓ ECB, Electronic Code Book
- ✓ CBC, Cipher Block Chaining
- ✓ OFB, Output Feed Back
- ✓ CFB, Cipher Feed Back

Stream-Cipher

Stream-cipher adalah algoritma sandi yang mengenkripsi data persatuan data, seperti bit, byte, nible atau per lima bit(saat data yang di enkripsi berupa data Boudout). Setiap mengenkripsi satu satuan data di gunakan kunci yang merupakan hasil pembangkitan dari kunci sebelum.

Algoritma-algoritma sandi kunci-simetris

Beberapa contoh algoritma yang menggunakan kunci-simetris:

DES - Data Encryption Standard
blowfish
twofish
MARS
IDEA
3DES - DES diaplikasikan 3 kali
AES - Advanced Encryption Standard, yang bernama asli rijndael

Algoritma Sandi Kunci-Asimetris

Skema ini adalah algoritma yang menggunakan kunci yang berbeda untuk proses enkripsi dan dekripsinya. Skema ini disebut juga sebagai sistem kriptografi kunci publik karena kunci untuk enkripsi dibuat untuk diketahui oleh umum (public-key) atau dapat diketahui siapa saja, tapi untuk proses dekripsinya hanya dapat dilakukan oleh yang berwenang yang memiliki kunci rahasia untuk mendekripsinya, disebut private-key. Dapat dianalogikan seperti kotak pos yang hanya dapat dibuka oleh tukang pos yang memiliki kunci tapi setiap orang dapat memasukkan surat ke dalam kotak tersebut. Keuntungan algoritma model ini, untuk berkorespondensi secara rahasia dengan banyak pihak tidak diperlukan kunci rahasia sebanyak jumlah pihak tersebut, cukup membuat dua buah kunci, yaitu kunci publik bagi para korensponden untuk mengenkripsi pesan, dan kunci privat untuk mendekripsi pesan. Berbeda dengan skema kunci-simetris, jumlah kunci yang dibuat adalah sebanyak jumlah pihak yang diajak berkorespondensi.

Fungsi Enkripsi dan Dekripsi Algoritma Sandi Kunci-Asimetris

Apabila Ahmad dan Bejo hendak bertukar berkomunikasi, maka:

Ahmad dan Bejo masing-masing membuat 2 buah kunci

1. Ahmad membuat dua buah kunci, kunci-publik $K_{publik[Ahmad]}$ dan kunci-privat $K_{privat[Ahmad]}$
2. Bejo membuat dua buah kunci, kunci-publik $K_{publik[Bejo]}$ dan kunci-privat $K_{privat[Bejo]}$

Mereka berkomunikasi dengan cara:

1. Ahmad dan Bejo saling bertukar kunci-publik. Bejo mendapatkan $K_{publik[Ahmad]}$ dari Ahmad, dan Ahmad mendapatkan $K_{publik[Bejo]}$ dari Bejo.
2. Ahmad mengenkripsi teks-terang P ke Bejo dengan fungsi $C = E(P, K_{publik[Bejo]})$
3. Ahmad mengirim teks-sandi C ke Bejo
4. Bejo menerima C dari Ahmad dan membuka teks-terang dengan fungsi $P = D(C, K_{privat[Bejo]})$

Hal yang sama terjadi apabila Bejo hendak mengirimkan pesan ke Ahmad

1. Bejo mengenkripsi teks-terang P ke Ahmad dengan fungsi $C = E(P, K_{publik[Ahmad]})$
2. Ahmad menerima C dari Bejo dan membuka teks-terang dengan fungsi $P = D(C, K_{privat[Ahmad]})$

Algoritma -Algoritma Sandi Kunci-Asimetris
Knapsack
RSA - Rivert-Shamir-Adelman
Diffie-Hellman

Fungsi Hash Kriptografis

Fungsi hash Kriptografis adalah fungsi hash yang memiliki beberapa sifat keamanan tambahan sehingga dapat dipakai untuk tujuan keamanan data. Umumnya digunakan untuk keperluan autentikasi dan integritas data. Fungsi hash adalah fungsi yang secara efisien mengubah string input dengan panjang berhingga menjadi string output dengan panjang tetap yang disebut nilai hash.

