RANCANG BANGUN ENKRIPSI DENGAN METODE RC4 UNTUK KEAMANAN DATA DENGAN MENGGUNAKAN VISUAL BASIC 6.0

Eva Faja Ripanti, Alfred Nobel Maula
Dosen Universitas INDONUSA Esa Unggul
evaripanti@yahoo.com

Abstrak

Pada era teknologi informasi seperti saat ini, kebutuhan akan keamanan terhadap suatu data menjadi suatu hal yang sangat diperhatikan oleh para pemilik dan pengguna data dan informasi. Karena data yang berada pada sistem komputer atau bahkan pada suatu jaringan pada dasarnya tidak aman dan akan sangat mudah dibaca dan disadap oleh pihak yang tidak berhak.

Teknik enkripsi telah menjadi suatu alternatif dalam mengamankan suatu data dan informasi. Dengan data yang terenkripsi, data tidak akan mudah untuk dibaca karena data telah diacak sedemikian rupa dengan menggunakan algoritma RC4 data sehingga dapat mengurangi dampak dari berbagai ancaman keamanan data yang mungkin terjadi dan kunci enkripsi tertentu.

Kata Kunci : Kriptografi, Enkripsi, Dekripsi, RC4

Pendahuluan

Pemakaian teknologi komputer sebagai salah satu aplikasi dari teknologi informasi sudah menjadi suatu kebutuhan, karena banyak pekerjaan yang dapat diselesaikan dengan cepat, akurat, efektif dan efisien.

Seiring dengan meningkatnya industri komputer, maka untuk meningkatkan kenyamanan, data dapat disimpan pada komputer yang terhubung ke dalam *Local Area Networks* (LAN) atau *Internet.* Karena dengan demikian data dapat diakses dari *workstation* lain yang berada dalam LAN yang sama atau bahkan dapat diakses dari mana saja, sehingga peran komputer dalam menyimpan informasi pentingpun bertambah. Oleh sebab itu, untuk menjaga dari hal-hal yang tidak diinginkan maka faktor keamanan komputer menjadi suatu yang sangat penting dan harus diperhatikan.

Suatu sistem komputer dapat dikatakan aman apabila dalam segala keadaan, *resource* yang digunakan dan yang diakses sesuai dengan kehendak *user*. Sayangnya, tidak ada satu sistem komputer yang memiliki sistem keamanan yang sempurna. Setidaknya kita harus mempunyai suatu mekanisme yang dapat membuat pelanggaran jarang terjadi.

Namun pada kenyataannya, seringkali faktor keamanan ini diabaikan atau bahkan dihilangkan oleh para pemilik dan pengelola sistem informasi yang menganggap bahwa dengan peningkatan keamanan dapat mengganggu performansi dari sistem. Hal ini juga berkaitan dengan pendapat yang mengatakan bahwa faktor kenyamanan (*convenience*) berbanding terbalik dengan faktor keamanan dan

kenyataan bahwa faktor keamanan ini harus ditebus dengan biaya yang cukup tinggi.

Oleh sebab itu, untuk menjaga keamanan suatu data terdapat sebuah metode pengamanan data yang dikenal dengan nama kriptografi. Kriptografi merupakan salah satu metode pengamanan data yang dapat digunakan untuk menjaga kerahasiaan data, keaslian data, serta keaslian pengirim. Metode ini bertujuan agar informasi yang bersifat rahasia tidak dapat diketahui atau dimanfaatkan oleh orang yang tidak berkepentingan atau yang tidak berhak menerimanya.
Kriptografi biasanya dalam bentuk enkripsi. Proses enkripsi merupakan proses untuk meng-*encode* data dalam bentuk yang hanya dapat dibaca oleh sistem yang mempunyai kunci untuk membaca data tersebut. Proses enkripsi dapat dilakukan dengan menggunakan *software* atau *hardware*. Hasil enkripsi disebut *cipher*. *Cipher* kemudian didekripsi dengan *device* dan kunci yang sama tipenya (sama *software*nya serta sama kuncinya).

Pembahasan

Landasan Teori

Kriptografi (*cryptography*) merupakan ilmu dan seni untuk menjaga pesan agar aman. (Cryptography *is the art and science of keeping messages secure*). "*Crypto*" berarti "*secret*" (rahasia) dan "*graphy*" berarti "*writing*" (tulisan). Para pelaku atau praktisi kriptografi disebut *cryptographers*. Sebuah algoritma kriptografik (*Cryptographic algorithm*), disebut *cipher*, merupakan persamaan matematik yang digunakan untuk proses enkripsi dan deskripsi. Biasanya kedua persamaan matematik (untuk enkripsi dan dekripsi) tersebut memiliki hubungan matematis yang cukup erat.

Kriptografi adalah ilmu yang mempelajari bagaimana membuat suatu pesan yang dikirim pengirim dapat disampaikan kepada penerima dengan aman (Schneier, 1996).

Menurut Bruce Schneier dalam Applied Cryptography (John Wiley & Sons, 1996), "Kriptografi adalah seni dan ilmu untuk menjaga agar pesan rahasia tetap aman. Kriptografi merupakan salah satu cabang ilmu algoritma matematika". Para penggemar kriptografi sering disebut *cryptographer*, sedangkan kebalikannya adalah *crypt-analyst* yang berusaha memecahkan sandi kriptografi.

Proses yang dilakukan untuk mengamankan sebuah pesan (yang disebut *plaintext*) menjadi pesan yang tersembunyi (disebut *ciphertext*) adalah enkripsi (*encryption*). *Ciphertext* adalah pesan yang sudah tidak dapat dibaca dengan mudah. Menurut ISO 7498-2, terminologi yang lebih tepat digunakan adalah "*encipher*". Proses sebaliknya, untuk mengubah *ciphertext* menjadi *plaintext*, disebut dekripsi (*decryption*). Menurut ISO 7498-2, 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 *cryptanalysis*. *Cryptology* merupakan gabungan dari *Cryptography* dan *Cryptanalysis*.