Sifat-Sifat Fungsi Hash Kriptografi

Tahan preimej (Preimage resistant): bila diketahui nilai hash h maka sulit (secara komputasi tidak layak) untuk mendapatkan m dimana h = hash(m).

Tahan preimej kedua (Second preimage resistant): bila diketahui input m1 maka sulit mencari input m2 (tidak sama dengan m1) yang menyebabkan hash(m1) = hash(m2).
Tahan tumbukan (Collision-resistant): sulit mencari dua input berbeda m1 dan m2 yang menyebabkan hash(m1) = hash(m2)

Algoritma-Algoritma Fungsi Hash Kriptografi

Beberapa contoh algoritma fungsi hash Kriptografi:

- MD4
- MD5
- SHA-0
- SHA-1
- SHA-256
- SHA-512

Enkripsi Untuk Keamanan Data Pada Jaringan

Salah satu hal yang penting dalam komunikasi menggunakan computer untuk menjamin kerahasian data adalah enkripsi. Enkripsi dalah sebuah proses yang melakukan perubahan sebuah kode dari yang bisa dimengerti menjadi sebuah kode yang tidak bisa dimengerti (tidak terbaca). Enkripsi dapat diartikan sebagai kode atau chiper. Sebuah sistem pengkodean menggunakan suatu table atau kamus yang telah didefinisikan untuk mengganti kata dari informasi atau yang merupakan bagian dari informasi yang dikirim. Sebuah chiper menggunakan suatu algoritma yang dapat mengkodekan semua aliran data (stream) bit dari sebuah pesan menjadi cryptogram yang tidak dimengerti (unitelligible). Karena teknik cipher merupakan suatu sistem yang telah siap untuk di automasi, maka teknik ini digunakan dalam sistem keamanan komputer dan network.

Pada bagian selanjutnya kita akan membahas berbagai macam teknik enkripsi yang biasa digunakan dalam sistem sekuriti dari sistem komputer dan network.

A. Enkripsi Konvensional.

Proses enkripsi ini dapat digambarkan sebagai berikut :

Plain teks -> Algoritma Enkripsi -> Cipher teks ->Algoritma Dekrispsi -> Plain teks
User A | | User B
|————————-Kunci (Key) ————————|
Gambar 1

Informasi asal yang dapat di mengerti di simbolkan oleh Plain teks, yang kemudian oleh algoritma Enkripsi diterjemahkan menjadi informasi yang tidak dapat untuk

dimengerti yang disimbolkan dengan cipher teks. Proses enkripsi terdiri dari dua yaitu algoritma dan kunci. Kunci biasanya merupakan suatu string bit yang pendek yang mengontrol algoritma. Algoritma enkripsi akan menghasilkan hasil yang berbeda tergantung pada kunci yang digunakan. Mengubah kunci dari enkripsi akan mengubah output dari algortima enkripsi.

Sekali cipher teks telah dihasilkan, kemudian ditransmisikan. Pada bagian penerima selanjutnya cipher teks yang diterima diubah kembali ke plain teks dengan algoritma dan dan kunci yang sama.

Keamanan dari enkripsi konvensional bergantung pada beberapa faktor. Pertama algoritma enkripsi harus cukup kuat sehingga menjadikan sangat sulit untuk mendekripsi cipher teks dengan dasar cipher teks tersebut. Lebih jauh dari itu keamanan dari algoritma enkripsi konvensional bergantung pada kerahasian dari kuncinya bukan algoritmanya. Yaitu dengan asumsi bahwa adalah sangat tidak praktis untuk mendekripsikan informasi dengan dasar cipher teks dan pengetahuan tentang algoritma diskripsi / enkripsi. Atau dengan kata lain, kita tidak perlu menjaga kerahasiaan dari algoritma tetapi cukup dengan kerahasiaan kuncinya.

Manfaat dari konvensional enkripsi algoritma adalah kemudahan dalam penggunaan secara luas. Dengan kenyataan bahwa algoritma ini tidak perlu dijaga kerahasiaannya dengan maksud bahwa pembuat dapat dan mampu membuat suatu implementasi dalam bentuk chip dengan harga yang murah. Chips ini dapat tersedia secara luas dan disediakan pula untuk beberapa jenis produk. Dengan penggunaan dari enkripsi konvensional, prinsip keamanan adalah menjadi menjaga keamanan dari kunci.

Model enkripsi yang digunakan secara luas adalah model yang didasarkan pada data encrytion standard (DES), yang diambil oleh Biro standart nasional US pada tahun 1977. Untuk DES data di enkripsi dalam 64 bit block dengan menggunakan 56 bit kunci. Dengan menggunakan kunci ini, 64 data input diubah dengan suatu urutan dari metode menjadi 64 bit output. Proses yang yang sama dengan kunci yang sama digunakan untuk mengubah kembali enkripsi.

B. Enkripsi Public-Key

Salah satu yang menjadi kesulitan utama dari enkripsi konvensional adalah perlunya untuk mendistribusikan kunci yang digunakan dalam keadaan aman. Sebuah cara yang tepat telah diketemukan untuk mengatasi kelemahan ini dengan suatu model enkripsi yang secara mengejutkan tidak memerlukan sebuah kunci untuk didistribusikan. Metode ini dikenal dengan nama enkripsi public-key dan pertama kali diperkenalkan pada tahun 1976.

Plain teks -> Algoritma Enkripsi -> Cipher teks -> Algoritma Dekrispsi -> Plain teks
User A | | User B

Private Key B —-|
|————————-Kunci (Key) ————————|
Gambar 2

Algoritma tersebut seperti yang digambarkan pada gambar diatas. Untuk enkripsi konvensional, kunci yang digunakan pada prosen enkripsi dan dekripsi adalah sama. Tetapi ini bukanlah kondisi sesungguhnya yang diperlukan. Namun adalah dimungkinkan untuk membangun suatu algoritma yang menggunakan satu kunci untuk enkripsi dan pasangannya, kunci yang berbeda, untuk dekripsi. Lebih jauh lagi adalah mungkin untuk menciptakan suatu algoritma yang mana pengetahuan tentang algoritma enkripsi ditambah kunci enkripsi tidak cukup untuk menentukan kunci dekrispi. Sehingga teknik berikut ini akan dapat dilakukan :

Masing - masing dari sistem dalam network akan menciptakan sepasang kunci yang digunakan untuk enkripsi dan dekripsi dari informasi yang diterima.
Masing - masing dari sistem akan menerbitkan kunci enkripsinya (public key) dengan memasang dalam register umum atau file, sedang pasangannya tetap dijaga sebagai kunci pribadi (private key).

Jika A ingin mengisim pesan kepada B, maka A akan mengenkripsi pesannya dengan kunci publik dari B.

Ketika B menerima pesan dari A maka B akan menggunakan kunci privatenya untuk mendeskripsi pesan dari A.

Seperti yang kita lihat, public-key memecahkan masalah pendistribusian karena tidak diperlukan suatu kunci untuk didistribusikan. Semua partisipan mempunyai akses ke kunci publik (public key) dan kunci pribadi dihasilkan secara lokal oleh setiap partisipan sehingga tidak perlu untuk didistribusikan. Selama sistem mengontrol masing - masing private key dengan baik maka komunikasi menjadi komunikasi yang aman. Setiap sistem mengubah private key pasangannya public key akan menggantikan public key yang lama. Yang menjadi kelemahan dari metode enkripsi publik key adalah jika dibandingkan dengan metode enkripsi konvensional algoritma enkripsi ini mempunyai algoritma yang lebih komplek. Sehingga untuk perbandingan ukuran dan harga dari hardware, metode publik key akan menghasilkan performance yang lebih rendah. Tabel berikut ini akan memperlihatkan berbagai aspek penting dari enkripsi konvensional dan public key.

Enkripsi Konvensional

Yang dibutuhkan untuk bekerja :

Algoritma yang sama dengan kunci yang sama dapat digunakan untuk proses dekripsi - enkripsi.

Pengirim dan penerima harus membagi algoritma dan kunci yang sama.

Yang dibutuhkan untuk keamanan :

Kunci harus dirahasiakan.

Adalah tidak mungkin atau sangat tidak praktis untuk menerjemahkan informasi yang telah dienkripsi.

Pengetahuan tentang algoritma dan sample dari kata yang terenkripsi tidak mencukupi untu menentukan kunc.