Cryptographic System (*Cryptosystem*)

Cryptographic System atau *cryptosystem* adalah suatu fasilitas untuk mengkonversikan *plaintext* ke

ciphertext dan sebaliknya. Dalam sistem ini, seperangkat parameter yang menentukan transformasi pencipheran tertentu disebut suatu set kunci. Proses enkripsi dan dekripsi diatur oleh satu atau beberapa kunci.
kriptografi. Secara umum, kunci-kunci yang digunakan oleh proses pengenkripsian dan pendekripsian tidak perlu identik, tergantung pada sistem yang digunakan.

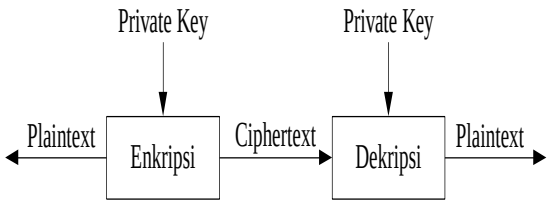
Suatu *cryptosystem* terdiri dari sebuah algoritma, seluruh kemungkinan *plaintext*, *ciphertext* dan kunci-kunci. Secara umum *Cryptosystem* dapat digolongkan menjadi dua, yaitu :

a. *Symmetric Cryptosystem*

Dalam *symmetric cryptosystem* ini, kunci yang digunakan untuk proses enkripsi dan dekripsi pada prinsipnya identik, tetapi satu buah kunci dapat pula diturunkan dari kunci yang lainnya. Kunci-kunci ini harus dirahasiakan. Oleh karena itulah sistem ini sering disebut sebagai sistem kripto kunci privat (*private key cryptosystem*). Jumlah kunci yang dibutuhkan umumnya adalah:

$$ {}_nC_2 = \frac{n \,.\, (n-1)}{2} $$

dengan n menyatakan banyaknya pengguna. Contoh dari sistem ini adalah RC4, Data Encryption Standard (DES), Blowfish, IDEA.

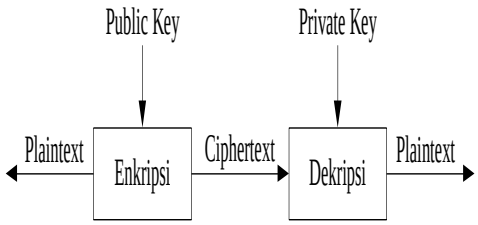


Sumber: Data Hasil Olahan

Gambar 1
Symmetric Cryptosystemb.

b. *Assymmetric Cryptosystem*

Dalam *assymmetric cryptosystem* ini digunakan dua buah kunci. Satu kunci yang disebut kunci publik (*public key*) dapat dipublikasikan, sedang kunci yang lain yang disebut kunci privat (*private key*) harus dirahasiakan. Proses menggunakan sistem ini dapat diterangkan secara sederhana sebagai berikut: bila A ingin mengirimkan pesan kepada B, A dapat menyandikan pesannya dengan menggunakan kunci publik B, dan bila B ingin membaca surat tersebut, ia perlu mendekripsikan surat itu dengan kunci privatnya. Dengan demikian kedua belah pihak dapat menjamin asal surat serta keaslian surat tersebut, karena adanya mekanisme ini. Sistem ini sering disebut sebagai sistem kripto kunci publik (*public key cryptosystem*). Contoh sistem ini antara lain RSA dan Merkle-Hellman Scheme.



Sumber: Data Hasil Olahan

Gambar 2
Assymmetric Cryptosystem

Setiap *cryptosytem* yang baik harus memiliki karakteristik sebagai berikut :

- Keamanan sistem terletak pada kerahasiaan kunci dan bukan pada kerahasiaan algoritma yang digunakan.
- *Cryptosystem* yang baik memiliki ruang kunci (*keyspace*) yang besar.
- *Cryptosystem* yang baik akan menghasilkan *ciphertext* yang terlihat acak dalam seluruh tes statistik yang dilakukan terhadapnya.

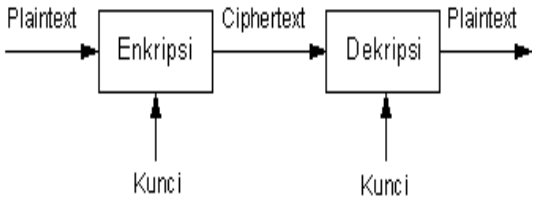
- *Cryptosystem* yang baik mampu menahan seluruh serangan yang telah dikenal sebelumnya.

Enkripsi

Salah satu hal yang sangat penting dalam menjaga kerahasiaan suatu data adalah dengan enkripsi. Enkripsi adalah sebuah proses yang melakukan perubahan sebuah kode dari yang bisa dimengerti menjadi sebuah kode yang tidak bisa dimengerti (tidak terbaca) (Anonim, 2003:43). Enkripsi dapat diartikan sebagai kode atau *cipher*. Sebuah sistem pengkodean menggunakan suatu tabel atau kamus yang telah didefinisikan untuk mengganti kata dari informasi atau yang merupakan bagian dari informasi yang dikirim. Sebuah *cipher* menggunakan suatu algoritma yang dapat mengkodekan semua aliran data (*stream*) bit dari sebuah pesan menjadi *cryptogram* yang tidak dimengerti.

Enkripsi dimaksudkan untuk melindungi informasi agar tidak terlihat oleh orang atau pihak yang tidak berhak. Informasi ini dapat berupa nomor kartu kredit, catatan penting dalam komputer, maupun *password* untuk mengakses sesuatu.

Enkripsi dibentuk berdasarkan suatu algoritma yang akan mengacak suatu informasi menjadi bentuk yang tidak bisa dibaca atau tidak bisa dilihat. Dekripsi adalah proses dengan algoritma yang sama untuk mengembalikan informasi teracak menjadi bentuk aslinya. Dengan enkripsi, data disandikan (*encrypted*) dengan menggunakan sebuah kunci (*key*). Untuk membuka (*decrypt*) data tersebut digunakan juga sebuah kunci yang dapat sama dengan kunci untuk mengenkripsi (untuk kasus *private key cryptography*) atau dengan kunci yang berbeda (untuk kasus *public key cryptography*).