Enkripsi Public Key

Yang dibutuhkan untuk bekerja :

Algoritma yang digunakan untuk enkripsi dan dekripsi dengan sepasang kunci, satu untuk enkripsi satu untuk dekripsi.

Pengirim dan penerima harus mempunyai sepasang kunci yang cocok.

Yang dibutuhkan untuk keamanan :

Salah satu dari kunci harus dirahasiakan.

Adalah tidak mungkin atau sangat tidak praktis untuk menerjemahkan informasi yang telah dienkripsi.

Pengetahuan tentang algoritma dan sample dari kata yang terenkripsi tidak mencukupi untu menentukan kunci.

Metode Enkripsi Simetris Rc4

Ketika internet menjadi salah satu media komunikasi yang banyak digunakan orang, sebagian orang kemudian berpikir untuk menjadikanya sebagai media untuk transaksi komersial semacan internet banking, e-comerce, dan lain sebagainya. Kebutuhan akan hal itu kemudian didukung dengan lahirnya berbagai metode ataupun algoritma – algoritma enkripsi untuk pengamanan data misalnya MD2,MD4,MD5,RC4,RC5, dan lain sebagainya. Pembakuan penulisan pada kriptografi dapat ditulis dalam bahasa matematika. Fungsi-fungsi yang mendasar dalam kriptografi adalah enkripsi dan dekripsi. Enkripsi adalah proses mengubah suatu pesan asli (plaintext) menjadi suatu pesan dalam bahasa sandi (ciphertext).

$$C = E(M)$$

dimana

M = pesan asli

E = proses enkripsi

C = pesan dalam bahasa sandi (untuk ringkasnya disebut sandi)

Sedangkan dekripsi adalah proses mengubah pesan dalam suatu bahasa sandi menjadi pesan asli kembali.

$$M = D(C)$$

D = proses dekripsi

Dalam setiap transaksi di internet , idealnya, setiap data yang ditransmisikan harusnya terjamin :

Integritas data

Jaminan integritas data sangat penting, sehingga data yang di kirimkan akan sama persis dengan data yang diterima, tanpa mengalami perubahan apapun pada selama ditransmisikan.

Kerahasiaan data

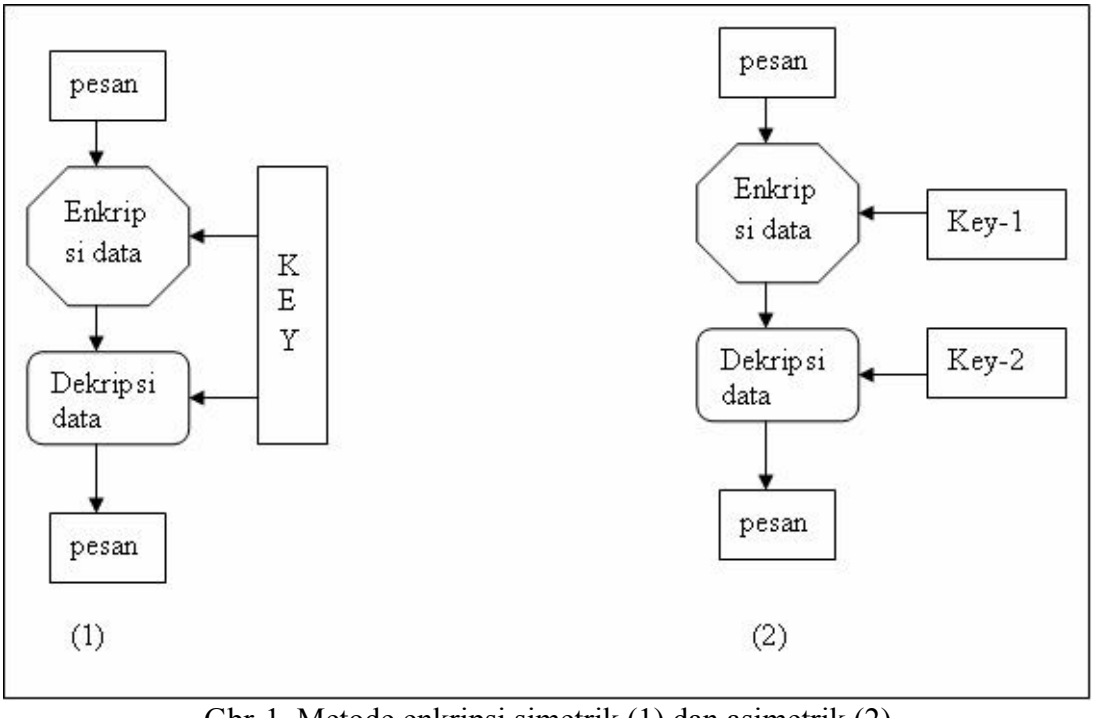
Jaminan kerahasiaan data juga penting karena dengan demikian tidak ada pihak lain yang bisa membaca data yang ada selama data tersebut ditransmisikan.

Otentikasi akse data

Mekanisme otentikasi akses data menjamin bahwa data ditransmisikan oleh pihak yang benar dengan tujuan transimisi yang benar pula.

Teknik kriptografi data untuk enkripsi ada dua macam yaitu:

1. Kriptografi simetrik, Dengan model kriptografi ini, data di enkripsi dan didekripsi dengan kunci rahasia yang sama.
2. Kriptografi asimetrik, Dengan model kriptografi ini, data dienkripsi dan didekripsi dengan kunci rahasia yang berbeda.pasangan kunci untuk enkripsi dan dekripsi dikenal dengan private key dan public key.



Gbr-1. Metode enkripsi simetrik (1) dan asimetrik (2)

Aplikasi kriptografi simetrik RC4 menggunakan Java

RC4 merupakan merupakan salah satu jenis stream cipher, yaitu memproses unit atau input data pada satu saat. Dengan cara ini enkripsi atau dekripsi dapat dilaksanakan pada panjang yang variabel. Algoritma ini tidak harus menunggu sejumlah input data tertentu sebelum diproses, atau menambahkan byte tambahan untuk mengenkrip. Metode enkripsi RC4 sangat cepat kurang lebih 10 kali lebih cepat dari DES.

Untuk melihat bagaimana metode enlripsi RC4 bekerja maka dalam tulisan ini dibuat aplikasi dengan menggunakan java, adapun source program tersebut adalah sebagai berikut :

Nama file : RC4Engine.java

————————-mulai————————-

```
class KeyParameter{
  private byte[] key;
  public KeyParameter(byte[] key){
    this(key,0,key.length);
  }
  public KeyParameter(byte[] key,int keyoff,int keyLen){
    this.key = new byte[keyLen];
    System.arraycopy(key,keyoff,this.key,0,keyLen);
  }
  public byte[] getKey(){
```

```
    return key;
  }
}

class EncRC4Engine{
  private final static int STATE_LENGTH = 256;
  private byte[] engineState = null,workingKey = null;
  private int x=0,y=0;
  private static final char[] kDigits = {'0','1','2','3','4','5','6','7','8','9','a','b','c','d','e','f'};
  EncRC4Engine(){}//constructor

  public void init(boolean forEncryption,KeyParameter params){
    if(params instanceof KeyParameter){
      workingKey = ((KeyParameter)params).getKey();
      setKey(workingKey);
      return;
    }
    throw new IllegalArgumentException("invalid parameter passed to RC4 init"+params.getClass().getName());
  }
public void processBytes(byte[] in,int inOff,int len,byte[] out,int outOff){
```

```
    if((inOff+len)>in.length){
      throw new RuntimeException(”output buffer too short”);
    }
    if((outOff+len)>out.length){
      throw new RuntimeException(”out put buffer too short”);
    }
    for (int i = 0 ; i < len ; i++){
      x = (x+1) & 0xff;
      y = ( engineState[x] + y ) & 0xff;
      //swap
      byte tmp = engineState[x];
      engineState[x] = engineState[y];
      engineState[y] = tmp;
      //xor
      out[i + outOff] = (byte)(in[i+inOff] ^ engineState[(engineState[x] +
engineState[y]) & 0xff]);
    }
  }
  public String bytesToHex(byte[] raw){
    int length = raw.length;
    char[]hex = new char[length*2];
    for (int i=0;i<length;i++){
```

```
    int value = (raw[i]+256) % 256;
    int highIndex = value >> 4;
    int lowIndex = value & 0×0f;
    hex[i*2+0] = kDigits[highIndex];
    hex[i*2+1] = kDigits[lowIndex];
  }
  return (new String(hex)).toString();
}
public void reset(){
  setKey(workingKey);
}
//private implementation
private void setKey(byte[] keyBytes){
  workingKey = keyBytes;
  x=0;y=0;
  if (engineState == null){
    engineState = new byte [STATE_LENGTH];
  }
  //reset the state of the engine
  for(int i = 0;i < STATE_LENGTH;i++){
    engineState[i]=(byte)i;
```

```
    }
    int i1=0;int i2=0;
    for(int i=0;i < STATE_LENGTH;i++){
      i2 = ((keyBytes[i1] & 0xff) + engineState[i]+ i2) & 0xff;
      //do the byte swap inline
      byte tmp = engineState[i];
      engineState[i]=engineState[i2];
      engineState[i2]=tmp;
      i1=(i1+1) % keyBytes.length;
    }
  }
}
```