Sumber: Data Hasil Olahan

Gambar 3. Proses enkripsi / deskripsi sederhana

Secara umum operasi enkripsi dan deskripsi dapat diterangkan secara matematis sebagai berikut :

EK (M) = C (proses Enkripsi)

DK (C) = M (proses Dekripsi)

Pada saat proses enkripsi kita menyandikan pesan M dengan satu kunci K lalu dihasilkan pesan C. Sedangkan pada proses dekripsi, pesan C tersebut diuraikan dengan menggunakan kunci K sehingga dihasilkan pesan M yang sama seperti pesan sebelumnya.

Dengan demikian keamanan suatu pesan tergantung pada kunci ataupun kunci-kunci yang digunakan, dan tidak tergantung pada algoritma yang digunakan sehingga algoritma-algoritma yang digunakan tersebut dapat dipublikasikan dan dianalisis, serta produk-produk yang menggunakan algoritma tersebut dapat diproduksi massal. Tidaklah menjadi masalah apabila seseorang mengetahui algoritma yang kita gunakan. Selama ia tidak mengetahui kunci yang dipakai, ia tetap tidak dapat membaca pesan.

Elemen dari Enkripsi

Ada beberapa elemen dari enkripsi yang akan dijabarkan di bawah ini: (Rahardjo, 2002).

a. Algoritma dari Enkripsi dan Dekripsi

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

Berdasarkan cara memproses teks (*plaintext*), *cipher* dapat dikategorikan menjadi dua jenis : *block cipher* dan *stream cipher*. *Block cipher* bekerja dengan memproses data secara blok, dimana beberapa karakter digabungkan menjadi satu blok. Setiap proses satu blok menghasilkan keluaran satu blok juga. Sementara itu *stream cipher* bekerja memproses masukan (karakter atau data) secara terus menerus dan menghasilkan data pada saat yang bersamaan.

Keamanan sebuah algoritma yang digunakan dalam enkripsi atau dekripsi bergantung kepada beberapa aspek. Salah satu aspek yang cukup penting adalah sifat algoritma yang digunakan. Apabila keluaran dari sebuah algoritma sangat tergantung kepada pengetahuan (tahu atau tidaknya) seseorang terhadap algoritma yang digunakan, maka algoritma tersebut disebut "*restricted algorithm*".

Apabila algoritma tersebut bocor atau 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 kurang aman, metoda pengamanan dengan *restricted algorithm* ini cukup banyak digunakan karena mudah implementasinya dan tak perlu diuji secara mendalam. Contoh penggunaan metoda ini adalah enkripsi yang menggantikan huruf yang digunakan untuk mengirim pesan dengan huruf lain.

b. Kunci yang digunakan dan panjang kunci

Kekuatan dari penyandian bergantung kepada kunci yang digunakan. Untuk itu, kunci yang lemah tersebut tidak boleh digunakan. Selain itu, panjangnya kunci yang biasanya dalam ukuran bit, 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* (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 diketahui.

c. *Plaintext*

Plaintext adalah pesan atau informasi yang akan dikirimkan dalam format yang mudah dibaca atau dalam bentuk aslinya.

d. *Ciphertext*

Ciphertext adalah pesan atau informasi yang sudah dienkripsi.

Jenis-jenis Metoda Kriptografi

Berikut adalah sebagian jenis-jenis metoda kriptografi yang dapat digunakan untuk melakukan enkripsi. Namun apabila akan menerapkan salah

satu dari metode enkripsi tersebut, maka harus dipilih metode enkripsi yang paling tepat untuk sistem informasi itu, sehingga tidak terjadi suatu kesalahan yang sangat besar mengingat yang akan diamankan adalah data yang pastinya sangat penting sekali.

1. Caesar Cipher

Caesar cipher adalah *cipher* pergeseran, dimana alfabet *ciphertext* diambil dari alfabet *plaintext*, dengan menggeser masing-masing huruf dengan jumlah tertentu (Anonim, 2003:51).

Caesar *cipher* digunakan oleh Julius Caesar dimana ia menjadi salah seorang yang pertama kali menggunakan enkripsi untuk meng-amankan pesannya dalam ber-komunikasi dengan para tentaranya.

Standar caesar *cipher* memiliki tabel karakter sandi yang dapat ditentukan sendiri. Ketentuan itu berdasarkan suatu kelipatan tertentu, misalnya tabel karakter sandi memiliki kelipatan tujuh dari tabel karakter aslinya. Contoh tabel karakter kelipata tujuh dapat dilihat pada tabel 2.1 di bawah ini.

Tabel 1. Tabel karakter kelipatan tujuh

Huruf asli	: 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
Huruf sandi	: h i j k l m n o p q r s t u v w x y z a b c d e f g

Sumber: Data Hasil Olahan

Dalam contoh ini huruf A diganti dengan huruf H, huruf B diganti huruf I, dan seterusnya sampai huruf Z diganti dengan huruf G. Dari sini kita bisa melihat bahwa pergeseran huruf menggunakan 7 huruf ke kanan.

Oleh sebab itu, jika dikirimkan berita asli "INDONUSA" akan menjadi "wlyihuhz". Ketentuan tabel karakter sandi dapat diubah sesuai dengan jumlah kelipatan dari huruf aslinya.

2. Enigma Cipher

Enigma *cipher* adalah suatu metode yang terkenal yang digunakan pada waktu Perang Dunia II bagi pihak Jerman. Waktu itu dikembangkan suatu metode atau model yang disebut dengan mesin Enigma. Mesin ini didasarkan pada sistem 3 rotor yang menggantikan huruf dalam *ciphertext* dengan huruf dalam *plaintext*. Rotor itu akan berputar dan menghasilkan hubungan antara huruf yang satu dengan huruf yang lain. Sehingga menampilkan berbagai substitusi seperti pergeseran caesar (Anonim, 2003:56).