————-selesai————-

nama file : TesRC4Engine.java

————-mulai————-

```
import java.io.*;

class tesRC4Engine {
  public static void main (String args[]){
    try{
      String myKey1, myKey2, text2Encrypt, text2Decrypt;
```

```
BufferedReader b;
byte[] ciphertext1;
byte[] ciphertext2;
EncRC4Engine s1 = new EncRC4Engine();
EncRC4Engine s2 = new EncRC4Engine();

b = new BufferedReader(new InputStreamReader(System.in));
System.out.print(”masukan data untuk encryption : “);
text2Encrypt = b.readLine();
System.out.print(”masukan kunci encryption : “);
myKey1 = b.readLine();
s1.init(true,new KeyParameter(myKey1.getBytes()));
ciphertext1 = new byte[text2Encrypt.length()];
s1.processBytes(text2Encrypt.getBytes(),0,text2Encrypt.length(),ciphertext1,0);
System.out.print(”hasil encrypt dari \”" + text2Encrypt);
System.out.println(”\” adalah ” + s1.bytesToHex(ciphertext1));

text2Decrypt = s1.bytesToHex(ciphertext1);
System.out.print(”masukan kunci untuk decrypt : “);
myKey2 = b.readLine();
s2.init(false,new KeyParameter(myKey2.getBytes()));
```

```
    ciphertext2 = new byte[ciphertext1.length];
    s2.processBytes(ciphertext1, 0, ciphertext1.length,ciphertext2,0);
    System.out.print("hasil decrypt dari \"" + text2Decrypt);
    System.out.println("\" adalah " + (new String(ciphertext2)).toString());
   }catch(IOException e){
    e.printStackTrace();
   }
  }
}
```

————selesai————

Kompilasi dan hasil eksekusi dari source program diatas afalah sebagai berikut:

```
C:\>java TesRC4Engine
masukan data untuk encryption : keamanan jaringan komputer
masukan kunci encryption : kjk
    hasil encrypt dari "keamanan jaringan komputer" adalah
    5ec874ee3cd7cd0b38dad37c7b684c0ccc589c2f1e689287a746

masukan kunci untuk decrypt : kjk
    hasil decrypt dari
    "5ec874ee3cd7cd0b38dad37c7b684c0ccc589c2f1e689287a746″ adalah
    keamanan jaringan komputer

C:\>java TesRC4Engine
masukan data untuk encryption : keamanan jaringan komputer
masukan kunci encryption : h
    hasil encrypt dari "keamanan jaringan komputer" adalah
    ca28c9e03687b13fd9f0273fb8c5d31eba8955371c82c3985647
masukan kunci untuk decrypt : kjk
    hasil decrypt dari
    "ca28c9e03687b13fd9f0273fb8c5d31eba8955371c82c3985647″ adalah
    ÿ…Ück>ZÁ@•1ªÃøsñ¢woš$k"s
```

Dari hasil diatas dapat diambil suatu analisi bahwa :

RC4 merupakan algoritma enkripsi simetris

Hal ini terbukti bahwa untuk melakukan enkripsi dan dekripsi diperlukan suatu kunci/key yang sama. Jika key untuk enkripsi dan key untuk dekripsi berbeda, hasil dekripsi tidak akan menghasilkan hasil yang sama dengan plain text yang sebenarnya. Hal ini dapat dilihat pada eksekusi program yang kedua.

Hasil enkripsi bergantung dari key

Hasil enkripsi dari plain text yang sama akan berbeda jika digunakan key atau kunci yang berbeda pula. Dengan kata lain hasil enkripsi merupakan fungsi dari plain text dan key.

Keamanan metode enkripsi RC4

Perbandingan kemanan metode enkripsi DES dengan RC4 menurut penulisnya pada ttp://www.geocities.com/amwibowo/resource/komparasi/komparasi.html adalah sebagai berikut :

Panjang kunci DES	Jaminan waktu untuk menemukan kunci
40-bit	0,4 detik
56-bit	7 jam
64-bit	74 jam 40 menit
128-bit	157.129.203.952.300.000 tahun

Tabel 1.1. Serangan brute-force pada DES

Panjang kunci RC4	Jaminan waktu untuk menemukan kunci
40-bit	15 hari
56-bit	2.691,49 tahun
64-bit	689.021,57 tahun
128-bit	12.710.204.652.610.000.000.000.000 tahun

Tabel 1.2. Serangan *brute-force* pada RC4

Dari tabel diatas dapat dilihat bahwa metode enkripsi dengan menggunkan RC4 masih lebih aman dibanding dengan metode enkripsi dengan DES.

"Analisa Kasus Hacking Situs Australia"
oleh : Donny B.U. M.Si *

Aksi pertama yang dilakukan oleh tarjo bukanlah aksi yang tergolong mahir/canggih dan tidak ada sangkut pautnya pernyataan sikap terhadap Australia. Yang dilakukan tarjo tersebut hanyalah "kebetulan" menemukan hole "hanya" di 1 server yang terletak di Australia, yaitu server milik perusahaan hosting ausinternet.com.au di IP 66.33.0.61. Jadi lantaran 1 server hostingnya tidak secure, maka puluhan situs yang berada dalam server itu secara otomatis terbuka/rawan untuk di-defaced. Jadi aksi tarjo tersebut bukanlah secara random memilih satu per-satu situs australia, tetapi kebetulan mengincarnya server hosting di Australia dan dia mendapatkan "1 pintu" untuk masuk ke banyak situs sekaligus.

Aksi tarjo tersebut tak lain hanyalah untuk mempromosikan dirinya atau komunitasnya. Seorang hacker yang menjebol suatu situs dengan tujuan "murni" untuk mengingatkan adminnya atau untuk tujuan "politik", dia tidak akan "menyapa" teman-temannya atau nama kelompoknya. Contohnya adalah aksi Fabian Clone dan K-Elektronik beberapa tahun lalu. Mereka hanya meninggalkan alamat e-mail mereka atau "hanya" nama kelompok mereka.

Sedangkan yang dilakukan tarjo adalah dengan menyapa teman-temannya (marshallz, pungky dan syzwz) dan menyebutkan nama tempat komunitasnya berkumpul (#cafeblue). Aksi ini adalah sekedar promosi nama channel mereka, serupa dengan aksi yang kerap dilakukan oleh kelompok #antihackerlink dan #medanhacking. Jadi pada awalnya ini bukan satu bentuk kepedulian hacker terhadap nasib Indonesia - Australia, tetapi mereka memanfaatkan isu tersebut untuk menaikkan nama mereka.

Saya akan postingkan beberapa data dari server IRC.
=====
-ChanServ- Info for □#cafeblue□:
-ChanServ- Founder : [CorLeoN] (~brandy@drinking.only.markbeer.com <mailto:~brandy@drinking.only.markbeer.com>)
-ChanServ- Registered : Tue 05/29/2001 13:57:55 GMT
-ChanServ- Last opping: Mon 11/04/2002 03:31:30 GMT
-NickServ- Info for tarjo:
-NickServ- Last seen address : dejava@202.152.12.153 <mailto:dejava@202.152.12.153> (isp IDOLA)
-NickServ- Last seen time : Mon 11/04/2002 13:36:56 GMT
-NickServ- Time registered : Mon 11/04/2002 13:36:56 GMT
-NickServ- Time now : Tue 11/05/2002 00:58:32 GMT
=====

Yang chanserv adalah data-data channel cafeblue di server DALnet. Yang nickserv adalah data nickname tarjo. Apakah tarjo ini adalah tarjo yang di cafeblue dan yang melakukan deface? Belum dapat dipastikan. karena tarjo "last seen" adalah kemaren. Jika saja pas saya masuk cafeblue ada nickname tarjo di dalamnya, maka hampir bisa dipastikan bahwa dialah tarjo "cafeblue" yang melakukan deface. kalau memang benar dia, maka dia bisa jadi berada di indonesia menggunakan ISP Idola.