Kondisi yang membuat Enigma kuat adalah putaran rotor. Karena huruf *plaintext* melewati rotor pertama, rotor pertama akan berputar 1 posisi. 2 rotor yang lain akan meninggalkan tulisan sampai rotor yang pertama telah berputar 26 kali (jumlah huruf dalam alfabet serta 1 putaran penuh). Kemudian rotor kedua akan berputar 1 posisi. Sesudah rotor kedua terus berputar 26 kali (26X26 huruf, karena rotor pertama harus berputar 26 kali untuk setiap waktu rotor kedua perputar), rotor ketiga akan berputar 1 posisi.

Siklus ini akan berlanjut untuk seluruh pesan yang dibaca. Dengan kata lain, hasilnya merupakan geseran yang digeser. Sebagai contoh, huruf s dapat disandikan sebagai huruf b dalam bagian pertama pesan, kemudian huruf m berikutnya dalam pesan. Dengan demikian, dari 26 huruf dalam

alfabet akan muncul pergeseran 26X26X26 yaitu 17576 posisi rotor yang mungkin.

3. Data Encryption Standard (DES)

Standard ini dibuat oleh National Beraue of Standard USA pada tahun 1977. DES menggunakan 56 bit kunci, algoritma enkripsi ini termasuk yang kuat dan tidak mudah diterobos. Cara enkripsi ini telah dijadikan standar oleh pemerintah Amerika Serikat sejak tahun 1977 dan menjadi standar ANSI tahun 1981.

Algoritma Enkripsi Data

Algoritma DES dirancang untuk menulis dan membaca berita blok data yang terdiri dari 64 bit di bawah kontrol kunci 64 bit (Anonim, 2003:59). Dalam pembacaan berita harus dikerjakan dengan menggunakan kunci yang sama dengan waktu menulis berita, dengan penjadwalan alamat kunci bit yang diubah sehingga proses membaca adalah kebalikan dari proses menulis.

Sebuah blok ditulis dan ditujukan pada permutasi dengan inisial IP, kemudian melewati perhitungan dan perhitungan tersebut sangat tergantung pada kunci kompleks dan pada akhirnya melewati permutasi yang invers dari permutasi dengan inisial IP^{-1}.

Perhitungan yang tergantung pada kunci tersebut dapat didefinisikan sebagai fungsi f, yang disebut fungsi *cipher* dan fungsi KS, yang disebut *Key Schedule*.

Sebuah deskripsi perhitungan diberikan pada awal, sepanjang algoritma yang digunakan dalam penulisan berita. Berikutnya, penggunaan algoritma untuk pembacaan berita didekripsikan. Akhirnya, definisi dari fungsi *cipher* f menjadi fungsi seleksi S_i dan fungsi permutasi adalah P.

4. Triple DES

Metoda ini dipakai untuk membuat DES lebih kuat lagi, yaitu dengan melakukan enkripsi DES tiga kali dengan menggunakan dua kunci yang berbeda.

Panjang kunci yang digunakan lebih panjang sehingga dapat mematahkan serangan yang tiba-tiba datang.

Triple DES ini merupakan model yang lain dari operasi DES yang mungkin lebih sederhana. Cara kerja dari model enkripsi ini adalah mengambil 3 kunci sebanyak 64 bit dari seluruh kunci yang mempunyai panjang 192 bit (Anonim, 2003:68). Triple DES memungkinkan pengguna memakai 3 sub kunci dengan masing-masing panjangnya 64 bit. Prosedur untuk enkripsi sama dengan DES, tetapi diulang sebanyak 3 kali. Data dienkrip dengan kunci pertama kemudian didekrip dengan kunci kedua dan pada akhirnya dienkrip lagi dengan kunci ketiga.

5. Rivest Code (RC4)

Algoritma RC4 ini dikembangkan oleh Ronald Rivest untuk RSA *data security* pada tahun 1987 dan baru dipublikasikan untuk umum pada tahun 1994. RC4 merupakan salah satu algoritma kunci simetris yang berbentuk *stream cipher* dimana algoritma ini melakukan proses enkripsi/dekripsi dalam satu byte dan menggunakan kunci yang sama.

RC4 menggunakan variabel yang panjang kuncinya dari 1 sampai 256 bit yang digunakan untuk

menginisialisasikan tabel sepanjang 256 bit. Tabel ini digunakan untuk generasi yang berikut dari *pseudo random bit* dan kemudian untuk menggenerasikan aliran *pseudo random* digunakan operasi XOR dengan *plaintext* untuk menghasilkan *ciphertext*. Masing-masing elemen dalam tabel ditukarkan mini mal sekali (Anonim, 2003:69).

Kunci RC4 seringkali dibatasi hingga 40 bit, namun kadang-kadang digunakan kunci 128 bit. Kunci RC4 ini memiliki kemampuan meng-gunakan kunci antara 1 sampai 2048 bit.

6. Gost Block Cipher

GOST ("Gosudarstvennyi Standard" = Standar Pemerintah) merupakan blok *cipher* dari bekas Uni Sovyet. Standar ini bernomor 28147-89 sehingga metode ini sering disebut sebagai GOST 28147-89.