Saya akan postingkan logs #cafeblue.

```
=====
Session Start: Tue Nov 05 07:59:29 2002
Session Ident: #cafeblue
[07:59] *** Now talking in #cafeblue
[07:59] *** Topic is 'CafeBlue□'
[07:59] *** Set by Rayvan on Thu Oct 17 12:56:49
[08:03] <desktop--> ada tarjo gak disini?
[08:03] <Buaya|Kurang|Ajar> desktop--: abis baca detik.com ya`
[08:04] * Buaya|Kurang|Ajar juga mau belajar deface site nya`....
[08:05] <desktop--> buaya siapa?
[08:05] <Buaya|Kurang|Ajar> orang`
[08:05] <desktop--> apakah tarjo sering kemari?
[08:05] <Buaya|Kurang|Ajar> yg pengen belajar`
[08:07] <Ini__budi> om TaRJo MaNa Yah?
[08:07] <Buaya|Kurang|Ajar> heran iks`
[08:07] <Buaya|Kurang|Ajar> pada join ke sini semua`
[08:07] <Buaya|Kurang|Ajar> ?
[08:07] * Ini__budi SeNaNG LiaT Dia NgeHaCk siTus aUsTraLI
[08:07] <Ini__budi> hahahhahaahhaha
[08:07] <Ini__budi> sYuKuR
[08:08] * Buaya|Kurang|Ajar juga`
[08:11] <Ini__budi> ah masak
[08:11] <Ini__budi> buktiin dong kayak om TaRjo
[08:12] <Buaya|Kurang|Ajar> maka nya ... aku join ke sini
[08:12] <Buaya|Kurang|Ajar> mau belajar`
[08:12] <Ini__budi> huahuahuahhahauhauhauhuhauhahauh
[08:12] <Ini__budi> wah kiranya mau belajar kesini Yah
=====
```

Log di atas mengasumsikan bahwa informasi tentang keberhasilan melakukan deface kerap memotivasi orang untuk belajar teknik hacking kepada pelakunya, dan hal tersebut dapat membuat channel yang sepi menjadi ramai. Hal tersebutlah yang

dilakukan oleh wenas saat pertama kali melakukan promosi #antihackerlink beberapa tahun silam.

Satu hal yang pasti, aksi balas-balasan ini akan merugikan pihak-pihak yang justru tidak ada kaitannya dengan kepentingan politik Indonesia - Australia ataupun kepentingan si hacker itu sendiri. Kemungkinan terburuk adalah kita akhirnya siap-siap saja situs-situs internet di Indonesia dan Australia akan kena aksi saling deface, yang sudah bergeser dari niatan awal untuk menyuarakan kepedulian (kalaupun memang benar) menjadi ajang adu gengsi antar hacker newbiee.

Motivasi awal memang bisa karena tersentuh patriotismenya, tetapi seperti "peperangan" yang sudah-sudah, ketika di medan pertempuran (battle field), yang ada di benak kita bukan lagi soal "patriotisme" dan "bela bangsa", tetapi bagaimana kita dan rekan seperjuangan kita bisa selamat dan tetap hidup seusai pertempuran tersebut. Sehingga, yang kita perjuangkan akhirnya adalah keselamatan diri kita dan rekan seperjuangan kita.

Demikian yang akan terjadi nanti pada aksi deface-men-deface. Patriotisme mungkin memang benar ada, tetapi pada proses rekruitmen awal. Ketika di tengah pertempuran "cyber" seperti saat ini, gengsi dan popularitas adalah hidup dan mati yang harus diperjuangkan. Sayangnya, yang jadi korban (atau dikorbankan) adalah "penduduk sipil".....