GOST merupakan blok *cipher* 64 bit dengan panjang kunci 256 bit (Anonim, 2003:77). Algoritma ini mengiterasi algoritma enkripsi sederhana sebanyak 32 putaran (*round*). Untuk mengenkripsi pertama-tama *plaintext* 64 bit dipecah menjadi 32 bit bagian kiri, L dan 32 bit bagian kanan, R. Subkunci (*subkey*) untuk putaran I adalah K_i. Pada satu putaran ke-I operasinya adalah sebagai berikut:

$L_i = R_{i-1}$

$R_i = L_{i-1}$ xor $f(R_{i-1}, K_i)$

Sedangkan pada fungsi f, mula-mula bagian kanan data ditambah dengan subkunci key modulus 2^{32}. Hasilnya dipecah menjadi delapan bagian 4 bit dan setiap bagian menjadi input s-box yang berbeda. Di dalam GOST terdapat 8 buah s-box kemudian dikombinasikan menjadi bilangan 32 bit kemudian bilangan ini dirotasi 11 bit ke kiri. Akhirnya, hasil operasi ini di-xor dengan data bagian kiri yang kemudian menjadi bagian kanan dan bagian kanan menjadi bagian kiri (*swap*). Pada implementasinya nanti, rotasi pada fungsi f dilakukan pada awal saat inisialisasi sekaligus membentuk s-box 32 bit dan dilakukan satu kali saja sehingga lebih meng-hemat operasi dan mempercepat proses enkripsi/dekripsi.

7. RSA (Rivest Shamir Adleman)

RSA adalah singkatan dari huruf depan dari 3 orang yang menemukannya pada tahun 1977 di MIT, yaitu Ron Rivest, Adi Shamir, dan Len Adleman. Algoritma ini merupakan cara enkripsi publik yang paling kuat saat ini. Algoritma RSA melibatkan seleksi digit angka prima dan mengalikan secara bersama-sama untuk mendapatkan jumlah, yaitu n. Angka-angka ini dilewati algoritma matematis untuk menentukan kunci publik KU={e,n} dan kunci pribadi KR={d,n} yang secara matematis berhubungan (Anonim, 2003:82). Ini merupakan hal yang sulit untuk menentukan e dan atau d diberi n. Dasar inilah yang menjadi algoritma RSA.

Sekali kunci telah diciptakan, sebuah pesan dapat dienkrip dalam blok dan melewati persamaan berikut ini :

$$C = M^e \bmod n \quad (1)$$

di mana C adalah *ciphertext*, M adalah *plaintext*, sedangkan e adalah kunci publik penerima. Dengan demikian, pesan di atas dapat dienkrip dengan persamaan berikut :

$$C = M^d \bmod n \quad (2)$$

Di mana d adalah kunci pribadi penerima. Sebagai contoh, kita mengasumsikan bahwa M=19. Kita akan menggunakan angka 7 sebagai huruf p dan angka 17 sebagai huruf q. Jadi n=7X17=119. Kemudian, e dihitung menjadi 5 dan dihitung lagi menjadi 77. KU={5, 119} dan KR={77, 119}. Kita dapat melalui nilai yang dibutuhkan dengan persamaan (1) untuk mencari nilai C. Dalam hal ini C=66, kemudian hasil dekrip C(66) dapat digunakan untuk mendapatkan nilai *plaintext* yang asli. Untuk persamaan (2) juga mendapat nilai 19 dan *plaintext* yang asli.

Implementasi Algoritma RC4

Dalam mengimplementasikan algoritma RC4 ini, penulis menggunakan panjang kunci 16 bit. Disini penulis mengasumsikan bahwa pesan yang akan dienkripsi adalah "**UNGGUL**" dan kunci yang digunakan adalah "**INDONUSA**". Khusus pada kunci "INDONUSA" ini akan dilakukan pengulangkan untuk memenuhi panjang kunci 16 bit sehingga keseluruhan kunci yang akan digunakan dalam proses enkripsi/dekripsi adalah "**INDONUSAINDONUSA**". Nilai Ascii yang didapat dari kunci tersebut adalah "73 78 68 79 78 85 83 65 73 78 68 79 78 85 83 65" yang kemudian akan digunakan sebagai kunci dari enkripsi/dekripsi.

1. Fase Key Setup

Pertama ciptakan 16 bagian bit yang terdiri dari jumlah angka 0 sampai dengan 15.

Tabel 2. Tabel variabel awal

Si =	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	S0	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	S13	S14	S15

Sumber: Data Hasil Olahan

Kemudian ciptakan 16 kunci bit yang terdiri dari pengulangan 8 bit kunci utama untuk mengisi panjang kunci.

Tabel 3. Tabel kunci

Ki =	73	78	68	79	78	85	83	65	73	78	68	79	78	85	83	65
	K0	K1	K2	K3	K4	K5	K6	K7	K8	K9	K10	K11	K12	K13	K14	K15

Sumber: Data Hasil Olahan

Pada operasi pencampuran yang akan dilakukan, penulis menggunakan varibel i dan f untuk index S_i dan K_i. Pertama kita memberi nilai i dan f dengan 0. Operasi pencampuran adalah iterasi dari rumus ($f + S_i + K_i$) mod 16 yang diikuti dengan penukaran S_i dan S_f.

Iterasi ke-1:

Untuk i = 0 (0 + 0 + 73)

Mod 16 = 9 = f

f S_0 K_0

Tukar S_i dengan S_f (tukar S_0 dengan S_0)

Tabel 4. Tabel variabel iterasi ke-1

S_i =	**0**	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	S_0	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8	S_9	S_{10}	S_{11}	S_{12}	S_{13}	S_{14}	S_{15}

Sumber: Data Hasil Olahan

Iterasi ke-2:

Untuk i = 1 (0 + 1 + 78)

mod 16 = 15 = f

f S_1 K_1

Tukar S_1 dengan S_6

Tabel 5. Tabel variabel iterasi ke-2

Si =	0	**6**	2	3	4	5	**1**	7	8	9	10	11	12	13	14	15
	S0	**S1**	S2	S3	S4	S5	**S6**	S7	S8	S9	S10	S11	S12	S13	S14	S15

Sumber: Data Hasil Olahan

Iterasi ke-3:

Untuk i = 2 (6 + 2 + 68)

mod 16 = 12 = f

f S_2 K_2

Tukar S_2 dengan S_{10}

Tabel 6. Tabel variabel iterasi ke-3

S_i =	0	6	**10**	3	4	5	1	7	8	9	**2**	11	12	13	14	15
	S_0	S_1	**S_2**	S_3	S_4	S_5	S_6	S_7	S_8	S_9	**S_{10}**	S_{11}	S_{12}	S_{13}	S_{14}	S_{15}

Sumber: Data Hasil Olahan

Iterasi ke-4:

Untuk i = 3 (10 + 3 + 79)

mod 16= 12 = f

Tukar S_3 dengan S_{15}

Tabel 7. Tabel variabel iterasi ke-4

S_i =	0	6	10	**15**	4	5	1	7	8	9	2	11	12	13	14	**3**
	S_0	S_1	S_2	**S_3**	S_4	S_5	S_6	S_7	S_8	S_9	S_{10}	S_{11}	S_{12}	S_{13}	S_{14}	**S_{15}**

Sumber: Data Hasil Olahan

Iterasi ke-5:

Untuk i = 4 (15 + 4 + 78)

mod 16 = 1 = f

Tukar S_4 dengan S_4

Tabel 8. Tabel variabel iterasi ke-5

S_i =	0	6	10	15	**4**	5	1	7	8	9	2	11	12	13	14	3
	S_0	S_1	S_2	S_3	**S_4**	S_5	S_6	S_7	S_8	S_9	S_{10}	S_{11}	S_{12}	S_{13}	S_{14}	S_{15}

Sumber: Data Hasil Olahan

Iterasi ke-6:

Untuk i = 5 (4 + 5 + 85)

mod 16 = 14 = f

Tukar S_5 dengan S_7

Tabel 9. Tabel variabel iterasi ke-6

S_i =	0	6	10	15	4	**7**	1	**5**	8	9	2	11	12	13	14	3
	S_0	S_1	S_2	S_3	S_4	**S_5**	S_6	**S_7**	S_8	S_9	S_{10}	S_{11}	S_{12}	S_{13}	S_{14}	S_{15}

Sumber: Data Hasil Olahan

Iterasi ke-7:

Untuk i = 6 (7 + 1 + 83)

mod 16 = 11 = f

Tukar S_6 dengan S_9

Tabel 10. Tabel variabel iterasi ke-7

S_i =	0	6	10	15	4	7	**9**	5	8	**1**	2	11	12	13	14	3
	S_0	S_1	S_2	S_3	S_4	S_5	**S_6**	S_7	S_8	**S_9**	S_{10}	S_{11}	S_{12}	S_{13}	S_{14}	S_{15}

Sumber: Data Hasil Olahan

Iterasi ke-8:

Untuk i = 7 (9 + 5 + 65)

mod 16 = 15 = f

Tukar S_7 dengan S_1

Tabel 11. Tabel variabel iterasi ke-8

S_i =	0	**5**	10	15	4	7	9	**6**	8	1	2	11	12	13	14	3
	S_0	**S_1**	S_2	S_3	S_4	S_5	S_6	**S_7**	S_8	S_9	S_{10}	S_{11}	S_{12}	S_{13}	S_{14}	S_{15}

Sumber: Data Hasil Olahan

Iterasi ke-9:

Untuk i = 8 (1 + 8 + 73)

mod 16 = 2 = f

Tukar S_8 dengan S_9

Tabel 12. Tabel variabel iterasi ke-9

S_i =	0	5	10	15	4	7	9	6	**1**	**8**	2	11	12	13	14	3
	S_0	S_1	S_2	S_3	S_4	S_5	S_6	S_7	**S_8**	**S_9**	S_{10}	S_{11}	S_{12}	S_{13}	S_{14}	S_{15}

Sumber: Data Hasil Olahan

Iterasi ke-10:

Untuk i = 9 (9 + 8 + 78)

mod 16 = 15 = f

Tukar S_9 dengan S_6

Tabel 13. Tabel variabel iterasi ke-10

S_i =	0	5	10	15	4	7	**8**	6	1	**9**	2	11	12	13	14	3
	S_0	S_1	S_2	S_3	S_4	S_5	**S_6**	S_7	S_8	**S_9**	S_{10}	S_{11}	S_{12}	S_{13}	S_{14}	S_{15}

Sumber: Data Hasil Olahan

Iterasi ke-11:

Untuk i = 10 (6 + 2 + 68)
mod 16 = 12 = f
Tukar S_{10} dengan S_{10}

Tabel 14. Tabel variabel iterasi ke-11

S_i =	0	5	10	15	4	7	8	6	1	9	**2**	11	12	13	14	3
	S_0	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8	S_9	**S_{10}**	S_{11}	S_{12}	S_{13}	S_{14}	S_{15}

Sumber: Data Hasil Olahan

Iterasi ke-12:

Untuk i = 11 (10 + 11 + 79)
mod 16 = 4 = f
Tukar S_{11} dengan S_7

Tabel 15. Tabel variabel iterasi ke-12

S_i =	0	5	10	15	4	7	8	**11**	1	9	2	**6**	12	13	14	3
	S_0	S_1	S_2	S_3	S_4	S_5	S_6	**S_7**	S_8	S_9	S_{10}	**S_{11}**	S_{12}	S_{13}	S_{14}	S_{15}

Sumber: Data Hasil Olahan

Iterasi ke-13:

Untuk i = 12 (7 + 12 + 78)
mod 16 = 7 = f
Tukar S_{12} dengan S_4

Tabel 16. Tabel variabel iterasi ke-13

S_i =	0	5	10	15	**12**	7	8	11	1	9	2	6	**4**	13	14	3
	S_0	S_1	S_2	S_3	**S_4**	S_5	S_6	S_7	S_8	S_9	S_{10}	S_{11}	**S_{12}**	S_{13}	S_{14}	S_{15}

Sumber: Data Hasil Olahan

Iterasi ke-14:

Untuk i = 13 (4 + 13 + 85)
mod 16 = 6 = f
Tukar S_{13} dengan S_{15}

Tabel 17. Tabel variabel iterasi ke-14

S_i =	0	5	10	15	12	7	8	11	1	9	2	6	4	**3**	14	**13**
	S_0	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8	S_9	S_{10}	S_{11}	S_{12}	**S_{13}**	S_{14}	**S_{15}**

Sumber: Data Hasil Olahan

Iterasi ke-15:

Untuk i = 14 (15 + 14 + 83)
mod 16 = 0 = f

Tukar S_{14} dengan S_{14}

Tabel 18. Tabel variabel iterasi ke-15

S_i =	0	5	10	15	12	7	8	11	1	9	2	6	4	3	**14**	13
	S_0	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8	S_9	S_{10}	S_{11}	S_{12}	S_{13}	**S_{14}**	S_{15}

Sumber: Data Hasil Olahan

Iterasi ke-16:

Untuk i = 15 (14 + 13 + 65)

mod 16 = 12 = f

Tukar S_{15} dengan S_{14}

Tabel 19. Tabel variabel iterasi ke-16

S_i =	0	5	10	15	12	7	8	11	1	9	2	6	4	3	**13**	**14**
	S_0	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8	S_9	S_{10}	S_{11}	S_{12}	S_{13}	**S_{14}**	**S_{15}**

Sumber: Data Hasil Olahan

Setelah proses pencampuran selesai maka ciptakan sebuah bit acak untuk enkripsi. Karena algoritma RC4 merupakan algoritma stream cipher yang melakukan enkripsi/dekripsi bit per bit maka harus dihasilkan bit acak sebanyak panjang dari karakter pesan yang akan dienkrip/didekrip.

Beri nilai i dan f dengan 0 lalu beri nilai i = (i + 1) mod 16 dan beri nilai f = (f + S_i) mod 16. Kemudian, tukarkan S_i dengan S_f sehingga hasilnya menjadi: (S_i + S_f) mod 16. Hasil dari rumus ini kemudian disimpan dalam S_t. Jadi bila dikirimkan pesan yang berisi "**UNGGUL**" yang memiliki panjang 6 karakter dengan menggunakan kunci enkrip/dekrip "**INDONUSA**" maka akan menghasilkan *ciphertext* yang berisi "**WUMBLF**".

2. Fase Ciphering

Tukar S_i dengan S_f (tukar S_1 dengan S_5)

Tabel 20. Tabel bit acak ke-1

S_i =	0	**7**	10	15	12	**5**	8	11	1	9	2	6	4	3	13	14
	S_0	**S_1**	S_2	S_3	S_4	**S_5**	S_6	S_7	S_8	S_9	S_{10}	S_{11}	S_{12}	S_{13}	S_{14}	S_{15}

Sumber: Data Hasil Olahan

Tukar S_i dengan S_f (tukar S_2 dengan S_{15})

Tabel 21. Tabel bit acak ke-2

S_i =	0	7	**14**	15	12	5	8	11	1	9	2	6	4	3	13	**10**
	S_0	S_1	**S_2**	S_3	S_4	S_5	S_6	S_7	S_8	S_9	S_{10}	S_{11}	S_{12}	S_{13}	S_{14}	**S_{15}**

Sumber: Data Hasil Olahan

Tukar S_i dengan S_f (tukar S_3 dengan S_{14})

Tabel 22. Tabel bit acak ke-3

S_i =	0	7	14	**13**	12	5	8	11	1	9	2	6	4	3	**15**	10
	S_0	S_1	S_2	**S_3**	S_4	S_5	S_6	S_7	S_8	S_9	S_{10}	S_{11}	S_{12}	S_{13}	**S_{14}**	S_{15}

Sumber: Data Hasil Olahan

Tukar S_i dengan S_f (tukar S_4 dengan S_{10})

Tabel 23. Tabel bit acak ke-4

S_i =	0	7	14	13	**2**	5	8	11	1	9	**12**	6	4	3	15	10
	S_0	S_1	S_2	S_3	**S_4**	S_5	S_6	S_7	S_8	S_9	**S_{10}**	S_{11}	S_{12}	S_{13}	S_{14}	S_{15}

Sumber: Data Hasil Olahan

Tukar S_i dengan S_f (tukar S_5 dengan S_{15})

Tabel 24. Tabel bit acak ke-5

S_i =	0	7	14	13	2	**10**	8	11	1	9	12	6	4	3	15	**5**
	S_0	S_1	S_2	S_3	S_4	**S_5**	S_6	S_7	S_8	S_9	S_{10}	S_{11}	S_{12}	S_{13}	S_{14}	**S_{15}**

Sumber: Data Hasil Olahan

Tukar S_i dengan S_f (tukar S_6 dengan S_7)

Tabel 25. Tabel bit acak ke-6

S_i =	0	7	14	13	2	10	**11**	**8**	1	9	12	6	4	3	15	5
	S_0	S_1	S_2	S_3	S_4	S_5	**S_6**	**S_7**	S_8	S_9	S_{10}	S_{11}	S_{12}	S_{13}	S_{14}	S_{15}

Sumber: Data Hasil Olahan

Kelebihan dan Kelemahan Algoritma RC4

Semua algoritma enkripsi/ dekripsi memiliki kelebihan dan kelemahan masing-masing. Dalam proses enkripsi/dekripsi data dengan menggunakan metode RC4,. Berikut kelebihan dan kelemahan dari algoritma enkripsi/dekripsi RC4:

– Kelebihan:

- Para Cryptanalyst akan kesulitan mengetahui sebuah nilai dalam tabel.
- Para Cryptanalyst akan kesulitan mengetahui lokasi mana di dalam tabel yang digunakan untuk menyeleksi masing-masing nilai.
- Kecepatan proses enkripsi/dekripsi 10 kali lebih cepat dari algoritma DES.

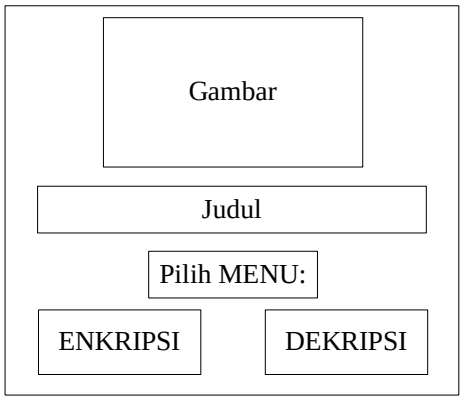
– Kelemahan:

- Algoritma RC4 mudah diserang dengan menggunakan analisis dari bagian dalam tabel.
- Salah satu dari 256 kunci dapat menjadi kunci yang lemah. Kunci ini diidentifikasi oleh kriptoanalisis yang dapat menemukan keadaan dimana salah satu dari bit yang dihasilkan mempunyai korelasi yang kuat dengan sedikit bit kunci.

Perancangan Input Output

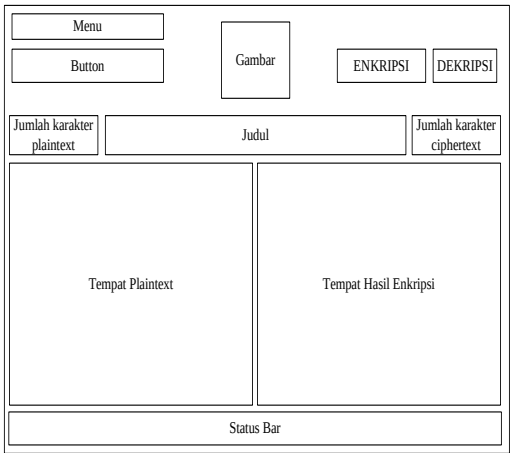
Dengan hasil rancangan *input output* ini penulis mengimplemen-tasikannya pada perancangan aplikasi dengan menggunakan *software* bantuan yaitu Microsoft Visual Basic 6.0 sehingga memudahkan penulis dalam membuat form-form yang diinginkan.

Untuk aplikasi enkripsi ini penulis membuat lima buah rancangan *input-output* yang nantinya akan diimplementasikan pada Microsoft Visual Basic 6.0. Lima buah rancangan *input output* itu adalah sebagai berikut:



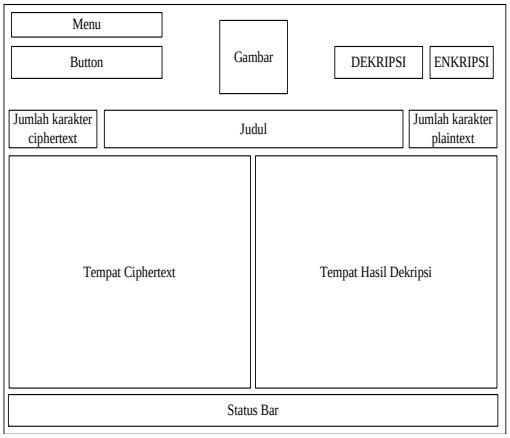
Sumber: Data Hasil Olahan

Gambar 4. Rancangan Form Menu Utama



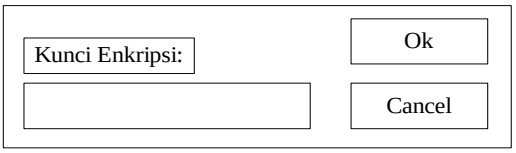
Sumber: Data Hasil Olahan

Gambar 5. Rancangan Form Enkripsi



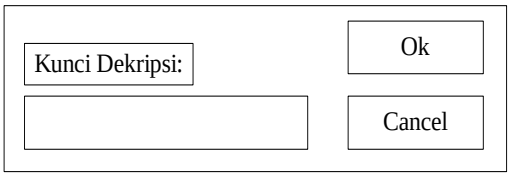
Sumber: Data Hasil Olahan

Gambar 6. Rancangan Form Dekripsi



Sumber: Data Hasil Olahan

Gambar 7. Rancangan Form Kunci Enkripsi



Sumber: Data Hasil Olahan

Gambar 8. Rancangan Form Kunci Dekripsi

Kesimpulan

Penggunaan aplikasi enkripsi untuk mengamankan data dan informasi menjadi suatu hal yang harus dilakukan bagi para pemilik dan pengguna data dan informasi apabila data dan informasi mereka tidak ingin diketahui oleh pihak-pihak yang tidak berhak.

Daftar Pustaka

Anonim, "Memahami Model Enkripsi dan Security Data", Andi Offset Yogyakarta, 2003.

Davis, Gordon B, "Management Information Systems", McGraw-Hill Book Co, Singapore, 1984.

Fathansyah, "Basis Data", Informatika, Bandung, 1999.

Halvorson, Michael, "Microsoft Visual Basic 6.0, Step by Step" Terjemahan oleh Adi kurniadi, PT. Elex Media Komputindo, 2000.

http://www.tedih.com/papers/p_kripto.html

http://id.wikipedia.org/wiki/kriptografi.html

INDOCISC, "Pengantar Kriptografi" www.indocisc.com, 2004.

Jogiyanto, "Analisis dan Disain Sistem Informasi", ANDI, Yogyakarta, 2001.

Kurniawan Tjandra, "Tip Trik Unik Visual Basic" Buku Ketiga, PT. Elex Media Komputindo, Jakarta, 2005.

McLeod, Jr. Raymond and George Schell, "*Management Information Systems",* Prentice Hall, New Jersey, 2001.

Rahardjo, Budi, "*Keamanan Sistem Informasi Berbasis Internet",* PT. Insan Infonesia, Bandung, 2002.

Schneier, Bruce, "*Applied Cryptography - Protocol, Algorithm, and Source Code in C",* Edisi Kedua, 1996.

Simmons, G.J, "*Contemporary Cryptology : The Science of Information Integrity,* IEEE Press, New York, 1993.

Sukmawan, Budi, *"RC4 Stream Cipher"*, 1998.

Supardi, Yuniar, "Pascal dan Flowchart Lewat Praktek", DINASTINDO, Jakarta, 2000.

Sutedjo, Budi, "Kamus++ Jaringan Komputer", ANDI, Yogyakarta, 2003.